F. Acar Savacı (Ed.)

LNAI 3949

# Artificial Intelligence and Neural Networks

14th Turkish Symposium, TAINN 2005
Izmir, Turkey, June 2005
Revised Selected Papers

Lecture Notes in Artificial Intelligence 3949

Edited by J. G. Carbonell and J. Siekmann

Subseries of Lecture Notes in Computer Science

F. Acar Savacı (Ed.)

# Artificial Intelligence and Neural Networks

14th Turkish Symposium, TAINN 2005
Izmir, Turkey, June 16-17, 2005
Revised Selected Papers

Springer

Series Editors

Jaime G. Carbonell, Carnegie Mellon University, Pittsburgh, PA, USA
Jörg Siekmann, University of Saarland, Saarbrücken, Germany

Volume Editor

F. Acar Savacı
Izmir Institute of Technology
Electrical-Electronics Engineering
Gülbahçe, Urla, Izmir, Turkey
E-mail: acarsavaci@iyte.edu.tr

Library of Congress Control Number: 2006929223

CR Subject Classification (1998): I.2, I.5, F.4.1, H.3, F.1

LNCS Sublibrary: SL 7 – Artificial Intelligence

ISSN 0302-9743
ISBN-10 3-540-36713-6 Springer Berlin Heidelberg New York
ISBN-13 978-3-540-36713-0 Springer Berlin Heidelberg New York



Typesetting: Camera-ready by author, data conversion by Scientific Publishing Services, Chennai, India
Printed on acid-free paper SPIN: 11803089 06/3142 5 4 3 2 1 0

# Preface

The Turkish Artificial Intelligence and Neural Network Symposium (TAINN) is an annual meeting where scientists present their new ideas and algorithms on artificial intelligence and neural networks with either oral or poster presentation. The TAINN-Turkish Conference on AI and NN Series started in 1992 at Bilkent University in Ankara, envisioned by various researchers in AI and NN then at the Bilkent, Middle East Technical, Boğaziçi and Ege universities as a forum for local researchers to get together and communicate. Since then, TAINN has been held annually around early summer. This year the 14th TAINN conference was organized by the EE and CE departments of the İzmir Institute of Technology with an emphasis on international contributions.

Among the 75 papers, 41 were accepted for oral presentation and 12 for poster presen-tation. In addition to the presentations, invited lectures were given by Burhan Türkşen (Knowledge/Intelligence Systems Laboratory, University of Toronto) and Joerg Siekmann (German Research Center for Artificial Intelligence DFKI, Saarland University), providing us with new developments in the field. We are very grateful to the contributions of such pioneer scientists.

The careful reviewing process by the well-recognized reviewers contributed to the quality of TAINN 2005, and therefore we thank them very much. We thank each member of the Organizing Committee and advisory board. We also thank Bora Mocan and Emre Çek, who are research assistants in the Electrical-Electronics Engineering Department of IYTE, for their efforts in the organization and in the preparation of the proceedings. We are grateful to the Turkish Scientific and Research Council of Turkey (TUBITAK) and the İzmir Branch of the Chamber of Electrical and Electronics Engineers (EMO) for their financial support.

Ferit Acar Savacı
Conference Chair

# Organizing Committee

## Chair

| | |
|---|---|
| Ferit Acar Savacı | Izmir Institute of Technology |

## Co-chair

| | |
|---|---|
| Kayhan Erciyes | Izmir Institute of Technology |

## Technical Program Chairs

| | |
|---|---|
| Bora Kumova | Izmir Institute of Technology |
| Bora Mocan | Izmir Institute of Technology |
| Deniz Çokuslu | Izmir Institute of Technology |
| Mehmet Emre Çek | Izmir Institute of Technology |

## Advisory Board

| | |
|---|---|
| Cem Say | Bogazici University |
| Cüneyt Güzeliş | Dokuz Eylul University |
| Ethem Alpaydın | Bogazici |
| H. Altay Güvenir | Bilkent |
| Kemal Oflazer | Sabancı |
| Uğur Halıcı | Middle East Technical University |

# Table of Contents

# A Case Study on Logging Visual Activities: Chess Game

Şükrü Ozan and Şevket Gümüştekin

Izmir Institute of Technology, Urla Izmir 35430 Turkey
sukruozan@iyte.edu.tr, sevketgumustekin@iyte.edu.tr
www.iyte.edu.tr

**Abstract.** Automatically recognizing and analyzing visual activities in complex environments is a challenging and open-ended problem. In this study this task is performed in a chess game scenario where the rules, actions and the environment are well defined. The purpose here is to detect and observe a FIDE (Fédération International des Èchecs) compatible chess board, generating a log file of the moves made by human players. A series of basic image processing operations have been applied to perform the desired task. The first step of automatically detecting a chess board is followed by locating the positions of the pieces. After the initial setup is established every move made by a player is automatically detected and verified. Intel® Open Source Computer Vision Library (OpenCV) is used in the current software implementation.

## 1 Introduction

Interpreting visual activities is an open-ended research problem in Computer Vision. The recent availability of necessary computational power to process vast amount of video data motivated research on several aspects of video processing. The temporal component of video introduces an additional dimension compared to static images enabling us extraction of meaningful events. In [1] temporal segmentation techniques are reviewed in a general terms. Several interrelated applications such as content based image retrieval [2], video annotation [3], video indexing [4] has been a major focus of research. Most of these applications are based on low level processing of pixel data. Since there is an unlimited number of visual events that need to be recognized it is impossible to expect a general purpose algorithm to extract the meaning of all the actions in a video stream. The high level information in video data can be better interpreted if the application domain is limited. Several successful studies has been carried out in sports video applications (e.g. [5][6]).

In this paper, a board game: chess has been considered as the domain of study. Compared to the other board games chess is more interesting. Besides its popularity it introduces some challenge as a vision problem with its three dimensional pieces and relatively complex rules. Most of the work on chess has been in the area of Artificial Intelligence aiming to create computer chess players. The progress of these studies ended up as a computer [7] defeating a human world champion. Other related work involves reading printed chess moves [8]. The studies similar to the one in this paper also attack the problem of interpreting a chess game in action but they are focused on

F.A. Savacı (Ed.): TAINN 2005, LNAI 3949, pp. 1 – 10, 2006.

designing and implementing robots [9][10] that can move pieces in the chess board. Here, our main concern is understanding the actions on the chess board.

By observing a standard chess board, the time instants where moves are performed by players have been detected, the moves are recognized using the visual data as well as the rules of the game and a log file of the moves is generated as a result. The camera is expected to be located with a clear view of the complete board and all pieces, but there is no further constraint on the perspective of the scene.

## 2 Chessboard Detection

A chessboard has a unique and uniform structure which makes it easily identifiable even in complex scenes. The first step of our scheme involves detection of the chessboard and extracting information about its position and orientation. A built in tool in Intel® Open Source Computer Vision Library (OpenCV) [11] is used to detect the chessboard and identify 49 inner corners.

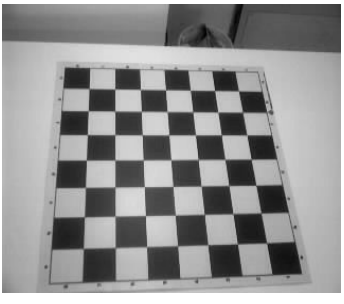


**Fig. 1.** Chess Board view example 1

The chessboard detection algorithm accepts grayscale images like the one in Figure 1. Initially, the algorithm applies a gray scale dilation operation [12]. By this operation touching corners of the squares are separated from each other. After the separation thresholding is applied to the image in order to convert the grayscale image to binary format. Generated binary image is used for a contour retrieving routine. While getting all the contour information, algorithm rejects contours with perimeters that are too small. So, contours with too small perimeters are regarded as noise in the image.

Contour analysis continues with searching for quadrangles. All the contours are checked and non-quadrangle contours are rejected. After these operations all the quadrangles and their corresponding contours in the source image are used for the next step which involves corner finding.

By using the retrieved contour plot as input, a corner finding algorithm is applied looking for sudden turns. The midpoint between the detected corners that are very close to each other are interpreted as the inner corners in the chessboard.

The 49 inner corners are verified and other corners are rejected by checking with the expected 7x7 uniform structure of the corner set.

## 3 Projection of the View

The only constraint on the position of the chessboard is that the angle between the surface normal of the board and the inverted viewing direction is sufficiently small (i.e. less than $30^o$). Each piece is expected to be visible. But, some undesired perspective effects are allowed. Among such effects are: slight occlusion and coverage of more than one squares by a single piece. In this section we describe how the view of the chess board is rectified.

After the inner corners of the chessboard are detected and ordered as in Figure 2, the far corner points labeled as 1,7,43 and 49 are used for the perspective projection.

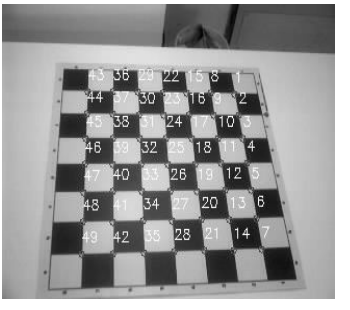


**Fig. 2.** Detected and labeled corners after chessboard detection algorithm applied

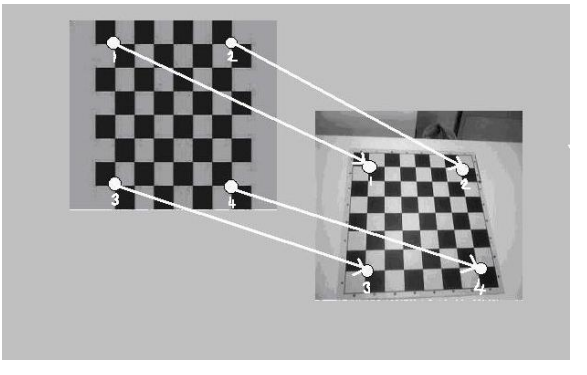



**Fig. 3.** 4 point correspondences for remapping

### 3.1 Finding the Parameters of the Perspective Transformation

A perspective projection can be formulized by the homographies [13]:

$$x_{im} = \frac{a * X + b * Y + c}{g * X + h * Y + 1}$$

$$y_{im} = \frac{d * X + e * Y + f}{g * X + h * Y + 1} \tag{1}$$

Where in a backwards projection scenario (X,Y) are the coordinates on the desired projected view image and ($x_{im}$,$y_{im}$) is the coordinates on the source image. Matching 4 points (8 coordinate parameters) in the target and source images allow us to find the unknown parameters of the perspective transformation.

8 linearly independent equations can be written in matrix representation as:

$$\begin{bmatrix} X_1 & Y_1 & 1 & 0 & 0 & 0 & -X_1 * x_{im1} & -Y_1 * x_{im1} \\ X_2 & Y_2 & 1 & 0 & 0 & 0 & -X_2 * x_{im2} & -Y_1 * x_{im2} \\ X_3 & Y_3 & 1 & 0 & 0 & 0 & -X_3 * x_{im3} & -Y_1 * x_{im3} \\ X_4 & Y_4 & 1 & 0 & 0 & 0 & -X_4 * x_{im4} & -Y_1 * x_{im4} \\ 0 & 0 & 0 & X_1 & Y_1 & 1 & -X_1 * y_{im1} & -Y_1 * y_{im1} \\ 0 & 0 & 0 & X_2 & Y_2 & 1 & -X_1 * y_{im2} & -Y_1 * y_{im2} \\ 0 & 0 & 0 & X_3 & Y_3 & 1 & -X_1 * y_{im3} & -Y_1 * y_{im3} \\ 0 & 0 & 0 & X_4 & Y_4 & 1 & -X_1 * y_{im4} & -Y_1 * y_{im4} \end{bmatrix} \cdot \begin{bmatrix} a \\ b \\ c \\ d \\ e \\ f \\ g \\ h \end{bmatrix} = \begin{bmatrix} x_{im1} \\ x_{im2} \\ x_{im3} \\ x_{im4} \\ y_{im1} \\ y_{im2} \\ y_{im3} \\ y_{im4} \end{bmatrix} \tag{2}$$

The 8x8 matrix can be represented as P for convenience, hence the unknown parameters can be computed as:

$$\begin{bmatrix} a \\ b \\ c \\ d \\ e \\ f \\ g \\ h \end{bmatrix} = P^{-1} \cdot \begin{bmatrix} x_{im1} \\ x_{im2} \\ x_{im3} \\ x_{im4} \\ y_{im1} \\ y_{im2} \\ y_{im3} \\ y_{im4} \end{bmatrix} \tag{3}$$

After the chessboard is detected and the projected view is calculated, players put their pieces in their places, and the orientation of the board is analyzed. This analysis is done by checking the horizontal and vertical projection of pixels to see if the

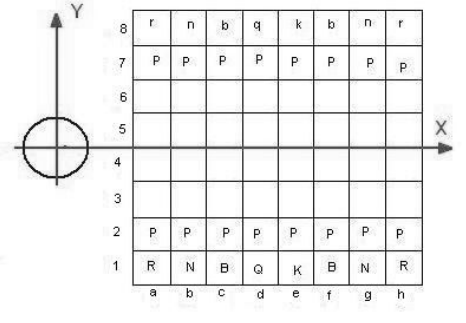



**Fig. 4.** Assumed chessboard standard, small letters are for black pieces, capital letters are for white pieces

horizontal projection shows an uneven distribution with a high number of black pixels towards the top of the board. If its orientation is not compatible with the standard given in Figure 4, then the projection is repeated as to show black pieces at the top and white pieces at the bottom in the projected view.

This is done by changing the orders of $(x_{im1},y_{im1})$, $(x_{im2},y_{im2})$, $(x_{im3},y_{im3})$ and $(x_{im4},y_{im4})$ in circular order, i.e. Instead of giving the reference points in the order:

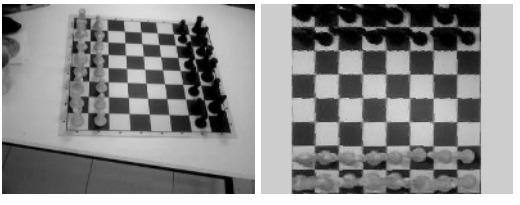
$\{ (x_{im1},y_{im1}), (x_{im2},y_{im2}), (x_{im3},y_{im3}),(x_{im4},y_{im4}) \}$

One of these sets is used:

$\{ (x_{im2},y_{im2}), (x_{im3},y_{im3}), (x_{im4},y_{im4}),(x_{im1},y_{im1}) \}$
$\{ (x_{im3},y_{im3}), (x_{im4},y_{im4}), (x_{im1},y_{im1}),(x_{im2},y_{im2}) \}$
$\{ (x_{im4},y_{im4}), (x_{im1},y_{im1}), (x_{im2},y_{im2}),(x_{im3},y_{im3}) \}$

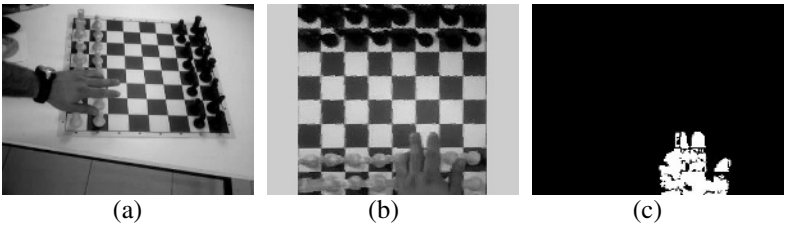
## 4 Move Detection

After the pieces are placed in order a typical scene looks like Figure 5.a. Figure 5.b is the projected view of the scene which is obtained by perspective transformation described earlier.



(a) (b)

**Fig. 5.** (a) A starting view (b) Projected view of the chess board



(a) (b) (c)

**Fig. 6.** (a)The hand in the scene (b)Projected view (c)Difference image

When the initial setup of pieces is complete the system enters in a “wait” mode for a play action. The next mode is the “alert” mode which is the state entered when a human activity is observed. The expected activity is the movement of a human hand

on the chessboard. In Figure 6.a, player's hand is in the view of camera. Figure 6.b is the projected view. The difference between Figure 5.b and 6.b is shown in Figure 6.c.

A typical activity is detected from the video stream by observing the difference from the reference frame. The difference images are thresholded and number of foreground pixels are recorded. This operation forms a plot shown in Figure 7. The horizontal axis here is the time axis or frame number and the vertical axis is the number of the white pixels in difference images. This plot can easily be interpreted within the context of the chess move. An increase at the beginning of the graph indicates the beginning of action. A peak value suddenly drops to low levels when human hand stops to pick up a piece. Another peak is observed when it stops to put the piece on the board. When the initial level is reached we can assume that the move has ended.

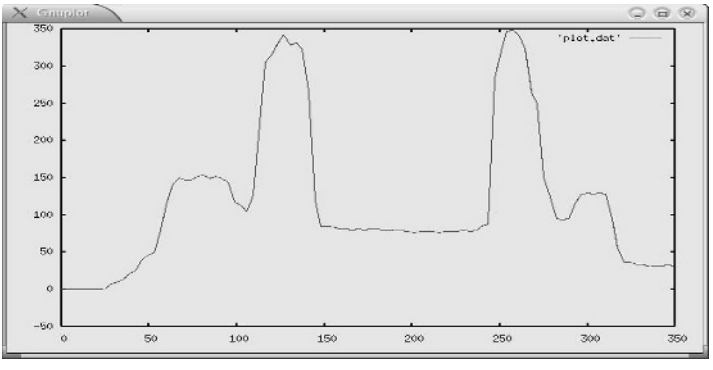



**Fig. 7.** A typical temporal plot of a move

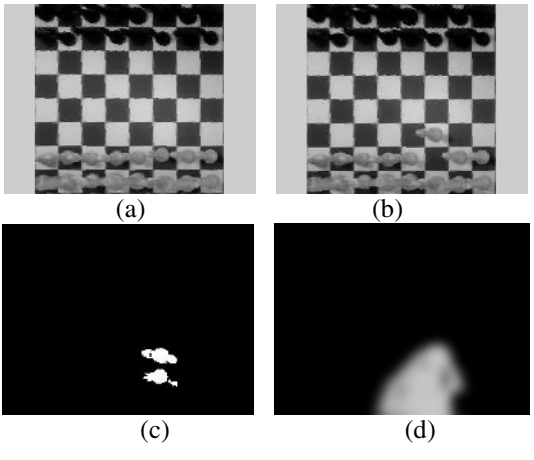



**Fig. 8.** (a) Before the move (b) After the move (c) Difference image (d) Cumulative difference

When the move is detected the system enters into the state of "analyze move". This step verifies whether or not a move is made and it determines the new placement of pieces on the board. This is done by using a cumulative difference image (Figure 8.d)

which shows the position of the hand during the observed interval as well as the static frames before (Figure 8.a) and after the move (Figure 8.b). Figure 8. c is the difference between Figs (a) and (b).

## 5 Move Analysis

When a possible move is detected the next step is to verify the move and determine the new positions of pieces. The decision is based on the information about the static (e.g. in Figure 8.c) and dynamic (e.g. in Figure 8.d) changes in the scene. The ambiguity in these images is encoded by representing each square of the chessboard with a degree of compliance with the possible move. These degrees are computed by normalizing the brightness values corresponding to a square by maximum brightness value observed in the difference images since the beginning of the game (E.g. Figure 10.b & c).

The possible movements indicated by difference images are checked to see if they are valid by using the moves allowed by chess rules at the current state of the game. This is done by using a move table representation (E.g. Figure 9) for each piece which is updated after each move.

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **8** | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 0 |
| **7** | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| **6** | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 |
| **5** | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| **4** | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| **3** | 0 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| **2** | 1 | 0 | 1 | 0 | 1 | 0 | 0 | 0 |
| **1** | 0 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| | **a** | **b** | **c** | **d** | **e** | **f** | **g** | **h** |

**Fig. 9.** Move table of a queen located at c4 with no other piece blocking

Move tables are constructed according to the chess rules (E.g. pawns only move forward; the pieces can not jump with the exception of knights, etc.). After a move table is constructed it is stored in an array of 8 unsigned characters using C language. This is made by representing each line in the move table as decimal numbers. Making this conversion reduces the memory usage and simplifies computation. For example the table in figure 9 can be stored as:

[34 36 168 112 255 112 168 36 34]

Here every number stands for a raw of packed bits converted to decimal form.

This high level decision making mechanism involving knowledge representation of chess domain improves the reliability of the algorithm and it enables us to recognize special moves like en-passant or castling.

The algorithm comprises the following steps:

- Combine the information encoded in two types of difference images (e.g. in Figure 10.b and c). We use MINIMUM operator which stand for fuzzy AND [14].

- For each position with a value greater than a threshold (e.g. 0.5) combine the array resulting from previous step by the array representing the rules for the piece at that position.
- Compare the pairs of possible moves and accept the one with highest degree of likelihood.

In Figure 10 (a)Original view of a sample game can be seen. Figures 10.b and 10.c shows the movement of white queen in d1 to d3 respectively.

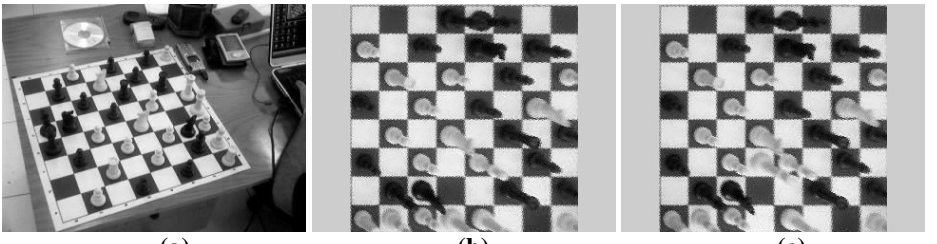



**Fig. 10.** (a) original view of a sample game (b) reference image taken before the move (c) reference image taken after the move

Figure 11 shows the difference image between 10.b and 10.c. To extract position of a piece before and after a move occurrence, bottom of the piece gives accurate information. In the difference images (E.g. 11.a), bottom of a piece can be distinguished as a circular shape. By using this morphology piece positions can be approximated. By using a circular structuring element (shown in figure 11.b), correlation for each square can be calculated. This structuring element has 30x30 pixels in size.

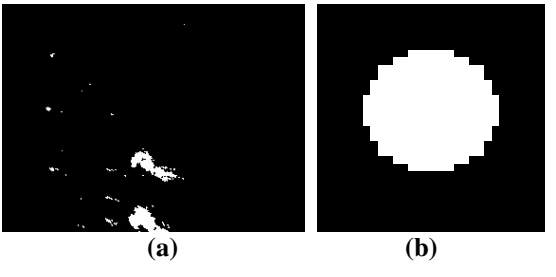



**Fig. 11.** (a) Image of the difference between 10.b 10.c (b) morphological element

Due to the perspective of the camera, in the difference image pieces like Queen and King generates some difference in neighbor squares. This situation may cause some faulty results. In order to eliminate these results, a little modification has been added to the correlation mechanism. Simple correlation requires a convolution of difference image with the structuring element for each square on the chessboard. The modification reduces correlation of a square if a black pixel in the element coincides with a white pixel (a difference pixel). By this way the faulty characteristic due to the

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **8** | 0 | 0 | 0 | 0 | 0.0021 | 0 | 0 | 0 |
| **7** | 0.0231 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **6** | 0 | 0.0042 | 0 | 0 | 0 | 0 | 0 | 0 |
| **5** | 0.0378 | 0.0021 | 0.0084 | 0 | 0 | 0 | 0 | 0 |
| **4** | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| **3** | 0.0462 | 0 | 0.0252 | 0.721 | 0.275 | 0 | 0 | 0 |
| **2** | 0.0021 | 0.0042 | 0.042 | 0.0084 | 0.174 | 0.0063 | 0 | 0 |
| **1** | 0 | 0.00196 | 0.00313 | 1 | 0.197 | 0 | 0 | 0 |
| | **a** | **b** | **c** | **d** | **e** | **f** | **g** | **h** |

**Fig. 12.** (a) correlation table of the difference image in 11.a

perspective is eliminated. In figure 12, the correlation table, which has been calculated by using the difference image 11.b and structuring element 11.b, can be seen.

By using the result obtained from the table in Figure 12 in parallel with move table information and the cumulative difference information, the move can be predicted accurately.

## 6 Conclusion

A computer vision system which identifies a chessboard, and interprets the actions on the board is described. The domain knowledge is represented and updated at each state of the game. The visual data is combined with the encoded domain knowledge in order to recognize actions. This study aims to be a first step into task oriented understanding of actions in video data. The well defined environment and the rules of the chess game provide us a good framework in order to use domain knowledge in a vision guided decision making process. The experience gained here can be extended to more complicated vision tasks involving more complex environments. Besides attacking more complicated problems, the future work involves making the algorithm more robust in terms of extraction of reliable data and decision making.

## References

1. I. Koprinska, S. Carrato, "*Temporal video segmentation: A survey*"; Signal Processing: Image Communication, Volume 16, Issue 5, January 2001, Pages 477-500
2. A. W. M. Smoulders, M. Worring, S. Santini, A. Gupta, and R. Jain, "*Content based image retrieval at the end of the early years*" IEEE Trans. Pattern Anal. Machine Intell., vol. 22, pp. 1349–1380, Dec. 2000.
3. A. Dorado, J. Calic, E. Izquierdo, "*A rule-based video annotation system*"; IEEE Transactions on Circuits and Systems for Video Technology, Volume 14, Issue 5, May 2004 Page(s): 622 – 633
4. R. Brunelli, O. Mich and C. M. Modena, "*A Survey on the Automatic Indexing of Video Data*", Journal of Visual Communication and Image Representation, Volume 10, Issue 2, June 1999, Pages 78-112

5. J. Assfalg, M. Bertini, C. Colombo, A. D. Bimbo, W. Nunziati, "*Semantic annotation of soccer videos: automatic highlights identification*". Computer Vision and Image Understanding, Volume 92, Issues 2-3, November-December 2003, Pages 285-305
6. A. Ekin, A.M. Tekalp, R. Mehrotra, "*Automatic soccer video analysis and summarization*"; IEEE Transactions on Image Processing, Volume 12, Issue 7, July 2003 Page(s):796 - 807
7. M. Campbell, A. J. Hoane, and F.H. Hsu, "*Deep Blue*", Artificial Intelligence, Volume 134, Issues 1-2, January 2002, Pages 57-83
8. H.S. Baird, K. Thompson, "*Reading chess*", IEEE Trans. Pattern Anal. Machine Intell., Volume 12, Issue 6, June 1990 Page(s):552 - 559
9. F.C.A. Groen, G.A. Den Boer, A. Van Inge, R. Stam, "*A chess-playing robot: lab course in robot sensor integration*", IEEE Transactions on Instrumentation and Measurement, Volume 41, Issue 6, Dec. 1992 Page(s):911 - 914
10. E. Uyar, Ş. Gümüştekin, Ş. Ozan, L. Çetin, "*A Computer Controlled Vision Oriented Robot Manipulator for Chess Game*", Proc. of Workshop on Research and Education in Control and Signal Processing, REDISCOVER 2004, Cavtat, Croatia
11. http://www.intel.com/research/mrl/research/opencv/
12. R.C. Gonzales, R.E. Woods, "Digital Image Processing", Addison Wesley, 1993
13. E. Trucco, A. Verri, "*Introductory Techniques for 3D Computer Vision*", Prentice Hall, 1998.
14. J.Z. Zimmermann, "Fuzzy Set Theory and its Applications", Kluwer Academic Publishers, 1993.

# Multiple Robot Path Planning for Robot Soccer

Çağdaş Yetişenler[1] and Ahmet Özkurt[2]

[1] Yuksek Teknoloji A.Ş., Izmir, Turkey
[2] Electrical and Electronics Eng. Dept., Dokuz Eylul University, Izmir, Turkey
cagdasyet2@yahoo.com, ozkurt@eee.deu.edu.tr

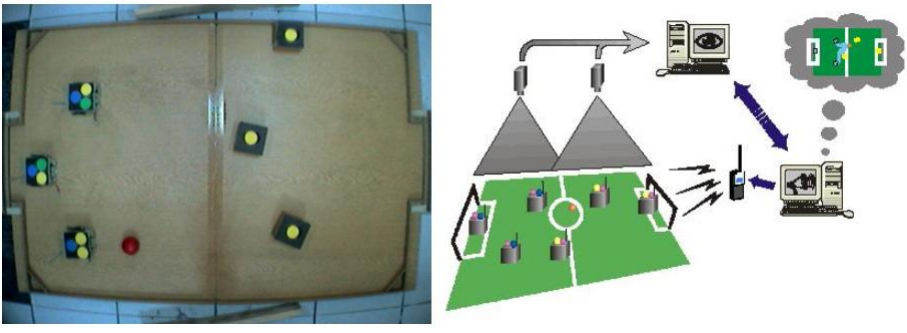
**Abstract.** The Robot World Cup Initiative (RoboCup) is an international joint project to promote AI, robotics, and related field. It provides a standard platform for robotic soccer game which includes a vision system, a strategic play algorithm and small mobile robots. The aim of the vision system for RoboCup small robot league is to track and predict the motion states of 6 agents and a ball, and send the states information to the Artificial Intelligence module. This paper presents a design and implementation of a real-time, robust global vision system. The two main efforts are realized in this study, design and implementation of multi robot structure and the construction and testing of the tracking and path planning algorithm. In the tracking phase, the color-based segmentation is used to locate the interesting objects in the image; the blob analysis is adopted to calculate the positions of the objects in the image and the noise is filtered out; and the linear filter is adopted to track and predict information of the states. It has been planned that the information gathered using this study can be used in multiple agent robotics applications.

## 1 Introduction

In the RoboCup Small-Size league, robustness and accuracy of vision localization is the key features for successful performance in competition. According to F180 rules, five robots on each team play soccer on a green field marked with white lines. The ball is orange and robots, as well as goals, are marked either yellow or blue to recog-nize teams. Besides yellow and blue, more colors are used to distinguish different teammates. One or more high-resolution color CCD video camera systems are allowed to mount on the top of the soccer field to capture real-time image of the field. Normally, using only one camera system will result in some problems, such as distortion, increase in invisible ball play area, size of the color patch too small and become prone to noise. Because of these drawbacks for one camera system, it is quite hard to tune the vision system. Field Ranger team has adopted two-camera vision system, each covering half the field as shown in Figure 1-a. Coordinates of a robot are obtained through linear transformation from robot's image position. In order to localize the robots and the ball, color patches of robots and ball must be captured and processed in real time. This is not a simple task. Firstly, RoboCup vision system requires fast image segmentation and identification; secondly, the color patches are sensitive to lighting

F.A. Savacı (Ed.): TAINN 2005, LNAI 3949, pp. 11–23, 2006.

conditions, and thirdly ball color is very similar to other noise like hands. In this paper, the tracking of home and competitor players with ball has been accomplished and the required path planning decisions have been made. For tracking, the color-based segmentation is used to locate the interesting objects in the image; the blob analysis is adopted to calculate the positions of the objects in the image and the noise is filtered out; and the linear filter is adopted to track and predict information of the states. The designed system can be used in multiple agent robotics applications. This paper gives description of our methods for the above challenges.



(a) (b)

**Fig. 1.** (a)Robot Soccer Field and (b) Autonomous Robocup system

### 1.1 The Small Robot League

The Robot World Cup Initiative (RoboCup) is an international research and education initiative. It is an attempt to foster AI and intelligent robotics research by providing a standard problem where wide range of technologies can be integrated and examined, as well as being used for education.

Our robotic team is an autonomous system consisting of a preset processing cycle (see Fig. 1-b). Vision module receives the raw image sequence from a camera and processes it, giving the motion states of the ball and each robot. This information is sent to a control module, which evaluates the world state and decide what to do in each step based on its specific strategy. In our system, global vision is the only way to collect information provided for high-level control. It means that if the vision module cannot work properly, the whole system will have a poor performance or even crash.

## 2 Tracking

Our tracking methodology is based on color base segmentation of color codes for robot identification, blob analysis to find out color placements, and robot identification based on preset robot color scheme determined earlier.

### 2.1 Color-Based Segmentation

Colors are distributed on the color space in the experiment in RGB. It is a natural choice to apply a threshold to segment the image. The thresholding is done based on the interested color. For an interested RGB color, there are 6 parameters we should know before segmentation, the minimum and maximum threshold values of each of 3 bands.

### 2.2 Blob Analysis

Blob analysis allows you to identify connected regions of pixels (blobs) within an image, and then to calculate selected features of those regions. Typical features de-scribe the size, shape, and location of the blobs. Therefore, the results of blob analysis can be used for a variety of purposes, for example, to distinguish between objects in an image, to locate objects, and to measure objects. The center of the blob is the location of the object in the image. And the area and the perimeters are often used to filter out unwanted noise.

After segmentation, the next step is to connect the segmented pixels into regions or blobs. This process can be expensive and can impact in real time performance. The practical benefit that the blob generation algorithm will only need to look for vertical connectivity. Every run generates a data structure that contains all the information of the pixels and a pointer used to connect them with other runs. The objective of the merging procedure is to connect all the runs that belong to a region. The idea is that every run of a region points to the region's run parent. Initially we have a disjoint forest of runs, every run points to itself. The merging procedure looks into adjacent rows and merge runs that are of the same color and overlap under four connectedness. Every run points towards a region's global parent, if overlapping occurs then a second pass is needed to merge the overlapped runs. The process is shown in Figure 2.

### 2.3 Robot Identification

Once the lists of regions are generated the recognition process begins. As mentioned before, to recognize the ball we simply take the region of red color whose area is nearest to 60 pixels (ball area with a 240x320 resolution). To do this we look at the absolute value of the difference between the region's area and 60.The most favorable area is the ball.

To recognize our robots we first select candidates following the same procedure as the robots of the opposite team. Then every candidate is further examined searching for additional patches used in the robots for individual identification. Every team decides which pattern of patches to us. In this study,it is decided to use a butterfly pattern [1]. The additional patches in a window around the central patch (candidate) are searched in this algorithm. If less than four additional patches are found, the candidate is discarded; if more pathes are found, four regions whose area is nearest to 80 pixel are selected. Once there are four patches in the window area around the central patch, the next step is to order them in a counter clockwise direction to obtain the color encoded ID of the robot. Every

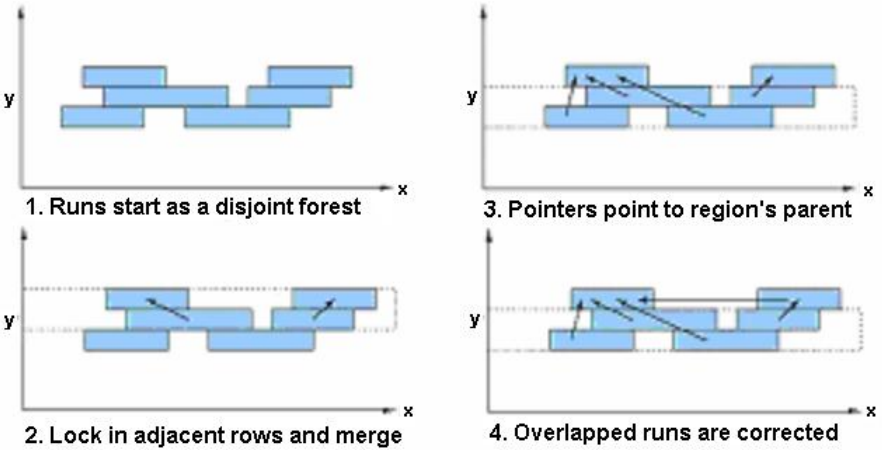



**Fig. 2.** Blob analysis algorithm

robot has a unique ID encoded in the additional patches, one color (green) is interpreted as a zero and the other color (Yellow) as a one. So a patch pattern of green, green, yellow and green is interpreted as 0010 or a decimal 2.

To order the additional patches in counter clockwise direction, first a simple closed path is computed. A simple closed path is found when a set of points are connected without crossing the paths between the points. Figure 3 shows an example of a simple closed path.

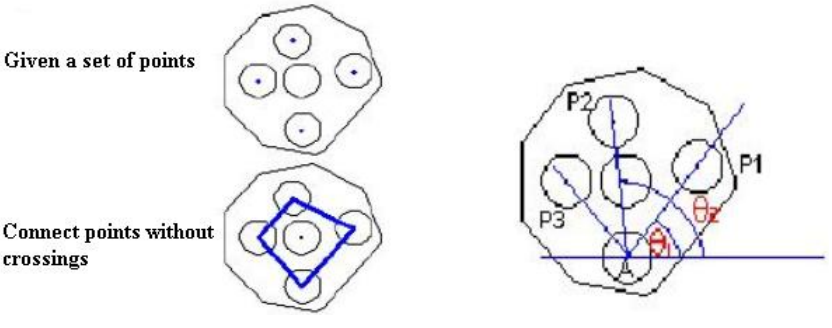



**Fig. 3.** Example of color based identification scheme used in the study

The first step for producing the path is to find the lowest point and take it as an anchor, then the angles of the line segments between all the points and the anchor are computed. The points are ordered form smaller to greater angle. Figure 3 shows the anchor point A, points Pi, line segments and computed angles.

Once the patches are ordered the distance between the points are computed, the pair of points whose distance is greater correspond to the frontal patches and the ID can be decoded.

## 3 Path Planning

Several path planning methods have been presented in the literature [2],[3],[4]. In the present work a cellular automata (CA) approach is applied [5]. For this purpose a free flying robot without dynamic and kinematic constrains is considered. It consists therefore a single point without any consideration about its orientation. The workspace is decomposed in a finite collection of non intercepting square cells. As previously noted this cell array is the cellular space of a CA and constitutes the configuration space, where the objects in the scene occupy a certain number of cells. Assuming a grid of given size, in the present case a 240x320 grid where each cell corresponds to approximately 1cm2 of real space, the discrete configuration space can be defined by the coordinates of each cell.

$$C = \{(x, y)|x \in \{0, \ldots, x_{max}\} \quad y \in \{0, \ldots, y_{max}\}\} \tag{1}$$

In this representation each object is a particular set of cells and the free space is the set of cells not belonging to any object. A path in free space is a finite discrete sequence of cells. The input to the algorithm is the CA representation of the workspace provided by the object recognition system, it contains information about the location of each colored object in the workspace. The position coordinates of each object is calculated as the centroid of the corresponding colored pixels.

In first phase of the algorithm, the image information is transformed to the discrete cellular space with four possible initial states: (0) free, (1) strange object, (2) initial position (position of the robot considered), (3) goal position (position of the ball or strategic position). In phase two, a predefined template of cells in state (1) is placed over the position of the strange objects in order to account for the physical size of the robot. The size of the template depends on the size of the robot and of the cell, for the typical resolutions used the template is about 24x24 cells. The schematic diagram in Figure 4 shows these processes.

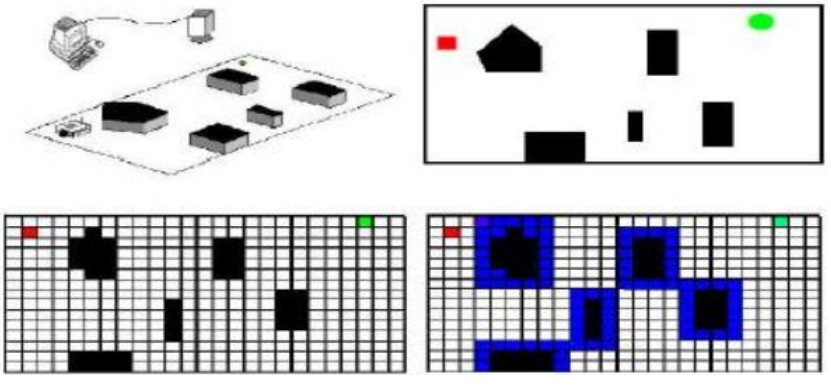


**Fig. 4.** Path Planning preparation algorithm

In the third phase, the final configuration resulting in the prior phase is used as initial configuration for a CA dynamics that computes the Manhattan distance between the initial and goal positions. In this cellular automata the possible states for the cells are the following: (0) free space, (1) obstacles (strange objects), (2) initial position, (3) goal position, (4) Manhattan distance 1 to the goal,..., ) 3 ( l + Manhattan distance l to the goal. The transition rule applied to evolve this cellular automata is:

$$s_i^{t+1} = \begin{cases} s_x^t + 1 & s_i^t = 0 \wedge \exists x \in \eta_i^t | s_x^t \geq 3 \\ s_i^t & otherwise \end{cases} \quad (2)$$

In Figure 5 this process that resembles a flood from the goal to the initial position is shown.

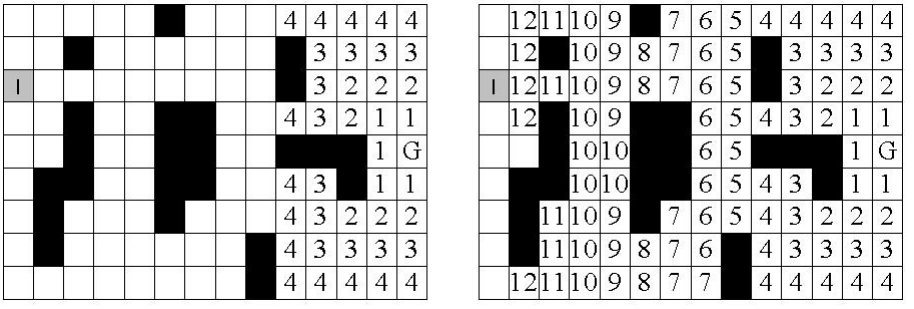



**Fig. 5.** Distance creation to determine possible paths

The flood dynamics is stopped either when the cell with the initial position is reached or when all cells in the cellular space are different from 0 in which case no path is possible. In the former case a path is calculated by going backwards from the goal to the start positions in a descending manner. Due to the existence of saddle points in the navigation function, the shortest path between an initial and goal position is not well defined. Often a cell has more than one neighbor with the same Manhattan distance to the goal. To follow always the steepest descend of the function may not work because there could be cases where the gradient may have the same value in several directions. In order to select a reasonable good path two heuristics are applied among the neighbors of a cell: (I) The cells that fulfill the condition of being in direction of the steepest descend and allows the conservation of the direction of movement for the robot are selected in the path with the highest priority or (II) those cells which are in direction North, South, West and East and have a small angle to the moving direction of the robot are also selected with high priority. In general it is found experimentally that these two heuristics efficiently produces reasonable good paths with few minimal changes of the direction of movement (commands for the robot). Figure 6 shows this process.

The knowledge of the sequence of cells that define a path together with the initial direction of movement of the robot allows, in a straightforward manner,

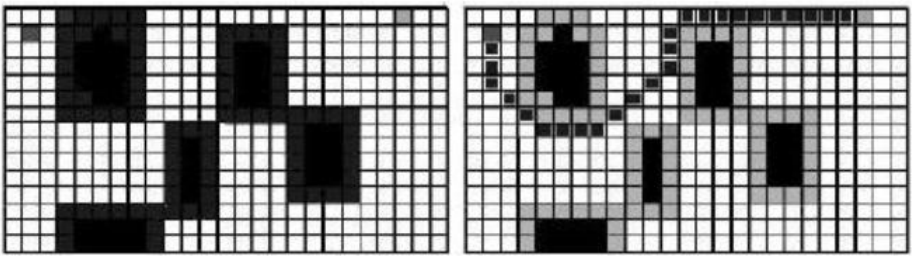


**Fig. 6.** Path decision based on distances calculated before

the production of a list of commands which guides the robot along the path to its goal. The experiments indicate that the CA based path planning algorithm allow real time motion planning. In effect, using as input an image of 240*320 pixels resolution, an average response of 10 ms per path was measured.

## 4 Organization and Strategy

The functionality of Team robots are divided into three levels based on the level of abstraction from basic motion: (1) at the lowest level, agents must be able to follow specified behaviors. (2) In second level it uses combination of these behaviors to build some complex abilities roles.(to get behind a ball, move toward a goal, take the Ball from corner, and so on). (3) Robots must have individual and team-oriented strategies and roles that create goals for the Robot (places the agent desires to be and courses of action the Robot desires to take). The following subsections describe each of these levels in turn.

### 4.1 Behavior Based Control

To build a strategy, for the beginning it will be much easy to use some simple movements and abilities, which can be added together to build up much complex abilities and coordination. Behavior Based Control is already used in Robocup soccer, every team built up its own strategy and uses this method in their team architecture. It is tried to build a strategy which is based on Behaviors and with their combinations Roles. Strategy algorithm is constructed with three layers. Lowest layer is Behaviors (Level 1) second level is Roles (Level 2) and last level is the Director level which gives duties to robots (Level3), all levels can control its sub level and will change its application. In following chapters all simple behaviors and their combination (Roles) will be examined. Figure 7 shows this strategy levels.

### 4.2 Simple Behaviors (Level 1)

Behaviors are constructed with combinations of basic robot movements (Turning, Running forward and so on). In strategy algorithm, robots have nine behaviors

these behaviors are basic and also have the ability to handle some hard situations. These behaviors are; Shoot, Go to Destination, Do Angle, Turn, Forward Turn, Opponent Turn, Forward Opponent Turn, Stop and Keeper Path. All these

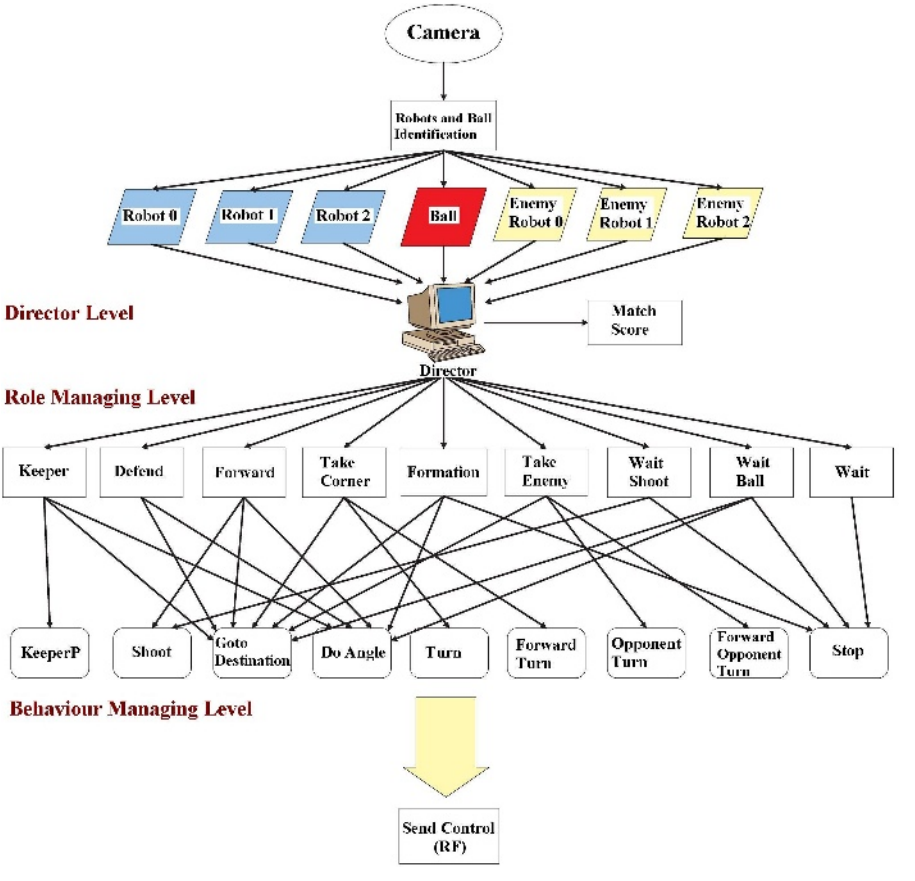



**Fig. 7.** Strategy Levels

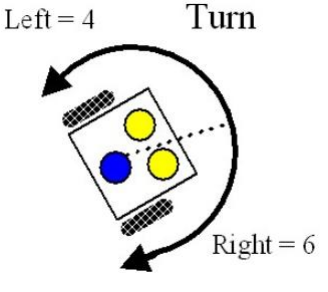



**Fig. 8.** Turn Behavior

behaviors are controlled, set and cleared by Behavior manager (Level 1). Lets examine Turn Behavior for sample.

**Turn Behavior.** This behavior is used for two purposes; taking ball from corners and intercepting opponent team members. Robot will turn left or right with full speed and drive circles. This movement is done with high speed to increase the chance of touch the ball and to change its direction. If this behavior is selected, it can't be switched off for 500ms this is controlled by the behavior manager. Figure 8 shows this movement. If ball is stuck at corner, according to Role selection Turn Behavior is loaded to Robot. After this, Robot will start rotating to hit the ball and take it from corner.

### 4.3 The Role Manager (Level 2)

Role selection algorithm is controlled by the Role Manager which controls role timeouts and subroutines. It works similar to Behavior manager only difference is Role manager controls Roles. Again Role manager is not the brain of the team it just controls the organization of roles. Role assignment is done in level 3 Director Level part. It also controls robot if it's doing behavior or not, if Robot is doing a behavior there is no need to go to Role selection part because as we discussed before all behaviors take time and until it is finish they can't be changed. This idea is also used in Role selection if a Robot is assigned to a role it can't be changed for a specific time this method helps strategy for not changing Roles so often and so quickly. There are nine roles in strategy program these are; Forward, Defend, Keeper, Take Corner, Formation, Take Enemy, Wait Shoot, Wait Ball, Wait. Let's examine forward.

**Forward Role.** When forward role is selected for a Robot there will be one condition in which the Robot must occupy in opponent's half field. In Figure 9 Robot is in opponent team's half field and is preparing for scoring. As it is mentioned before when robot takes a role it will do some basic movements (behaviors) and use them in some order. In Forward behavior Robot can use three behaviors, these are Shoot, Do Angle and Go to Destination. Considering the Figure 9 this process can be seen, in first part (1) Robot travels to a position which is behind the ball, to do this it uses Go To Destination behavior (That point is calculated by subroutine Take Score). In second part (2) robot uses Do angle behavior to turn its face to the ball. In third part (3) robot will use shoot behavior for hitting the ball. Take Score subroutine has another duty which is a calculation that finds the empty sides of the opponent's goal. It controls the opponent keeper (which is the nearest opponent robot to the goal) and determines the empty corners which can be G1, G2, G3 in Figure 9.

This Role also have a panic shoot ability, if the ball some how passes between the goal and Robot it will shoot instantly without any movements. This ability will increase the scoring chance. This Role takes 500ms and can't be changed by Role manager.

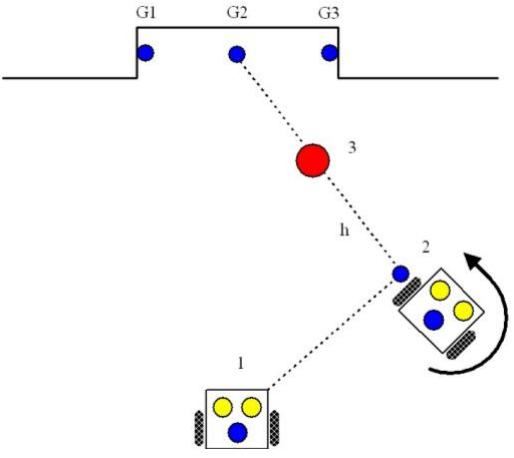



**Fig. 9.** Forward role

### 4.4 Strategy Director (Level 3)

When all Behaviors and all Roles are constructed it is time to use a role selector algorithm. Main idea in Role selection is based on Match field regions which are shown in Figure 10. There are 9 regions and according to ball's place, strategy will be developed. For declarations, it is preferred to use some extra identification names which are N1, N2, NK. Robot N1 is assigned to a robot which is the nearest robot to the ball. Most aggressive roles are given to Robot N1 which are the Forward, Defend, Take Corner and so on. Any Robot in the team members can take this identification. N2 name identification is used for the robot which will help to Robot N1 for any of its duty. For example if Robot N1 is shooting or doing an aggressive attack Robot N2 will wait for the ball. Robot N2 will also wait for shooting when Robot N1 has the ball. RobotN2's duty is waiting the ball in order to protect home robots for running to same target (ball). Again any robot in team can take label N2. Robot NK is assigned to keeper which will be given to any robot in team but for constant it is preferred using it as Robot 0 (Robot 0 is labelled as Robot NK). Robot 1 and Robot 2 are set to either N1 or N2 but this decision is made by Director Level according to place of the ball and place of the robots.

In the first part of the program, Robot N1 and Robot N2 is assigned by looking to Ball and its place in field. If ball is in regions 8 or 9 (according to Robot Near Ball Subroutine) nearest Robot to ball is selected as Robot N1 and other player robot is selected as Robot N2; as it mentioned before Robot NK is always selected as Robot 0. If the ball is not in regions 8 or 9, again some constant declarations are used for the ball in upside of the field (regions 3 or 5). Robot 1 is labelled as Robot N2 and Robot 2 is labelled as Robot N1 if ball is in regions 4 or 6, Robot 1 is labelled as Robot N1 and Robot 2 is labelled as Robot N2.

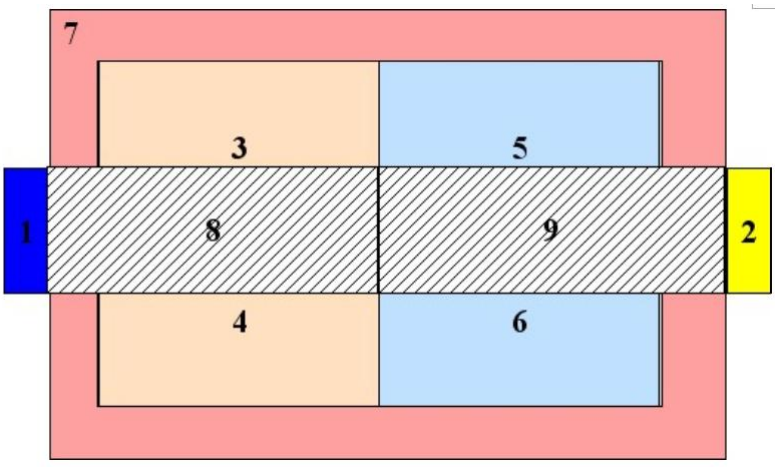



**Fig. 10.** Match field areas for strategy

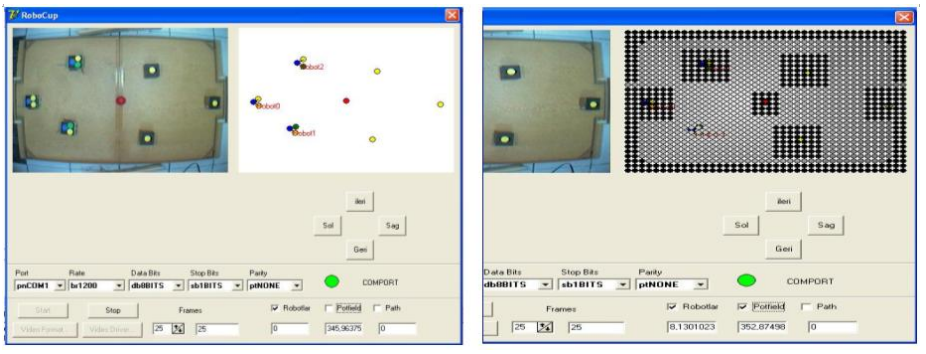
After Robot labelling part is finished, it is time to look at robot role selection algorithm. According to the ball field regions (Figure 10) role assignments are chosen. If ball is in region 1 or 2, match score subroutine is called (for score board calculations) and Director level assigns Formation role to all robots. If an opponent Robot is near to ball, Robot N1 is charged with Take Enemy Role and Robot N2 is charged with Wait Ball Role. If ball is in regions 3 or 4, Robot N1 is charged with Defend Role and Robot N2 is charged with Wait Ball Role. If ball is in regions 5 or 6, when a condition of the ball which is hoped from the opponent keeper and approaching to Robot N2. Director manager will charge Robot N2 with Wait Shoot role and Robot N1 will be charged with Wait Ball role. If ball is not approaching to Robot N2 then Robot N1 is charged with Forward and Robot N2 is charged with Wait Ball Role.

## 5 Results

The system which have been implemented includes six small robots designed for near cost-free strategy. The another goal of this study was to develop own equipment and to control the problems which may be caused by the hardware. Mainly, those small machines based on twin gearbox system, PIC based controller, an RF transceiver pair, and light-weight structured mechanic parts.

The data grabbed from the camera is processed to extract the colors used top of the robots as mentioned in section 2.1. Then, Robot identification is implemented by blob analysis and matching phase. The robot identification example is shown Figure 11-a. showing three home robots, three competitors and one red ball.

After the robot and position recognition, before the path planning problem, the area should be determined in order to run robots for playing in playground. For that purpose, potential regions are calculated by the distances to aim which




(a) (b)

**Fig. 11.** (a)Robot recognition and identification example and (b) Potential Fields Determination

may be the ball to capture or hit for offense and/or taking position for defense. But knowing robot dimensions, some areas are masked not to travel on. The remaining area are divided into small parcels and distances to aims are calculated. Those potential areas are shown in Figure 11-b. After robots and positions determined, and the real time distances to aim we mentioned by the playing rule, robot paths are solved by tracking possible distance maps in free playground. Several of tracks may be chosen by the selector function based on the shortness to the goal. This decision is important and determines the performance. This path planning example is given below in Figure 12.

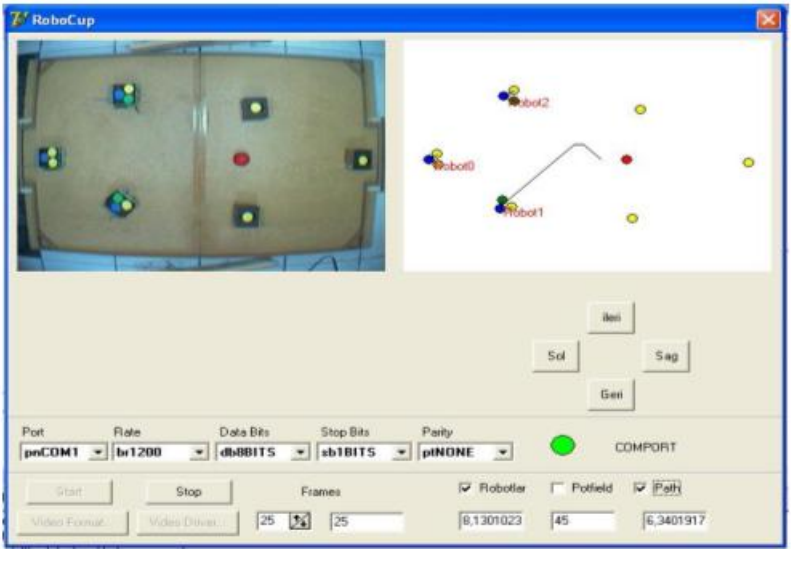



**Fig. 12.** Final Path Planning Results

## 6 Conclusions

In this study, a cellular automata approach for solving robot path planning problem which requires an efficient experimental performance on robocup type multiple robot path planning applications. This work presents a design and implementation of the real-time vision system and required multilevel strategy and simple behavior algorithms.

The main idea in this system is to design an algorithm in an dynamic environment which includes multiple robots. User robots can be managed according to positions of the team members, the ball and the opponent robots.

The performance of the system is based on the response of the algorithm in the dynamical system. For this purpose, a self-tuned and robust system is aimed to design.

As a conclusion, the practical performance is satisfactory, however, further studies including more improvements and new strategies are necessary.

## References

1. Bruce, J., , Veloso, M.: Fast and accurate vision-based pattern detection and identification. In: Proc. of the IEEE Int. Conf. on Robotics and Automation, ICRA'03, Taiwan (2003) 1277 – 1282
2. Latombe, J.: Robot Motion Planning. Kluwer Academic Publishers, Boston, MA (1991)
3. Lee, S., Kardaras, G.: Elastic string based global path planning using neural networks. In: Proc. of the IEEE Int. Symp. on Computational Intelligence in Robotics and Automation, CIRA '97, Monterey, CA (1997) 108 – 114
4. de Rodriguez, M., Moreno, J.: Heuristic algorithm for robot path planning based on real space renormalization. Lecture Notes in Computer Science **1952** (2000) 379
5. Behring, C., Bracho, M., Castro, M., Moreno, J.: An algorithm for robot path planning with cellular automata. In: Proc. of the Fourth Int. Conf. on Cellular Automata for Research and Industry, ACRI'2000 ACRI'2000 in Theoretical and Practical Issues on Cellular Automata. (2000) 11–19

# Navigation and GPS Based Path Control of an Autonomous Vehicle

Erol Uyar, Levent Çetin, and Aytaç Gören

Dokuz Eylül University Engineering Faculty
Mechanical Engineering Department
35110 Bornova İzmir Turkiye
erol.uyar@deu.edu.tr

**Abstract.** In this work the navigation and GPS based trajectory control of an unmanned autonomous vehicle is introduced. The caterpillar tread like mobile vehicle with duo cycle drive is equipped with a mobile GPS and micro controller. To get accurate position data, a DGPS (Differential GPS) with a local base station is used. In order to let the vehicle to follow a desired trajectory, a reference path is given by using Way Points as GPS data's to micro controller, which provide the position and orientation control of the vehicle. The instant position and orientation of the vehicle in accordance to an assigned plane coordinate frame are calculated by micro controller from detected latitude and longitude values (land coordinates) of mobile GPS, which receive corrected on line data's from base station.The duo cycle vehicle is driven with geared DC motors, which are equipped with chained caterpillar tread drives, so that it can have better motion, clambering and tracking performances. Successful navigation and path following results are obtained with several experiments by various control applications

## 1 Introduction

In the automation of manufacturing processes the use of Autonomous Guided Vehicles (AGV) has been an important task for the efficiency, quality and just in time working. Various theoretical algorithms are developed [1],[2] and simulation studies are made with relevance in this task. But the non-linear nature of the vehicle-dynamic besides applied navigation system requires a great effort to achieve an accurate on line trajectory guidance. In this work an effective real time oriented, successfully applied control system both for navigation and motion planning is introduced.

Three basic concepts for an autonomous vehicle to let follow a designed path are navigation, guidance and control. To achieve satisfying results, firstly the problem of an accurate positioning and localization must be solved. In many cases a sensor fusion consisting of different sensors (like internal and external sensors) supporting each other, are used to obtain reliable data for control system. [3].

Generally two basic position-estimation methods for navigation is applied; absolute and relative positioning. The absolute positioning, which is based on landmarks, map matching or satellite, connected navigation signals (like GPS), where absolute

F.A. Savacı (Ed.): TAINN 2005, LNAI 3949, pp. 24 – 31, 2006.

positioning sensors interact with dynamic environment. Relative positioning is usually based on inertial sensors as odometers or camera-oriented systems.

For positioning internal and external type of sensors are generally used. The internal sensors like encoders gyroscopes, accelerometers etc. provide generally direct measurable variables on the vehicle External sensors such as sonar's, radars, camera or laser based systems, measure relationship between the vehicle and its environment.

For short periods internal sensors are generally quite accurate, but for long-term estimation they usually produce a typical drift, which is not caused by the use of external sensors. But because of their characteristics the measured values are not always available. These facts guided to the idea of using multi-sensor fusion in navigation systems.

In the following the principles of a GPS based navigation and positioning control of an autonomous vehicle in plain by using remote guidance through a PC is introduced.

## 2 Implementation and Fomulation of the Applied Controls

A general schematic representation of the implicated trajectory planning control on duo cycle mobile vehicle is introduced in Fig.1.

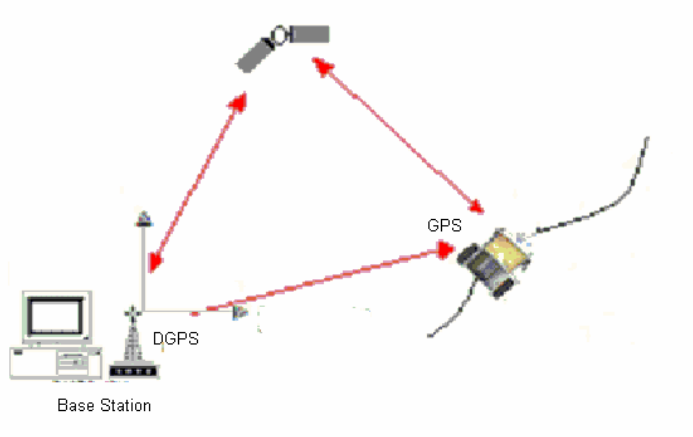



**Fig. 1.** General view of trajectory planning and obstacle avoidance

A GPS base station, stationary PC, and RF Module is placed on initial coordinates. They are then used to give the reference waypoints of the trajectory and to control the motion of the vehicle via serial bilateral data communication.The mobile vehicle is equipped with a mobile GPS, RF module and a micro controller, which controls the motion of the DC motors with caterpillar drive.

In order to let the vehicle to follow a given trajectory in the plane coordinates, a PC as master controller, GPS Base Station connected through an RF Modem are implemented on the zero point of the coordinate frame. The instant position of the vehicle

from longitude and latitude values in accordance to an assigned plane coordinate frame is detected by using a mobile GPS, which is connected and mounted with an RF on the vehicle.

Through the communication between base and mobile GPS's the position and orientation of the vehicle rear axis midpoint will be calculated in given sampling time differences by the PC and the motors then will be guided in accordance to the calculated navigation error.

**Fig. 2.** General view of DGPS-System

As first step a correlation between measured latitude and longitude values with plain coordinates (X-Y) as real distances are investigated through various measurements.

These are then used for identification of the vehicle-location in real plain and so for trajectory control. Details of the calculations and control process are given in equitation's in below.

For further use of obstacle avoidance, an Ultrasonic Sensor for distance estimation is also used which can be turned by a stepper motor fixed to the vehicle. When the sensor detects an object (obstacle) on the trajectory the vehicle will be stopped ca.40 cm before the obstacle .The stepper motor turns the sensor to left and right sides to identify the location of the obstacle, so that it can be passed away optimally. Software based on a special algorithm runs PC, which executes sub control of the motion.

## 3 Position Estimation Using GPS System

To find out the accuracy and repeatability of the GPS System, measured land coordinates as latitude and longitude and their correspondence with absolute plain coordinates (X-Y) are tested by some measurements.

For this reason the absolute navigation data's of some known fixed points within a distance of 5 m are compared with measured GPS values by using statistical evaluation of the calculated variance and standard deviation.

It is seen that the absolute error changed between 2-3 centimeters, which was an excellent accuracy for control distances of 1-2 meters. Also a very reliable repeatability is achieved with the applied GPS System. A general view of the DGPS System is given in Fig.2

Also the time response of the vehicle to motion input changes are investigated, so that a consequent sampling time (here 3 seconds) can be chosen for an efficient control algorithm. Firstly the angular velocities of the rear wheels of the vehicle are exactly measured and in accordance the velocity of the midpoint C of the rear axis is investigated with the equations as given in chapter 4. The taken out way of C in a given time can so be calculated easily and with enough accuracy.

In this application a constant velocity of the point C is assumed, although a velocity control for a accurate path following is required. But using a very simple control algorithm with an intermittent correction by appropriate choice and investigation of sampling distances, a very good tracking accuracy within few centimeters is reached.

The position (location and orientation) of the vehicle is dedicated continuously by the mobile GPS with appropriate sampling measurements and are sent then via RF Module to the main (stationary) PC, where after some calculations in accordance to the control algorithm first the orientation is corrected and then the vehicle is guided to the next destination point by the micro controller driven DC Motors. This process is then continuously repeated until the last destination point on the path is reached.

## 4 Kinematics of the Motion

Tracking of a given path (path following) with global coordinates, is the main aspect for the guidance of the vehicle. As first step, the vehicle as a holonomic model and its non-linear kinematics equations are given with following assumptions:

The vehicle is built up as a caterpillar track of length L, equipped with geared chain mechanisms, which are controlled from rare wheels. (Duo cycle steering) Fig.3

The center point of rear axle C and its linear velocity $v$ is taken as the base aspects to the motion. The velocities ω of the both rear wheels are chosen as main control parameters.

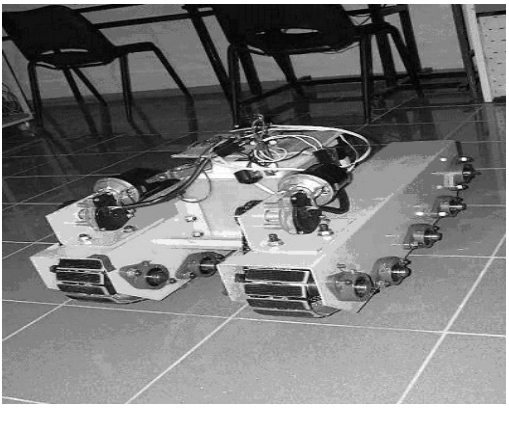


**Fig. 3.** General view of the Vehicle

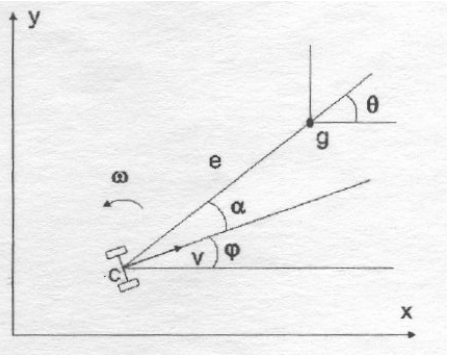



**Fig. 4.** The kinematical model of the vehicle

The kinematics equations that relate the linear and angular velocity of the robot with the angular velocity of each wheel can be expressed as: Fig.4

$$\begin{bmatrix} v \\ w \end{bmatrix} = \begin{bmatrix} R/2 & R/2 \\ -R/D & R/D \end{bmatrix} * \begin{bmatrix} w_L \\ w_R \end{bmatrix} \tag{1}$$

$$\dot{x} = v.\cos\varphi \tag{2}$$

$$\dot{y} = v.\sin\varphi \tag{3}$$

$$\dot{\varphi} = w \tag{4}$$

Where $w_R$ and $w_L$ represent the right and left wheel angular velocities, $v$ and $w$ denote the linear and angular velocities of a coordinate frame attached at center point C. The parameter $R$ stands for the wheel radius and the inter-wheel distance is $2D$. Now, the vehicle can be considered as a point at c which has to follow given waypoints in an absolute coordinate system $(x, y)$. If the coordinates of some successive points are given as $P_1(x_1,y_1)$, $P2(x_2,y_2)$ and $P3(x_3,y_3)$ then the vector equations between successive points can be written as;

$$R1 = (x1)\ i + (y1)\ j \tag{5}$$

$$R2 = (x2\text{-}x1)\ i + (y2\text{-}y1)j \tag{6}$$

$$R3 = (x3\text{-}x2)\ i + (y3\text{-}y2)\ j \tag{7}$$

$$R2*R3 = r_2.r_3.\sin.\alpha_{23}\ k \tag{8}$$

The distances and orientation angles between the waypoints can then be calculated from the coordinates instantaneously.

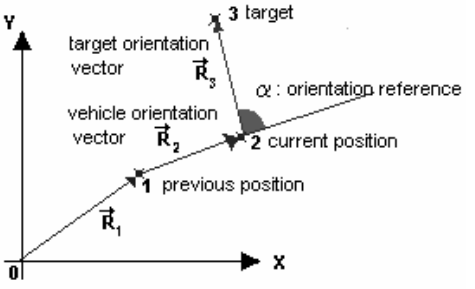



**Fig. 5.** Position and orientation of the vehicle

## 5 Control of the System

We consider the dominant dynamics of the two independent DC motors, which drive the wheels of the vehicle. The dynamic relation between the input voltage u(t) and the angular velocity of the motor shaft w(t) is expressed by the following differential equation (7). To correct the orientation in sampling increments, the time for a turn angle of 180 degrees around midpoint C is dedicated. Also the way of the midpoint by the time is dedicated with some measurements.

Assuming that both motors have the same dynamics Equation (1) can be used to obtain a dynamic model relating the differential/common mode voltages applied to the motors and the vehicle's angular/linear velocities:

$$Tw_i'(t) + w_i(t) = Ku_i(t); \quad i = 1,2 \tag{9}$$

$$v(t) + Tv'(t) = \frac{KR}{2} u_+(t) \tag{10}$$

$$w(t) + Tw'(t) = \frac{KR}{D} u_-(t) \tag{11}$$

For the guidance of vehicle, reference way points are given and with estimating an optimal sampling time in accordance to the given trajectory, the vehicle is guided by classical P algorithms with changing Gain. By Software written in visual basic language whole trajectory calculations and control actions are executed by the stationary computer. Assuming a motion in the plane, the plane coordinates are evaluated from measured mobile GPS values for orientation of the vehicle.

The measured values are fed back to be compared with the given reference coordinate values, which are specified as next following point on the desired path as shown in Fig.5. According to the position error, the turning direction and linear velocities? for the both wheels of the vehicle are controlled in accordance to chosen gain. (Fig.6) Various tests with different Gain Values are made to obtain available and optimally path following with changing curvatures. Successful results are obtained in control applications with an accuracy of 20-30 centimeters.

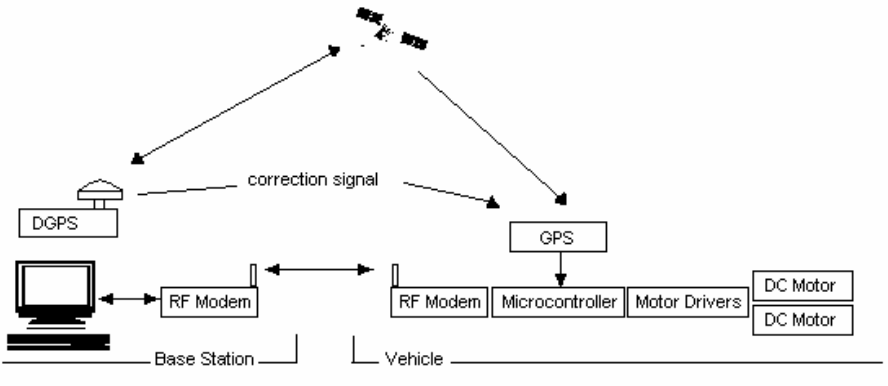



**Fig. 6.** Block diagram of control system

## 6 Experimental Results

Some experiments have been carried out to investigate the dynamic of the control system. First of all the response and data capture ability of the system are investigated so that, reasonable control parameters can be estimated

To investigate the effects of the output parameters on control ability, experiments are being made with various turn angles and velocities. The open and closed loop responses of the system are experimented separately using various test distances and paths; also to observe the tracking ability and accuracy of the system. Fig.7.a.b
The closed loop response of the system is tested by giving some reference waypoints in land plane.The tracking ability of the vehicle in accordance some given way points are shown in Fig.7 a.b

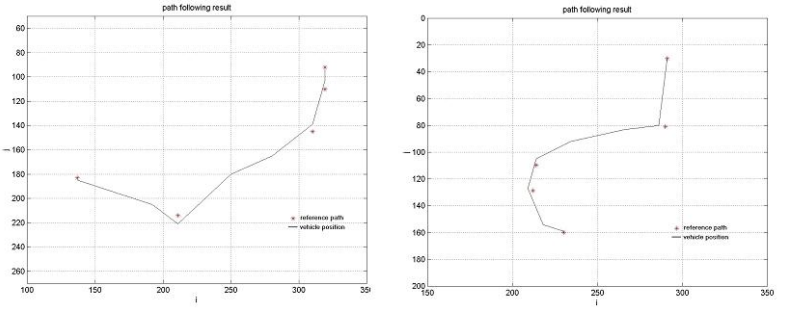



**Fig. 7.** Closed loop control and trajectory following of the vehicle

## 7 Conclusion

The trajectory following of a caterpillar duo cycle vehicle with chain drive is implemented and tested in this work. The non- linearity's of the system in an on line

application is overcome using special control algorithms. In a former work using camera (image effects) [4] in closed residences to estimate the distances, a nonlinear transformation table was used and the accurate measuring distances were limited till to 2 m because of the nonlinearities and sight area of the camera. These are overcome here using a DGPS system which has a great accuracy even in distances till to Kilometers with linear relations.So the dependence of the camera images to light conditions are also eliminated.On the other hand the direct detection and correction of the coordinates in every sampling instance made it possible to compansate the position (location) and orientation errors and so it provided to a great trajectory following accuracy in spite of an intermittent control algorithmus.

As a further study of trajectory following with speed control, a polynomial path from 6th order is thought to be used to fit any given trajectory optimally. To achieve a desired path tracking primarily the radius of curvatures are calculated so that the centre point of the rear axe can be hold on the given path.

## References

1. Uyar E. and Cetin L., Gören A., Trajectory Planning and Obstacle Avoidance Control Of A Mobile Vehicle. Third Triennial International Conference on Applied Automatic Systems Ohrid, Rep. Macedonia, 18-20.Sept.2005
2. Cordesses L,Thuilot B., Martinet P.,Cariou C.Curved Path Following of a Farm Tractor Using a CP-DGPS, Syroco '00, Vienna, 2000.
3. Banta A. Model Creation for a Mobile Robot Using 2 Data and Features. 6th IFAC Symposium on Robot Control Vol.2. Sept 2000.Vienna Austria
4. Groavac S.. A New Visual Navigation Algortihm Using Nonlinear Accelaration Measurements. 10th Mediterrenian Conference on control&automation.June, 2002, Lisbon Portugal

# A Generative Model for Multi Class Object Recognition and Detection

Ilkay Ulusoy

METU, Electrical and Electronics Eng. Department, Ankara Turkey
ilkay@metu.edu.tr

**Abstract.** In this study, a generative type probabilistic model is proposed for object recognition. This model is trained by weakly labelled images and performs classification and detection at the same time. When test on highly challenging data sets, the model performs good for both tasks (classification and detection).

## 1 Introduction

In recent years object recognition is being approached using machine learning techniques based on probability theory. From basic decision theory [2] we know that the most complete characterization of the solution is expressed in terms of the set of posterior probabilities $p(k|\mathbf{X})$ where $k$ is one of the classes and $\mathbf{X}$ is the vector which describes the image. Once we know these probabilities it is straightforward to assign the image $\mathbf{X}$ to a particular class to minimize the expected loss.

Since detailed hand-segmentation and labelling of images is very labour intensive, learning object categories from 'weakly labelled' data is studied in recent years. Weakly labelled data means that training images are labelled only according to the presence or absence of each category of object. A major challenge presented by this problem is that the foreground object is accompanied by widely varying background clutter, and the system must learn to distinguish the foreground from the background without the aid of labelled data.

A key issue in object recognition is the need for predictions to be invariant to a wide variety of transformations of the input image. Any member of a category of objects should be recognized in spite of wide variations in visual appearance due to variations in the form and colour of the object, occlusions, geometrical transformations (such as scaling and rotation), changes in illumination, and potentially non-rigid deformations of the object itself. As a soltion to these variations, many of the current approaches rely on the use of local features obtained from small patches of the image. Informative features selected using some information criterion versus generic features were compared in [11] and although the informative features used were shown to be superior to generic features when used with a simple classification method, they are not invariant to scale and orientation. By contrast, generic interest point operators such as saliency [6], DoG (Difference of Gaussians) [8] and Harris-Laplace [9] detectors are invariant to location, scale and orientation, and some are also affine invariant [8] [9] to some degree.

In the hierarchy of object recognition problems, one upper level is the localization of the objects in the view. Most of the methods are good either in classification [3] or

F.A. Savacı (Ed.): TAINN 2005, LNAI 3949, pp. 32–40, 2006.

detection [7] but not both. In [7] an implicit shape model is provided for a given object category and this model consists of a class specific alphabet of local appearances that are prototypical for the object category and of a spatial probability distribution which specifies where each codebook entry may be found on the object. Fergus et al. [5] learn jointly the appearances and relative locations of a small set of parts. Since their algorithm is very complex, the number of parts has to be kept small. [4] tried to find out informative features (i.e. object features) without considering spatial relationship between the features based on information criteria. However, in this supervised approach, hundreds of images were hand segmented. Finally, Xie and Perez [12] extended the GMM based approach of [4] to a semi-supervised case. A multi-modal GMM was trained to model foreground and background features where some uncluttered images of foreground were used for the purpose of initialization.

In this study we will also follow a probabilistic approach for object classification using weakly labelled data set and details of our model is given in Section 2. Our model also has the ability to localize the objects in the image. We do not consider a spatial model but we will label each feature as belonging to the object or background. We will focus on the detection of objects within images by combining information from a large number of patches of the image.In this study we will use DoG interest point detectors, and at each interest point we extract a 128 dimensional SIFT feature vector [8]. Following [1] we concatenate the SIFT features with additional colour features comprising average and standard deviation of $(R, G, B)$, $(L, a, b)$ and $(r = R/(R+G+B)$, $g = G/(R+G+B))$, which gives an overall 144 dimensional feature vector [10]. In Section 3 we will provide experiments we have performed and discuss our results.

## 2 The Generative Model

We used a probabilistic generative model and model the joint distribution of image label and image $p(\mathbf{t}, \mathbf{X})$. Here $\mathbf{X}$ denotes the image and $\mathbf{t}$ denotes the image label vector with independent components $t_k \in \{0, 1\}$ in which $k = 1, \ldots K$ labels the class. Each class can be present or absent independently in an image. $\mathbf{X}$ denotes the observation for image. $\mathbf{X}$ comprises a set of $J$ patch vectors $\{\mathbf{x}_j\}$ where $j = 1, \ldots, J_n$. $\tau_\mathbf{j}$ denotes the patch label vector for patch $j$ with components $\tau_{jk} \in \{0, 1\}$ denoting the class of the patch. These are mutually exclusive, so that $\sum_{k=1}^{K} \tau_{jk} = 1$.

By decomposing the joint distribution into $p(\mathbf{t}, \mathbf{X}) = p(\mathbf{X}|\mathbf{t})p(\mathbf{t})$ we model the two factors separately. The prior probability $p(\mathbf{t})$ is specified in terms of $K$ parameters $\psi_k$ where $0 \leqslant \psi_k \leqslant 1$ and $k = 1, \ldots, K$:

$$p(\mathbf{t}) = \prod_{k=1}^{K} \psi_k^{t_k} (1 - \psi_k)^{1-t_k}. \tag{1}$$

The $\psi_k$ parameters can be taken directly as the real world frequencies.

The remainder of the model, $p(\mathbf{X}|\mathbf{t})$, is specified in terms of the conditional probabilities $p(\mathbf{X}|\mathbf{t}) = p(\boldsymbol{\tau}|\mathbf{t})p(\mathbf{X}|\boldsymbol{\tau})$. The probability of generating a patch from a particular class is governed by a set of parameters $\pi_k$, one for each class, such that $\pi_k \geqslant 0$, constrained by the subset of classes actually present in the image. Thus

$$p(\boldsymbol{\tau}|\mathbf{t}) = \left(\sum_{l=1}^{K} t_l \pi_l\right)^{-1} \prod_{k=1}^{K} (t_k \pi_k)^{\tau_k} \tag{2}$$

Note that there is an overall undetermined scale to these parameters, which may be removed by fixing one of them, e.g. $\pi_1 = 1$.

For a given class, the distribution of patch feature vector $\mathbf{x}$ is governed by a separate mixture of Gaussians which we denote by

$$p(\mathbf{x}|\boldsymbol{\tau}) = \prod_{k=1}^{K} \phi_k(\mathbf{x}; \boldsymbol{\theta}_k)^{\tau_k} \tag{3}$$

where the model $\phi_k(\mathbf{x}; \boldsymbol{\theta}_k)$ is given by

$$\phi_k(\mathbf{x}; \boldsymbol{\theta}_k) = \sum_{m=1}^{M} \rho_{km} \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_{km}, \boldsymbol{\Sigma}_{km}).$$

If we assume $N$ independent images, and for image $n$ we have $J_n$ patches drawn independently, then the joint distribution of patch labels and patch feature vectors is

$$\prod_{n=1}^{N} p(\mathbf{t}) \prod_{j=1}^{J_n} p(\mathbf{x}_{nj}|\boldsymbol{\tau}_{nj}) p(\boldsymbol{\tau}_{nj}|\mathbf{t}_n). \tag{4}$$

Since the $\{\boldsymbol{\tau}_{nj}\}$ are unobserved we use the EM algorithm. The expected complete-data log likelihood is given by

$$\sum_{n=1}^{N} \sum_{j=1}^{J_n} \left\{ \sum_{k=1}^{K} \langle \tau_{njk} \rangle \ln \left[ t_{nk} \pi_k \phi_k(\mathbf{x}_{nj}) \right] - \ln \left( \sum_{l=1}^{K} t_{nl} \pi_l \right) \right\}. \tag{5}$$

The expected values of $\tau_{nkj}$ are computed in the E-step using

$$\begin{aligned} \langle \tau_{njk} \rangle &= \sum_{\{\boldsymbol{\tau}_{nj}\}} \tau_{njk} p(\boldsymbol{\tau}_{nj}|\mathbf{x}_{nj}, \mathbf{t}_n) \\ &= \frac{\sum_{\{\boldsymbol{\tau}_{nj}\}} \tau_{njk} p(\boldsymbol{\tau}_{nj}, \mathbf{x}_{nj}|\mathbf{t}_n)}{\sum_{\{\boldsymbol{\tau}_{nj}\}} p(\boldsymbol{\tau}_{nj}, \mathbf{x}_{nj}|\mathbf{t}_n)} \\ &= \frac{t_{nk} \pi_k \phi_k(\mathbf{x}_{nj})}{\sum_{l=1}^{K} t_{nl} \pi_l \phi_l(\mathbf{x}_{nj})}. \end{aligned} \tag{6}$$

Notice that the first factor on the right hand side of (2) has cancelled in the evaluation of $\langle \tau_{njk} \rangle$.

Setting the derivative with respect to one of the parameters equal to zero and re-arranging the equality, the re-estimation equations are obtained and following the EM algorithm all parameters are estimated. For details of the method please refer to [10].

This model can be viewed as a generalization of that presented in [12] in which a parameter is learned for each mixture component representing the probability of that component being foreground. This parameter is then used to select the most informative $N$ components in a similar approach to [4] and [11] where the number $N$ is chosen heuristically. In our case, however, the probability of each feature belonging to one of the $K$ classes is learned directly.

Given all patches $\mathbf{X} = \{\mathbf{x}_j\}$ from an image, for the inference of the patch labels, the posterior probability of the label $\boldsymbol{\tau}_j$ for patch $j$ can be found by marginalizing out all other hidden variables

$$\begin{aligned} p\left(\boldsymbol{\tau}_j|\mathbf{X}\right) &= \sum_{\mathbf{t}} \sum_{\boldsymbol{\tau}/\boldsymbol{\tau}_j} p\left(\boldsymbol{\tau}, \mathbf{X}, \mathbf{t}\right) \\ &= \sum_{\mathbf{t}} p\left(\mathbf{t}\right) \frac{1}{\left(\sum_{l=1}^{K} \pi_l t_l\right)^J} \prod_{k=1}^{K} \left(\pi_k t_k \phi_k\left(\mathbf{x}_j\right)\right)^{\tau_{jk}} \prod_{i \neq j} \left[\sum_{k=1}^{K} \pi_k t_k \phi_k\left(\mathbf{x}_i\right)\right] \end{aligned} \tag{7}$$

where $\boldsymbol{\tau} = \{\boldsymbol{\tau}_j\}$ denotes the set of all patch labels, and $\boldsymbol{\tau}/\boldsymbol{\tau}_j$ denotes this set with $\boldsymbol{\tau}_j$ omitted. Note that the summation over all possible $\mathbf{t}$ values, which must be done explicitly, is computationally expensive.

Inference of image label needs almost as much computation as patch label inference where the posterior probability of image label $\mathbf{t}$ can be computed using

$$p\left(\mathbf{t}|\mathbf{X}\right) = \frac{p\left(\mathbf{X}|\mathbf{t}\right) p\left(\mathbf{t}\right)}{p\left(\mathbf{X}\right)} \tag{8}$$

where $p(\mathbf{t})$ is computed from the data set, $p(\mathbf{X})$ is the normalization factor and $p\left(\mathbf{X}|\mathbf{t}\right)$ is calculated by integrating out patch labels

$$\begin{aligned} p\left(\mathbf{X}|\mathbf{t}\right) &= \sum_{\boldsymbol{\tau}} \prod_{j=1}^{J} p\left(\mathbf{X}, \boldsymbol{\tau}|\mathbf{t}\right) \\ &= \prod_{j=1}^{J_n} \frac{\sum_{k=1}^{K} t_k \pi_k \phi_k\left(\mathbf{x}_j\right)}{\sum_{l=1}^{K} t_l \pi_l}. \end{aligned} \tag{9}$$

## 3 Results and Discussion

In this study, we have used four test sets (motorbikes, bicycles, people and cars) in which the objects vary widely in terms of number, pose, size, colour and texture. Some examples from each set are given in Figure 1. The sets are obtained from PASCAL challenge[1] and are formed by combining some other sets. We used "train and validation" set for training and "test" set for testing for each category. Test and train image

[1] Detailed information can be reached at $(http : //www.pascal - network.org/challenges/VOC/)$.

**Fig. 1.** Example images from test and training sets of motorbikes (first row), bicycles (second row), people (third row) and cars (last row)

**Table 1.** Train and test set image numbers for each set

| Set | Train | Test |
|---|---|---|
| Motorbikes | 216 | 216 |
| Bicycles | 114 | 114 |
| People | 84 | 84 |
| Cars | 272 | 275 |

numbers are given in Table 1. As can be observed from the images, the sets are highly challenging.

An image is labelled by the objects present in the image and also the bounding boxes of the objects in the image are given. In this study, we aim to learn the object categories from weakly labelled images so we do not intend to use bounding box information. The bounding box information is used to test the localization success of the model.

Our model can handle multiple categories to be present in the same image. However, since the train sets (but not the test sets) we used in this study have objects from a single category only, we construct a model for each category. Each model has a separate Gaussian mixture for foreground object and background, each of which has 10 components with diagonal covariance matrices.

Initial results showed that with random initialization of the mixture model parameters it is incapable of learning a satisfactory solution. We conjectured that this is due to

the problem of multiple local maxima in the likelihood function (a similar effect was found by [12]). Thus we used one-tenth of training images with their bounding box information for initialization purposes. Feautures inside the bounding box in an image are labelled as foreground and other features are labelled as background. Features belonging to each class were clustered using the K-means algorithm and the component centers of a class mixture model were assigned to cluster centers of the respective class. The mixing coefficients were set to the number of points in the corresponding cluster divided by the total number of points in that class. Also, covariance matrixes were computed using the data points assigned to the respective center.

After training a model for each category, we test it for classification and detection. By classification we mean predicting presence/absence of an example of that class in the test image as been defined by The PASCAL Visual Object Classes Challenge 2005. The classifier produces a real valued output which is actually the probability of image being

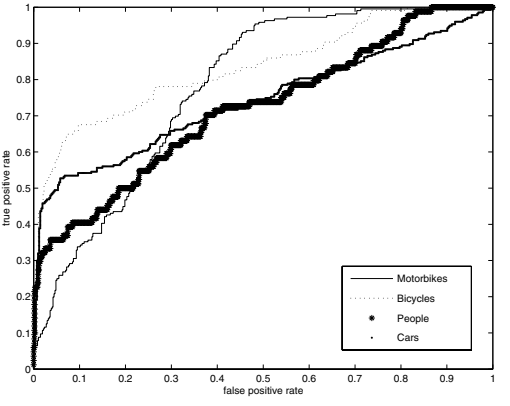



**Fig. 2.** ROC curves for motorbikes, bicycles, people and cars

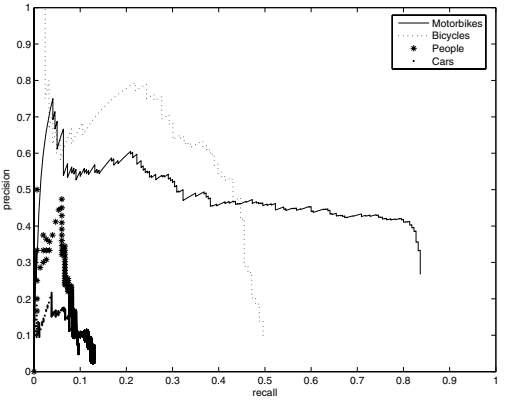



**Fig. 3.** Precision recall curves for motorbikes, bicycles, people and cars

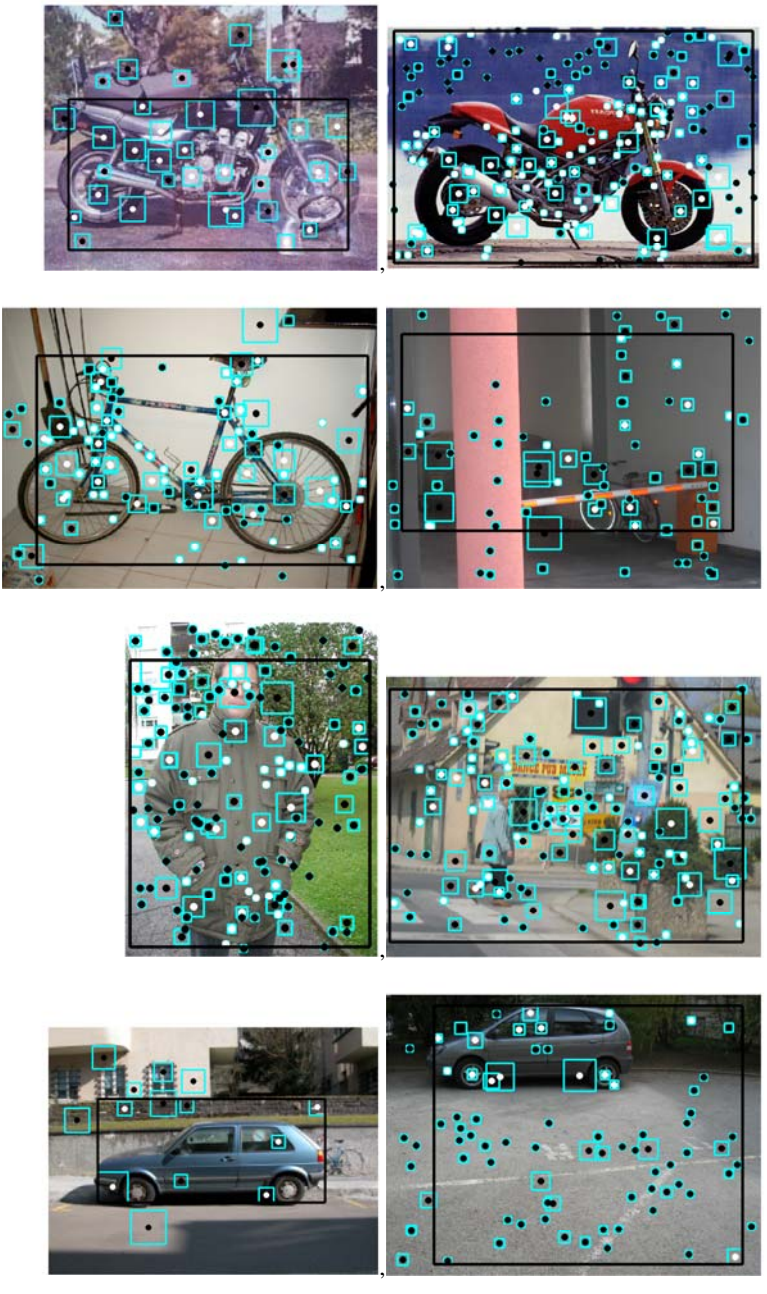


**Fig. 4.** Example images from test and training sets of motorbikes (first row), bicycles (second row), people (third row) and cars (last row).

labelled as belonging to the category for which the model was trained. By applying different threshold values an ROC curve (true positive versus false positive curves) is

plotted for each category. In Figure 2 ROC curves for all categories are plotted. The equal error rates are $70, 75, 65$ and $68\%$ for motorbikes, bicycles, people and cars sets respectively.

By detection we mean predicting the bounding box and label of each object from four target classes in the test image as been defined by The PASCAL Visual Object Classes Challenge 2005. Our model produces a label for each patch as foreground or background. We construct a bounding box as a rectangle which contains all the patches labelled as foreground. Note that in this case we did not consider multiple objects in an image although we have many such images in the test sets. To be considered a correct detection, the area of overlap between the predicted bounding box and ground truth bounding box must exceed $50\%$. The results of detection are given as precision (true positive over false positive and true positive) recall (true positive over allpositives in the database) curves in Figure 3.

Our model does not have the best classification and detection performances. However, for these challenging sets the model performs well for both localization and classification at the same time.

Both classification and detection are based on patch labelling. Thus patch labelling performance defines the classification and detection performances. In our model, the number of components of Gaussian Mixtures has to be kept small and diagonal matrix instead of full covariance matrix has to be used due to computational problems. Using full covariance matrix brings a very high computational load for 144 dimensional feature vector. Thus we have to use 10 components for each mixture and a diagonal covariance matrix for each component which limit the success of the model. Some patch labelling results for different categories are given in Figure 4. Here patches are shown with squares and each patch centre includes a coloured circle which denotes patch label (white for foreground and black for background). Each row is for a different category (motorbikes, bicycles, people, cars starting from top to bottom). The images in the left column show some good detections whereas right column is for bad detections. A bounding box is shown with a rectangle in each image.

Poor detection is because of our bounding box definition for which we do not consider spatial relationship between patches. All foreground patches are included into the bounding box to give the object location. Any wrong labelled background patch causes wrong detection. For example, the image in the fourth row and right column has a single patch which is labelled as car although it is a background patch. Since this patch is far away from other foreground patches, it causes wrong bounding box placement. Thus bounding box construction should be improved by including spatial relationship between the patches. Also the generative model can be improved by including the spatial relations between the patches.

## References

1. K. Barnard, P. Duygulu, D. Forsyth, N. Freitas, D. Blei, and M. I. Jordan. Matching words and pictures. *Journal of Machine Learning Research*, 3:1107–1135, 2003.
2. C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, 1995.
3. G. Csurka, C. R. Dance, L. Fan, J. Willamowski, and C. Bray. Visual categorization with bags of keypoints. In *Workshop on Statistical Learning in Computer Vision, ECCV*, 2004.

4. G. Dorko and C. Schmid. Selection of scale invariant parts for object class recognition. In *ICCV*, 2003.
5. R. Fergus, P. Perona, and A. Zisserman. Object class recognition by unsupervised scale invariant learning. In *CVPR*, 2003.
6. T. Kadir and M. Brady. Scale, saliency and image description. *International Journal of Computer Vision*, 45(2):83–105, 2001.
7. B. Leibe, A. Leonardis, and B. Schiele. Combined object categorization and segmentation with an implicit shape model. In *Workshop on Statistical Learning in Computer Vision, ECCV*, 2004.
8. D. Lowe. Distinctive image features from scale invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004.
9. K. Mikolajczyk and C. Schmid. Scale and affine invariant interest point detectors. *International Journal of Computer Vision*, 60:63–86, 2004.
10. I. Ulusoy and C. M. Bishop. Generative versus discriminative methods for object recognition. In *CVPR*, 2005.
11. M. Vidal-Naquet and S. Ullman. Object recognition with informative features and linear classification. In *ICCV*, 2003.
12. L. Xie and P. Perez. Slightly supervised learning of part-based appearance models. In *IEEE Workshop on Learning in CVPR*, 2004.

# Depth of General Scenes from Defocused Images Using Multilayer Feedforward Networks

Veysel Aslantas and Mehmet Tunckanat

Erciyes University, Faculty of Engineering, Department of Computer Eng., 38039
Melikgazi, Kayseri, Turkey
{Aslantas, Tunckanat}@erciyes.edu.tr

**Abstract.** One of the important tasks in computer vision is the computation of object depth from acquired images. This paper explains the use of neural networks to calculate the depth of general objects using only two images, one of them being a focused image and the other one a blurred image. Having computed the power spectra of each image, they are divided to obtain a result which is independent from the image content. The result is then used for training Multi-Layer Perceptron (MLP) neural network (NN) trained by the backpropagation algorithm to determine the distance of the object from the camera lens. Experimental results are presented to validate the proposed approach

## 1 Introduction

The depth of the visible surface of a scene is the distance between the surface and the sensor. Obtaining this information of a scene from its two-dimensional images can assist numerous applications such as object recognition, navigation and inspection.
Based on focus blur information, three different depth computation techniques have been developed: Depth from Automatic Focusing (DFAF) [1-3], Depth from Defocusing (DFD) [4-16] and Depth from Automatic Defocusing (DFAD) [17].

Previous work on depth from defocusing usually required knowledge of the relation among lens parameters, depth and blur circle radius (e.g., in [3, 5]). The main limitation of this is that those parameters are based on the ideal thin lens model and can never be measured precisely for any camera since real imaging systems have several lenses.

Most of the DFD algorithms found in the literature are also based on assuming that the PSF of the camera is either a Gaussian or a circularly symmetric function to obtain a relation between depth and the spread parameter of the PSF which is potentially the most erroneous part of many of the earlier methods.

NNs have found many applications in vision-related areas. Recently, they have also been employed to compute depth using focus blur [15, 16, 18, 19]. In the proposed method, a neural network [20] is employed to compute the depth of a scene. The technique requires two images of the same scene one of which is sharp the other is defocused. Power spectrum of each images are computed. The spectrums are then divided to acquire a result which is independent from the image content. The result is employed for training NN trained by the BP algorithm to determine the distance of

F.A. Savacı (Ed.): TAINN 2005, LNAI 3949, pp. 41 – 48, 2006.

the object from the camera lens. Therefore, the proposed method is independent of the PSF of the camera and does not require a relation between blur and depth.

The technique does not require special scene illumination and needs only a single camera. Therefore, there are no correspondence and occlusion problems as found in stereo vision and motion parallax techniques or intrusive emissions as with active depth computation techniques.

The remainder of the paper comprises five sections. Section 2 presents the theory underlying the proposed technique. Section 3 explains the experimental procedure for acquiring data to train and test MLPs in depth computation. Section 4 discusses the results obtained. Section 5 concludes the paper.

## 2 Problem Formulation

The transformation effected by an optical system can be modeled as a convolution operation [21]. The image of a blurred object may then be written as:

$$I(x,y) = \int\int H(x-\xi, y-\eta, d(\xi,\eta))S(\xi,\eta)\, d\xi d\eta df \tag{1}$$

where $x$ and $y$ are image coordinates, $\xi$ and $\eta$ are two spatial variables, $S(x,y)$ and $I(x,y)$ are the sharp and defocused images of the source object respectively, $d(x,y)$ is the distance from the object to the plane of best focus (PBF) and $H(x,y,d)$ is the PSF. If the distance from the object to the PBF is constant, then the PSF $H(x,y,d)$ can be written as $H(x,y)$ and the defocusing process is defined as a convolution integral:

$$I(x,y) = \int\int H(x-\xi, y-\eta)S(\xi,\eta)\, d\xi d\eta \tag{2}$$

The convolution operation is usually denoted by the symbol $\otimes$. Therefore, Equation (2) can be abbreviated as:

$$I(x,y) = H(x,y) \otimes S(x,y) \tag{3}$$

In the Fourier domain, Equation (3) can be expressed as:

$$I(u,v) = H(u,v)S(u,v) \tag{4}$$

where: $\{I(x,y), I(u,v)\}$, $\{H(x,y), H(u,v)\}$ and $\{S(x,y), S(u,v)\}$ are Fourier pairs. The function $H(u,v)$ is often referred to as the optical transfer function (OTF).Most of the focus based techniques assume that the distance function $d(x,y)$ is slowly varying, so that it is almost constant over local regions. The defocus is then modeled by the convolution integral over these regions.

### 2.1 Form of Point Spread Function

Figure 1 shows the basic geometry of image formation. Each point in a scene is projected onto a single point on the focal plane, causing a focused image to be formed on it. However, if the sensor plane does not coincide with the focal plane, the image formed on the sensor plane will be a circular disk known as a *"circle of confusion"* or *"blur circle"* with diameter 2R, provided that the aperture of the lens is also circular.

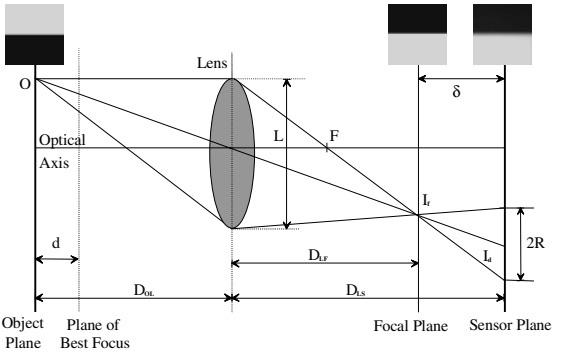



**Fig. 1.** Basic Image Formation Geometry

According to geometrical optics, the intensity distribution within the blur circle is assumed to be approximately uniform i.e., the PSF is a circular *"pillbox"*. In reality, however, diffraction effects and characteristics of the system play a major role in forming the intensity distribution within the blur circle. After examining the net distribution of several wavelengths and considering the effects of lens aberrations the net PSF is best described by a 2D Gaussian function [5, 6]:

$$H(x, y) = \frac{1}{2\pi\sigma^2} e^{-\frac{(x^2+y^2)}{2\sigma^2}} \tag{5}$$

where $\sigma$ is the spread parameter which is proportional to the radius $R$ of the blur circle.

$$R = k\sigma \tag{6}$$

The proportionality constant $k$ depends on the system and can be determined through calibration. The optical transfer function $H(u,v)$ for geometrical optics can be obtained by taking the Fourier transform (FT) of $H(x,y)$:

$$H(u, v) = e^{-\frac{(u^2+v^2)\sigma^2}{2}} \tag{7}$$

By substituting Equation (7) into Equation (4) and solving for $\sigma$, an equation can be written as:

$$\sigma^2 = \frac{-1}{(u^2+v^2)} \ln\left(\frac{I(u,v)}{S(u,v)}\right) \tag{8}$$

where $I(u,v)$ and $S(u,v)$ are the powers of $I(x,y)$ and $S(x,y)$ respectively.

In principle, it is sufficient to calculate $\sigma$ at a single point $(u,v)$ by employing Equation (7). However, a more accurate value can be obtained by averaging $\sigma$ over some domain in the frequency space:

$$\sigma^2 = \frac{-1}{A}\iint_P \frac{1}{(u^2+v^2)} \ln\left(\frac{I(u,v)}{S(u,v)}\right) dudv \tag{9}$$

where $P$ is a region in the $(u,v)$ space containing points where $\left(I(u,v)/S(u,v)\right) > 0$ and $A$ is the area of $P$ [3].

### 2.2 Relating Depth to Camera Parameters and Defocus

The object may be either in front of or behind the plane of best focus on which points are sharply focused on the focal plane. From Figure 1, by using similar triangles, a formula for a camera with a thin convex lens of focal length $F$ can be derived to establish the relationship between the radius $R$ of the blur circle and the distance $D_{OL}$ from a point in a scene to the lens [5]:

$$D_{OL} = \frac{FD_{LS}}{D_{LS} - F - 2fk\sigma} \tag{10}$$

where $D_{LS}$ is the distance between the lens and the sensor plane, $f$ is the $f$-number of a given lens. When the object is in front of the PBF, Equation (10) becomes:

$$D_{OL} = \frac{FD_{LS}}{D_{LS} - F + 2fk\sigma} \tag{11}$$

Equations (10) and (11) relate the object distance $D_{OL}$ to $\sigma$.

### 2.3 Window Effect

The convolution operation uses pixels that are out of the local region and this computation causes leaks. In order to reduce the effect of these image overlap problem [23], the image region must be multiplied by a center weighted function such as Gaussian mask:

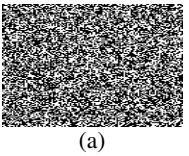
$$w(x,y) = \frac{1}{2\pi s^2} e^{-\frac{x^2+y^2}{2s^2}} \tag{12}$$

where $s$ is the spread parameter of the Gaussian and depends on the window size. $s$ can have a value of the half of the window size [23]. The mask must be centered at the region of interest. The resulting image is then used for depth computation. However, erroneous depth result can be obtained if the relations given above are used.

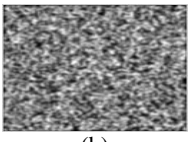
## 3 Collection and Preparation of MLP Training and Test Data

An object can have any form and intensity distribution. Therefore, it is very difficult for a neural network to learn intensity data since a large amount of data corresponding to different object will be needed for each distance. If an image content independent data set is obtained for a particular distance, it is possible to teach a NN using this data. Hence, the following process was made for computing this kind of data set.

For the simulations in this paper, a 100x100 binary random-dot-pattern (RDP) image (Figure 2(a)) was blurred by a 2-D Gaussian function with $\sigma$ =1. The result of this process was a grey level image which was used as the original focused image (Figure 2(b)).



(a)



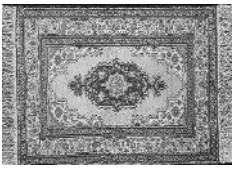
(b)

**Fig. 2.** Images used for the simulations (a) Binary Random Dot Pattern Image (b) Blurred image of (a) ( =1)

In the simulations, the camera was set such that the best focused plane was at infinity (thus the object would always be between the best focused plane and the lens). The focal length and *f*-number of the lens used in the experiments were set to 40mm and 2.8 respectively. The distance of the object (Figure 2(b)) from the camera varied for different images but the camera parameters were the same for all images.

For each distance, the defocused images were generated by convolving the original image with a 2-D Gaussian function given in Equation (5). The spread parameter of the Gaussian function is computed from Equation (11). The camera constant (*k*) had a value of 0.71. The value is a good approximation in most practical cases [9]. To reduce image overlapping effects, the images are multiplied with a Gaussian mask using Equation (12).

Image regions were selected randomly from the same location on sharp and defocused images. The size of the regions was 16x16 and 32x32 pixels for different experiments. The power spectrum of the each image region was calculated. The spectrums are then divided to acquire a result which is independent from the image content. Therefore, a set of data was obtained for a certain distance. Because the result is symmetric along *u* and *v* axes, only first quarter of the data was used in the experiments. The data size was reduced further by taking the square of each coefficient and summing them column wise. Hence, the size of data for the NN was reduced from 16x16 and 32x32 to 1x8 and 1x16 respectively. The calculation was made for each distance ranging from 600mm to 1400mm at intervals of 10mm. The data set computed for each depth was used as input to train the NN. In the experiments, all neural networks used had one output (for depth). The test data set was obtained in the same way from the images given in Figure 3.



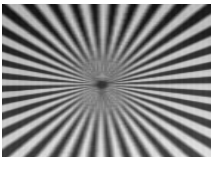


**Fig. 3.** Test Images (a) Rug (b) Test Pattern (c) Baboon

## 4 Experiments

For each distance, ten image regions were selected randomly from the sharp and blurred RDP image shown in Figure 2(b). 10 dB zero-mean Gaussian noise was added to the five of them. The computations described in the previous section were applied on each image region. The data were entered in rows in a text file to prepare the training set. The distance values corresponding to each row were added to the file to act as desired outputs. Several MLP structures were used to search for the best result. In the investigations reported here, the activation functions of the neural networks were tangent (T), linear (P) and logarithmic (L) functions. All the NNs used in this work are trained by the Levenberg-Marquardt (LM) algorithm [22]. After the training process had completed, each neural network was tested with the training and test data sets. The NN structures used for each experiment and their results for the noise free images can be seen in Table 1.

**Table 1.** Calculation and NN Parameters for each experiment

| Exp. No. | Input Neurons | Activation Functions | Hidden Neurons | %Error | | | |
|---|---|---|---|---|---|---|---|
| | | | | Rug | Test Pattern | RDP | Baboon |
| 1 | 16 | T-T-P | 5–13 | 0.4702 | 0.2915 | 0.2760 | 0.3054 |
| 2 | 16 | T-T-P | 20–16 | 0.1164 | 0.2266 | 0.1394 | 0.0574 |
| 3 | 16 | L-T-T-P | 16–12–14 | 0.3187 | 0.2394 | 0.1657 | 0.2056 |
| 4 | 8 | T-T-P | 16–12 | 0.4285 | 0.9963 | 0.1008 | 0.6718 |
| 5 | 8 | T-L-T-P | 8–4–2 | 0.5215 | 0.2964 | 0.4245 | 1.9330 |
| 6 | 8 | L-T-R-P | 13–15–17 | 0.5727 | 0.4076 | 0.1141 | 0.4597 |
| 7 | 16 | T-T-T-P | 5–13–4 | 0.0459 | 0.2008 | 0.0103 | 0.0477 |
| 8 | 8 | L-L-L-P | 18–12–10 | 0.1957 | 0.3159 | 0.1177 | 0.2501 |
| 9 | 16 | T-T-T-L-P | 5–13–4–6 | 0.3589 | 0.0937 | 0.2410 | 0.2005 |
| 10 | 8 | L-L-T-P | 19–11–13 | 0.5180 | 0.2239 | 0.1368 | 0.0752 |

The results of 10 dB noisy images for the seventh and ninth experiments are given in Table 2. It can be noted that the neural networks produced low percentage errors corresponding to high depth computation accuracies. The results for noisy test data sets produced from the Baboon image are plotted in Figure 4. It can be noted that the error in depth computation was about 0.6%.

**Table 2.** The results of 10 dB noisy images

| **Exp No.** | **Rug** | **Test Pattern** | **RDP** | **Baboon** |
|---|---|---|---|---|
| 7 | 0.4678 | 0.5407 | 0.3594 | 0.2310 |
| 9 | 0.4616 | 0.6480 | 0.4402 | 0.3127 |

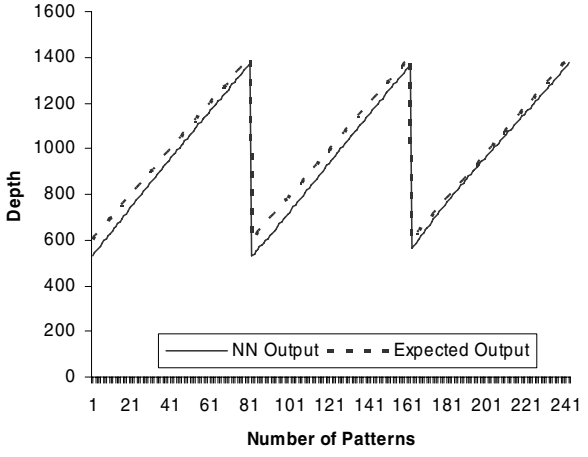



**Fig. 4.** The result of the noisy Baboon image for experiment 7

## 5 Discussion

In this paper, a simple method for computing the depth information of general objects from a sharp and a blurred image of them using a neural network. The experiments results given show that it is possible to compute depth of objects with reasonable accuracy. Previous work on DFD usually required the relation among lens parameters, the depth and blur circle radius. The main limitation of this is that those parameters are based on the ideal thin lens model and can never be measured precisely for any camera since real imaging systems have several lenses. Most of the DFD algorithms found in the literature are also based on assuming that the PSF of the camera is either a Gaussian or a circularly symmetric function to obtain a relation between depth and the spread parameter of the PSF which is potentially the most erroneous part of many of the earlier methods. The proposed method is independent of the PSF of the camera and does not require a relation between blur and depth. The method was implemented to compute the depth of general scenes using their sharp and defocused images. Experimental results have shown that the error was approximately 0.6%.

## References

1. Krotkov E.P.: Focusing, International Journal of Computer Vision, 1(3),(1987),.223-237.
2. Nair H.N., and Stewart C.V.: Robust Focus Ranging, IEEE Conference on Computer Vision and Pattern Recognition, Champaign, Illinois, (1992),309-314
3. Subbarao M., and Choi T.: Accurate Recovery of Three Dimensional Shape from Focus, IEEE Transaction on Pattern Analysis and Machine Intelligence" 17(3), (1995), 266-274
4. Grossmann P.: Depth from Focus, Pattern Recognition Letters, 5(1), (1987), 63-69
5. Pentland A.P.: A New Sense for Depth of Field, IEEE Transaction on Pattern Analysis and Machine Intelligence 9(4), (1987), 523-531

6. Subbarao M., and Gurumoorthy N.: Depth Recovery from Blurred Edges, IEEE Conference on Computer Vision and Pattern Recognition, Ann Arbor, Michigan, (1988),498-503
7. Holeva L.F.: Range Estimation from Camera Blur by Regularised Adaptive Identification, International Journal on Pattern Recognition and Artificial Intelligence 8(6), (1994), 1273-1300
8. Pentland A.P., Scherock S., Darrell T. and Girod B.: Simple Range Cameras based on Focal Error, Journal of Optical Society of America, 11(11),(1994),2925-2934
9. Subbarao M., and Surya G.: Depth from Defocus: A Spatial Domain Approach, International Journal on Computer Vision, 13(3), (1994), 271-294
10. Xu S., Capson D.W. and Caelli T.M.: Range Measurement from Defocus Gradient, Machine Vision Applications, 8, (1995), 179-186
11. Asada N., Fujiwara H. and Matsuyama T.: Edge and depth from focus, International Journal on Computer Vision, 26(2), (1998), 153-163
12. Watanabe M. and Nayar S.K.: Rational filters for passive depth from defocus, International Journal on Computer Vision, 27(3), (1998), 203-225
13. Asada N., Fujiwara H. and Matsuyama T.: Particle depth measurement based on depth-from-defocus, Optics & Laser Technology, 31(1),(1999),95-102
14. Chaudhuri S. and Rajagopalan A.N.: Depth from defocus: A real aperture imaging approach, Springer-Verlag New York, Inc. (1999)
15. Pham D.T. and Aslantas V.: Depth from defocusing using a neural network, Pattern Recognition, 32(5), (1999), 715-727
16. Asif M. and Choi T.S.: Shape from focus using multilayer feedforward neural networks, IEEE Transaction on Image Processing, 10(11), (2001), pp.1670-1675
17. Aslantas V.: Depth from automatic defocusing, sent for publication
18. Aslantas V.,Tunckanat M.: Depth From Image Sharpness Using a Neural Network, ICSP, Canakkale, Turkey, (2003) 260-265
19. Aslantas, V.: Estimation of Depth From Defocusing Using A Neural Network. ICIS International Conference on Signal Processing. Çanakkale, Turkey. (2003) 305-309
20. Rumelhart D.E. and McClelland J. L.: Parallel Distributed Processing: Explorations in the Microstructure of Cognition, MIT Press, Cambridge, MA., (1986)
21. Horn, B.K.P.: Robot Vision. McGraw-Hill, New York (1986)
22. Gill P. E., Murray W., and Wright H.: Practical Optimization, New York: Academic Pres., 1981
23. Wei C., Three Dimensional Machine Vision Using Image Defocus, Ph.D. Dissertation, Electrical Eng., State University of New York at Stony Brook, 1994

# Tracking Control Based on Neural Network for Robot Manipulator

Murat Sonmez[1], Ismet Kandilli[2], and Mehmet Yakut[3]

[1,3] Kocaeli University, Electronics and Telecommunication Dept,
41040 Kocaeli, Turkey
{mrsonmez, myakut}@kou.edu.tr
[2] Kocaeli University, Industrial Electronics Dept. 41500 Kocaeli, Turkey
kandilli@kou.edu.tr

**Abstract.** In this paper, a control algorithm based on neural networks is presented. This control algorithm has been applied to a robot arm which has a highly nonlinear structure. The model based approaches for robot control require high computational time and can result in a poor control performance, if the specific model structure selected does not properly reflect all the dynamics. The control technique proposed here has provided satisfactory results. A decentralized model has been assumed here where a controller is associated with each joint and a separate neural network is used to adjust the parameters of each controller. Neural networks have been used to adjust the parameters of the controllers, being the outputs of the neural networks, the control parameters.

## 1 Introduction

In the past decade, the applications of intelligent control techniques (fuzzy control or neural network control) to the motion control for robot manipulators have received considerable attention [5], [7], [9], [10], [14], [17],[22], [23]. A control system, which comprises PID control and neural network control, was presented by Chen et al. [5] for improving the control performance of the system in real time. Clifton et al. [6] and Misir et al.[17] designed fuzzy-PID controllers which were applied to the position control of robot manipulators. Huang and Lee [9] suggested a stable self organizing fuzzy controller for robot motion control. This approach has a learning ability for responding to the time-varying characteristic of a robot manipulator. However, the fuzzy rule learning scheme has a latent stability problem. Yoo and Ham [23] presented two kinds of adaptive control schemes for robot manipulator via fuzzy compensator in order to confront the unpredictable uncertainties. Though the stability of the whole control system can be guaranteed, some strict constrained conditions and prior system knowledge are required in the control process. On the other hand, Kim and Lewis [10] deal with the application of quadratic optimization for motion control of robotic systems using cerebella model arithmetic computer neural networks. Lewis et al. [14] developed a multilayer neural-net controller for a general serial-link rigid robot to guarantee the tracking performance. Both system-tracking stability and error convergence can be guaranteed in these neural-based-control systems [10], [14], [24], [25]. However the functional reconstructed error, the neural tuning weights and high

F.A. Savacı (Ed.): TAINN 2005, LNAI 3949, pp. 49 – 57, 2006.

order term in Taylor series are assumed to be known bounded functions, and some inherent properties of robot manipulator are required in the design process.

Since the dynamics of robot manipulators are highly nonlinear and may contain uncertain elements such as friction, many efforts have been made in developing control schemes to achieve the precise tracking control of robot manipulators. Conventionally, many control techniques for robot manipulators rely on proportional-integral-derivative (PID) type controllers in industrial operations due to their simple control structure ease of design, and low cost [1], [2], [8]. However, robot manipulators have to face various uncertainties in practical applications, such as pay load parameter internal friction and external disturbance [11], [13], [18]. All the uncertain or time varying factors could affect the system control performance seriously. Many control techniques have been investigated as viable means to improve the shortcomings of the conventional PID type controllers [4], [12], [15], [20]. Sun and Mills proposed an adaptive learning control scheme to improve trajectory performance and could guarantee convergence in single and repetitive operational modes. But the control scheme requires the system dynamics in detail. A model based PID controller was presented by Li et al. [15] to achieve the time varying tracking control of a robot manipulator. However it is difficult to establish an appropriate mathematical model for the design of a model based control system. Thus, the general claim of traditional intelligent control approaches is that they can attenuate the effects of structured parametric uncertainty and unstructured disturbance using their powerful learning ability without prior know ledge of the controlled plant in the design-processes.

## 2 Neural Networks for Approximation

Fig.1 depicts a neural network (ANN) which can be considered as one-layer feed forward neural network with input preprocessing element. The ANN architecture appears to be identical to that of the conventional two-layer neural network expect, for the critical differences that only weights in the output layer will be adjusted. ANN can be used to approximate a nonlinear mapping $z(x): X^n \longrightarrow \psi^m$ where $X^n$ is the application specific n dimensional input space and $\psi^m$ in the application specific output space. If $\sigma(.)$ is a continuous discriminant function, then finite sums of the form

$$y_j(x) = \sum (w_{ji}\sigma_i(p_i + \theta_i) + \theta_j) + \varepsilon_j(x) \text{ , j=1,..,m} \quad (1)$$

are dense in the space of continuous functions defined on hypercube or compact set, $w_{ji}$ output layer weight values, $\theta_i$ , $\theta_j$ threshold values and $p_i$ input to neuron. $\sigma(.)$ is called the sigmoid activation function and N is number of units (also called nodes and neurons) in the hidden layer. $\sigma_i(.)$is the output of the *i*th sigmoid function. We use notation $\sigma_i(.)$ to denote the application of the nonlinear function $\sigma_i(.)$ to each element of the vector *p*. Note that the activation function $\sigma_i(.)$ for each neuron are not necessary the same. Finally, $\varepsilon_j(x)$ is called as the neural network functional reconstruction error.

In general, even given the best possible weight values, the given nonlinear function is not exactly approximated and functional reconstruction errors $\varepsilon_j(x)$ are remaining. We assume that an upper limit of the functional reconstruction error is known. For

control purposes, it is assumed that the ideal approximated neural network weights exist for a specified value of $\varepsilon_M$.

$$\left\|\varepsilon(x)\right\| \le \varepsilon_M \tag{2}$$

Then the neural network equation can be expressed in a compact matrix form

$$y(x) = W^T \sigma(p) + \varepsilon(x) \tag{3}$$

With $W$, $\sigma(p)$, $y(x)$ and $\varepsilon(x)$. The inputs $p$ to neural network is augmented by input preprocessing method [14]. Notice that the threshold values $\theta$ are incorporated into weight matrices $W$ and sigmoid vector $\sigma(p)$. Then for suitable neural network approximation properties, some conditions must be satisfied by $\sigma(p)$. It must be a basis [3]. Then, an estimate $\hat{y}(x)$ of $y(x)$ can be written as

$$\hat{y}(x) = \hat{W}^T \sigma(p) \tag{4}$$

where $\hat{W}$ are estimates of the ideal neural network weights that are provided by some online weight tuning algorithms. The NN in [2], [10] will be considered in the remainder of this paper. It is proven that linearity in the unknown parameters has the so called best approximation property. In particular in Barron's paper [16], it was shown that neural network can serve as universal approximators for continuous functions more efficiently than traditional functional approximators, such as polynomials, trigonometric expansions, or splines, even though there exist fundamentally lower bound on the functional reconstruction error, $(1/N)^{2/n}$ for the static nonlinear mapping.

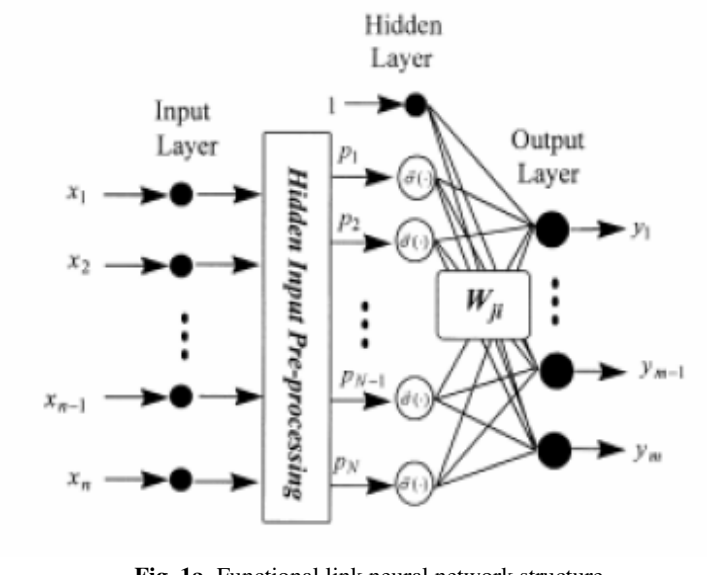



**Fig. 1a.** Functional link neural network structure

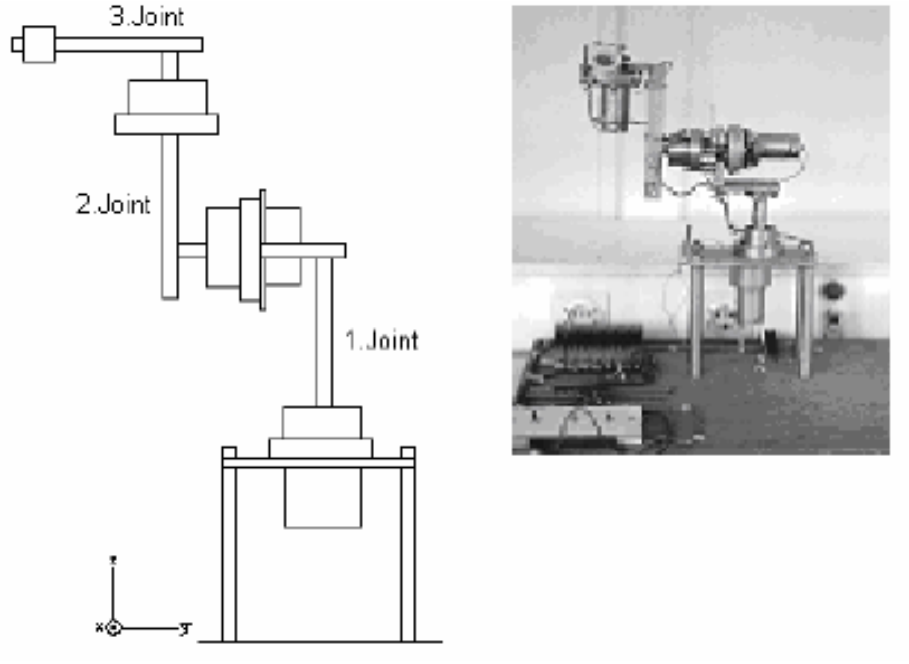



**Fig. 1b.** Robot Manipulator

## 3 Robot Dynamics

Consider the dynamic model of an n rigid-link robot manipulator in the following Lagrange form

$$M(q)\ddot{q} + V_m(q,\dot{q})\dot{q} + G(q) + F(\dot{q}) + T_L(t) = \tau(t) \tag{5}$$

where $q, \dot{q}, \ddot{q}$ denote the joint position, velocity and acceleration vectors, respectively: $M(q)$ is inertia matrix, which is symmetric and positive definite; $V_m(q,\dot{q})$ is centripetal-Coriolis matrix: $G(q)$ is the gravity vector; $F(\dot{q})$ represents the static and dynamic friction terms; $T_L(t)$ is the vector of disturbances and unmodelled dynamic and $\tau(t)$ is the control vector of torque by the joint actuators. The control problem is to find a control law so that the state $q(t)$ can track the desired command $q_d(t)$. To achieve this control objective define the tracking error vector $e(t)=q_m(t)-q(t)$, in which $q_m(t)$ represents the reference trajectory. The instantaneous performance measure is defined as

$$r(t) = \dot{e}(t) + \wedge e(t) \tag{6}$$

where $\wedge$ is the constant gain matrix or critic (not necessarily symmetric). The robot dynamic may be written as

$$M(q)(\ddot{q}_d + \wedge\dot{e}) + V_m(q,\dot{q})(\dot{q}_d + \wedge e) + g(q) + F_v(q) + f_c(\dot{q}) + \tau_d(t) \tag{7}$$

and for instance,

$$x = \left[e^T, \dot{e}^T, q_d^T, \dot{q}_d^T, \dot{q}_d^T\right]^T \tag{8}$$

this key function h(x) captures all the unknown dynamics of robot arm. Define now a control input torque as

$$\tau(t) = h(x) - u(t) \tag{9}$$

with u(t) an auxiliary control input to be optimized later. The closed loop system becomes

$$M(q)\dot{r}(t) = -V_m(q,\dot{q})r(t) + u(t) \tag{10}$$

## 4 Simulation Results

The dynamic equations for an n-link manipulator can be found in Lewis et al. [14]. The cost functional to be minimized is

$$J(u) = \frac{1}{2}\int_0^\infty (\tilde{z}^T Q\tilde{z} + u^T Ru)dt \tag{11}$$

An external disturbance and frictions are

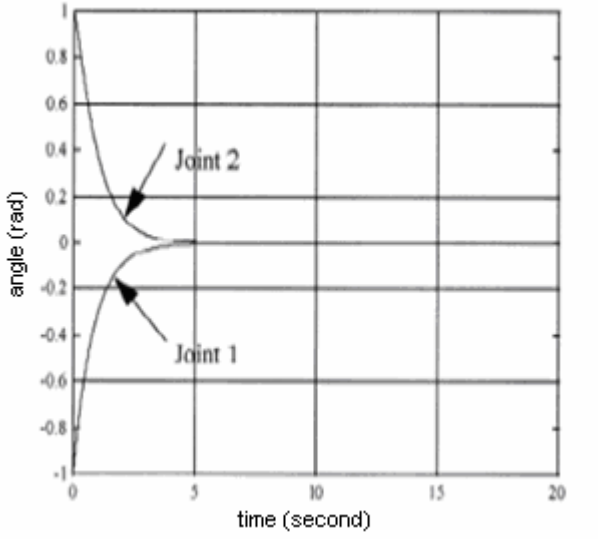



**Fig. 2.** Performance of Tracking Controller (a) tracking error for slow motion

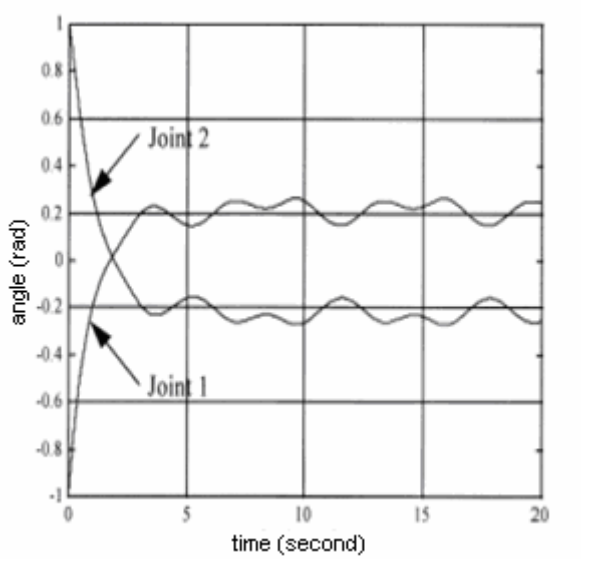



**Fig. 3.** Performance of Tracking Controller (a) tracking error for fast motion

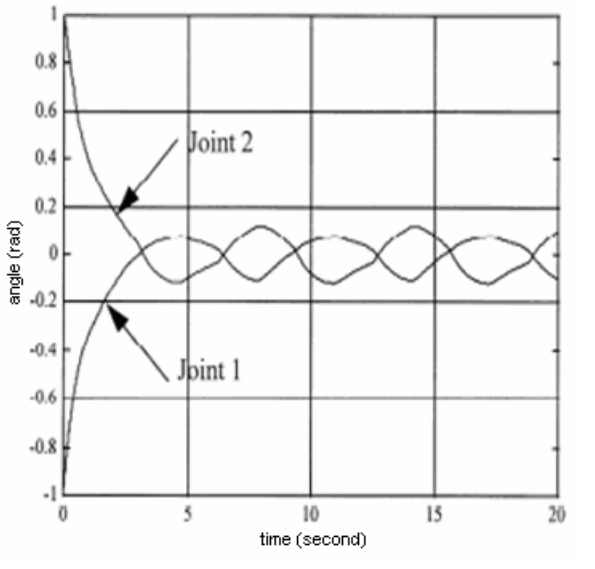



**Fig. 4.** Performance of Tracking Controller (b) tracking error for fast motion

$$\tau_d(t) = \begin{bmatrix} 8\sin(2t) \\ 8\cos(t) \end{bmatrix} \tag{12}$$

The weighting matrices are $Q_{11}$, $Q_{12}$ and $Q_{22}$.

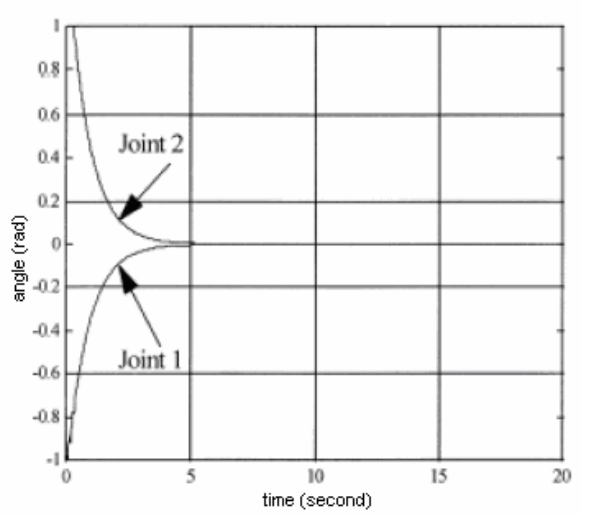



**Fig. 5.** Performance of ANN controller (b) tracking error for slow motion

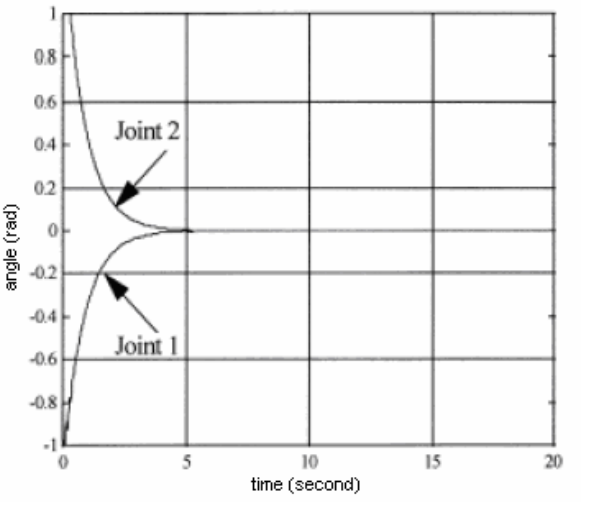



**Fig. 6.** Performance of ANN controller (b) tracking error for fast motion

The motion problem considered is for the robot end factor to track a point on a circle centered at x=y=0.05 m and radius 0.05 m which turns ½ times per second in slow motion and 2 times per second in fast motion. It was pointed out that control system performance may be quite different in low and high speed motion. Therefore, we carry out our simulation for two circular trajectories.

The desired positions in low speed are

$$x_d(t)=1+0.05\sin(t),\ y_d(t)=1+0.05\cos(t); \tag{13}$$

and the high-speed position are

$$x_d(t)=1+0.05\sin(4t),\ y_d(t)=1+0.05\cos(4t); \tag{14}$$

By solving the inverse kinematics, we obtain the desired joint angle trajectory in fast motion. The responses of the system, where all nonlinearities are exactly known, are shown in fig. 3 without disturbances and friction. The simulation was performed in low speed and high speed. After a transient due to error in initial conditions, the position errors tend asymptotically to zero. To show the effect of unstructured uncertainties, we dropped a term, $(m1+m2)gl_1c_1$ simulation results are shown in fig. 4 in low speed. Note that there is a steady-state error with system. Fig. 5 and 6 shows the effect of external disturbances and friction forces which is difficult to model and compensate. This is corrected by adding a neural network as follows.

Basically the ANN can be characterized by

Number of hidden neurons: 8
Hidden neuron activation functions: $\sigma(x)=1/(1+\exp(-x))$.
Output neuron activation functions: $\sigma(x)= x$;
Learning rate in the weight tuning law
Inputs to hidden neurons: p=x.
Simulation time: 20 s

## 5 Conclusion

Since the dynamic characteristics of on n rigid-link robot manipulator are highly nonlinear and coupling, and the complete dynamic model is difficult to obtain precisely, an ANN control system has been successfully designed in this study to control the joint position of the robot manipulator to achieve high-accuracy position control. In the ANN control system, all the system dynamic can be unknown and no strict constraints were required in the design process. All adaptive learning algorithms in the ANN control system were derived in the sense of Lyapunov stability analysis, so that system-tracking stability of the closed-loop system can be guaranteed whether the uncertainties occur or not. To verify the effectiveness of the proposed control scheme, the ANN control system was implemented to control a two-link robot manipulator from the simulation results, the joint-position tracking responses can be controlled to closely follow the reference trajectories under a wide range of operating conditions and the occurrence of uncertainties.

## References

1. Astrom, K.J., Hagglund, T.: Automatic tuning of simple regulators with specification on phase and amplitude margins, Automatica 20 (1984) 645.
2. Astrom, K.J., Hagglund, T.: PID Controller: Theory, Design, and Tuning, Research Triangle Park, NC, USA, (1995).
3. Astrom, K.J., Wittenmark, B.: Adaptive Control, Addition-Wesley, New York (1995).
4. Cha, I., Han, C.: The auto-tuning PID controller using the parameter estimation, IEEE=RSJ International Conference on Intelligent Robots and System (1999) 46.

5. Chen, P.C.Y., Mills, J.K., Vukovich, G.: Neural network learning and generalization for performance improvement of industrial robots, Canadian Conference on Electrical and Computer Engineering (1996) 566.
6. Clifton, C., Homaifar, A., Bikdash, M.: Design of generalized Sugeno controllers by approximating hybrid fuzzy PID controllers, IEEE International Conference on Fuzzy Systems (1996) 1906.
7. Gutierrez, L.B., Lewis, F.L., Lowe, and J.A.: Implementation of a neural network tracking controller for a single flexible link: comparison with PD and PID controller IEEE Trans. Ind. Electron 45 (1998) 307.
8. Hang, C.C., Astrom, K.J., Ho, W.K.: Refinements of the Ziegler-Nichols tuning formula, IEE Proc. Control Theory Appl. 138 (1991) 111.
9. Huang,S.J., Lee, J.S.: A stable self-organizing fuzzy controller for robotic motion control, IEEE Trans. Ind. Electron 47 (2000) 421.
10. Kim, Y.H., Lewis, F.L.: Optimal design of CMAC neural-network controller for robot manipulators, IEEE Trans. Systems Man Cybernet 30 (2000) 22.
11. Koivo, A.J.: Fundamentals for Control of Robotic Manipulators, Wiley New York (1989).
12. Kuc,T.Y., Han,W.G.: Adaptive PID learning of periodic robot motion, IEEE Conference on Decision and Control (1998) 186.
13. Lewis, F.L., Abdallah, C.T., Dawson,D.M.: Control of Robot Manipulators, Macmillan New York (1993).
14. Lewis, F.L., Yesildirek, A., Liu, K.: Multilayer neural-net robot controller with guaranteed tracking performance, IEEE Trans. Neural Networks 7 (1996) 388.
15. Li, Y., Ho, Y.K., Chua, C.S.: Model-based PID control of constrained robot in a dynamic environment with uncertainty, IEEE International Conference on Control Applications (2000) 74.
16. Lin, F.J., Hwang, W.J., Wai, R.J.: Ultrasonic motor servo drive with on-line trained neural network model-following controller, IEE Proc. Electric Power Appl. 145 (1998) 105.
17. Misir, D., Malki, H.A., Chen, G.: Graphical stability analysis for a fuzzy PID controlled robot arm model, IEEE International Conference on Fuzzy Systems (1998) 451.
18. Schilling, R.J.: Fundamentals of Robotic: Analysis and Control Prentice-Hall, N.J (1998).
19. Slotine, J.J.E., Li, W.: Applied Nonlinear Control, Prentice-Hall, Englewood N.J (1991).
20. Sun,D., Mills, J.K.: High-accuracy trajectory tracking of industrial robot manipulator using adaptive-learning schemes, American Control Conference (1999) 1935.
21. Taylor, D.: Composite control of direct-drive robots, Proceedings of the IEEE Conference on Decision and Control (1989) 1670.
22. Vemuri,A.T., Polycarpou, M.M.: Neural-network-based robust fault diagnosis in robotic systems, IEEE Trans. Neural Networks 8 (1997) 1410.
23. Yoo, B.K., Ham,W.C.: Adaptive control of robot manipulator using fuzzy compensator, IEEE Trans. Fuzzy Systems 8 (2000) 186.
24. Nil,M., Sönmez, M., Yüzgeç, U., Çakır, B.: Artificial Neural Network Based Control of 3 DOF Robot Arm, Int. Symposium on Intelligent Manufacturing System IMS (2004) 468-474
25. Sönmez, M., Yakut, M.: A Robotic System based on Artificial Neural Network, Mechatronics and Robotic (2004) 293-296

# Performance Evaluation of Recurrent RBF Network in Nearest Neighbor Classification

Mehmet Kerem Müezzinoğlu

Computational Intelligence Lab.
University of Louisville
Louisville, KY 40292, U.S.A.
mkerem@ieee.org

**Abstract.** Superposition of radial basis functions centered at given prototype patterns constitutes one of the most suitable energy forms for gradient systems that perform nearest neighbor classification with real-valued static prototypes. It has been shown in [1] that a continuous-time dynamical neural network model, employing a radial basis function and a sigmoid multi-layer perceptron subnetworks, is capable of maximizing such an energy form locally, thus performing almost perfectly nearest neighbor classification, when initiated by a distorted pattern. This paper reviews the proposed design procedure and presents the results of the intensive experimentation of the classifier on random prototypes.

## 1 Introduction

Nearest neighbor pattern classification is the problem of evaluating the association map

$$f(\mathbf{z}) = \arg\min_{\mathbf{y}\in M} d(\mathbf{z},\mathbf{y}), \tag{1}$$

defined on a pattern space $\mathbb{P}$, where $M \subseteq \mathbb{P}$ is a finite set of prototype patterns and $d(\cdot,\cdot)$ is a metric on $\mathbb{P}$. A system that calculates (1) for given $M$ and $\mathbf{z}$, called the Nearest Neighbor Classifier (NNC), is the focus of the design problem in this paper.

A straightforward way of evaluating exactly (1) for any given instance ($\mathbf{z} \in \mathbb{P}$, $M \subseteq \mathbb{P}$) requires computation of an array of $m = |M|$ distances from $\mathbf{z}$ to each $\mathbf{y} \in M$, then obtaining the index of the minimum element through comparisons, and finally extracting the pattern $\mathbf{y}^* \in M$ associated with the resulting index. This three-stage procedure can be implemented easily on digital computers and it necessitates $m$ evaluations of the metric $d(\cdot,\cdot)$ together with $m-1$ pairwise comparisons among the distance array. In addition to its high computational requirements, the method also requires the prototype patterns stored explicitly in the physical memory of the system to be extracted after the comparison phase. A particular case of the problem for $\mathbb{P} = \{0,1\}^n$, named the nearest codeword problem, has been reported in [2] as NP-complete. This result may be extended to an arbitrary $\mathbb{P}$.

In the development years of neural network theory, a single layer of $n$ discrete neurons within a feedback loop was facilitated to retrieve binary patterns from their distorted versions in [3]. This concept has triggered an enormous interest in analysis and design of finite-state recurrent neural networks, since these neurodynamical systems

F.A. Savacı (Ed.): TAINN 2005, LNAI 3949, pp. 58–69, 2006.

exploit the memory storage and recovery property, providing a motivation towards explaining the biological associative memory by collective operation of basic computational units. It is possibly for this reason that, artificial neural systems that demonstrate the pattern reconstruction property, even partially for some $\mathbf{z} \in \mathbb{P}$, have been accepted conceptually as associative memories in the related literature. However, there is still a significant gap from engineering viewpoint between NNC, i.e. the ideal associative memory, and the conventional neural associative systems, as detailed below. Based on this fact, we view the iterative map realized by a recurrent network model from its initial state vector to the steady state as an approximation of (1), setting formally the NNC as the objective in auto-associative memory design.

The way Hopfield Associative Memory (HAM) [3] operates has indeed three major advantages over the conventional implementation described above:

1. The computation of the associative map is left to the autonomous dynamics of the network, so no comparison is performed explicitly throughout the process.
2. The network structure is independent of the number $m$ of prototypes to be stored.[1]
3. Convergence to a fixed point is guaranteed in at most $n^2$ steps, when the network has symmetric weight parameters and is operated in asynchronous mode [4].

On the other hand, HAM performs so poor by evaluating (1) that it hardy qualifies as a binary NNC.

Most of the limitations of conventional HAM in approximating (1) apply in general for any recurrent network with a fixed structure independent of the cardinality of $M$. They can be explained by an energy function approach to the network dynamics: A *fixed network model* may be associated only to a *fixed form of energy function* defined over its state space, which is minimized locally along trajectories generated by the network dynamics. In particular, as proven in [4], the energy function associated to the HAM model has a quadratic form, and hence the network is able to recall a point only if it is designated as a local minimum to this energy form by a design method. In other words, a given set $M$ of $n$-dimensional prototype vectors cannot be stored altogether as fixed points of HAM, unless there exists a pair $(\mathbf{Q}, \mathbf{c})$ such that the quadratic $Q(\mathbf{x}) = \mathbf{x}^T\mathbf{Q}\mathbf{x} + \mathbf{c}^T\mathbf{x}$ has a local minimum at each element of $M$ [5]. Similar restrictions apply for energy forms associated with other recurrent models with a fixed structure. Therefore, no fixed network model is capable of handling all possible (unrestricted) prototype combinations $M \subseteq \mathbb{P}$ in general.

In [1], we have proposed a continuous-time gradient neural network model with adaptive structure that qualifies as an NNC. The dynamics of the model is defined in the bounded state space $[0, 1]^n$ such that it maximizes an associated scalar energy function, which is in the form of a sum of Gaussian functions. Since the maxima of such a form can be allocated at any point of the unit-hypercube $[0, 1]^n$ (not necessarily at the vertices), our approach relaxes the conventional restriction of binary prototypes in neural associative memory towards real-valued ones.

The proposed dynamic model employs a hybrid of two algebraic networks, namely a Radial Basis Function (RBF) network and a Multi-Layer Perceptron (MLP) network.

[1] This would have been a really valuable property from information theoretic point of view, if there would have existed a design method capable of mapping an arbitrary $M$ as the fixed point set of the network, which utilizes $n^2 + n$ parameters.

Each RBF node in the model inserts into the state space an open basin of attraction around its center, thus introducing a stable fixed point, whereas MLP is utilized merely as a multiplier.

Recurrent RBF networks have been applied successfully to such areas that require rapid and accurate data interpolation, such as adaptive control [6] and noise cancellation [7]. To our knowledge, the proposed approach constitutes the first application of the RBF network structure in a feedback loop working as a nearest neighbor classifier.

This paper presents the experimental validation of the RBF-based model. The outline of the paper is as follows. The design constraints, energy function-based design procedure, and network model of the originating work [1] are reviewed in the following section. In Section III, we present the results of computer experiments, which was conducted to assess the particular network parameters on the classification performance of the proposed model. The concluding remarks are given in Section IV.

## 2 Methodology

### 2.1 Design Problem and Constraints

The objective is to design a continuous-time autonomous neurodynamical system operating on the continuous state space $\mathbb{R}^n$ to evaluate (1) as a map from the initial state vector to the steady state response. This setting assumes initially $\mathbb{P} = \mathbb{R}^n$ with a metric induced by the Euclidean norm: $d(\mathbf{u}, \mathbf{v}) = \|\mathbf{u} - \mathbf{v}\|_2$.

We limit the search for the dynamical system within the gradient networks in order to utilize the one-to-one correspondence between the stable equilibria of a gradient network and the local extrema of its energy function. In fact, this allows to pose the design problem of the classifier solely in the function space: Assign each prototype to a local maximum of an energy form $E(\cdot)$ in a bijective way, rendering every $\mathbf{p} \in M$ a local maximum of $E(\cdot)$ without creating any extraneous one.

The design problem cast in this way reduces the feasible energy functions to the ones that assume multiple and easily-adjustable local maxima. The energy function must also allow the designer to tune the basins of attraction of such fixed points that they share the state-space of the system in the way the ideal classifier (1) quantizes $\mathbb{P}$. Finally, the gradient of $E(\cdot)$ must be well defined and assume a valid neural implementation. Fortunately, these design constraints are not contradicting as they are satisfied simultaneously by at least one particular class of functions as shown below.

### 2.2 Radial Basis Functions

Radial basis functions are extensively used to implement algebraic mappings that interpolate and/or generalize a finite set of samples [8,9]. The nonlinearities possess the localization property [10], which makes the network sensitive only to particular regions of the input space. The centers and widths of high-sensitivity regions are expressed explicitly as network parameters. Hence, the RBF network design problem can be cast directly in the input space by shaping the regions according to data distribution. Encoding the sample features as network-specific quantities, which is often tackled in MLP design, is not needed. We make use of this property of RBF networks to represent the

prototypes in the design of our neurodynamical NNC. However, instead of the regression capability of RBF networks, carefully studied in statistical learning literature, we are interested here in evaluating their input-output function as an energy form for dynamical NNCs.

An admissible RBF $\phi(\cdot) : \mathbb{R}^n \to \mathbb{R}^+$ possesses the following three attributes:

1. $\phi(\cdot)$ is continuous and bounded.
2. $\phi(\cdot)$ attains its unique maximum at a point $\mathbf{c} \in \mathbb{R}^n$, called center.
3. $\lim_{\|\mathbf{x}-\mathbf{c}\|\to\infty} \phi(\mathbf{x}) = 0$.

Note that, such a $\phi(\cdot)$ discriminates the points within a vicinity $\mathcal{D}$ of $\mathbf{c}$ from others by returning a higher value. The width of $\mathcal{D}$ is usually parameterized in the expression of $\phi(\cdot)$ as in the case of the most popular RBF, the Gauss function

$$\phi(\mathbf{x}) = \exp\left(-\frac{\|\mathbf{x}-\mathbf{c}\|_2^2}{\gamma}\right), \tag{2}$$

where $\gamma \geq 0$ is the width parameter[2]. Hereinafter we will assume this form for $\phi(\cdot)$ and maintain the design based on Gaussian RBF, though the arguments in the sequel could be derived in a similar way for any other admissible RBF as well.

A single RBF $\phi(\cdot)$ centered at $\mathbf{c}$ is viewed here as a continuous function to be maximized by the gradient system

$$\begin{aligned}\dot{\mathbf{x}}(t) &= \nabla_{\mathbf{x}}\phi(\mathbf{x})|_{\mathbf{x}=\mathbf{x}(t)} \\ &= -\frac{2}{\gamma}(\mathbf{x}(t)-\mathbf{c})\,\phi(\mathbf{x}(t))\end{aligned} \tag{3}$$

along all trajectories. Such a convergence to the unique fixed point $\mathbf{c}$ from all initial conditions $\mathbf{x}(0) \in \mathbb{R}^n$ is interpreted as a trivial instance of nearest neighbor classification for $M = \{\mathbf{c}\}$. Among infinitely many state equation forms on the same state space, all bearing a unique stable fixed point at $\mathbf{c}$,[3] we specifically adopt (3) as the basic building block of our design, since this setting allows proper concurrence of systems with a single fixed point to form a single dynamics with multiple fixed points.

### 2.3 Dynamical Classifier

Given $m$ prototypes $M = \{\mathbf{p}_i\}_{i=1}^m \subset \mathbb{R}^n$, let us set first $m$ distinct energy functions $\{\phi_i(\cdot)\}_{i=1}^m$ each centered at $\mathbf{c}_i = \mathbf{p}_i$ with the width parameter $\gamma_i > 0$. These energy forms yield $m$ separate dynamical classifiers in the form of (3). Note that each classifier is perfect in the sense that it evaluates (1) exactly for the (unique) prototype $\mathbf{c}_i = \mathbf{p}_i$. In order to obtain a single state equation with stable fixed points at the prototypes, we propose summing up merely the right-hand sides of the individual equations:

$$\dot{\mathbf{x}}(t) = -\sum_{i=1}^{m} \frac{2}{\gamma_i}(\mathbf{x}(t)-\mathbf{c}_i)\,\phi_i(\mathbf{x}(t)). \tag{4}$$

[2] The width of a Gauss curve has been traditionally denoted by the standard deviation, equal to $\sqrt{\gamma}$ in (2). For conciseness of the notation, we prefer parameterizing it merely by $\gamma$.

[3] The simplest form of such systems is $\dot{\mathbf{x}}(t) = -\mathbf{x}(t) + \mathbf{c}$, which maximizes the energy function $E(\mathbf{x}) = -\|\mathbf{x}-\mathbf{c}\|_2^2/2$.

The resulting state equation (4) defines a gradient system which now maximizes the continuous and bounded energy function

$$E(\mathbf{x}) = \sum_{i=1}^{m} \phi_i(\mathbf{x}). \tag{5}$$

This relation guarantees that the continuous-time system (4) is globally convergent, due to Lyapunov's indirect method [11], and its fixed points are the extremum points of (5).

In light of the discussion of Section 2.1, the qualitative performance of (4) as an NNC is dependent upon the degree to which (5) fits the design constraints.

In [1], we have accounted for the impact of the width parameters $\gamma_1, \ldots, \gamma_\ell$, the only adjustable parameters of the proposed model, on the qualitative properties of the dynamical system (4). It is also shown in [1] that the width parameters must be chosen all equal ($\gamma_1 = \cdots = \gamma_m = \gamma$) and sufficiently small due to three particular reasons: (1) The basins of attractions of the prototypes in (4) share the state space optimally only when $\gamma_1 = \cdots = \gamma_m$; (2) The dynamical system defined by (4) has exactly $m$ stable fixed points only if $\max_i \gamma_i$ is sufficiently small; (3) The distance between the stable fixed points of the state equation (4) and the original prototypes $\{\mathbf{c}_1, \ldots, \mathbf{c}_m\}$ diminish exponentially as $\gamma_i \to 0$ for all $i \in \{1, \ldots, m\}$.

### 2.4 Transform to Bounded State Space

The nearest neighbor classification problem considered initially on $\mathbb{P} = \mathbb{R}^n$ is equivalent to the one posed within the unit hypercube $\mathbb{P} = [0,1]^n$ through a bijective transform, which maps the prototypes and the distorted pattern to $[0,1]^n$ in a distance-preserving way. If the given prototypes $M = \{\mathbf{p}_i\}_{i=1}^m$ are not contained already within the unit-hypercube, then a linear contraction that scales them by the maximum entry $\max_{i,j}(\mathbf{p}_i)_j$ among all prototypes transforms the problem to $[0,1]^n$. The contracted prototypes reduce the state space of (4) to $[0,1]^n$ due to the following corollary:

**Corollary 1.** *If $\mathbf{c}_1, \ldots, \mathbf{c}_m \in [0,1]^n$, then, for all initial conditions $\mathbf{x}(0) \in [0,1]^n$, the solution of (4) is bounded by the unit hypercube.*

*Proof.* Let $\mathbf{x}$ be an arbitrary point on a face of the unit hypercube, i.e. $x_j = 0$ or $x_j = 1$ for some $j \in \{1, \ldots, n\}$. For $x_j = 0$, the right-hand side of the $j$-th differential equation in (4) is nonnegative due to the assumption that all prototypes lie within the hypercube. Similarly, $\dot{x}_j \leq 0$ for $x_j = 1$. Hence, the vector field defined by (4) never points toward the exterior of $[0,1]^n$ along the faces, constraining all solutions starting within the hypercube into $[0,1]^n$.

The unit-hypercube as a state-space has nice features, among them it offers a clearer analysis and a canonical implementation. Therefore, without loss of generality, we focus on the transformed classification problem by assuming that $M \in [0,1]^n$.

### 2.5 Gradient Network Model

In this subsection, we outline the neural network model of the proposed system (4). Its key component is the RBF network model, shown in Figure 1.

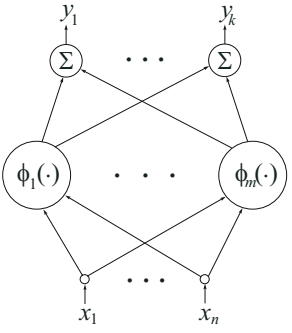



**Fig. 1.** RBF network model with $n$ inputs, $m$ RBF nodes, and $k$ outputs

We begin by rearranging the state representation (4) as

$$\dot{\mathbf{x}}(t) = -\frac{2}{\gamma}\left[\mathbf{x}(t)\sum_{i=1}^{m}\exp\left(-\frac{\|\mathbf{x}(t)-\mathbf{c}_i\|_2^2}{\gamma}\right) - \sum_{i=1}^{m}\mathbf{c}_i\exp\left(-\frac{\|\mathbf{x}(t)-\mathbf{c}_i\|_2^2}{\gamma}\right)\right], \tag{6}$$

and focus on realizing the right-hand side by an algebraic neural network. In the direct realization scheme adopted here, the network has the feedforward path and the feedback provided via $n$ integrators.

Note that the form (6) requires one straight and $n$ weighted summations of the RBFs, which could be achieved by an RBF network with $n+1$ output nodes. This network employs $m$ RBF nodes, to compute the vector

$$\mathbf{r} = \left[\exp\left(-\frac{\|\mathbf{x}-\mathbf{c}_1\|_2^2}{\gamma}\right)\cdots\exp\left(-\frac{\|\mathbf{x}-\mathbf{c}_m\|_2^2}{\gamma}\right)\right]^T,$$

where $\mathbf{x}$ is the input vector. The output node computing the straight sum is a single summing unit labelled by $s$, whereas the remaining $n$ output nodes perform the matrix multiplication $\mathbf{Cr}$, where

$$\mathbf{C} = \left[\mathbf{c}_1\cdots\mathbf{c}_m\right].$$

To realize the first term within the parenthesis in (6), $n$ additional blocks are required in order to multiply the state vector $\mathbf{x}(t)$ by the output of node $s$. Finally, a single linear layer subtracts the second term from the first one and scales the outcome by the constant $-2/\gamma$. The block diagram of the gradient model is shown in Figure 2.

Multiplication of variables is in general an expensive task in artificial neural architectures within the conventional and widely-accepted McCullogh-Pitts model [12], which assumes a neuron to be a computational unit capable of summing its weighted inputs, and then passing the result through a simple nonlinearity. It is not possible in general to fit the fundamental arithmetic operation of multiplication within this scheme. Fortunately, the bounded state-space of the classifier, as shown by Corollary 1, together with the upper bound $m$ on the sum of RBFs, constitute a particular case that allows a simple and efficient neural network realization of the multiplier blocks in Figure 2.

To achieve multiplication of two arbitrary scalar variables $a, b \in [0,1]$, we consider the two-layer perceptron network shown in Figure 3, which utilizes four sigmoidal

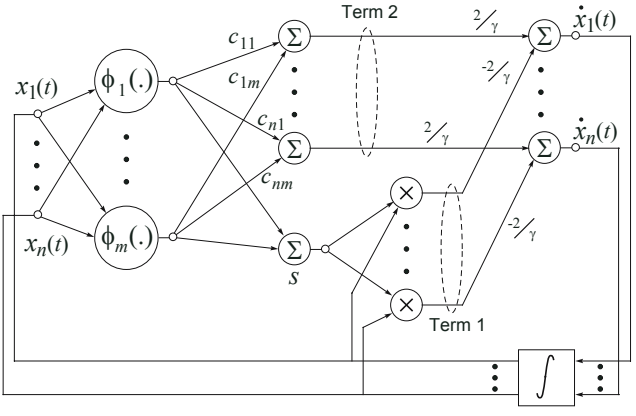



**Fig. 2.** Block diagram of the neurodynamical NNC proposed in [1]. The variables labelled by Term 1 and Term 2 denote the first and the second term in brackets of (6), respectively.

nodes and a linear one. The activation functions of the first layer nodes are all $\tanh(\cdot)$. To determine the parameters of this subnetwork to perform multiplication, we generated the training set

$$\mathcal{S} = \left\{ \left( \begin{bmatrix} x_1 & x_2 \end{bmatrix}^T, x_1 \cdot x_2 \right) : x_1 = 10^{-2}k, x_2 = 10^{-2}l, k, l = 0, 1, \ldots, 100 \right\},$$

then performed the classical backpropagation training algorithm with adaptive step-size and random initial conditions. The training results after 2000 epochs are shown in Table 1 (to be evaluated for $m = 1$ for now), yielding a mean-square error of less than $10^{-8}$ on $\mathcal{S}$. Since $\mathcal{S}$ represents the input space $[0, 1]^2$ effectively enough, having obtained such a small training error, we conclude that the proposed MLP network realizes successfully $a \times b$.

Due to Corollary 1, each state variable is constrained within $[0, 1]$ along the classification, so can be applied directly as an input to the resulting MLP network to be multiplied with the other operand. However, the sum of $m$ Gaussians (the node labelled by $s$ in Figure 2) is in general not within this bound, but definitely in $[0, m]$. Thus, it can be applied as the second input to the MLP subnetwork only after being scaled by $1/m$. The output of the multiplier should then be scaled by $m$ in order to realize the first term in the parenthesis. These two modifications on the second operand can be achieved by scaling the input layer weights $v_{i2}$, $i = 1, \ldots, 4$ and the output layer parameters $w_1, \ldots, w_4, b$ by $1/m$ and $m$, respectively, as given in Table 1. With these parameters, the resulting MLP could replace the multiplier blocks in Figure 2.

As a result, the overall model proposed to realize (4) employs a hybrid of one RBF and $n$ MLP networks on the feedforward path to implement the right-hand side of (4), plus $n$ integrator blocks on the feedback path to provide dynamics. The number of RBF nodes in the first layer equals the cardinality of $M$. This provides the key feature of the model, namely a structure adaptive to the given prototypes, as discussed in the introduction. Also note that the centers and the columns of the weight matrix $\mathbf{C}$ are set

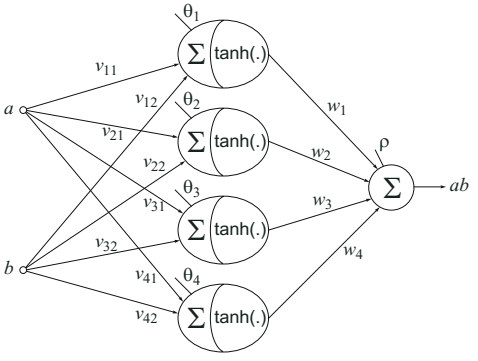



**Fig. 3.** MLP network model to multiply two bounded variables $a$ and $b$

**Table 1.** Parameters of the MLP network of Figure 2 to realize $a \times b$ for $a \in [0, 1]$ and $b \in [0, m]$

| | | | |
|---|---|---|---|
| $v_{11}$ | $0.02418$ | $\theta_2$ | $6.77450$ |
| $v_{21}$ | $16.34820$ | $\theta_3$ | $-0.78514$ |
| $v_{31}$ | $0.12108$ | $\theta_4$ | $-0.68511$ |
| $v_{41}$ | $0.09868$ | $w_1$ | $82.46650m$ |
| $v_{12}$ | $-0.10386/m$ | $w_2$ | $0.00004m$ |
| $v_{22}$ | $-58.58570/m$ | $w_3$ | $59.61670m$ |
| $v_{32}$ | $0.12528/m$ | $w_4$ | $-84.57270m$ |
| $v_{42}$ | $-0.02288/m$ | $\rho$ | $-60.5247m$ |
| $\theta_1$ | $0.68984$ | | |

directly to the given prototypes. This reduces the burden of adding a new prototype to the existing fixed points (i.e. memorizing) to incorporating merely a new RBF node and augmenting $\mathbf{C}$ by the new prototype, without requiring any computations. Similarly, removing (or forgetting) an existing prototype can be directly achieved by removing the associated RBF node and its connections. The only tunable parameter of the $n$ identical MLP subnetworks is $m$, the number of prototypes. Then the sole scalar parameter to be adjusted by the designer is $\gamma$ to improve the classification performance, as explained in [1] and summarized above.

We finally note that the proposed neurodynamic model does compute the distances from the current state vector to the prototypes,[4] but still performs no comparisons to evaluate (1), differing itself from the straightforward NNC.

## 3 Experimental Results

In this section, we present the experimental results related to classification performance of the proposed model in terms of randomly-generated classification instances.

[4] Such explicit distance calculations are avoided really in conventional neural associative memory models [13].

To simulate the continuous-time system (6) on a digital computer, we used the MATLAB® ODE Toolbox function ode23s [14]. This numerical solver implements a modified version of Rosenbrock method, a generalization of Runge-Kutta discretization, and is specialized in ordinary stiff differential equations, such as (6).

In order to validate experimentally the arguments of the preceding section on the width parameter $\gamma$, we evaluated the proposed NNC on random instances of the nearest neighbor classification problem. We denote an instance by the a pair $(\mathbf{z}, M)$, where $\mathbf{z}$ is an element (distorted pattern) and $M$ is a finite subset (prototype set) of the pattern space $\mathbb{P}$. In the experiments, we assumed $\mathbb{P} = [0,1]^n$, as was done throughout the design, and drew $1000$ instances independently from uniform distribution for each $n \in \{5, 10, 20\}$ and $m \in \{3, 10, 50\}$. To complete the simulations in reasonable durations, the right-hand side of (6) was scaled by $10^6$ in each trial.

Table 2 reports the classification accuracy of our neurodynamic NNC with respect to three different choices of $\gamma$, namely $0.8\gamma_{\text{sup}}$, $0.5\gamma_{\text{sup}}$, and, $0.1\gamma_{\text{sup}}$, where $\gamma_{\text{sup}} = 20$ in all experiments. Each entry on the columns labelled $C\%$ is the success rate of the classifier for the corresponding $n$, $m$, and, $\gamma$, in percents, i.e. number of instances classified correctly divided by the number of all instances $1000$. By the correct classification is meant such outcome of an experiment that, for the initial conditions set as the distorted pattern $\mathbf{z}$, the steady-state response of dynamical NNC is a fixed point, which is *closest* to the prototype $f(\mathbf{z})$ obtained by evaluating (1) exactly. In other words, we accept here a steady-state solution as a correct outcome if $\mathbf{x}(\infty)$ falls into the correct class by satisfying $f(\mathbf{x}(\infty)) = f(\mathbf{x}(0))$, even though $\mathbf{x}(\infty)$ itself is different than the exact outcome $f(\mathbf{x}(0))$.[5] This neglected difference is accounted in separate columns labelled by $\epsilon$ in the table. The quantity $\epsilon$ is the average gap between the exact classification result and the one obtained by our NNC for the corresponding $\gamma$, so is indeed a measure of the distortion in the energy representation (5). In particular,

$$\epsilon(n,m) = \frac{1}{1000} \sum_{i=1}^{1000} \left\| f(\mathbf{x}^i(0)) - \mathbf{x}^i(\infty) \right\|_2$$

where $\mathbf{x}^i(0)$ is the initial condition (which is set as the distorted pattern $\mathbf{z}$) and $\mathbf{x}^i(\infty)$ is the steady-state response of the system at $i$-th instance of the test group generated for $n, m$.

Table 2 shows that the classifications performed on the same group of instances turns out to be more accurate as $\gamma$ decreases in both factors represented by $C\%$ and $\epsilon$. It can also be seen from these results that the improvement in performance due to small $\gamma$ is indeed exponential, i.e., for large $\gamma$, the a small decrement in $\gamma$ causes a much greater improvement in $C\%$ and $\epsilon$ than in the case of a small $\gamma$. For the particular range of $n, m$ considered here, when $\gamma$ is set to $10$, the gaps between the actual fixed points and desired ones reduces significantly, making the outcome satisfactory.

The effect of $\gamma$ could be observed also on a single column of Table 2. For large $n$ and small $m$, the expected distances between the prototypes are larger than the opposite

[5] This distinguishes the concept of correct classification from exact classification, where the difference between the two outcomes is zero. In fact, exact classification can never be achieved by the proposed NNC.

**Table 2.** Correct classification rate $C\%$ and distortion $\epsilon$ on retrieved prototypes

| $n$ | $m$ | $\gamma = 0.8\gamma_{\sup}$ | | $\gamma = 0.5\gamma_{\sup}$ | | $\gamma = 0.1\gamma_{\sup}$ | |
|---|---|---|---|---|---|---|---|
| | | $C\%$ | $\epsilon$ | $C\%$ | $\epsilon$ | $C\%$ | $\epsilon$ |
| 5 | 3 | 74.6 | 0.14 | 98.3 | $0.3 \times 10^{-7}$ | 100 | $0.8 \times 10^{-10}$ |
| | 10 | 91.5 | 0.03 | 98.9 | $0.5 \times 10^{-8}$ | 100 | $1.2 \times 10^{-11}$ |
| | 50 | 98.8 | 0.01 | 99.8 | $1.2 \times 10^{-10}$ | 100 | $3.8 \times 10^{-16}$ |
| 10 | 3 | 58.0 | 0.25 | 97.5 | $2.3 \times 10^{-8}$ | 100 | $3.4 \times 10^{-10}$ |
| | 10 | 65.9 | 0.10 | 96.4 | $1.6 \times 10^{-9}$ | 100 | $1.7 \times 10^{-11}$ |
| | 50 | 85.5 | 0.03 | 97.3 | $4.2 \times 10^{-11}$ | 100 | $2.3 \times 10^{-15}$ |
| 20 | 3 | 52.2 | 0.42 | 93.4 | $2.0 \times 10^{-10}$ | 100 | $4.9 \times 10^{-10}$ |
| | 10 | 57.2 | 0.15 | 92.5 | $8.1 \times 10^{-11}$ | 100 | $6.7 \times 10^{-12}$ |
| | 50 | 78.4 | 0.08 | 92.7 | $6.1 \times 10^{-13}$ | 100 | $4.3 \times 10^{-16}$ |

case, making $\gamma_{\sup}$ large. Therefore, for fixed $\gamma$ and $n$, the classifier is more likely to produce accurate results, when dealing with larger prototype sets.

Note that our definition of correct classification above relates $C\%$ to $\epsilon$ so that the effects of $\gamma$ on these two performance criteria are in the same direction. On the other hand, the rate $C\%$ depends also on another phenomenon not represented in Table 2, namely the corruption of basins of attraction due to large $\gamma$. To visualize this corruption and how it is affected by $\gamma$, we generated two random classification instances in $[0, 1]^2$, one with 3 prototypes and the other with 5 prototypes, and obtained two dynamical NNCs to resolve these instances. The basins of attraction of the fixed points of each classifier are illustrated for two different $\gamma$ values in Figure 4.

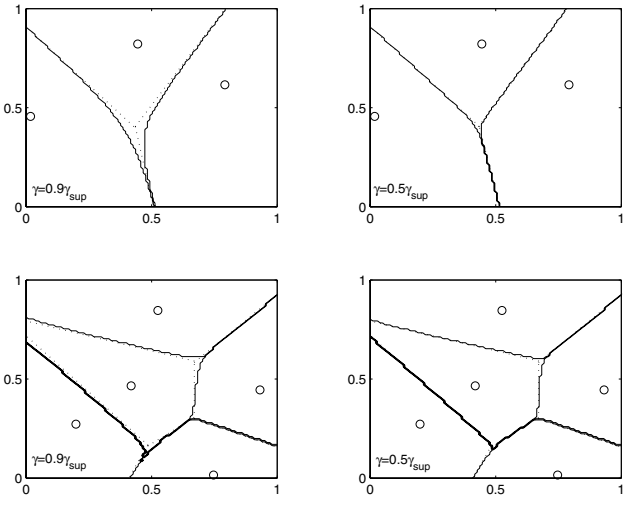



**Fig. 4.** The basins of attraction of stable fixed points obtained for two randomly-generated prototype sets for $m = 3$ (above) and $m = 5$ (below). The plots on the left show the partitioning for $\gamma = 0.8\gamma_{\sup}$ and the ones on the right for $\gamma = 0.5\gamma_{\sup}$. The dashed lines on each plot shows ideal partitioning.

Observe from Figure 4 that the widths of the RBFs impacts also the partitioning of the pattern space, even after all are set equal to $\gamma$. Choosing a small $\gamma$ in the design is advantageous from this point of view, too. By comparing the amounts of corruptions of the partitionings on two instances shown on the first column of Figure 4 (for $\gamma = 0.5\gamma_{\text{sup}}$), it can be concluded that the negative effect of nonzero $\gamma$ becomes more visible when the number of prototypes is fewer. Therefore, a large prototype set would more likely be handled better by the dynamical NNC from this aspect, too.

The cost of choosing a small $\gamma$ reveals itself on the processing time of the dynamical NNC. To process a test instance in constructing Table 2, it took the dynamical NNC $6\mu s$ for $\gamma = 0.8\gamma_{\text{sup}}$, $33\mu s$ for $\gamma = 0.5\gamma_{\text{sup}}$, and, $4.75s$ for $\gamma = 0.1\gamma_{\text{sup}}$ in average to converge to (i.e. to enter the $2\%$ band of) the steady-state response. These processing times depends neither on the dimension $n$ nor the number $m$ of fixed points of the dynamical system, but solely on the scalar parameter $\gamma$.

## 4 Conclusions

We have presented a novel continuous-time neurodynamical model that performs nearest neighbor classification on the continuous pattern space $[0, 1]^n$. The design procedure proposed for the model imposes no restriction on prototypes due to the adaptive structure of RBF network to the cardinality of $M$. The prototypes are represented explicitly as network parameters (weights and thresholds), so the given information needs not be encoded in the system. Moreover, this representation ensures localized parameters in the dynamical model associated to each fixed point so that a prototype can be added, removed, or modified independently, i.e. without affecting the parameters associated to rest of the fixed points. Although the network calculates all distances from the current state vector to prototypes, the convergence to the nearest one is guaranteed for each initial state vector without comparisons along the autonomous process. The only network parameter influencing the classification accuracy of the proposed NNC is the width parameter $\gamma$ of the Gaussian RBFs used in the design. The results show that the model implements the association map (1) almost perfectly, so it qualifies as a neurodynamic NNC.

## Acknowledgement

The author would like to thank Prof. Cüneyt Güzeliş for his suggestions that triggered the original idea of this work. This discussion presents partially the work [1] submitted to an archival journal.

## References

1. M. K. Muezzinoglu and J. M. Zurada, "RBF-based neurodynamic nearest neighbor classification in real pattern space", *submitted in 2005*.
2. M. R. Garey and D. S. Johnson, *Computers and Intractability.* New York: W.H. Freeman, 1979.

3. J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities", *Proc. National Acad. Sci.*, vol. 79, pp. 2554-2558, 1982.
4. J. Bruck and J. W. Goodman, "A generalized convergence theorem for neural networks", *IEEE Trans. Information Theory*, vol. 34, pp. 1089-1092, 1988.
5. M. K. Müezzinoğlu, C. Güzeliş, and J. M. Zurada, "An energy function-based design method for discrete Hopfield associative memory with attractive fixed points", *IEEE Trans. Neural Networks*, vol. 16, pp. 370-378, 2005.
6. R. M. Sanner and J. J. M. Slotine, "Gaussian networks for direct adaptive control", *IEEE Trans. Neural Networks*, vol. 3, pp. 837-863, 1992.
7. S. A. Billings and C. F. Fung, "Recurrent radial basis function networks for adaptive noise cancellation", *Neural Networks*, vol. 8, pp. 273-290, 1995.
8. T. Poggio and F. Girosi, "Networks for approximation and learning", *Proceedings of the IEEE*, vol. 78, pp. 1481-1497, 1990.
9. J. Park and I. W. Sandberg, "Universal approximation using radial-basis-function networks", *Neural Computation*, vol. 3, pp. 246-257, 1991.
10. N. B. Karayiannis, "Reformulated radial basis neural networks trained by gradient descent", *IEEE Trans. Neural Networks*, vol. 10, pp. 657-671, 1999.
11. H. K. Khalil, *Nonlinear Systems.* 3rd Edition, Prentice-Hall, 2001.
12. J. M. Zurada, *Introduction to Artificial Neural Systems.* St. Paul, MN: West, 1992.
13. A. N. Michel and D. Liu, *Qualitative Analysis and Sythesis of Recurrent Neural Networks.* New York: Marcel Dekker, 2002.
14. L. F. Shampine and M. W. Reichelt, "The MATLAB ODE Suite", *SIAM Journal on Scientific Computing*, vol. 18, pp 1-22, 1997.

# Tracking Aircrafts by Using Impulse Exclusive Filter with RBF Neural Networks

Pınar Çivicioğlu

Erciyes University, Civil Aviation School, Department of Aircraft Electrics and Electronics, 38039, Kayseri, Turkey
civici@erciyes.edu.tr

**Abstract.** Target Tracking based on Artificial Neural Networks has become a very important research field in *Dynamic Signal Processing*. In this paper, a new Target Tracking filter, entitled RBF neural network based Target Tracking Filter, RBF-TT, has been proposed. The tracking performance of the proposed filter, RBF-TT, has also been compared with the classical *Kalman Filter* based Target Tracking algorithm. Predictions during experiments have been made for the civil aircraft positions, one step ahead in real time. Extensive simulations revealed that the proposed filter supplies superior tracking performances to the Kalman Filter based comparison filter.

## 1 Introduction

One of the most important requirements for surveillance systems, e.g. *Civil Aviation Air Traffic Control*, is Target Tracking (TT), which employs one or more sensors, together with computer subsystems, to interpret the environment [1,2,3,4,5,6,7,8,9,10,11]. Typical sensor systems, such as *Sonar*, *Tracking-Radar* and *Infrared*, report measurements from diverse sources: targets of interest, background noise sources such as clutter, or internal error sources such as *thermal noise* and *impulsive noise*. The objective of TT is based on the collection of sensor data from a field of view containing one or more potential targets of interest and then to partition the sensor data into sets of observations, or tracks, that are produced by the same sources. After the tracks are formed and confirmed, the number of targets can be estimated and quantities, such as target velocity, future predictions, and target classification characteristics, can be computed for each of the track [1,2,3].

There are three main areas in which neural network technology is applied to tracking, data association, and related problems [1,4,5]. These applications are the solutions of assignment-type problems via the Hopfield neural network approach [6], the use of neurons to represent bins in target state space for the TT application discussed in the [1,2], and tracking filter design and data association in TT [1,2,6].

The purpose of TT is to estimate various aspects of motion such as 3D spatial location of the target based on information obtained by Tracking-Radar sensors

F.A. Savacı (Ed.): TAINN 2005, LNAI 3949, pp. 70–77, 2006.

in *Civil Aviation* [1,2]. Many discussions are available on TT using active sensor systems [1,2,3,4,5,7,8,9,10,11,12].

On the other hand, due to recent advances in radar technology [13], it is now possible to use passive sensors such as infrared sensors and sonar systems for TT. When the passive sensors are used, only the information on the target angle seen from the sensor is obtained with a single sensor. A successful performance of TT is obtained when the optimal extraction of useful information about the state of the target is selected from the noisy observations [1,2,3,7,8,9,10,11,12]. A good model of the target certainly facilitates this extraction of information to a great extent. Hence, one can easily say that a good model is worth a thousand pieces of data.

Most of the TT algorithms are *model-based* because a good model-based tracking algorithm greatly outperforms any model-free tracking algorithm if the underlying model turns out to be a good one. *Target detection*, *tracking*, *classification* and *identification* (recognition) are closely interrelated areas of TT, with significant overlaps [7,8,9,10,11,12].

In this paper, a new TT filter based on impulse exclusive filter with RBF neural networks has been proposed. The remaining part of the paper is organized as follows. Section 2 briefly gives *Radar Technology and Uncertainty of Target Positioning*. Section 3 describes *Kalman Filter Based Target Tracking*. The presentation of the *Proposed Method*, *Experiments* and *Results and Conclusions* are given in Sections 4, 5 and 6, respectively.

## 2 Radar Technology and Uncertainty of Target Positioning

One important function of each radar surveillance system is to keep and improve target tracking maintenance performance [13]. It becomes a crucial and challenging problem especially in complicated situations of closely spaced, or crossing targets [1,2,3,4,5,7,8,9,10,11].

There are approximately 11,000 aircrafts on the *Civil Aircraft Registry* [14]. Therefore, TT has a vital importance in *Civil Aviation Safety*. The scientific researches on Avionics were concentrated on *Communications*, *Radar* and *TT*, which are main research areas of Ultra-Wideband technologies [12,13]. The radar sensor is probably the most used active sensor for TT applications [1,2,3,12,13]. The technical details of the Radar Systems have not been mentioned in this paper due to paper length limitations. An excellent tutorial of Radar Systems can be found in [13].

The radar sensor emits *microwave-energy* and then measures the returned energy. Hence, it is possible to calculate the distance to the target from the time delay and the speed of light. The sensor can be used in a passive mode, searching for targets emitting radar signals. This is often referred to as a radar warning receiver. Depending on the particular application and radar sensor, different features are measured by the sensor. The most common one is measuring angles (azimuth ($\theta$), and elevation) and range ($R$) or 3D spatial positions, $x$-$y$-$z$, of

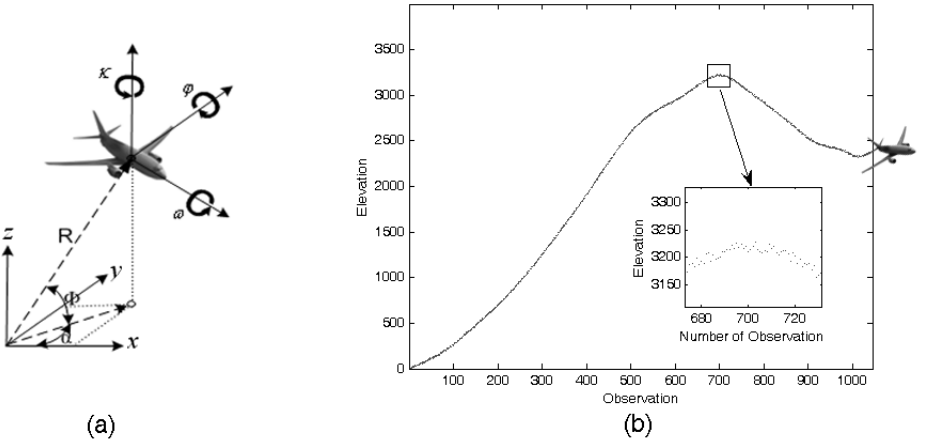



**Fig. 1.** (a) Target and Cartesian Coordinate System, (b) Observations

the target [12,13]. If a doppler-radar is used, the range rate is available. The range and resolution of the measurements are dependent on the used signal processing technique and also on the physical constraints, especially the antenna aperture [13]. Different signal processing techniques used in radar applications are constant false-alarm rate and *Synthetic Aperture Radar*.

Selection of coordinate systems for TT problem has been examined in various studies. In general, $\theta$ and $R$ have been used for radar measurements and $x$-$y$-$z$ spatial-cartesian coordinates have been used for TT purposes [12,13]. Some of the basic tracking models, such as Tracking-Radars, use cartesian coordinates in order to trace targets. In this paper a total of 1050 $x$-$y$-$z$ data of a Tracking-Radar system have been employed. The spatial relations of the Target (e.g., Aircraft) and cartesian coordinate system have been illustrated in Fig. 1(a). The tracking data, which have the standard deviation of $\sigma = 5.00$, have been illustrated with pseudo-scaling in Fig. 1(b).

## 3 Kalman Filter Based Target Tracking

Kalman filtering [12] is a relatively recent development in filtering, although it has roots going far back to Gauss [1,2,3,12]. Kalman filtering has been applied in many areas as diverse as *aerospace*, *marine navigation*, *nuclear power plant instrumentation*, *demographic modeling* and so on.

The Kalman filter is an *on-line*, recursive algorithm which tries to estimate the true state of a system where noisy observations or measurements are available. In Bayesian terms, it is wished to propagate the conditional probability density of the true state, given knowledge on previous measurements. The Kalman filter contains a linear model for the process and a linear model for the measurement. The former model describes how the current state of the tracker is changing, given the previous instance:

$$\begin{pmatrix} x_k \\ y_k \end{pmatrix} = \begin{pmatrix} 1\ 0 \\ 0\ 1 \end{pmatrix} \begin{pmatrix} x_{k-1} \\ y_{k-1} \end{pmatrix} + \begin{pmatrix} \tilde{x}_{k+1} \\ \tilde{y}_{k+1} \end{pmatrix} \tag{1}$$

or in a shorter notation:

$$p_k = Ap_{k-1} + \tilde{p}_{k-1} \tag{2}$$

with A the state transition matrix and $\tilde{p}$ the process noise vector.

The measurement model of the filter is straightforward as well:

$$\begin{pmatrix} u_k \\ v_k \end{pmatrix} = \begin{pmatrix} H_x\ 0 \\ 0\ \ H_y \end{pmatrix} \begin{pmatrix} x_k \\ y_k \end{pmatrix} + \begin{pmatrix} \tilde{u}_{k-1} \\ \tilde{v}_{k-1} \end{pmatrix} \tag{3}$$

or shorter:

$$m_k = Hp_k + \tilde{m}_{k-1} \tag{4}$$

with H the measurement matrix and $\tilde{m}$ the measurement noise vector. H describes how the measured data relate (linearly) to the state. We define,

$$Q = E\left[\tilde{p} * \tilde{p}^T\right] = \begin{pmatrix} Q_x\ 0 \\ 0\ \ Q_y \end{pmatrix} \tag{5}$$

and

$$\xi = E\left[\tilde{m} * \tilde{m}^T\right] = \begin{pmatrix} \xi_x\ 0 \\ 0\ \ \xi_y \end{pmatrix} \tag{6}$$

as the covariance matrices of the process and measurement noise respectively.

We start the initial state: $p_0 = Hm_0$ and define $P_0 = \begin{pmatrix} e\ 0 \\ 0\ e \end{pmatrix}$ with e and update the Kalman variables in a two-step predict-correct loop, until we run out of data.

Predict the next state: $p_k^- = Ap_{k-1}$. Predict the next error covariance: $P_k^- = AP_{k-1}A^T + Q$.

Compute Kalman gain: $K = P_k^- H^T \left(HP_k^- H^T + R\right)^{-1}$. Update estimated state with measurement: $p_k = p_k^- + K\left(m_k - Hp_k^-\right)$. Update the error covariance: $P_k = (I - K|H)\, P_k^-$.

## 4 Proposed Method

The capability of neural networks [4,5,6] for approximating arbitrary *input-output* mappings give a simple way to identify unknown dynamic functions in order to predict the needed output one step ahead or more. In a tracking system [15], measured radar signals are mostly mixed with additive white noise [1,2,3,16]. In order to filter out or minimize this measured noise on-line and to predict the aircraft position one step ahead, a simple back propagation algorithm has been used.

The proposed RBF-TT has three-input neurons, and three-output neurons. The inputs use the spatial positions, *x-y-z*, of the target for the times of (t-2,t-1,t), and the outputs represent the spatial positions of the target, *x-y-z*, at times

of (t-1,t,t+1).Therefore, RBF-TT is a sliding system [16,17,18] for tacking over the time domain. The flowing chart of the RBF-TT has been illustrated in Fig. 2.

When the size of the training set is not big enough, target tracking performance decreases due to the deviations from the target positions that the RBF network produces. RBF-TT uses a traning set which has only 3-elements with three parameters, *x-y-z*. Therefore, in order to improve the performance of the RBF-TT a smoothing operator, namely impulse exclusive filter [16,17,18], is required for the post-processing of the neural network outputs. The impulse exclusion algorithms have been previously studied in detail in [16,17,18]. The impulse exclusive filter, which has been employed in this paper, is based on the filter proposed in [18], but two of the properties have been used as diferent from [18]: The sliding window size is equal to 3x1 instead of 3x3 and the extreme value within the window is accepted as corruption. The mathematical details of the impulse exclusive filter can be found in [18].

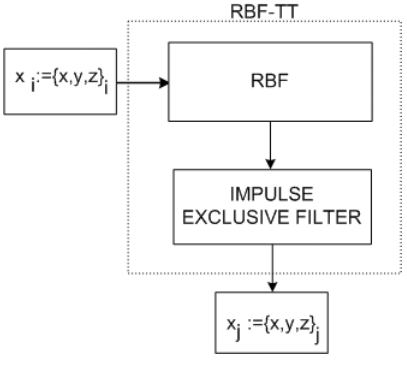



**Fig. 2.** Flowing Chart of Proposed Method, where the indices of $i$ and $j$ denote the observations at the times of (t-2,t-1,t) and (t-1,t,t+1), respectively

### 4.1 Radial Basis Function Artificial Neural Network

An alternative network architecture to the Multilayered Perceptrons is the Radial Basis Function Artificial Neural Network (RBFN), which can be represented with radially symmetric hidden neurons [6,19]. The topology of the RBFN is similar to the three-layered MLP but the characteristics of the hidden neurons are different. RBFNs require more neurons than standard feed-forward backpropagation networks, but they can be usually designed in a less time than it takes to train standard feed-forward networks. RBFNs can be employed to approximate functions and they work best when many training arrays are used. Basically, the structure of an RBFN involves three different layers. The input layer is made up of source neurons. The second layer is a hidden layer of high dimension serving a different purpose from that of an MLP. This layer consists of an array of neurons where each neuron contains a parameter vector called a center. The neurons calculate the *Euclidean Distance* between the center and the network

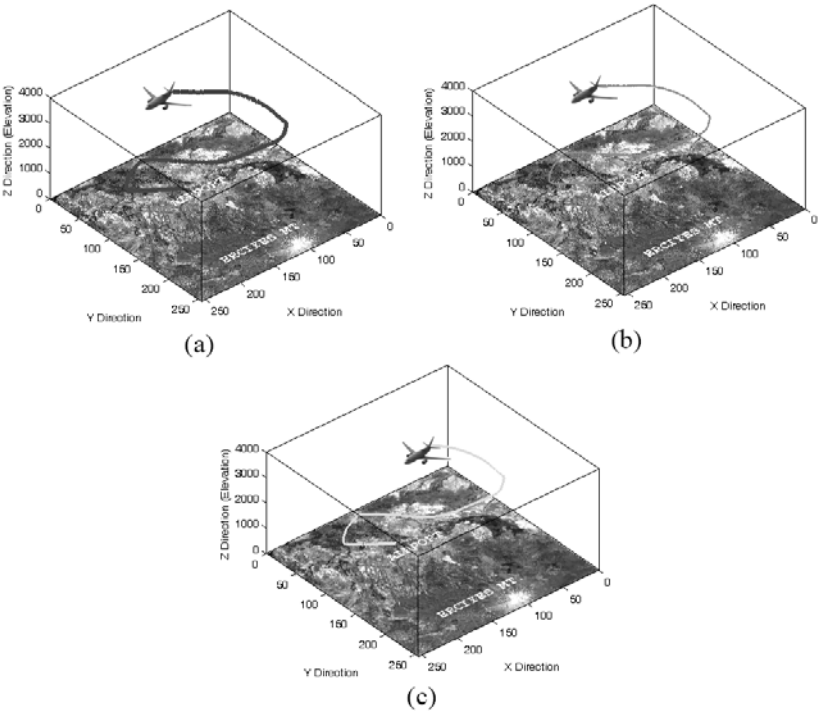



**Fig. 3.** (a) Radar Observations, (b) KALMAN Filter based TT, (c) RBF-TT

input vector, and then pass the result through a nonlinear function. The output layer, which supplies the response of the network, is a set of linear combiners. The transformation from input layer to the hidden layer is nonlinear but the transformation from the hidden layer to the output layer is linear. The output of a hidden layer is a function of the distance between the input vector and the stored center, which is calculated as,

$$O_s = \|B - Cs\| = \sqrt{\sum_{j=1}^{T} (B_j - C_{sj})^2} \tag{7}$$

The learning phase consists of using a clustering algorithm and a nearest neighbor heuristic in order to determine the $C_s$ cluster centers. The weights from the hidden layer to the output layer are determined by using linear regression or a gradient descent algorithm. Exact design of RBFN constitutes a radial basis network very quickly with zero error on the design vectors [18,19].

## 5 Experiments

In this paper, the flight data of a Tracking-Radar of a single Civil Aircraft, which has taken-off from Kayseri Erkilet Airport, have been used. Radar data

**Table 1.** Quality measures of target tracking algorithms

| **Tracking Filters** | **Comparison Measures** | |
|---|---|---|
| | **MSE** | **MAE** |
| KALMAN Filter based TT | 53.59 | 5.77 |
| RBF-TT | 4e-25 | 5e-13 |

association process is performed by using a nearest point search algorithm but data-gating [1,2,3] is not required for the single targets.

In multi-target tracking, data association and data-gating are important issues for improving tracking performance. However, tracking filter performance is the basis of TT in both multi-target TT and maneuvering TT. Therefore, TT performance for the single target of RBF-TT is examined in this paper. The Tracking-Radar data ($\theta$, $R$) were converted into three dimensional spatial positions, $x$-$y$-$z$.

The performances of the proposed RBF-TT and Kalman filter based TT have been quantitatively evaluated with the quality measures of *Mean Square Error* (MSE) and *Mean Absolute Error* (MAE) and tabulated in Table 1. The visual performances are illustrated in Fig. 3.

## 6 Results and Conclusion

In this paper, a novel RBF neural network based target tracking filter, RBF-TT, has been proposed for the prediction of one step ahead position of maneuvering targets. RBF-TT is effective and simple to implement. The tracking success of the proposed filter has been validated over the real data of civil aircraft positions in real time and the performance of the RBF-TT has been compared with the performance of the classical Kalman Filter based Target Tracking algorithm.

Extensive simulations reveal that the proposed filter, RBF-TT, supplies superior tracking performances to the Kalman Filter based comparison filter and tracking is achieved with high precision. The computational burden and computational complexity of the RBF-TT is also smaller than the Kalman filter based TT.

## References

1. Blackman, S.: Popoli, R., Design and Analysis of Modern Tracking Systems, Artech House, USA, (1999).
2. Bar-Shalom, Y., Blair, W., D.: Multitarget-Multisensor Tracking: Applications and Advances Volume III, Artech House, Inc., USA, (2000).
3. Bar-Shalom, Y., Li, X., R., Kırubarajan, T.: Estimation with Applications to Tracking and Navigation, Theory Algorithms and Software, John Wiley & Sons, Inc., USA, (2001).
4. Bierman, G.: Vector Neural Network Signal Integration for Radar Application, Signal and Data Processing of Small Targets, **2235**, (1994), 290–302.

5. Shams, S.: Neural Network Optimization for Multi-Target Multi-Sensor Passive Tracking, Proc. IEEE, **84**, (10), (1996), 1442–1457.
6. Haykın, S.: Neural networks, Macmillan, New York, (1994).
7. Li, X. R., Jilkov, W. P.: Survey of Maneuvering Target Tracking-Part I: Dynamic Models, IEEE Transactions on Aerospace and Electronic Systems, **39**,(4), (2003), 1333-1364.
8. Li, N., Li, X. R.: Target Perceivability and its Applications, IEEE Transactions on Signal Processing, **49**, (11), (2001), 2588–2604.
9. Wang, X., Challa, S., Evans, R., Li, X. R.: Minimal Sub-Model-Set Algorithm for Maneuvering Target Tracking, IEEE Transactions on Aerospace and Electronic Systems, **39**, (4), (2003).
10. Chen, H., Kirubarajan, T., Bar-Shalom, Y., Pattipati, K. R.: An MDL Approach for Multiple Low Observable Track Initiation, IEEE Trans. Aerospace and Electronic Systems, **39**, (3), (2003), 862–882.
11. Tartakovsky, A. G., Li, X. R., Yaralov, G.: Sequential Detection of Targets in Multichannel Systems, IEEE Transactions on Information Theory, **49**, (2), (2003), 425–445.
12. Brookner, E.: Tracking and Kalman Made Easy, John Wily & Sons., (1998), 60-62.
13. Mahafza, B., R.: Radar Systems Analysis and Design Using Matlab, Chapman & Hall/CRC, USA, (2000).
14. Civil Aviation Safety Authority (CASA), Civil Aircraft Register, http://www.casa.gov.au/casadata/register/seven.htm
15. Li, N., Li, X. R.: Tracker Design based on Target Perceivability, IEEE Transactions on Aerospace and Electronic Systems, **37**, (1), (2001), 214–225.
16. Çivicioğlu, P., Alçı, M: Impulsive Noise Suppression from Highly Distorted Images with Triangular Interpolants. AEU International Journal of Electronics and Communications, **58** (5), (2004), 311–318.
17. Çivicioğlu, P., Alçı, M: Edge Detection of Highly Distorted Images Suffering from Impulsive Noise. AEU International Journal of Electronics and Communications, **58** (6), (2004), 413-419.
18. Çivicioğlu, P., Alçı, M, Beşdok, E.: Using an Exact Radial Basis Function Artificial Neural Network for Impulsive Noise Suppression from Highly Distorted Image Databases. Lecture Notes in Artificial Intelligence, **3261**, (2004), 383-391.
19. MathWorks, Neural Networks Toolbox, MATLAB v7.00, Function Reference, New York, The MathWorks, Inc., (2004).

# A Multilayer Feedforward Fuzzy Neural Network

Aydoğan Savran

Ege University, Department of Electrical and Electronics Engineering,
35100, İzmir, Turkey
aydogan.savran@ege.edu.tr

**Abstract.** This paper describes the architecture and learning procedure of a multilayer feedforward fuzzy neural network (FNN). The FNN is designed by replacing the sigmoid type activation function of the multilayer neural network (NN) with the fuzzy system (FS). The Levenberg-Marquardt (LM) optimization method with a trust region approach is adapted to train the FNN. Simulation results of a nonlinear system identification problem are given to show the validity of the approach.

## 1 Introduction

Multilayer feedforward neural network (NN), also referred as multilayer perceptron (MLP), is one of the most popular NN architecture. The MLP consist of weighted connections and neurons arranged in layers. Each neuron collects the values from all of its input connections and produces a single output passing through an activation function. The sigmoid type activation functions are usually used. One way to improve the NN performance is to modify the activation function. The modifications of the activation function are highly investigated in the literature [1],[2],[3],[4].

A different approach to define activation functions is to use the fuzzy logic methodology. A fuzzy system can transform a knowledge base into a nonlinear mapping. When a fuzzy system designed or trained, the IF-THEN rules which are easily interpreted can be obtained. Recently, Olivas et al. proposed to use a fuzzy based activation function to increase the interpretation and simplify the hardware implementation of the neurons [5]. Oysal et al. developed a dynamic fuzzy network consist of the fuzzy activation function to model dynamic systems [6].

In the present paper, we propose the use of the fuzzy system as the activation function of the MLP. We aim to combine the MLP computational power with the fuzzy system interpretability in the same scheme. Since the MLP and the fuzzy system we used are both feedforward and universal approximators, we can conclude that the resulting FNN also conveys these features. The FNN is trained with a similar procedure with the MLP. So the learning methods of the MLP can be modified for the proposed FNN. We adapted the Levenberg-Marquardt (LM) optimization method with a trust region approach which provides fast convergence to train the FNN [7].

F.A. Savacı (Ed.): TAINN 2005, LNAI 3949, pp. 78 – 83, 2006.

## 2 The FNN architecture

The architecture of the proposed FNN is depicted in Fig. 1. The FNN is a modified model of the MLP with the fuzzy system (FS) activation function. We chose the FS with product inference engine, singleton fuzzifier, center average defuzzifier, and Gaussian membership functions (MFs) are of the following form

$$f_j(x_j) = \frac{\sum_{r=1}^{R_j} \bar{y}_{jr} \exp(-\frac{1}{2}(\frac{x_j - \bar{x}_{jr}}{\sigma_{jr}})^2}{\sum_{r=1}^{R_j} \exp(-\frac{1}{2}(\frac{x_j - \bar{x}_{jr}}{\sigma_{jr}})^2} \tag{1}$$

where $x_j$ is the input of the FS, $R_j$ is the number of the rules in the fuzzy rule base, $\bar{y}_{jr}$, $\bar{x}_{jr}$, and $\sigma_{jr}$ are the free fuzzy sets parameters [8]. $\bar{x}_{jr}$ and $\sigma_{jr}$ are the center and the spread of the $r^{th}$ rule of the $j^{th}$ neuron, respectively. An output membership function which has a fixed spread of 1, can be characterized with only the center parameter ($\bar{y}_{jr}$).

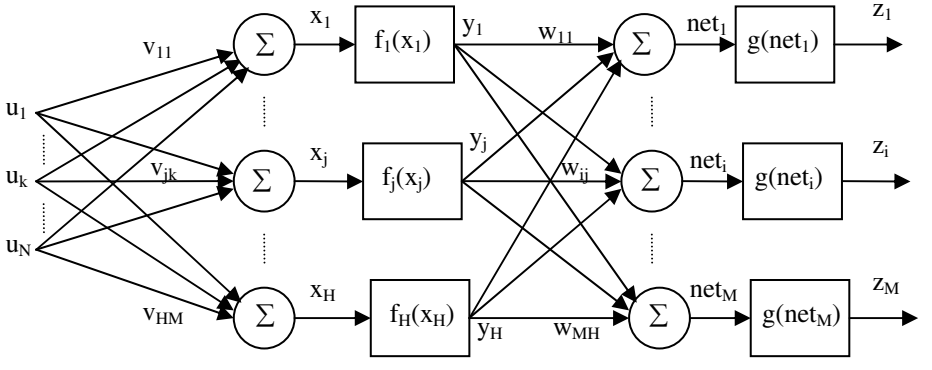



**Fig. 1.** The FNNN architecture

The output of the FNN computed as:

$$\begin{aligned} &x_j = \sum_k v_{jk} u_k, \quad y_j = f_j(x_j), \quad net_i = \sum_j w_{ij} f_j(x_j), \quad z_i = g_i(net_i) \\ &z_i = g_i(\sum_j w_{ij} f_j(\sum_k v_{jk} u_k)) \\ &k = 1,...,N \qquad j = 1,...,H \qquad i = 1,...,M \end{aligned} \tag{2}$$

where f(.) and g(.) are the hidden layer and the output layer activation functions respectively. In the hidden layer, the activation function is the fuzzy system. In the output layer, fuzzy, sigmoid or linear type activation functions can be used.

## 3 Learning

The output sum squared error can be defined as the cost function

$$E(p) = \frac{1}{2}\sum_i (z_i - d_i)^2 = \frac{1}{2}\sum_i e_i^2 \tag{3}$$

where z is the FNN output, d is the desired output, and p represents the free parameters of the FNN ($v_{jk}$, $w_{ij}$, $\bar{y}_{jr}$, $\bar{x}_{jr}$ and $\sigma_{jr}$). In order to improve the convergence rate and accuracy of the steepest descent algorithm, we adapted the Levenberg-Marquardt algorithm with a trust region approach to train the FNN. The LM method requires the Jacobian matrix J that contains the partial derivatives of the error terms with respect to free parameters of the FNN as

$$J = \begin{bmatrix} \frac{\partial e_1}{\partial w_{ij}} & \frac{\partial e_1}{\partial v_{jk}} & \frac{\partial e_1}{\partial \bar{x}_{jr}} & \frac{\partial e_1}{\partial \sigma_{jr}} & \frac{\partial e_1}{\partial \bar{y}_{jr}} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ \frac{\partial e_M}{\partial w_{ij}} & \frac{\partial e_M}{\partial v_{jk}} & \frac{\partial e_M}{\partial \bar{x}_{jr}} & \frac{\partial e_M}{\partial \sigma_{jr}} & \frac{\partial e_M}{\partial \bar{y}_{jr}} \end{bmatrix} \tag{4}$$

where ($\bar{y}_{jr}$, $\bar{x}_{jr}$ and $\sigma_{jr}$) represent all of the parameters of the fuzzy systems and ($v_{jk}$ and $w_{ij}$) represent the weights in the hidden and output layers. The sensitivity of the error with respect to weights of the output layer can be computed as

$$\begin{aligned} &\frac{\partial e_i}{\partial w_{ij}} = \frac{\partial e_i}{\partial z_i}\frac{\partial z_i}{\partial net_i}\frac{\partial net_i}{\partial w_{ij}} \\ &\text{where} \quad \frac{\partial e_i}{\partial z_i} = 1, \quad \frac{\partial z_i}{\partial net_i} = g_i'(net_i) = \delta_i, \quad \frac{\partial net_i}{\partial w_{ij}} = z_j \\ &\frac{\partial e_i}{\partial w_{ij}} = g_i'(net_i) y_j = \delta_i y_j \end{aligned} \tag{5}$$

Then, we can carry out the similar process at the hidden layer.

$$\begin{aligned} &\frac{\partial e_i}{\partial v_{ij}} = \frac{\partial e_i}{\partial z_i}\frac{\partial z_i}{\partial net_i}\frac{\partial net_i}{\partial y_j}\frac{\partial y_j}{\partial x_j}\frac{\partial x_j}{\partial v_{jk}} \\ &\text{where} \quad \frac{\partial net_i}{\partial z_j} = w_{ij}, \quad \frac{\partial y_j}{\partial x_j} = f_j'(x_j), \quad \frac{\partial x_j}{\partial v_{jk}} = u_k \\ &\frac{\partial e_i}{\partial v_{jk}} = (f_j'(x_j)\sum_i \delta_i w_{ij}) u_k \end{aligned} \tag{6}$$

where $f'(x)$ is the input-output sensitivity of the fuzzy system. It is computed as

$$f'_j(x_j)=\sum_{r=1}^{R_j}\frac{y_j-\bar{y}_{jr}}{\sum_{r=1}^{R_j}\exp(-\frac{1}{2}(\frac{x_j-\bar{x}_{jr}}{\sigma_{jr}})^2)}\exp(-\frac{1}{2}(\frac{x_j-\bar{x}_{jr}}{\sigma_{jr}})^2)(\frac{x_j-\bar{x}_{jr}}{\sigma_{jr}}) \tag{7}$$

where $x_j$ and $y_j$ are the fuzzy system input and output, respectively.

The parameters of the fuzzy systems have to be also adjusted. So, the sensitivity of the error with respect to the fuzzy systems parameters computed as

$$\begin{aligned}
\frac{\partial e_i}{\partial \bar{y}_{jr}}&=\frac{\partial e_i}{\partial z_i}\frac{\partial z_i}{\partial net_i}\frac{\partial net_i}{\partial y_j}\frac{\partial y_j}{\partial \bar{y}_{jr}}=\frac{\partial y_j}{\partial \bar{y}_{jr}}\sum_i \delta_i w_{ij}\\
\frac{\partial e_i}{\partial \bar{x}}&=\frac{\partial y_j}{\partial \bar{x}_{jr}}\sum_i \delta_i w_{ij}\\
\frac{\partial e_i}{\partial \sigma_{jr}}&=\frac{\partial y_j}{\partial \sigma_{jr}}\sum_i \delta_i w_{ij}
\end{aligned} \tag{8}$$

The derivatives ($\frac{\partial y_j}{\partial \bar{y}_{jr}}$, $\frac{\partial y_j}{\partial \bar{x}_{jr}}$ and, $\frac{\partial y_j}{\partial \sigma_{jr}}$) can be found in [8].

After computing the Jacobian matrix J the FNN parameters (p) which consist of the weigths($w_{jk}$, $v_{ij}$,) and fuzzy systems parameters ($\bar{y}_{jr}$, $\bar{x}_{jr}$, $\sigma_{jr}$) are updated with the LM method as

$$\begin{aligned}
(J_n^T J_n+\eta_n I)\Delta p_n&=-J_n^T e_n\\
p_{n+1}&=p_n+\Delta p_n
\end{aligned} \tag{9}$$

where $\eta_k \geq 0$ is a scalar, n is the epoch number, and I is the unit matrix. The trust region approach of Fletcher is used to determine $\eta$ at every epoch [7].

## 4 Simulation Results

In this section, the simulation results of the identification of a nonlinear plant using the FNN are presented. The plant to be identified is a bioreactor. The dynamical equations of the bioreactor are given by [9]

$$\begin{aligned}
\frac{dc_1}{dt}&=-c_1 u+c_1(1-c_2)e^{c_2/\gamma}\\
\frac{dc_2}{dt}&=-c_2 u+c_1(1-c_2)e^{c_2/\gamma}(1+\beta)/(1+\beta-c_2)
\end{aligned} \tag{10}$$

where the state variables $c_1$ and $c_2$ are, respectively, dimensionless cell mass and substrate conversion, u is the flow rate through the bioreactor, $\beta$=0.02 is the cell growth rate constant, and $\gamma$=0.48 is the nutrient consumption constant. The simulation data sets are obtained integrating the dynamics equations of the bioreactor. They are integrated with a fourth-order Runge-Kutta algorithm using an integration step size of

Δ=0.01 time units. We define 50Δ as a macro time step. We generated the training and the test data sets by applying the different series of step signal input to the bioreactor model. The series-parallel identification model with three inputs which are the plant input and the past values of the plant states was set. The FNN has six neurons with fuzzy activation functions in he hidden layer and two linear neurons in the output layer. The fuzzy systems with three rules are used as the activation functions. The initial parameters of the fuzzy systems are chosen so that the membership functions uniformly cover the interval [-2,2] at the input and [-1,1] at the output. The Levenberg-Marquardt optimization method with the trust region approach of Fletcher is used to update network parameters.

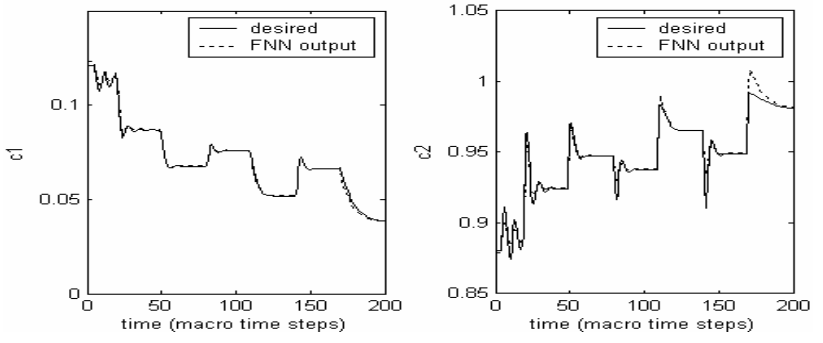



**Fig. 1.** The identification performance of the FNN

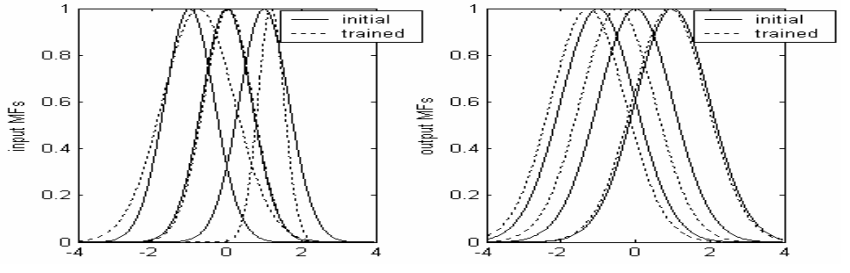



**Fig. 2.** The evolutions of the fuzzy rules with the training

The identification performance of the FNN is depicted in Fig. 1. The FNN outputs track the bioreactor states very well. The evolutions of the fuzzy rules with the training are shown in Fig. 1. The center and the spread of the input membership function and the center of the output membership function are adjusted with training. We can generate the IF-THEN rules from the trained fuzzy systems in the hidden layer. We define the Gaussian membership functions ($\mu_{A_j^r}(x_j;\overline{x}_{jr},\sigma_{jr})$ and $\mu_{B_j^r}(y_j;\overline{y}_{jr},1)$) with the center and spread parameters. $R_j^r$ represents the $r^{th}$ rule of the $j^{th}$ neuron. As an example, the rules of the $1^{nd}$ neuron are generated as

$$
\begin{aligned}
&R_1^1: \quad \text{IF} \quad x_1 = A_1^1 \quad \text{THEN} \quad y_1 = B_1^1 \\
&\text{where} \quad \mu_{A_1^1}(x_1;-0.7622,0.9677), \quad \mu_{B_1^1}(y_1;-1.2727,1) \\
&R_1^2: \quad \text{IF} \quad x_1 = A_1^2 \quad \text{THEN} \quad y_1 = B_1^2 \\
&\text{where} \quad \mu_{A_1^2}(x_1;-0.0383,0.6575), \quad \mu_{B_1^2}(y_1;-0.4795,1) \\
&R_1^3: \quad \text{IF} \quad x_1 = A_1^3 \quad \text{THEN} \quad y_1 = B_1^3 \\
&\text{where} \quad \mu_{A_1^3}(x_1;1.1857,0.3532), \quad \mu_{B_1^3}(y_1;0.9226,1)
\end{aligned}
\tag{11}
$$

## 5 Conclusions

We have proposed to combine the fuzzy system paradigm with the MLP. The resulting FNN was designed by replacing the sigmoid type activation function of the MLP with the fuzzy system. We adapted the LM optimization method with the trust region approach of Fletcher for the fast update of the FNN parameters. A typical problem in ANNs was simulated in order to show the validity of the approach. A good tracking performance is obtained.

The IF-THEN rules from the trained fuzzy systems in the hidden layer have been generated. We can conclude that interpreting these rules together with the rule reduction techniques may lead to optimize the number of neurons in the hidden layer

## References


1. Yamada, T., and Yabuta, T.: Neural Network Controller using Autotuning Method for Nonlinear Functions. IEEE Transactions on Neural Networks.(1992), 3, 595-601.
2. Chen, C.T. and Chang, W.D.: A Feedforward Neural Network with Function Shape Autotuning. Neural Networks, (1996) 9(4), 627-641.
3. Guarnieri, S., and Pizza. F.: Multilayer Feedforwardt Network with Adaptive Spline Activation Function. IEEE Transsactions on Neural Networks, (1999), 10(3), 672-683.
4. Trentin, E.: Networks with Trainable Amplitude of Activation Functions. Neural Networks, (2001), 14(4/5), 471-493.
5. Olivas, E.S., Guerrero, J.D.M., Valls, G.C., Lopez, A.J.S., Maravilla, J.C., and Chova, L.G.: A Low-Complexity Fuzzy Activation Function for Artificial Neural Networks. IEEE Transsactions on Neural Networks, (2003), 14(6), 1576-1579.
6. Oysal, Y., Becerikli, Y., and Konar, A.F.: Generalized modeling Principles of A Nonlinear System with a Dynamic Fuzzy Network. Computers & Chemical Engineering. 27 (2003), 1657-1664.
7. Scales, L.E.: Introduction to Non-Linear Optimization. Springer-Verlag New York Inc., USA, (1985) 115-118.
8. Wang, L.X.: A Course in Fuzzy Systems and Control. Prentice-Hall, Inc., (1997)
9. Ungar, L.H.: A Bioreactor Benchmark for Adaptive Network-based Process Control. In Neural Networks for Control, eds. Miller III, W.T., Sutton, R.S., and Werbos, P.J., MIT Press, London, (1990), 387-402.

# Neural Networks and Cascade Modeling Technique in System Identification

Erdem Turker Senalp[1], Ersin Tulunay[1,2], and Yurdanur Tulunay[3]

[1] Department of Electrical and Electronics Engineering, Middle East Technical University, 06531, Balgat, Ankara, Turkey
{Senalp, Ersintul}@metu.edu.tr
[2] TUBITAK Marmara Research Center, Information Technologies Institute, Gebze, Kocaeli, Turkey
Ersin.Tulunay@mam.gov.tr
[3] Department of Aerospace Engineering, Middle East Technical University, 06531, Balgat, Ankara, Turkey
Ytulunay@metu.edu.tr

**Abstract.** The use of the Middle East Technical University Neural Network and Cascade Modeling (METU-NN-C) technique in system identification to forecast complex nonlinear processes has been examined. Special cascade models based on Hammerstein system modeling have been developed. The total electron content (TEC) data evaluated from GPS measurements are vital in telecommunications and satellite navigation systems. Using the model, forecast of the TEC data in 10 minute intervals 1 hour ahead, during disturbed conditions have been made. In performance analysis an operation has been performed on a new validation data set by producing the forecast values. Forecast of GPS-TEC values have been achieved with high sensitivity and accuracy before, during and after the disturbed conditions. The performance results of the cascade modeling of the near Earth space process have been discussed in terms of system identification.

## 1 Introduction

Most of the dynamical systems can be represented by nonlinear modeling techniques. Nonlinear modeling is capable of describing the global system behavior for the overall operating range. Applying nonlinear model identification is inevitable for most of the real complex nonlinear processes including the near Earth space processes.

For many nonlinear dynamic processes, cascade models based on Hammerstein system modeling provide sufficient approximation [1] [2] [3] [4] [5] [6]. For many nonlinear dynamic processes it is required to present nonlinearities in the gain of the processes and provide dynamics in a linear block. This can be achieved by cascade modeling. In Hammerstein system modeling a nonlinear static block is cascaded with a linear dynamic block as shown in Figure 1.

These types of dynamic nonlinear process modeling provide some important features. The process identification task is simplified by modeling the dynamic part in the linear block, so data collection, computation of parameters and dynamic system

F.A. Savacı (Ed.): TAINN 2005, LNAI 3949, pp. 84 – 91, 2006.

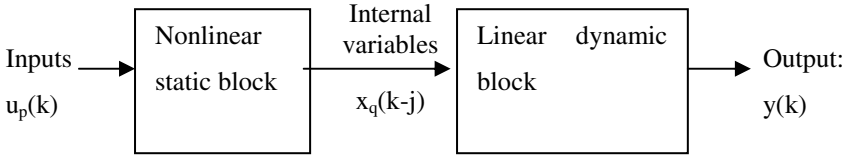



**Fig. 1.** Cascade models based on Hammerstein system modeling

analysis are simplified. To present nonlinearity in only the static gain decreases the degrees of freedom in the nonlinear system identification. In addition to this, it has accurate and robust approximations for a large class of real complex processes [1].

The internal variables are state-like variables and they are estimated by METU-NN. The static nonlinearity and the dynamic linearity in METU-C are estimated by using the cascade modeling technique.

It is important to identify ionospheric parameters, because satellites in low Earth orbit travel through it, and it is the medium through which radio waves used for communications propagate. Forecasting the number of electrons in a column of one meter-squared cross-section along a path through the ionosphere [7], the total electron content (TEC) values, is vital for telecommunications and satellite based navigation systems especially in the disturbed space weather conditions [8] [9] [10]. The interaction between electromagnetic waves and the ionospheric plasma has been studied both from scientific and engineering points of view [11]. Since 1990 a group of aerospace and electrical engineers have been developing Near Earth Space data driven models for system designers, users, planners as part of the EU-COST-TIST Actions at the METU [12]. The authors have studies on Neural Network based approaches, in modeling of the ionospheric processes [8] [9] [12] [13] [14] [15] [16] [17]. Those studies have provided insight on the system identification of the near Earth space processes.

In this work, to the best knowledge of the authors, it is the first time special models based on Hammerstein system modeling, METU-NN-C, with significant inputs have been developed for near Earth space processes. This paper outlines TEC forecasting problem and preparation of data, explains the METU-NN-C Hammerstein models as a system identification approach for forecasting ionospheric processes, gives the results with error tables, cross correlation coefficients and scatter diagrams, and discusses the generalized and fast learning and operation of the METU-NN-C Models.

## 2 Preparation of Data

For the training, test and validation within the development mode of the METU-NN-C, TEC data evaluated from GPS measurements in 1 April – 31 May 2000 and 2001 at Chilbolton (51.8º N; 1.26º W) receiving station are used. Operation has been performed on another data set by producing the forecast TEC values at Hailsham (50.9º N; 0.3º E) GPS receiving station for selected months in 2002 [18].

In the model intrinsic information about the solar activity is achieved by choosing the time periods for input data with the similar solar activity. This is the basic criterion in the selection of the train, test and validation years. The current high solar

activity time periods are selected in the time intervals. For training and validation phases within development procedure data sets of same month but different year are used to take the seasonal dependency into account.

## 3 Construction of the Neural Network Model

In METU-NN, for the current process, Feedforward Neural Network architecture with six neurons in one hidden layer is used. The activation functions in the hidden layer are hyperbolic tangent sigmoid functions and the activation function in the output layer is a linear function, so that the hidden layer outputs can represent the static part of the state-like internal variables in cascade modeling. Levenberg-Marquardt Backpropagation algorithm is used during training [19] [20]. The METU-NN is used to estimate the internal variables. The 5 inputs used for the METU-NN are as follows:

1. The present value of the TEC:

$$u_1(k) = \text{f(k)} \tag{1}$$

2. Cosine component of the minute, *m*, of the day:

$$u_2(k) = C_m = -Cos(2.\pi.m / 1440) \tag{2}$$

3. Sine component of the minute of the day:

$$u_3(k) = S_m = Sin(2.\pi.m / 1440) \tag{3}$$

4. Cosine component of the day, *d*, of the year:

$$u_4(k) = C_d = -Cos(2.\pi.d / 366) \tag{4}$$

5. Sine component of the day of the year:

$$u_5(k) = S_d = Sin(2.\pi.d / 366) \tag{5}$$

The output layer of the METU-NN hosts the value of the TEC being observed 60 minutes later than the present time. The outputs of the hidden layer in METU-NN are six of the internal variables for the METU-C.

## 4 Construction of the Cascade Model

The static nonlinearity in METU-C is described by polynomial representation of inputs. If the inputs are denoted by $u_p(k)$ then the outputs of the nonlinear element, i.e. the internal variables $x_q(k)$, may be expressed as in Equation 6.

$$\hat{\hat{x}}_q(k) = \sum_{p=1}^{R} f\left[u_p(k)\right] = \sum_{p=1}^{R}\sum_{i=0}^{m} \gamma_{pi}.u_p^i(k) \tag{6}$$

where R is the number of inputs, m+1 is the length of the series and $\gamma_{pi}$ are coefficients to be determined.

The output $u_1(k+1)$ is represented as shown in Equation 7 by using a dynamic linearity by optimizing a linear relationship for the internal variables $x_q(k)$ and their past values $x_q(k-j)$ which constitute their history.

$$\hat{\hat{u}}_1(k+1) = \sum_{q=1}^{S} \sum_{j=0}^{n} h_q(j).\hat{\hat{x}}_q(k-j) \tag{7}$$

where S is the number of the static internal variables and n is the number representing the history of the stored internal variables in memory. Thus, the product S.(n+1) gives the number of dynamic internal variables. The coefficients of the linearity in Equation 7, i.e. $h_q(j)$, are also determined in the development mode.

In the development mode, the construction work of the METU-C is carried out. It is composed of "training phase" and "test phase" as in the Neural Network approach [21]. In the training phase the parameters of the cascaded static nonlinear block and dynamic linear block are optimized. Figure 2 demonstrates the development modes of the METU-NN-C blocks.

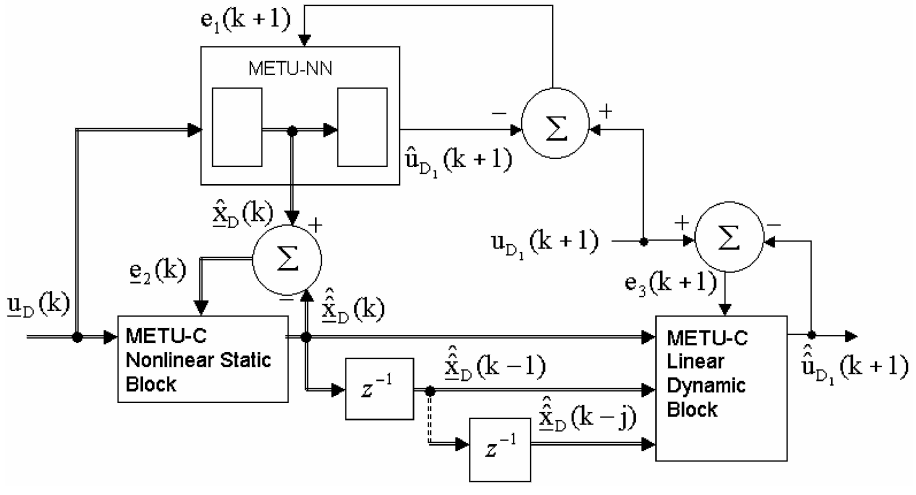



**Fig. 2.** Development of the METU-NN-C Models

The "Levenberg-Marquardt" optimization algorithm is used within training in the development mode for fast learning of the process with input data of very large size. For preventing the memorization, independent validation data are used and when the gradient of the error in the validation phase becomes near zero the training is terminated. The model is then ready for its use in the operation mode for forecasting of the TEC values. In the operation mode another data set is used for calculating the errors, point by point, to measure the performance of the model. Figure 1 demonstrates the operation mode of the METU-C.

For considering the first, second and third order terms in the polynomial representation, m is selected to be 3 for the TEC input, i.e. m=3 for p=1 in the model. For the temporal inputs m is selected to be 1, i.e. m=1 for p>1 in the model. Let the time instant *k* be in terms of minutes. The value of the TEC at the time instant *k* is designated by f(k). The 7 inputs used for the METU Cascade Model are the present value of the TEC, second and third powers, Cosine and Sine components of the minute of the day and day of the year.

The outputs of the first stage, i.e. 6 outputs for the static nonlinear block designated by $x_q(k)$, and their one hour past and two hours past values are stored as internal variables so that S=6 and n=2 in Equation 7. These internal variables are the inputs to the second stage of the cascade model, i.e. 18 inputs for the dynamic linear block of the METU-C model.

The output of the cascade model is designated by $u_1(k+60)=f(k+60)$ which is the value of the TEC to be observed 60 minutes later than the present time.

## 5 Results

In performance analysis an operation has been performed on an independent validation data set by producing the forecast values of the TEC. The operation mode performance analyses and results of the TEC forecast cover the time interval between April and May 2002 for the Hailsham receiving station. Forecast of the TEC values one hour in advance is performed for the validation data set in 10 minutes interval. Then the cross correlation coefficients between the observed GPS TEC and forecast TEC are calculated. The root mean square, normalized and absolute error values are also calculated. Table 1 is the error table displaying the results.

**Table 1.** Error Table

| | |
|---|---|
| Root Mean Square Error (TECU) | 1.7908 |
| Normalized Error | 0.0639 |
| Absolute Error (TECU) | 1.1708 |
| Cross Correlation Coefficient | 0.9863 |

Figure 3 is the scatter diagram of the forecast and observed TEC values. Figure 4 is the enlarged portion of some data points, i.e. the diurnal variations of the observed and forecast TEC values during 18-22 April 2002.

In the scatter diagram the fitted line has a slope close to one. Therefore the forecasting errors are small. This fact is the indication of the system reaching the correct operating point within the system identification. In other terminology, the system is succeeded to reach the global minimum of the error cost function. The correlation coefficients are very close to unity, which means that the METU-C model learned the shape of the inherent nonlinearities. Therefore, the deviations from straight line are small in the scatter diagram. When Figure 4 is observed it can be concluded that the model gives accurate forecasts before, during and after the disturbed conditions.

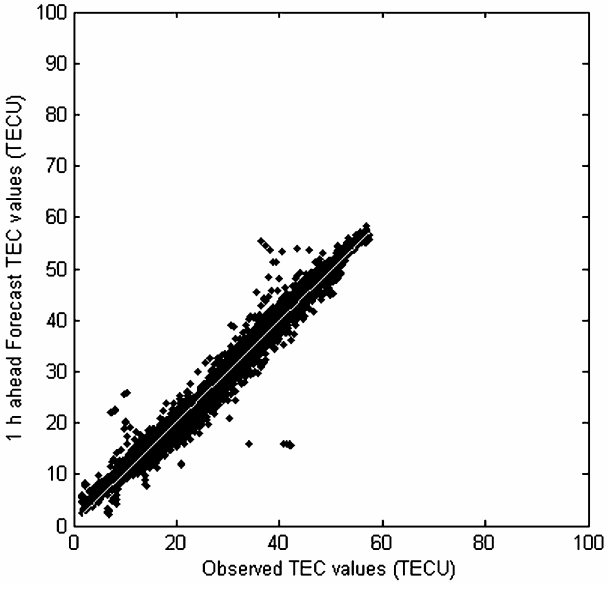



**Fig. 3.** 1 hour ahead Forecast TEC versus Observed GPS TEC values for April and May 2002

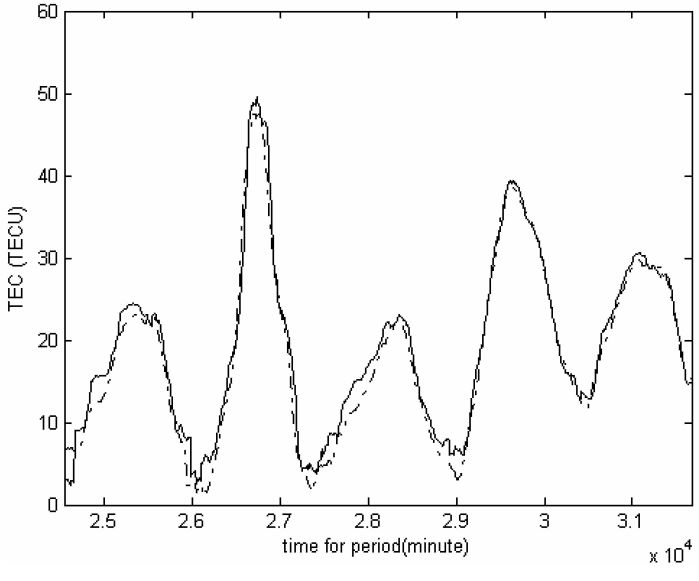



**Fig. 4.** Observed (dotted), and 1 hour ahead Forecast (solid) TEC values for 18-22 April 2002

## 6 Conclusions

Forecasting of the TEC values, especially in the disturbed space weather conditions, is crucial for communication, radar and navigation systems employing HF radio

waves to cope with the effects of unpredictable variability of the ionospheric parameters.

Neural Network and cascade modeling technique in METU-NN-C are used in system identification. In this work, to the best knowledge of the authors, cascade modeling of Hammerstein form has been used first time for the forecast of an ionospheric-plasmaspheric process, namely the TEC variation 1 hour in advance. The model learned the shape of the inherent nonlinearities and the system reached the correct operating point. The cascade modeling of the process is also capable of forecasting the TEC values for disturbed solar-terrestrial conditions.

It is demonstrated that the identification of the complex nonlinear processes, such as the TEC variation, can be achieved by cascading a static nonlinear block and a linear dynamic block of Hammerstein form.

Summary of the main contributions of this work may be given as follows:

1) Organization of representable data for learning complex processes,
2) Estimation of the internal variables of METU-C by using NN,
3) Cascade modeling of a highly complex nonlinear process such as the TEC variation, and
4) General demonstration of learning capability by calculating cross correlations and general demonstration of reaching a proper operating point by calculating errors.

## Acknowledgement

Dr. Lj. Cander who hosted Mr. Senalp and provided the GPS data during a STM as a joint COST 271 EACOS Action between UK and Turkey is acknowledged. [18].

## References

1. Ikonen E., and Najim K.: Learning control and modelling of complex industrial processes, Overview report of the activities within the European Science Foundation's programme on Control of Complex Systems (COSY) Theme 3: Learning control, February (1999)
2. Narendra K.S., and Gallman P.G.: An Iterative Method for the Identification of Nonlinear Systems Using a Hammerstein Model, IEEE Transactions on Automatic Control, (1966), 546-550
3. Fruzzetti, K.P., Palazoglu A., McDonald K.A.: Nonlinear model predictive control using Hammerstein models, J. Proc. Cont., Vol. 7, No. 1, (1997), 31-41
4. Marchi, P.A., Coelho L.S., Coelho A.R.: Comparative Study of Parametric and Structural Methodologies in Identification of an Experimental Nonlinear Process, Proceedings of the 1999 IEEE International Conference on Control Applications, Hawaii, USA, 22-27 August (1999), 1062-1067
5. Bai E.W., and Fu M.: A Blind Approach to Hammerstein Model Identification, IEEE Transactions on Signal Processing, Vol. 50, No. 7, July (2002), 1610-1619
6. Westwick, D.T., and Kearney, R.E.: Identification of a Hammerstein model of the stretch reflex EMG using separable least squares, Engineering in Medicine and Biology Society, 2000. Proceedings of the 22nd Annual International Conference of the IEEE, Vol. 3, 23-28 July (2000), 1901-1904

7. Chilbolton Weather Web, Space Weather – Total Electron Content of the Ionosphere, Rutherford Appleton Laboratory, http://www.rcru.rl.ac.uk/weather/tec.htm, (2001)
8. Senalp E.T., Tulunay E., Tulunay Y.: Neural Network Based Approach to Forecast the Total Electron Content Values, EGS 2002 Conference, 27th General Assembly of the European Geophysical Society, CD of Abstracts, Nice, France, 21-26 April (2002), EGS02-A-00867
9. Tulunay E., Senalp E.T., Cander Lj.R., Tulunay Y.K., Bilge A.H., Mizrahi E., Kouris S.S., Jakowski N.: Development of algorithms and software for forecasting, nowcasting and variability of TEC, Annals of Geophysics, Vol. 47, N. 2/3, (2004) 1201-1214
10. Cander Lj.R., Milosavlijevic, M.M., Stankovic S.S., Tomasevic S.: Ionospheric Forecasting Technique by Artificial Neural Network, Electronics Letters, Vol. 34, No. 16, Online No: 19981113, 6 August (1998), 1573-1574
11. Rycroft, M.: Lecture Notes, TUBITAK FGI Research Institute, Kandilli, Istanbul, Turkey (2004)
12. Tulunay Y., Tulunay E., Kutay A.T., Senalp E.T.: Neural Network Based Approaches for Some Nonlinear Processes, URSI-TURKIYE'2002, ITU – Istanbul, Turkey, 18-20 September (2002), 403-406
13. Tulunay E., Tulunay Y., Senalp E.T.: Neural Network Based Approach to Forecast Ionospheric Parameters, SPECIAL Workshop, Lindau, Germany, 8-11 November (2000)
14. Tulunay Y., Tulunay E., Senalp E.T., Ozkaptan C.: Neural Network Modeling of the Effect of the IMF Turning on the Variability of HF Propagation Medium, AP 2000, Millennium Conference on Antennas & Propagation, ICAP&JINA, Davos, Switzerland, 9-14 April (2000), 132
15. Tulunay E., Senalp E.T., Tulunay Y.: Forecasting the Total Electron Content By Neural Networks, COST 271 "Effects of the Upper Atmosphere on Terrestrial and Earth-Space Communications" Workshop, Ionospheric Modeling and Variability Studies for Telecommunications Applications, Book of Abstracts, Sopron, Hungary, 25-27 September (2001), 47
16. Tulunay Y., Tulunay E., Senalp E.T.: An Attempt to Model the Influence of the Trough on HF Communication by Using Neural Network, Radio Science, Vol. 36, No. 5, September - October (2001), 1027-1041
17. Tulunay Y., Tulunay E., Senalp E.T.: The Neural Network Technique-2: An Ionospheric Example Illustrating its Application, Adv. Space Res., Vol. 33, No. 6, (2004), 988-992
18. COST271 WG 4 STM, "Effects of the Upper Atmosphere on Terrestrial and Earth-Space Communications", Short term scientific mission work on the TEC data available at RAL to organize the input data for the METU NN model by E.T. Senalp under the supervision of Dr. Lj. Cander, Terms of Reference, RAL, Chilton, Didcot, U.K., 30 June – 7 July (2002)
19. Hagan M.T., and Menhaj M.B.: Training Feedforward Networks with the Marquard Algorithm, IEEE Transactions on Neural Networks, Vol. 5 No. 6, (1994), 989-993
20. Haykin, S.: Neural Networks: A Comprehensive Foundation, 2nd ed., Prentice-Hall, Inc., New Jersey, USA, (1999), pp. 2, 10, 21-22, 83-84, 169, 215
21. Tulunay Y., Tulunay E., Senalp E.T., The Neural Network Technique-1: A General Exposition, Adv. Space Res., Vol. 33, No. 6, (2004), 983-987

# Comparison of Complex-Valued Neural Network and Fuzzy Clustering Complex-Valued Neural Network for Load-Flow Analysis

Murat Ceylan, Nurettin Çetinkaya, Rahime Ceylan, and Yüksel Özbay

Selçuk University, Electrical & Electronics Dept.
42075-Konya/Turkey
{mceylan, ncetinkaya, rpektatli,yozbay}@selcuk.edu.tr

**Abstract.** Neural networks (NNs) have been widely used in the power industry for applications such as fault classification, protection, fault diagnosis, relaying schemes, load forecasting, power generation and optimal power flow etc. Most of NNs are built upon the environment of real numbers. However, it is well known that in computations related to electric power systems, such as load-flow analysis and fault level estimation etc., complex numbers are extensively involved. The reactive power drawn from a substation, the impedance, busbar voltages and currents are all expressed in complex numbers. Hence, NNs in the complex domain must be adopted for these applications. This paper proposes the complex-valued neural network (CVNN) and a new fuzzy clustering complex-valued neural network (FC-CVNN) to estimate busbar voltages in a load-flow problem. The aim of this paper is to present a comparative study of estimation busbar voltages in load-flow analysis using the conventional neural network (real-valued neural network, RVNN), the CVNN and the new FC-CVNN. The results suggest that a new proposed FC-CVNN and CVNN architecture can generalize better than ordinary RVNN and the FC-CVNN is also learn faster.

## 1 Introduction

Load-flow studies are extremely important in evaluating the operation of power systems, controlling them and planning for future expressions. [1,2]. Steady state of power systems may be determined by solving the power flow (PF) equations that matematically are represented by a set of nonlinear algebraic equations. This problem is known as the load-flow or PF problem and its main objective is the calculation of all bus voltages magnitudes and angles, and consequently the PFs over the transmission lines [3,4].

The principles involved in power-flow studies are straightforward, but a study relating to a real power system can be carried out only with a digital computer [5,6]. NNs represent the promising new generation of information-processing networks [7,8]. Advances have been made in applying such systems for problems that have been found to be intractable or difficult for traditional computation. NNs have been proved to capable of learning from raw data. They can be used to identify internal relationships within raw data not explicitly given or even known by human experts, and there is no need to assume any linear relationship between data. NNs have widely

F.A. Savacı (Ed.): TAINN 2005, LNAI 3949, pp. 92 – 99, 2006.

been used in electric power engineering. For energy management, load-flow and optimal-power-flow problems have been solved by NNs [9,10]. However, most existing NNs for electric power applications have been designed using real numbers. In power engineering, applications such as load-flow analysis, phasor evaluation, signal processing and image processing mainly involve complex numbers. Although conventional NNs are able to deal with complex numbers by treating the real and imaginary parts independently, it will be shown in this paper that their behaviour is not so satisfactory [11].

This paper was proposed using complex-valued neural network (CVNN) and a new fuzzy clustering complex-valued neural network (FC-CVNN) to make a neural network system more effective for manipulation of complex numbers in electric power system. CVNN, the first approach proposed in this paper, particularly is designed for computational complex numbers. So, number of input and output nodes of neural network is reduced using this approach. FC-CVNN, the second approach proposed in this paper, is been composed of two subnetworks: fuzzy self organizing layer and CVNN. The fuzzy self organizing layer performing the data reduction and the following CVNN working for recognition. A number of sets in training patterns is reduced using fuzzy c-means clustering in fuzzy self organizing layer before inputs are presented to the CVNN system. Therefore, training period of a neural network is decreased. The two methods proposed in this paper, CVNN and FC-CVNN, were used to estimate busbar voltages in a load-flow problem.

## 2 Complex-Valued Neural Network (CVNN)

In this paper, we used complex back-propagation (BP) algorithm for load-flow analysis. In this section, the theory of the complex-BP algorithm is applied to a multilayer (complex-valued) neural network [12]. This algorithm enables the network to learn complex-valued patterns in a natural way. We will first describe the complex-BP model. The input signals, weights, thresholds and output signals are all complex numbers.

The activity $Y_n$ of neuron n is defined as:

$$Y_n = \sum_m W_{nm} X_m + V_n \tag{1}$$

where $W_{nm}$ is the weight connecting neuron $n$ and $m$, $X_m$ is the input signal from neuron $m$, and $V_n$ is the threshold value of neuron $n$. To obtain the output signal, the activity value $Y_n$ is converted into its real and imaginary parts as follows :

$$Y_n = x + iy = z \tag{2}$$

where $i$ denotes $\sqrt{-1}$. Although various output functions of each neuron can be considered, the output function defined by the following equation will be used :

$$f_C(z) = f_R(x) + i.f_R(y) \tag{3}$$

where $f_R(u)=1/(1+exp(-u))$ and is called the sigmoid function.

For the sake of simplicity, the networks used both in the analyses and experiments will have three layers. We will use $W_{ml}$ for the weight between the input neuron $l$ and

the hidden neuron $m$, $V_{nm}$ for the weight between the hidden neuron $m$ and the output neuron $n$, $\theta_m$ for the threshold of the hidden neuron $m$, and $\gamma_n$ for the threshold of the output neuron $n$. Let $I_l$, $H_m$, $O_n$ denote the output values of the input neuron $l$, the hidden neuron $m$, and the output neuron $n$, respectively. Let also $U_m$ and $S_n$ denote the internal potentials of the hidden neuron $m$ and the output neuron $n$, respectively. That's, $U_m=\Sigma_l W_{ml}.I_l+\theta_m$, $S_n=\Sigma_m V_{nm}H_m+\gamma_n$, $H_m=f_C(U_m)$ and $O_n=f_C(S_n)$. Let $\delta_n=T_n-O_n$ denote the error between the actual pattern $O_n$ and the target pattern $T_n$ of output neuron $n$. We will define the square error for the pattern $p$ as $E_p=(1/2)\Sigma_n |T_n-On|^2$.

We can show that the weights and the thresholds should be modified according to the equations in [13].

Complex activation function is a superposition of real and imaginary logarithmic sigmoids, as shown by

$$\varphi(Z)=\frac{C}{1+\exp(-Z_r)}+j.\frac{C}{1+\exp(-Z_i)} \tag{4}$$

where $Z_r$ and $Z_i$ are, respectively, the real and imaginary parts of $Z$ [12-14].

$\varepsilon$ is learning rate parameter in above equations. It may be noted here that a large value of the learning rate may lead to faster convergence but may also result in oscillation. In this paper, learning rate parameter was found as 1.0 via experimental study.

## 3 The Fuzzy Clustering Complex-Valued Neural Network

This paper presents a new fuzzy clustering complex-valued neural network (FC-CVNN) for load-flow analysis. The proposed structure is been composed of two subnetworks: fuzzy self organizing layer and complex-valued neural network.

The idea of fuzzy clustering is to divide the data into fuzzy partitions that overlap with one another. Therefore, the inclusion of data in a cluster is defined by a membership grade in [0,1]. Formally, clustering an unlabeled data $X=\{x_1,x_2,...,x_N\} \subset R^h$, where $N$ represents the number of data vectors and $h$ the dimension of each data vector, is the assignment of $c$ partition labels to the vectors in $X$. $c$–partition of $X$ constitutes sets of $(c\cdot N)$ $\{u_{ik}\}$ membership values that can be conveniently arranged as a $(c \times N)$ matrix $U=[u_{ik}]$. The problem of fuzzy clustering is to find the optimum membership matrix $U$. The most widely used objective function for fuzzy clustering is the weighted within groups sum of squared errors $J_m$, which is used to define the following constrained optimization problem [14].

$$\min\left\{ J_m(U,V,X)=\sum_{k=1}^{N}\sum_{i=1}^{c}(u_{ik})^m \left\|x_k-v_i\right\|_A^2 \right\} \tag{5}$$

where $V=\{v_1,v_2,...,v_c\}$ is the vector of (unknown) cluster centers. Fuzzy partitions are carried out using the fuzzy c-means (FCM) algorithm through an iterative optimization of according to the following steps [14]:

Step 1) Choose the number of clusters $(c)$, weighting exponent $(m)$, iteration limit $(iter)$, termination criterion $(\varepsilon>0)$, and norm for error $\left\| V_t-V_{t-1} \right\|$.

Step 2) Guess initial position of cluster centers:

$$V_0 = \{v_{1,0}, v_{2,0}, ....., v_{c,0}\} \subset \Re^{ch} . \left(\sum_{i=1}^{c} u_{ik} = 1 \forall k\right)$$

Step 3) Iterate for *t=1* iter , calculate

$$u_{ik,t} = \left[\sum_{j=1}^{c}\left(\frac{\|X_k - V_{i,t-1}\|_A}{\|X_k - V_{j,t-1}\|_A}\right)^{2/m-1}\right]^{-1}, \quad V_{i,t} = \frac{\sum_{k=1}^{N}(u_{ik,t})^m x_k}{\sum_{k=1}^{N}(u_{ik,t})^m} \tag{6}$$

IF error=$\| V_t - V_{t-1} \| \leq \varepsilon$, THEN stop, and put $(U_f, V_f) = (U_t, V_t)$ for NEXT *t*.

In this study, weighting exponent was chosen as 5. It can be seen in Fig.1 that training patterns are decreased using fuzzy c-means clustering algorithm in the architecture of FC-CVNN before the CVNN training. In other words, fuzzy clustering algorithm is used as data reduction algortihm. The clustered data is applied as input to the CVNN. Finally, the CVNN is trained for learning clustered data. Training is done with the complex back-propagation algorithm.

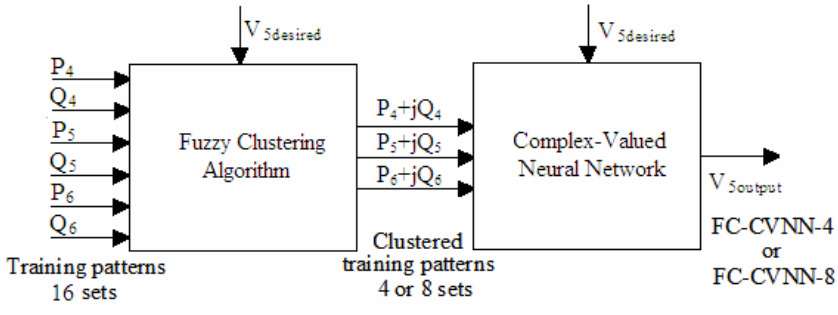



**Fig. 1.** Fuzzy Clustering Complex-Valued Neural Network (FC-CVNN) architecture

## 4 Results

It has been shown that a multilayer feedforward neural network with one or more hidden layers is sufficient to approximate any continuous nonlinear function arbitrarily well on a compact interval, provided sufficient hidden neurons are avaliable [7]. The power load-flow problem is a nonlinear problem and, hence, it can be analysed with the help of a NN. A 6-busbar network, as shown in Fig.2 [7], has been used to test the performance of the new developed CVNN and FC-CVNN. Busbar 1, busbar 2 and busbar 3 are generator busbars while busbar 1 is the swing busbar. Busbar 4, busbar 5 and busbar 6 are ordinary load busbars where the P (active power) and Q (reactive power) are to be specified. The training example is generated by the Power World Simulator 9.0 Educational Version Program used Newton-Raphson algorithms. This is just an illustrative example. The details of the network parameters are shown in Table 1 and 2. Network parameters, training and test patterns in Table 1,2,3 and 4 were taken from study of Chan et al. [7].

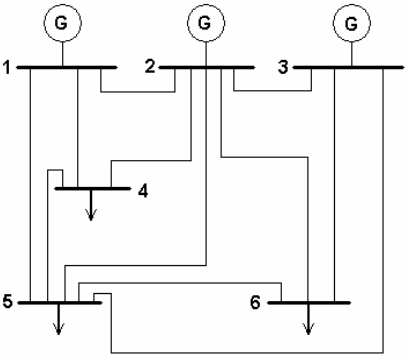



**Fig. 2.** The 6-busbar network for load-flow computation

**Table 1.** Busbar power for load-flow study

| Busbar | $P_{load}$ | $Q_{load}$ | $P_{gen}$ | $V_{spec}$ |
|---|---|---|---|---|
| 1 | 0 | 0 | - | 1,05 |
| 2 | 0 | 0 | 0,5 | 1,05 |
| 3 | 0 | 0 | 0,6 | 1,07 |
| 4 | $P_4$ | $Q_4$ | - | - |
| 5 | $P_5$ | $Q_5$ | - | - |
| 6 | $P_6$ | $Q_6$ | - | - |

**Table 2.** Network parameters for load-flow study

| From busbar | To busbar | R, p.u. | X, p.u. | B, p.u. |
|---|---|---|---|---|
| 1 | 2 | 0,1 | 0,2 | 0,02 |
| 1 | 4 | 0,05 | 0,2 | 0,02 |
| 1 | 5 | 0,08 | 0,3 | 0,03 |
| 2 | 3 | 0,05 | 0,25 | 0,03 |
| 2 | 4 | 0,05 | 0,1 | 0,01 |
| 2 | 5 | 0,1 | 0,3 | 0,02 |
| 2 | 6 | 0,07 | 0,2 | 0,025 |
| 3 | 5 | 0,12 | 0,26 | 0,025 |
| 3 | 6 | 0,02 | 0,1 | 0,01 |
| 4 | 5 | 0,2 | 0,4 | 0,04 |
| 5 | 6 | 0,1 | 0,3 | 0,03 |

Training patterns generated by the software package for learning by RVNN, CVNN and FC-CVNN are shown at Table 3. In this case, the voltage at busbar 5 is to be estimated while the three generators maintain constant voltages at the corresponding busbars. $P_i$ and $Q_i$, i=4, 5 and 6, are inputs of NN. Number of input nodes for RVNN, CVNN and FC-CVNN are 6, 3 and 6, respectively. $V_5$ is output of NN. Number of output nodes for RVNN, CVNN and FC-CVNN are 2, 1 and 1, respectively. Number of training sets is 16 for RVNN and CVNN. Number of training sets was decreased as using fuzzy clustering algortihm in FC-CVNN proposed in this paper. Therefore, two training patterns were obtained in two different lengths of them as 4 and 8 sets by fuzzy clustering. The aim of fuzzy clustering is to decrease training time of CVNN by decreasing length of training sets. Then, these training sets (FC-4, FC-8) were applied as input to the CVNN.

The number of hidden nodes was determined via experimentation. The experimental results show that the optimum number of hidden nodes was four with the minimum test error of 0.38 % for 10000 iterations, can be seen in Fig. 3. Therefore, CVNN was determined as 3:4:1 network, so the RVNN was determined as 6:8:2 network. Similarly, optimum learning rate was found as 1.0 in experimental study.

The initial values of all weights were randomly selected. In this study, logaritmic sigmoid function formulated Eq. (4) was used as activation function. So, training and testing patterns were normalised in range of [0,1]. Maximum iteration number was set to 34000 for all of simulations. The power load-flow network was learned by RVNN, CVNN and FC-CVNN under 16 combinations of $P_4$, $P_5$, $P_6$, $Q_4$, $Q_5$, $Q_6$ and $V_5$. The variation of the total squared error of four structures, RVNN, CVNN, FC-CVNN-4, FC-CVNN-8, with respect to the number of iteration in training is shown in Fig. 4.

**Table 3.** Training patterns for the neural networks

| Case | $P_4$+j$Q_4$ p.u. | $P_5$+j$Q_5$ p.u. | $P_6$+j$Q_6$ p.u. | $V_5$ p.u. |
|---|---|---|---|---|
| 1 | 0,7+j0,7 | 0,7+j0,7 | 0,7+j0,7 | 0,9542 - j0,113 |
| 2 | 0,9+j0,9 | 0,9+j0,9 | 0,9+j0,9 | 0,9182 - j0,117 |
| 3 | 0,9+j0,7 | 0,7+j0,7 | 0,7+j0,7 | 0,9518 - j0,113 |
| 4 | 0,7+j0,9 | 0,7+j0,7 | 0,7+j0,7 | 0,9502 - j0,111 |
| 5 | 0,7+j0,7 | 0,9+j0,7 | 0,7+j0,7 | 0,9455 - j0,126 |
| 6 | 0,7+j0,7 | 0,7+j0,9 | 0,7+j0,7 | 0,9383 - j0,106 |
| 7 | 0,7+j0,7 | 0,7+j0,7 | 0,9+j0,7 | 0,9526 - j0,117 |
| 8 | 0,7+j0,7 | 0,7+j0,7 | 0,7+j0,9 | 0,9513 - j0,112 |
| 9 | 0,9+j0,7 | 0,9+j0,9 | 0,9+j0,9 | 0,9225 - j0,119 |
| 10 | 0,7+j0,9 | 0,9+j0,9 | 0,9+j0,9 | 0,9207 - j0,118 |
| 11 | 0,9+j0,9 | 0,9+j0,7 | 0,9+j0,9 | 0,9345 - j0,125 |
| 12 | 0,9+j0,9 | 0,7+j0,9 | 0,9+j0,9 | 0,9273 - j0,105 |
| 13 | 0,9+j0,9 | 0,9+j0,9 | 0,9+j0,7 | 0,9213 - j0,119 |
| 14 | 0,9+j0,9 | 0,9+j0,9 | 0,7+j0,9 | 0,9197 - j0,115 |
| 15 | 0,9+j0,9 | 0,8+j0,8 | 0,7+j0,7 | 0,9355 - j0,113 |
| 16 | 0,7+j0,7 | 0,8+j0,8 | 0,9+j0,9 | 0,9376 - j0,117 |

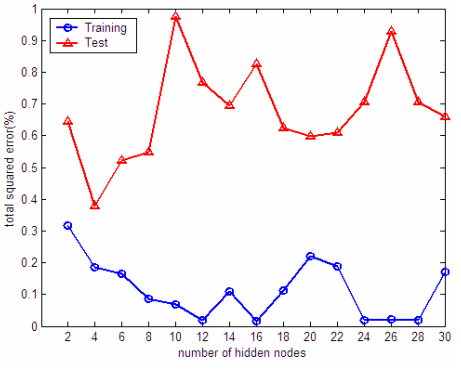



**Fig. 3.** The experimental results show that optimum number of hidden nodes was 4

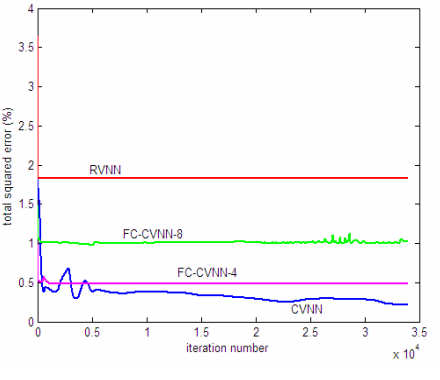



**Fig. 4.** The variation of total squared error of four structures with respect to number of iteration in training for power load-flow problem

**Table 4.** Test Cases

| Case | $P_4$+j$Q_4$ | $P_5$+j$Q_5$ | $P_6$+j$Q_6$ | $V_5$ |
|---|---|---|---|---|
| 1 | 0,77 + j0,82 | 0,75 + j0,79 | 0,84 + j0,83 | 0,9402 – j0,114 |
| 2 | 0,72 + j0,76 | 0,88 + j0,81 | 0,77 + j0,80 | 0,9341 – j0,120 |
| 3 | 0,83 + j0,87 | 0,72 + j0,79 | 0,82 + j0,89 | 0,9375 – j0,110 |
| 4 | 0,75 + j0,77 | 0,82 + j0,89 | 0,80 + j0,76 | 0,9303 – j0,115 |
| 5 | 0,84 + j0,81 | 0,71 + j0,77 | 0,79 + j0,82 | 0,9420 – j0,110 |
| 6 | 0,88 + j0,81 | 0,83 + j0,87 | 0,75 + j0,82 | 0,9285 – j0,113 |
| 7 | 0,80 + j0,80 | 0,80 + j0,80 | 0,80 + j0,80 | 0,9366 – j0,115 |
| 8 | 0,61 + j0,69 | 0,92 + j0,95 | 0,78 + j0,67 | 0,9256 – j0,120 |
| 9 | 0,58 + j0,69 | 0,76 + j0,94 | 0,97 + j0,88 | 0,9293 – j0,112 |
| 10 | 0,79 + j0,87 | 0,61 + j0,57 | 0,94 + j0,68 | 0,9621 – j0,114 |
| 11 | 0,60 + j0,60 | 0,60 + j0,60 | 0,60 + j0,60 | 0,9719 – j0,099 |
| 12 | 1,00 + j1,00 | 1,00 + j1,00 | 1,00 + j1,00 | 0,8949 – j0,147 |

**Table 5.** Comparison of four structures in testing

| $V_5$-CORRECT | $V_5$-RVNN | $V_5$-CVNN | $V_5$-FC-CVNN-8 | $V_5$-FC-CVNN-4 |
|---|---|---|---|---|
| 0,9402 – j0,114 | 0.4585+j0.5020 | 0.9094-j0.1051 | 0.9331-j0.1274 | 0.9167-j0.1230 |
| 0,9341 – j0,120 | 0.4585+j0.5020 | 0.9601-j0.1448 | 0.9618-j0.1469 | 0.9169-j0.1241 |
| 0,9375 – j0,110 | 0.4585+j0.5020 | 0.9109-j0.1038 | 0.9331-j0.1274 | 0.9177-j0.1170 |
| 0,9303 – j0,115 | 0.4585+j0.5020 | 0.9591-j0.1413 | 0.9331-j0.1274 | 0.9168-j0.1236 |
| 0,9420 – j0,110 | 0.4585+j0.5020 | 0.9133-j0.1029 | 0.9331-j0.1274 | 0.9167-j0.1232 |
| 0,9285 – j0,113 | 0.4585+j0.5020 | 0.9052-j0.1098 | 0.9331-j0.1274 | 0.9183-j0.1239 |
| 0,9366 – j0,115 | 0.4585+j0.5020 | 0.9078-j0.1086 | 0.9068-j0.0992 | 0.9169-j0.1243 |
| 0,9256 – j0,120 | 0.4585+j0.5020 | 0.9577-j0.1453 | 0.9331-j0.1274 | 0.9173-j0.1248 |
| 0,9293 – j0,112 | 0.4585+j0.5020 | 0.9069-j0.1044 | 0.9331-j0.1274 | 0.9161-j0.1192 |
| 0,9621 – j0,114 | 0.4585+j0.5020 | 0.9204-j0.1007 | 0.9167-j0.1012 | 0.9494-j0.1469 |
| 0,9719 – j0,099 | 0.4585+j0.5020 | 0.9698-j0.1210 | 0.9331-j0.1274 | 0.8982-j0.1052 |
| 0,8949 – j0,147 | 0.4585+j0.5020 | 0.8980-j0.1268 | 0.9167-j0.1012 | 0.9532-j0.1215 |
| Test Error (%) | 22.55 | 0.91 | 0.89 | 0.99 |
| CPU Time (sec) | 346.31 | 316.70 | 180.50 | 88.76 |

After the four structures were trained, they were used to estimate $V_5$ for different testing patterns of $P_i$ and $Q_i$, i=4, 5 and 6. There are two categories of testing patterns, first set (cases 1 to 7) being those values of P and Q randomly selected in between the limits of values of P and Q included in Table 3. The other set (cases 8 to 12) is randomly selected outsides the limits to test the ability of generalization of the four structures [7]. The $P_i$ and $Q_i$ values in test sets are shown in Table 4 while the test results are shown in Table 5.

Although CVNN was better than FC-CVNN and RVNN in training (see Fig. 4), more accurate recognition rate (99 %) was obtained with CVNN and FC-CVNN to estimate voltages of busbar 5 in testing (see Table 5). In other words, the generalization ability of CVNN and FC-CVNN is better than RVNN. Also, it is noted in the results below that the FC-CVNN converges to a determined error goal faster than RVNN and CVNN.

## 5 Conclusions

It is generally accepted that the learning power of NNs is useful in electric-power system analysis. However, it is well known that in computations related to electric

power systems, complex numbers are extensively involved, due to the existence of phase differences between different parameters. Therefore, in this paper, the complex-valued neural network and the new fuzzy clustering complex-valued neural network has been developed and presented to estimate busbar voltages in load-flow analysis. For training and testing, patterns generated by the power flow software package are used.

A comparative assessment of the performance of RVNN, CVNN and FC-CVNN show that more reliable results are obtained with the FC-CVNN and CVNN for the estimation voltage of busbars in load-flow analysis. The aim in developing FC-CVNN was to achieve more optimum results with relatively few training patterns in less training time. It has demonstrated that the training time of the FC-CVNN was 52% of the time required by the RVNN.

## References


1. Luciana M.C.B., Carlos A.C., Carlos A.F.M.: A Critical Evaluation of Step Size Optimization Based Load Flow Methods, IEEE Transaction on Power Systems, Vol.15, No.1, February , 202-207, (2000)
2. Nasar, S. A.: Theory and Problems of Electric Power Systems, 1990, McGraw-Hill, 172p.
3. Arrilliga, J., Arnold, C.P., Harker, B.J.: Computer Analysis of Power Systems, Wiley, (1991).
4. Paucar, V.L., Rider, M.J.: Artificial neural networks for solving the power flow problem in electric power systems, Electric Power System Research, Vol.62, 139-144, (2002).
5. Olle I.E.; Electric Energy System Theory, McGraw-Hill Book Company, 1985, 526p.
6. William D. Stevenson, Jr., Elements of Power System Analysis, International Edition, McGraw-Hill, Singapore, 421p., (1982)
7. Chan, W.L., So, A.T.P., and Lai L.L.: Initial applications of complex artificial neural networks to load-flow analysis, IEE Proc. Gener. Transm. Distrib.,vol. 147, No. 6, 361-366, November, (2000).
8. Zurada, J.M.: Introduction to artificial neural systems, Info Access and Distribution Pte Ltd., 1-3, Singapore, (1992)
9. Nguyen, T.T.: Neural network optimal power flow, proceedings of the fourth international conference on Advances in power system control, operation and management, IEE Conf. Publ.450, 266-271, November, (1997)
10. Nguyen, T.T.: Neural network load-flow, IEE Proc. Gener.Transm. Distrib., Vol.142, 51-58,(1995).
11. Chan, W.L., and So, A.T.P.: Development of a new artificial neural network in complex space, Proceedings of 2nd Biennial Australian Engineering Mathematics Conference, 225-230, Sydney, July, (1996).
12. Haykın, S.: Adaptive filter Theory, Prentice-Hall, (2002).
13. Nitta, T.: An extension of the back-propagation algortihm to complex numbers, Pergamon Neural Networks, Vol.10, 1391-1415, (1997).
14. Jang, J.-S. R, Sun, C.-T, and Mizutani, E.: Neuro-Fuzzy and Soft Computing, Prentice Hall, USA, (1997).

# A New Formulation for Classification by Ellipsoids

Ayşegül Uçar[1], Yakup Demir[1], and Cüneyt Güzeliş[2]

[1] Electrical and Electronics Engineering Department, Engineering Faculty, Fırat University, Elazig, Turkey
agulucar@firat.edu.tr,ydemir@firat.edu.tr
[2] Electrical and Electronics Engineering Department, Dokuz Eylül University, Kaynaklar Campus, İzmir, Turkey
guzelis@eee.deu.edu.tr

**Abstract.** We propose a new formulation for the optimal separation problems. This robust formulation is based on finding the minimum volume ellipsoid covering the points belong to the class. Idea is to separate by ellipsoids in the input space without mapping data to a high dimensional feature space unlike Support Vector Machines. Thus the distance order in the input space is preserved. Hopfield Neural Network is described for solving the optimization problem. The benchmark Iris data is given to evaluate the formulation.

## 1 Introduction

Many methods for binary classification were proposed. Support Vector Machines (SVMs) are one of the optimization-based approaches for solving supervised machine learning problems [1-2]. The basic ability of SVM is to implicitly transform data to a high dimensional space and to construct a linear binary classifier without having to perform any computations in the high dimensional space. Hence SVM doesn't suffer from problems such as the dimensionality of the data and the sparsity of data in the high-dimensional space. The hyperplanes in the high dimensional transform space result in complex decision surfaces in the input data space. However the SVM classifiers can not be preserve the distance in input space since they separate data in the feature space [3].

In this paper, the classification in the input space is aimed. An ellipsoid is used as main tool. Few authors attempted to solve the pattern separation problem by ellipsoids. Among these, Barnes, Boyd and Astorino presented valid the formulations and algorithms for especially ellipsoidal separable problem [4-7]. On the other hand, Frigue developed semidefinite programming formulations for both ellipsoidal separable and non-ellipsoidal separable problems. However a robust classification remains still a need for pattern separation area. We give a new formulation by underlying both structural error and empirical error in SVM classifiers in this work.

The remainder of this paper is organized as follows. Section II reviews the SVM classifiers and their some limitations. The proposed formulation and solution algorithm are shown in Section III. In Section IV, the performance of the method is illustrated on Iris data. Conclusions are summarized in Section V.

F.A. Savacı (Ed.): TAINN 2005, LNAI 3949, pp. 100 – 106, 2006.

## 2 Reviews to SVM Classifier

The SVM learns the decision function that maximally separates two classes of training samples by the prime optimization problem [1]:

$$\begin{aligned} \min_{w,b,\xi} \quad & J(\vec{w},\vec{\xi}) = \frac{1}{2} w^T w + C\sum_{i=1}^{l} \xi_i \\ st. \quad & y_i\left[w^T \varphi(x_i) + b\right] \geq 1 - \xi_i, \\ & \xi_i \geq 0, i = 1,...,l \end{aligned} \tag{1}$$

where $\varphi(\mathbf{x})$ is a nonlinear transformation vector from the input space to the feature space, b is a bias, w is an adjustable weight vector, and C>0 is a tradeoff constant between training error and margin for slack variables $\vec{\xi}$ allowing to the misclassification of training sample. The optimization problem is solved constructing the following dual quadratic programming:

$$\begin{aligned} \min_{\alpha} \quad & J_D(\alpha) = -\frac{1}{2}\sum_{i,j=1}^{l} y_i y_j K(x_i, x_j)\alpha_i \alpha_j + C\sum_{i=1}^{l} \alpha_i \\ st. \quad & \sum_{i=1}^{l} \alpha_i y_i = 0 \\ & 0 \leq \alpha_i \leq C, i = 1,...,l \end{aligned} \tag{2}$$

where the kernel function is expressed by $K(x_i, x_j) = \varphi^T(x_i)\varphi(x_j)$. $\alpha_i$ are Lagrange multipliers.

### 2.1 Some Limitations of SVM Classifiers

SVM classifiers have some limitations. Firstly, there is a question whether the feature space distances describe meaningful structures of the input space data. In this section, this issue is investigated for Radial Basis Function (RBF) kernels and polynomial kernels. If the distance is preserved in both spaces, then the distance in the feature space is given as a function of the distance in the input space.

In case of RBF kernel, $k(x_i, x_j) = \exp\left(-\left\|x_i - x_j\right\|^2 / 2\sigma^2\right)$, the distance in feature space is given as:

$$d(\Phi(x_i), \Phi(x_j)) = \sqrt{2 - 2\exp\left(-\frac{d(x_i, x_j)^2}{2\sigma^2}\right)}. \tag{3}$$

This implies that the distance between two observations in a RBF feature space is bounded by $\lim_{d(x_i,x_j)\to\infty} d\left(\Phi(x_i),\Phi(x_j)\right)=\sqrt{2}$ and $\sigma$ regulates the sensitivity regarding their input space distance. Hence as the distance increases, classification results in error since they approximate to each as seen by Fig. 1(a) [8-10].

On the other hand, in the polynomial kernels, $k\left(x_i,x_j\right)=\left(\left\langle x_i,x_j\right\rangle+\theta\right)^d$, the distance in the feature space depends on the absolute position in input space. In the standardized polynomial kernels, $k\left(x_i,x_j\right)=\left(\left\langle x_i,x_j\right\rangle\Big/\sqrt{\left\langle x_i,x_j\right\rangle\left\langle x_i,x_j\right\rangle}+\theta\right)^d$, the feature space distance is a function of the angle α between $x_i$ and $x_j$ in the input space

$$d\left(\Phi(x_i),\Phi(x_j)\right)=\sqrt{2(1+\theta)^d-2(\cos\alpha+\theta)^d}\ . \tag{4}$$

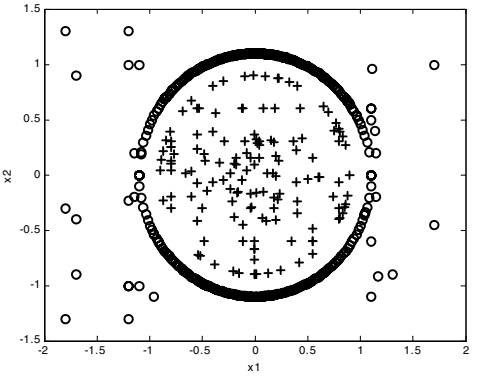



**Fig. 1.** An artificial data constructed for example

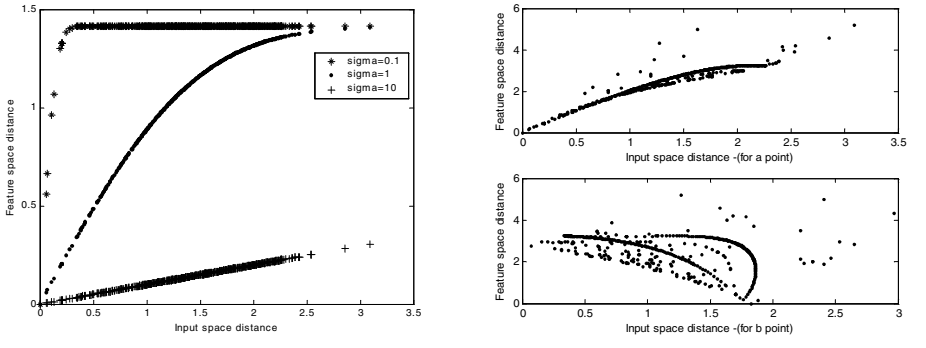



**Fig. 2.** Distance order input vs. feature space. (a) RBF kernel. (b) Polynomial kernel.

In order to show whether the distance preserve, we generated an artificial data of two-class including 494 data in Fig. 1(b). Choosing two reference points, a=[-1.1500-0.2000] and b=[0.5 0.5798], we computed the distances in input space and feature space to this points. In addition we enquired whether the distance order is preserved in the feature space. As seen from the distance to a (b) points tabulated in Table 1 (2) and the Euclidean distances to a (b) points illustrated in Fig. 2, the distance order in the input space is not preserved for a polynomial kernel. In this paper, we present a new formulation in the input space to get rid of this disadvantage. SVM has also the other limitations in that an analytical/structural point of view, the details can be found in [11].

**Table 1.** The distances to a point

| x1 | x2 | Input space | Feature space |
|---|---|---|---|
| -1.2000 | -1.0000 | 2.3207 | 3.7795 |
| 1.7000 | 1.0000 | 1.2714 | 3.8236 |
| -1.7000 | -0.4000 | 2.4083 | 4.3480 |
| -1.8000 | 1.3000 | 2.4101 | 6.0186 |
| -1.8000 | -1.3000 | 2.9705 | 6.0684 |

**Table 2.** The distances to b point

| x1 | x2 | Input space | Feature space |
|---|---|---|---|
| 0.6500 | 0.6225 | 1.9790 | 2.9706 |
| 0.8600 | -0.2653 | 2.0111 | 2.9748 |
| 0.4600 | 0.9992 | 2.0075 | 3.2122 |
| 1.1500 | -0.2000 | 2.3000 | 3.3171 |
| 0.8800 | -0.1887 | 2.0300 | 2.9759 |

## 3 Proposed formulation

A new optimization formulation for a robust classification is proposed by using ellipsoid in this study. In the proposed formulation, each class assigned with data is represented by an ellipsoid. It is aimed that the ellipse includes the data belongs to the class and the other to be excludes. For $c \in \Re^n$ and $M \in \Re^{nxn}$, the separating ellipsoid is defined as

$$\varepsilon_{M,c} = \left\{ x \in \Re^n \middle| (x-c)^T M (x-c) \leq 1 \right\}; \tag{5}$$

where c is the center of the ellipsoid and M is a symmetric positive matrix that determines its shape.

If the problem is separable, the empirical error equal to zero and the structural error is defined as 1/det(M) similar to SVM classifier. If the problem is non-separable, then misclassified data must be penalized. In this case, the empirical error is not equal to zero [8],[12]. Hence the proposed formulation is written as

$$\min_{c,M} \mathrm{E} = \frac{B}{2}\sum_{i=1}^{L} g\left(y_i.\left((x_i-c)^T M\left(x_i-c\right)-1\right)\right)+\frac{A}{2}.\frac{1}{|M|}, \tag{6}$$

where A and B parameters give the tradeoff between the volume of the optimal separation and error. An appropriate selection of these parameters is very important to minimize the objective function. Here the loss function and its derivative is chosen to be

$$\begin{array}{ll} g(\xi)=\xi & \xi>0 \\ g(\xi)=0 & \xi\le 0 \end{array} \qquad \begin{array}{ll} g'(\xi)=1 & \xi>0 \\ g'(\xi)=0 & \xi\le 0 \end{array}. \tag{7}$$

First term, if a data is outside of ellipsoid, a positive term adds to the objective function. Second term is used to find the minimum volume ellipsoid covering the data in the class. This term can be easily obtained taking into consideration the geometric interpretation of the problem. While the algorithm carries on minimization 1/(det(M), the data number excluding of ellipsoid tries to minimize in same time. In other words, even when data is not perfectly separated by ellipsoids, our formulation can find an optimal solution.

### 3.1 Hopfield Network Approach

To solve the optimization problem, we construct a Hopfield network [8]. The Hopfield neural network is fully connected continuous time network that employs units with analog input and output. The basis idea is to encode the objective function and the problem constraints in terms of an appropriate energy function, Lyapunov function that can be minimized by the network architecture. In this study, synchronously operating two Hopfield networks are constructed. The network weights to be estimated are centers, c and covariance matrixes, M of ellipsoids. The behavior of each Hopfield network is evaluated by

$$\frac{dU}{dt} = -\frac{\partial E}{\partial v}, \tag{8}$$

$$v = f(U), \tag{9}$$

where E is the energy function, U is the input neural units, and v is the output of neural units. In this work, f is chosen as linear activation function for both networks. The input of each neural unit is updated by

$$U_{new} = U_{old} - \eta\frac{\partial E}{\partial v}, \tag{10}$$

where $\eta$ represents learning rate.

The dynamics of the network is given according to the following differential equations:

$$\frac{\partial U_c}{\partial t} = B\sum_{i=1}^{L} y_i M(x_i - c) g'\left(y_i\left((x_i - c)^T M(x_i - c) - 1\right)\right) \tag{11}$$

$$\frac{\partial U_M}{\partial t} = -\frac{B}{2}\sum_{i=1}^{L} y_i (x_i - c)(x_i - c)^T g'\left(y_i\left((x_i - c)^T M(x_i - c) - 1\right)\right) + \frac{A}{2}\frac{\mathrm{M}^{-\mathrm{T}}}{|M|} \tag{12}$$

## 4 Examples

To see how the proposed formulation works we used Fisher's Iris data set. This data set contains 150 feature vectors of dimension 4, which belong to three classes representing different IRIS subspecies. Each class contains 50 feature vectors. One of the three classes is well separated from the other two, which are not easily separable due to the overlapping of their convex hulls. In the first example, we consider only two-class separable problem for two characters of the data set. In the second example, we carry out multi-class separation by reducing the problem separating classes to independent separation problems involving two classes each. We separate each class from one class consisting of the other two. We used the first 75 examples contained in the IRIS data set for the training the remaining for testing. Then we present comparatively results the classifications by SVM on the iris data.

In two-class separable example, we chosen as B=9, A=9 and learning rates: 0.08 and 0.0002 for c and M. respectively and operated algorithm for 1000 epoch. In multi-class example, we chosen as B=4, A=5 and learning rates: 0.08 and 0.0001 for c and M. respectively and operated algorithm for 4000 epoch. For SVM, we accepted as C=5000 as in [13]. We carried out the simulations in MATLAB.

For two-class separable problem, both SVM and our method result in % 100 accuracy in both test and training. On the other hand, in multi-class problem, our method yields % 94.667 accuracy in test and %94.667 in training, respectively while SVM yields %100 and %93.333 accuracy in test and training. However SVM can give

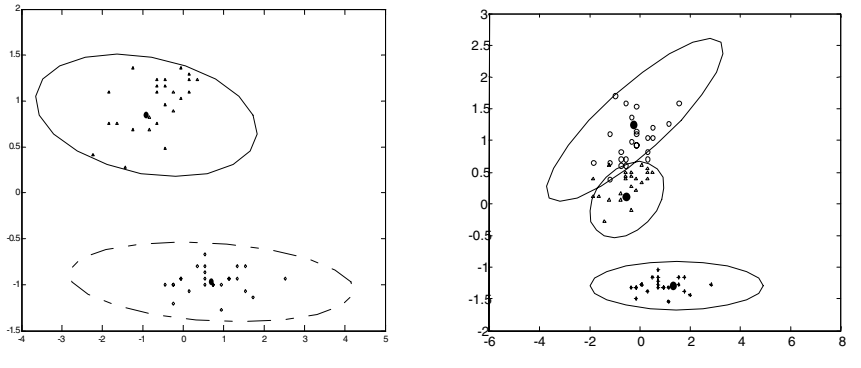


**Fig. 3.** Separating ellipsoids obtained using the proposed method. (a) Two-class problem. (b) Three-class problem.

better performance in terms of smaller margin. The performance of our method is illustrated in Fig. 3.

## 5 Conclusions

The SVM classifiers cannot preserve in the feature space the distance order in input space. In this paper we have investigated this limitation of the SVM classifier. This idea has activated us to obtain SVM like classifiers in input space. We have proposed a new optimization formulation for a robust classification in the input space. We have carried out the solution of the objective function including centers and covariance matrixes of separating ellipsoids by the Hopfield Neural Network. In particular, it is remarkable that this formulation allows us to robust classification by minimum volume ellipsoids in the input space.

## References

1. Vapnik, V.: Statistical Learning Theory. John Wiley, New York (1998)
2. Cristianini, N., Shawe-Taylor, J.: An Introduction to Support Vector Machines, Cambridge University Press (2000)
3. Zhang, B.: Is the Maximal Margin Hyperplane Special in a Feature Space? Hewlett-Packard Research Laboratories Palo Alto (2001)
4. Barnes, E.R.: An Algorithm for Separating Patterns by Ellipsoids, IBM. J. Res. Develop, Vol. 26. (1982) 759-764
5. Vandenberghe, L., Boyd S.: Applications of Semidefinite Programming. Technical Report of California University (1998)
6. Glineur, F.: Pattern Separation Via Ellipsoids and Conic Programming, Mémoire de D.E.A., Faculté Polytechnique de Mons, Mons, Belgium, September (1998)
7. Astorino, A. Gaudioso, M.: Ellipsoidal Separation for Classification Problems. Optimizations Methods and Software. Vol. 20. (2005) 267-276
8. Doğan, H.: Gradient Networks Design for Clustering in Novel Optimization Frameworks. Dokuz Eylül University, PhD. Thesis, December (2004)
9. Kruss, M.: Nonlinear Multivariate Analysis with Geodesic Kernels. Berlin Technical University, Thesis, February (2002)
10. Zhang, Z.: Learning Metrics Via Discriminant Kernels and Multidimensional Scaling: Toward Expected Euclidean Representation. ICML (2003) 872-879
11. Lyhyaoui, A., Martinez, M., Mora, I., Vaquez, M., Sancho, J.-L., Figueiras-Vidal, A.R.Sample Selection Via Clustering to Construct Support Vector-Like Classifiers. IEEE Trans. Neural Networks, Vol. 10. (1999) 1474-1481
12. Tax, D.M.J., Duin, R.P.W.: Support Vector Domain Description. Pattern Recognition Letters, Vol. 20. (1999) 1191-1199
13. Vincent, W.: SVM and Co-Training. Honc Konc Bapist University, Technical Report, (2002)

# DSP Based Fuzzy-Neural Speed Tracking Control of Brushless DC Motor

Çetin Gençer[1], Ali Saygin[2], and İsmail Coşkun[2]

[1] Fırat University Faculty of Technical Education
23100, Elaziğ
cgencer@firat.edu.tr
[2] Gazi University Faculty of Technical Education
06500 Teknikokullar, Ankara
asaygin@gazi.edu.tr, icoskun@gazi.edu.tr

**Abstract.** In this paper, Fuzzy-Neural control architecture is applied in order to construct precise speed control of brushless DC motor (BLDC) in high performance applications. BLDC motors have been becoming popular owing to high torque density, large power to weight ratio, high efficiency, high power factor, and robustness. The proposed controller is successfully implemented in real time using a digital signal processor (DSP) board DS1104 for BLDC motor. The effectiveness of the proposed controller is verified by as well as experimental results of different dynamic operating conditions. The software of system is developed in dSPACE Control Desk.

## 1 Introduction

In recent years, the Brushless DC (BLDC) motors are basically a permanent magnet synchronous motor have become a popular choice in industrial applications such as machine tool drives, computer peripherals, robotics and electric propulsion. Much of this popularity is due to the BLDC motors reduced maintenance (no brushes), superior power density, efficiency, and lower rotor inertia [1]. The model of BLDC motor is however nonlinear [2]. In order to achieve the precise BLDC motor speed control in high performance applications, an accurate modeling for BLDC motor is necessary [3].

There are some difficulties in the speed, position or torque control of BLDC motors, as they sensitive to parameter and load variations. In order to achieve high performance, the vector control of BLDC motor drive is employed. However, the controller design of such system plays crucial role in the system performance [4]. The decoupling characteristics of vector controlled BLDC motors are adversely affected by the parameter changes. Conventional controllers, PI and other controllers such as model reference adaptive controller, sliding mode controller, variable structure controller have been widely utilized as speed controllers in the BLDC motors [5]. But, conventional controllers, need accurate linear mathematical model of system. Even if the BLDC motor can be modeled, unknown parameter and load variations, nonlinearities and no modeled dynamics reduce the control performance of these controllers. Moreover, the difficulties of obtaining the exact d-q axes reactance parameters of the BLDC motors leads to cumbersome design approach for these controllers.

F.A. Savacı (Ed.): TAINN 2005, LNAI 3949, pp. 107 – 116, 2006.

The use of DSP in motor control is rapidly increasing as the price of DSPs continues to fall and the requirements for control rise. DSP's can execute sophisticated algorithms to improve motor performance in areas such as noise control, variable speed, energy efficiency and enabling the selection of less expensive motors. New low-cost industrial DSP chips such as the dSPACE 1104, which used this experimental application, have on-chip peripherals that provide all the necessary I/0 to sense and actuate the BLDC motor.

During the past decade, neural Networks (NN) and fuzzy logic control (FLC) are widely used in the control of nonlinear systems including electrical drives [3-6]. The NN has several key features that make it suitable for nonlinear system control. Neural networks can be trained to learn any continuous nonlinear function in desired accuracy and they are robust. However, long training time, difficulties on training of NN for complex nonlinear dynamic systems and determining the structure and size of the NN are the disadvantageous of the NN controllers [7,8]. FLC incorporates the expert experience of a human operator in the design of the controller in controlling a system whose input-output relationship is described by a collection of fuzzy control rules involving linguistic variables rather than a complicated dynamic model. This provides a means to incorporate human expert experience in designing the controller. The lack of systematic design procedure is the main disadvantageous of the fuzzy control. Thus, the trail and error approach is usually required in most fuzzy control applications [9].

Fuzzy-Neural systems have the benefits of neural networks and fuzzy logic systems. That is, neural networks provide connectionist structure and learning abilities to the fuzzy logic systems, and the fuzzy logic systems provide the neural networks with a structural framework with high-level fuzzy IF-THEN rule thinking and reasoning [10].

In recent years, neural network-based fuzzy systems, which utilize the learning ability of NN to realize the fuzzy logic inference system, are used in the control of nonlinear systems [9]. Various FNC structure are proposed to increase the approximation, learning and adaptation capabilities of the neuro fuzzy systems. Sugeno type Fuzzy-Neural system has been used in control applications [10,11]. However, wide variety of neuro fuzzy structures based on the selection of the type and the number of membership functions and feed forward or feedback connections between the layers are also proposed. Fuzzy-Neural control of electrical drives is an attractive and new research area. In this study, Sugeno type Fuzzy-Neural control (FNC) structure for the BLDC motors is proposed. Speed control performance of the motor under the variable loads is simulated and implemented. In this paper, BLDC motor dynamics are given in section 2 FNC structure is explained in section 3 and real time implementation FNC for the BLDC motors is given in section 4. Simulation and experimental results showing the control performance of the system under the variable load conditions are given in section 5.

## 2 Motor Dynamics

In this study, surface-mounted BLDC motor has been used. Stator windings of the motor are wye connected as shown in Fig. 1.

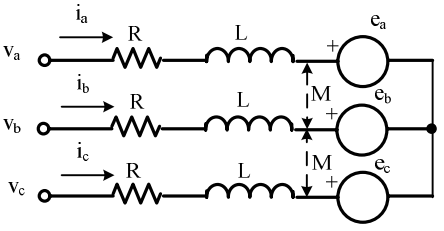



**Fig. 1.** Equivalent circuit of stator windings of PMSM

Assuming stator winding are balanced and stator phase currents in state-space form are written as follows.

$$\frac{d}{dt}\begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} = \frac{1}{L-M}\left\{\begin{bmatrix} v_a \\ v_b \\ v_c \end{bmatrix} - \begin{bmatrix} R & 0 & 0 \\ 0 & R & 0 \\ 0 & 0 & R \end{bmatrix}\begin{bmatrix} i_a \\ i_b \\ i_c \end{bmatrix} - \begin{bmatrix} e_a \\ e_b \\ e_c \end{bmatrix}\right\} \tag{1}$$

Where $e_{a,b,c}$ are Back EMF's and written as below

$$\begin{bmatrix} e_a \\ e_b \\ e_c \end{bmatrix} = -\omega_r \lambda_m \begin{bmatrix} \mathrm{Sin}(\theta_r) \\ \mathrm{Sin}\left(\theta_r - \frac{2\pi}{3}\right) \\ \mathrm{Sin}\left(\theta_r + \frac{2\pi}{3}\right) \end{bmatrix} \tag{2}$$

Torque of the PMSM is

$$T_e = -\frac{P}{2}\lambda_m\left(i_a \mathrm{Sin}(\theta_r) + i_b \mathrm{Sin}\left(\theta_r - \frac{2\pi}{3}\right) + i_c \mathrm{Sin}\left(\theta_r + \frac{2\pi}{3}\right)\right) \tag{3}$$

The equation of motion is expressed as

$$T_e = J\left(\frac{2}{P}\right)\frac{d}{dt}\omega_r + B_m\left(\frac{2}{P}\right)\omega_r + T_l \tag{4}$$

## 3 Fuzzy-Neural Controller

Various Fuzzy-Neural controller structures have been recommended [9, 10]. In the present study, Fuzzy-Neural controller which has been shown in the Fig. 2 has been used. Fuzzy-Neural control has two inputs to be error between the actual and desired

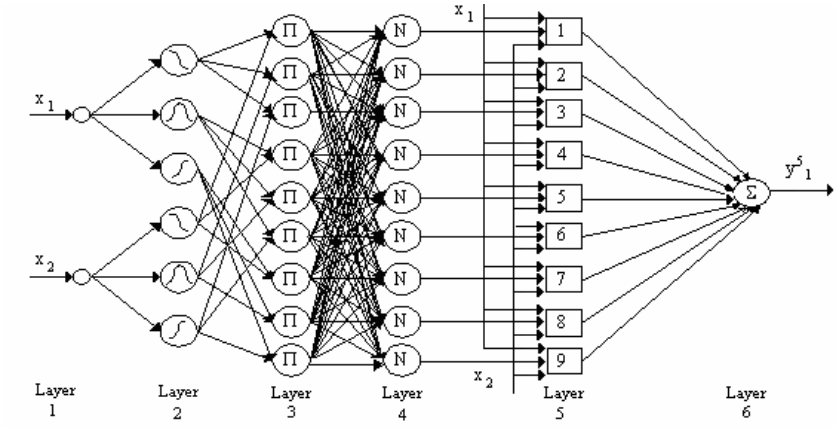



**Fig. 2.** Fuzzy-Neural controller

speed and error derivative and has one output to be quadrature components of the desired speed.

In Fig.2, in first layer of Fuzzy-Neural controller, each node of this layer is input signal, transmitted to the other layers. Any mathematical operation is not performed in this node. Second layer of Fuzzy-Neural controller includes degrees of the membership functions for input values. Membership functions were elected to be bell and sigmoid type and output of FNC $y_{ij}$ , were calculated from i input to j output as Eq. 5 [11].

$$y_{ij}^{1} = \frac{1}{1+e^{-a_{ij}(x_i^1-c_{ij})}} \quad i=1,2 \text{ ve } j=1,3$$

$$y_{i2}^{1} = \frac{1}{1+\left|\frac{x_i^1-c_{i2}}{a_{i2}}\right|^{2b_{i2}}} \tag{5}$$

Input $x_1$ shows, error and input $x_2$ shows error derivative in Fig. 2.a, b and c show parameters of membership functions trained in Eq.5.

Third layer of FNC includes the fuzzy rule base and nodes in this layer determine the fuzzy rules. Output of the any node k (shown as Figure 2) is the product of the inputs and it is calculated as,

$$y_k^2 = \prod_i y_{ij}^1 \quad k=1,2...9 \tag{6}$$

The fourth layer is a normalization layer and it computes the firing strength of the fuzzy rules. The normalized output of the any node in this layer is the division of the firing strength of the fuzzy rule k to the sum of the firing strength of the rules.

$$y_k^3 = \frac{y_k^2}{\sum_k y_k^2} \tag{7}$$

The fifth layer gives the firing strength of a rule and the output of this layer is the product of the normalized firing strength and a pre-selected function f. The fifth layer output and the function f is defined as,

$$\begin{aligned} y_k^4 &= y_k^3 f_k \\ f_k &= p_k x_1 + q_k x_2 + r_k \end{aligned} \tag{8}$$

Here, p, q and r are output parameters to be trained. Sixth layer is the output layer of FNC and it products a output by collecting inputs. Thus, output this layer gives d-q currents to be desired into the Fuzzy-Neural control of the BLDC motor.

$$y_o^5 = \sum_k y_k^4 \quad o=1,2 \tag{9}$$

To set the desired controller performance of FNC, it is necessary to adaptation of parameters obtained. Thus, FNC needs to train to neural network. Speed follow error for on-line training of FNC and reduced function according to harmony parameters could be writing as Eq. 10.

$$\begin{aligned} e(k) &= \left(\omega_r(k) - \omega(k)\right)^2 \\ E &= \frac{1}{2} e^2(k) \end{aligned} \tag{10}$$

Here $\omega_r$ shows reference speed, $\omega$ shows actual motor speed, e is the tracking error and E is the cost function to be minimized. If n learning ratio and $\theta$ trained are parameters, harmony rule parameters according to learning algorithms back propagation, defines as Eq. 11.

$$\theta(k) = \theta(k-1) + \Delta\theta(k) = \theta(k-1) + \left(-\frac{\partial E(k)}{\partial \theta}\right) \tag{11}$$

If parameter vector $\theta$ are described $\theta=[a,b,c,p,q,r]^T$ for a membership group and output function, gradient of function according to parameters defined as Eq. 12.

$$\delta^1 = \frac{\partial E}{\partial e}\frac{\partial e}{\partial \omega}\frac{\partial \omega}{\partial y^5}, \quad \frac{\partial E}{\partial p} = \delta^1 \frac{\partial y^5}{\partial p}$$

$$\frac{\partial E}{\partial a} = \delta^1 \frac{\partial y^5}{\partial y^4}\frac{\partial y^4}{\partial y^3}\frac{\partial y^3}{\partial y^2}\frac{\partial y^2}{\partial y^1}\frac{\partial y^1}{\partial a} \tag{12}$$

In the present study, delta adaptation rule which is more effective than others methods has been used and described as Eq. 13.

$$\delta^1 = Ae + \dot{e} \tag{13}$$

Here, A is positive number. As a conclusion delta parameter which is in Eq. 9 could be calculated and parameters of trained rules could be produced [12].

## 4 Real Time Implementation

In the present study, Fuzzy-Neural algorithms for speed control and PI control for current control have been used. PI current controllers are designed according to electrical dynamics of the motor. FNC uses the tracking error and the derivative of the error as inputs and gives the desired q current for the vector control of the motor. FNC is trained using the back propagation algorithm. The SIMULINK block diagram with FNC is shown in Fig.3.

Figure 4 shows the SIMULINK blocks diagram of drive system using dSPACE 1104 controller card. Six PWM outputs from the dSPACE 1104 are used to drive a 3-phase power inverter. Two current sensors on inverter legs are used to sense motor a and b

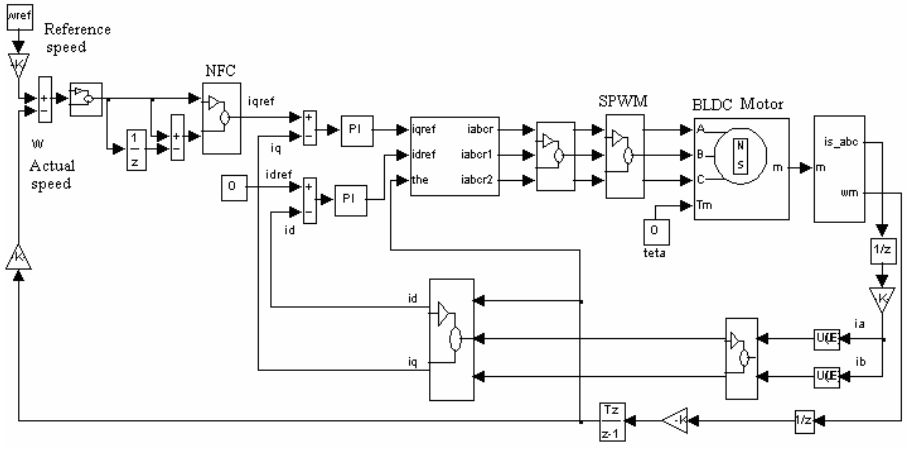



**Fig. 3.** Control architecture for Fuzzy-Neural speed control of BLDC motors

phase currents. The voltages across the current sensors are amplified and fed in the ADC input channels of the dSPACE 1104. The dSPACE 1104 samples these voltages right at the right time of a shaft of the motor for rotor angle measurement. The outputs from the encoder are interfaced directly to the controller card encoder block inputs of the dSPACE 1104. The dSPACE 1104 obtains the rotor angle by reading the QEP counter. It then uses the sampled motor currents and obtained rotor angle to implement a Vector Control algorithm to control the motor [13].

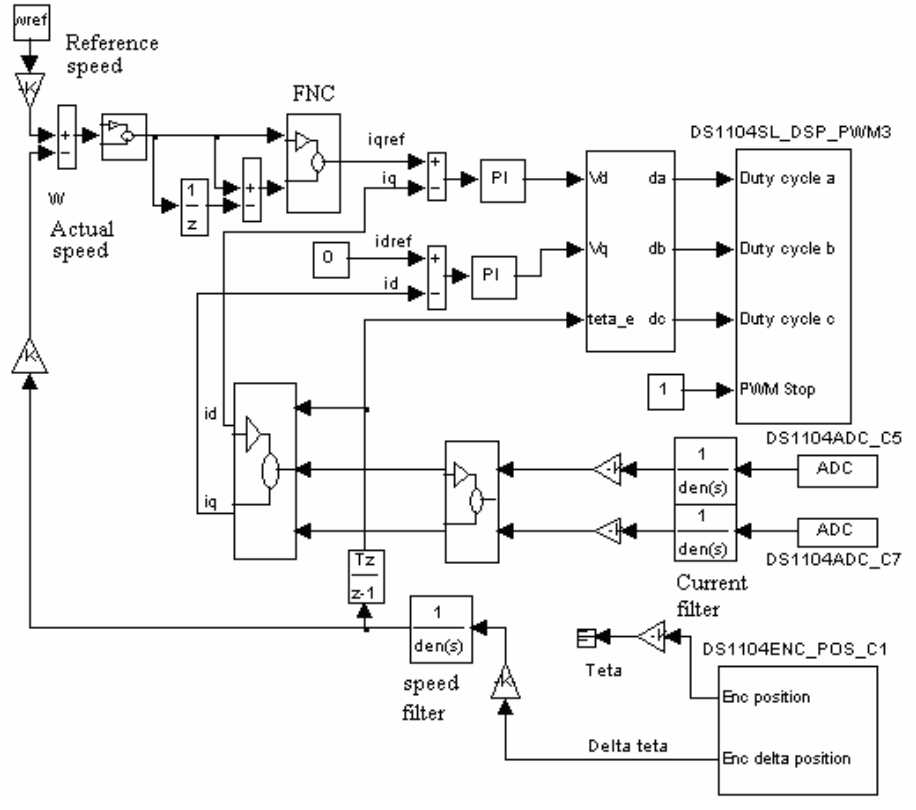



**Fig. 4.** dSPACE 1104 DSP based FNC for 3-phase BLDC motor drive

## 5 Experimental Results

Parameters of BLDC motor used in simulation and experiment are determined approximately as R=11.05 Ω, L=21.5 H, J=0.0001kg.m$^2$, P=6, $T_n$=0.9 Nm, $E_z$=39 v/1000 rpm., n=3000 d/dak., $I_n$=1.7 A.

In order to assess the speed tracking capability of the proposed FNC controller for the BLDC motor two speed commands (namely, step and sine wave) were used for experimentation. Fig. 5 shows steady-state operation speed response of FNC controller under 2000 rpm reference speed command. The rising time is 45 ms, settling time

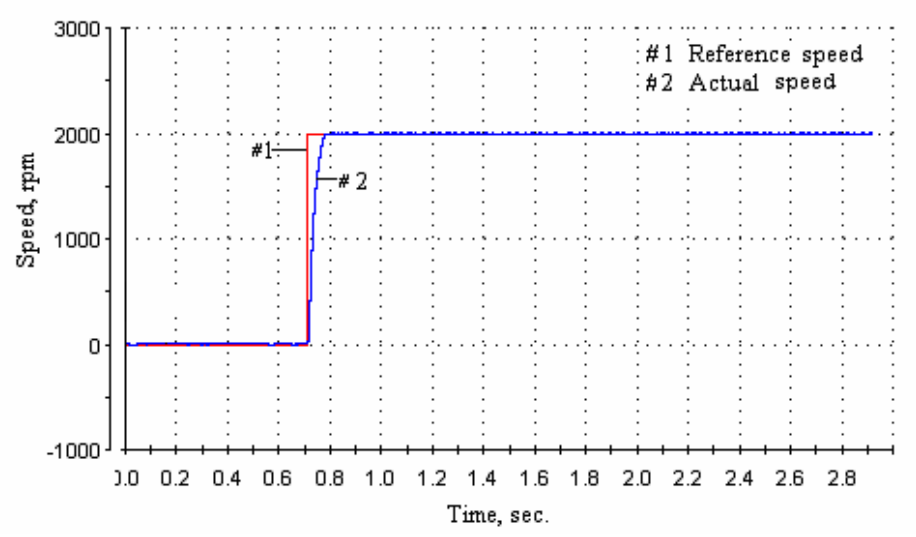



**Fig. 5.** Speed response of FNC controller at 2000 d/d step input (no load)

is 65 ms and overshoot is 0%. As seen from Fig. 6 BLDC motor tracking the reference speed smoothly.

Fig.6 shows steady-state operation speed response of FNC controller at 2000 rpm reference speed command under loading condition. As seen from Fig. 6, the rising time is 0.5 s, settling time is 0.65 s and overshoot is 0%. As seen from Fig. 7 BLDC motor tracking the reference speed under load smoothly.

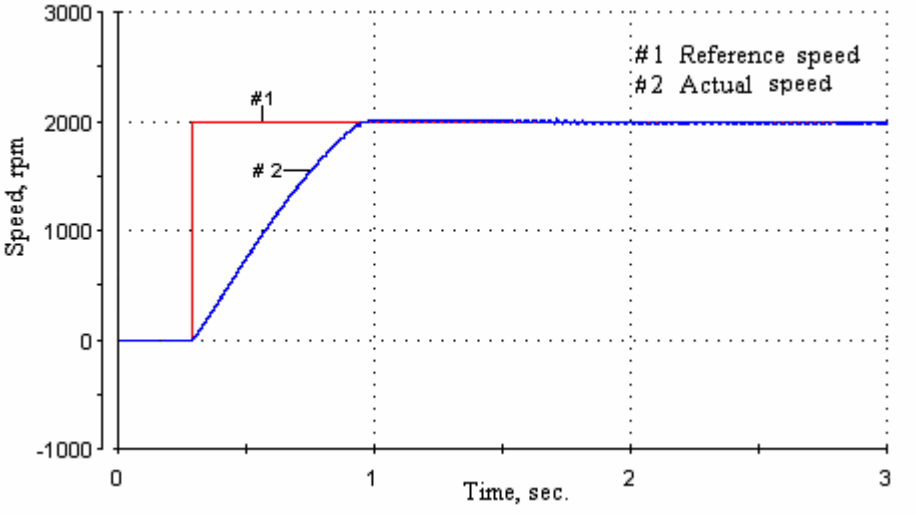



**Fig. 6.** Speed response of FNC controller at 2000 d/d step input under loading condition

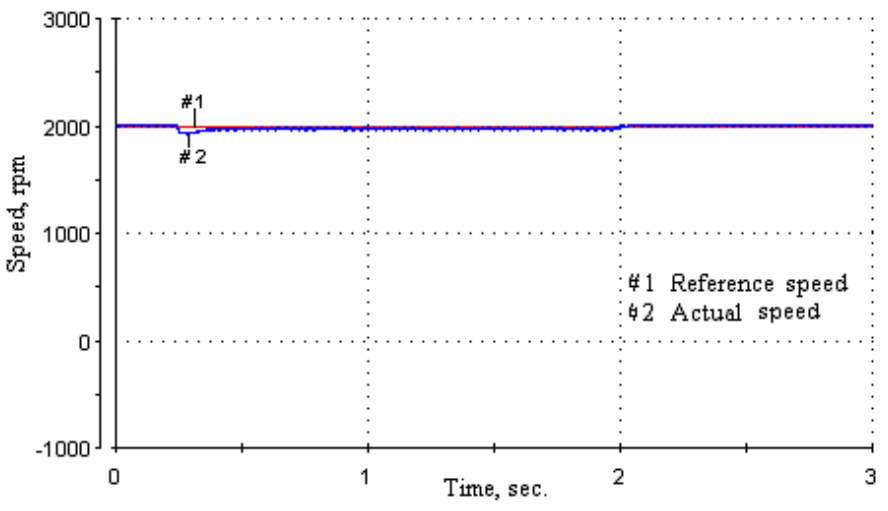



**Fig. 7.** Speed response of FNC controller at 2000 d/d step input under unexpected loading condition

In order to seen that performance of FNC controller under unexpected loading condition, nominal load torque was also applied to the motor at the times between 0.23 - 2 seconds. As seen from Fig. 7, the performance of the FNC is favorable.

In order to seen that performance of the FNC, sine reference speed was applied to BLDC motor. Fig. 8 illustrates sine reference speed and sine actual speed response of the BLDC motor. The motor is under load. High tracking accuracy is observed at Fig. 8.

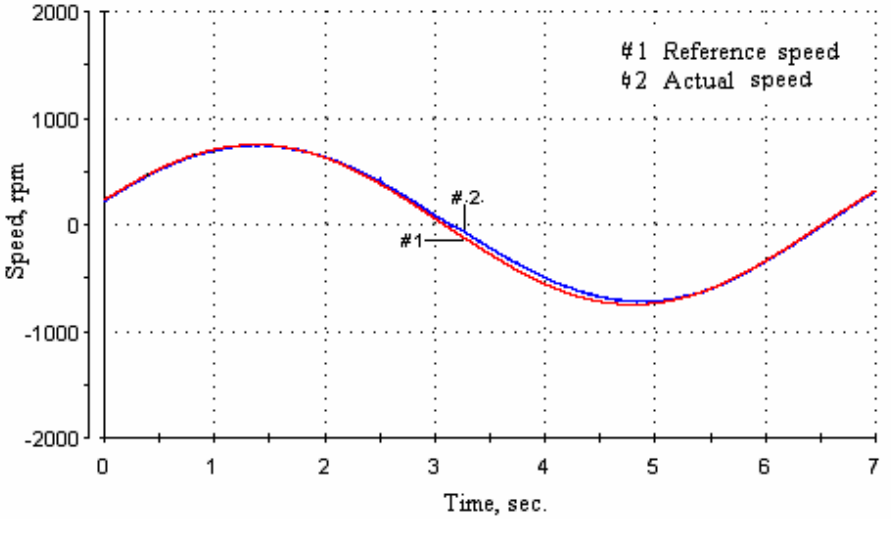



**Fig. 8.** Speed tracking performance of the BLDC motor for the sinusoidal speed reference

## 6 Conclusions

A speed control technique based on a FNC controller for BLDC motor drive has been presented in this paper. The closed loop vector control of BLDC motor drive incorporating the proposed FNC controller has been successfully implemented in real time with dSPACE 1104 DSP card. The FNC controller parameters have been optimized off-line using back propagation learning algorithm. Experimental results show the effectiveness of the control system owing to no overshoot, minimum settling time and zero steady-state error. The BLDC drive has been found robust in terms of quick response and disturbance rejection. Although the complex and nonlinear model of the BLDC motors, FNC is fairly favorable and design of the FNC does not need the exact mathematical model of the system. Main drawback of the Fuzzy-Neural systems is that it has many parameters to be trained.

## Acknowledgements

The authors acknowledge the financial support of "Fırat University Scientific Research Project Department (FUBAB)", through project number 659.

## References

1. Kim, K-H.,Young, M-J.: DSP-Based High-Speed Sensorless Control for a Brushless DC Motor Using a Link Voltage Control, Electric Power Compenents and Systems, Taylor & Francis (2002) Vol.30. 889-906
2. Rahman M.A. and Zhow P.: Analysis of Brushless Permanent Magnet Synchronous Motors, IEEE Transactions on Industrial Electronics (1996), Vol.43. 256-267
3. Lin F.J., Wai R. J. and Chen H. P.: A PM Synchronous Servo Motor Drive With an On-Line Trained Fuzzy Neural Network Controller, IEEE Transactions on Energy Conversion, (1998), Vol.13. 319-325
4. Lee C.H. and Teng C. C.: Identification and Control of Dynamic Systems Using Recurrent Fuzzy Neural Networks, IEEE Transactions on Fuzzy Systems, (2000), Vol. 8, 349-366
5. Ibrahim, Z. And Levi, E.: Fuzzy Logic Versus PI Speed Control in High-Performance AC Drives: A Comparison, Electric Power Compenents and Systems, Taylor & Francis, (2003), 31:403-422
6. Lazerini B., Reyneri L. M. and Chiaberge M.: A Neuro-Fuzzy Approach to Hybrid Intelligent Control, IEEE Transactions on Industry Applications, (1999), Vol.35, 413-425
7. Barsoum N.: Artificial Neuron Controller for DC Drive**,** IEEE Power Engineering Society Winter Meeting, (2000), Vol.1, 398 -402
8. Rubaai A., Kotaru R. and Kankam, M.D.: A Real-Time Neural Network Based Controller For Brushless DC Motor Drives., Industry Applications Conference, 1997. Thirty-Second IAS Annual Meeting, IAS '97., Conference Record of the 1997 IEEE , Vol. 2, 828 -835
9. Jang J.-S.R. and Sun C.-T.: Neuro-fuzzy Modelling and Control, Proceeding of the IEEE,(1995), Vol. 83, 378-405
10. Jang J.-S.R., Sun C.-T. and Mizutani E.: Neuro-Fuzzy and Soft Computing, Prentice Hall, (1997)
11. Lin C.T. and Lee C.S.G., Neural Fuzzy Systems, Prentice Hall, (1996)
12. Chen Y.C. and Teng C.C.: A Model Reference Control Structure Using A Fuzzy Neural Network, Fuzzy Sets and Systems, (1995), Vol. 73, 291-312
13. DS 1104 R&D Controller Board Features, dSPACE GmbH, Germany (2003)

# Fault Diagnosis with Dynamic Fuzzy Discrete Event System Approach

Erdal Kılıç[1], Çağlar Karasu[2], and Kemal Leblebicioğlu[1]

[1] Middle East Technical University, Electrical and Electronic Engineering Department
Computer Vision and Intelligent Systems Laboratory,
06531 Ankara, Turkey
erdal@eee.metu.edu.tr, kleb@metu.edu.tr
[2] Tübitak, Sage,
06531 Ankara, Turkey
ckarasu@tubitak.sage.gov.tr

**Abstract.** Determining faults is a challenging task in complex systems. A discrete event system (DES) or a fuzzy discrete event system (FDES) approach with a fuzzy rule-base may resolve the ambiguity in a fault diagnosis problem especially in the case of multiple faults. In this study, an FDES approach with a fuzzy rule-base is used as a means of indicating the degree and priority of faults, especially in the case of multiple faults**.** The fuzzy rule-base is constructed using event-fault relations. Fuzzy events occurring any time with different membership degrees are obtained using k-means clustering algorithm. The fuzzy sub-event sequences are used to construct super events. The study is concluded by giving some examples about the distinguishability of fault types (parameter, actuator) in an unmanned small helicopter.

## 1 Introduction

Today, fault detection and diagnosis are very important tasks in complex systems. There are two main approaches to a fault detection and diagnosis (FDD) problem: model based [1], [2], [17] and knowledge based [3]. A summary of these approaches is given in Willsky [1] and their developments are summarized by Isermann [2]. The extension of knowledge-based approaches (i.e., neural networks, adaptive neural networks, neuro-fuzzy systems and hybrid neuro-fuzzy systems) and design methodologies can be seen in [3], [18] and [19].

The chosen diagnosis method is important to isolate multiple faults in complex systems. If no information is available on the fault-event relation, classification methods (i.e., fuzzy clustering, artificial neural network and probabilistic methods) can be used for fault diagnosis. If more information about event-fault relations is available, different methods of reasoning (i.e., probabilistic reasoning, probabilistic reasoning with fuzzy logic and reasoning with artificial intelligence [21], [22], [24]) can be applied. When fuzzy reasoning is utilized, it is possible to present the results in the form of possibility of faults and their sizes [23]. The adaptive neuro-fuzzy systems can be used in order to improve the rule-base further [3], [18].

F.A. Savacı (Ed.): TAINN 2005, LNAI 3949, pp. 117 – 124, 2006.

Conventional DES approaches [7], [16] are used to model systems that cannot be described by differential equations or difference equations, but must be described by sequences of events that record significant qualitative changes in the state of the system. Although they have been applied in many engineering fields, they may not be adequate for fault diagnosis applications, in which the state (e.g., a component health status) is somewhat uncertain (e.g., degree of fault) and vague even in a deterministic sense [18], [19]. Furthermore, the determination of the faulty component's set could be too restrictive since users may want to identify different levels of faults. Usually, the state (healthy or unhealthy) of components, obtained from measurements, expert experience, or analysis using probabilistic schemes cannot be determined accurately. The research on the diagnostic problem for such systems with fuzziness is interesting and important. Furthermore, the transition from one state to another is also vague. It is hard to say how exactly an actuator's condition has changed from "good" to "bad".

Recently, the failure diagnosis problem has been investigated via DES approach [7]. Two basic practices for this purpose are automata theory and Petri nets. Some modeling practices using the above theories can be found in [7] and [8]. Sometimes one may need to model systems that cannot be modeled by the current DES modeling methods due to the vagueness in the definitions of the states and/or events. In order to overcome these difficulties, the concepts of fuzzy state and fuzzy event can be used [9], [10]. Lin and Ying [12] initiated the study of FDES by combining fuzzy set theory with DES to solve problems, which are not possible to be solved by conventional DES. They then applied their results about FDES on HIV/AIDS treatment planning problem [25].

In this study to solve the fault diagnosis problem, an FDES approach based on fuzzy rule-base (i.e., same other fuzzy reasoning methods) is proposed. The proposed approach then has been applied to a failure diagnosis problem in an unmanned small helicopter. In literature one can find many fault diagnosis methods about helicopters. Among them [5], [13], [20] are focused on detecting and identifying helicopter (CH-46) gearbox faults. Signal analysis techniques with pattern classification (Kalman filter approach [6], decision trees, learning vector quantization, multi-layer perceptrons, fuzzy ARTMAP, and Gaussian mixtures [20]) are used to diagnose helicopter gearbox faults. Feature vectors used to isolate helicopter faults are based on neuro-fuzzy system, reasoning [15] and signal processing approaches [13], [20] (Mean square RMS, kurtosis maximization). Two studies related to the actuator fault compensation and actuator and sensor faults in a helicopter can be seen in [14] and [6], respectively. Multiple faults also may occur in a helicopter at the same time. Hence, the FDES concept is more convenient in the investigation of this problem. It also allows one to first build each component model separately. Moreover, to construct features (events-fault relations) the k-means classification algorithm can be used since it is simple. In the literature there are a few FDD applications employing DES [7] but no FDES based application, so far.

The rest of the paper is organized as follows. In section 2, DES and FDES concepts are given. In section 3, a case study is introduced. In section 4, the results obtained are presented. Finally, in the last section, a summary of the present work and some conclusions and future studies are indicated.

## 2 Fuzzy Discrete Event Systems

State, event and event transition function values are crisp in a DES. Before an FDES structure is modelled, let's recall a model for a DES structure first.

***Definition 1:*** Discrete event systems can be modeled by a five-tuple $G = (Q, \Sigma, f, h, q_0)$ [11], where

- $Q$ is the set of states,
- $\Sigma$ is the set of events containing detectable (i.e., an event is *detectable* if it produces a measurable change in the output) and undetectable events, which are generally fired asynchronously,
- $f : Q \times \Sigma \to Q$ is the state transition function,
- $h : Q \times \Sigma \to \hat{\Sigma}$ is the output equation, where $\hat{\Sigma}$ is a set of detectable events, $\hat{\Sigma} \subseteq \Sigma$,
- $q_0$ is a $1 \times n$ ($n$: the number of places) initial state vector, whose elements are zero or 1.

In order to generalize a DES structure into an FDES structure, the concepts of *fuzzy state* and *event* are proposed in this study. We combine fuzzy set theory with (crisp) DES structure in which events and states have crisp values. In an FDES structure, events and state transition functions are fuzzy vectors whose components take values between zero and 1. All the events in FDES occur continuously at the same time with different membership degrees (i.e., events firing at the same time with different degrees); hence the system can be in many places (states) at a given instant.

If we reformulate the crisp DES into fuzzy DES we may write below the definition for FDES as given in [12].

***Definition 2:*** Fuzzy discrete event systems can be modeled by a five-tuple $G = (\overline{Q}, \overline{\Sigma}, \bar{f}, \bar{h}, \bar{q}_0)$ where,

- $\overline{Q}$ is the set of states $(q_1 \ldots q_n)$, which occur continuously with different membership degree $(\mu_{q_1}(t) \ldots \mu_{q_n}(t))$ at an instant of time, where $n$ is the number of states,
- $\overline{\Sigma}$ is the set of events $(e_1 \ldots e_m)$ containing detectable and undetectable events, all events occurs continuously with different membership degrees, $(\mu_{e_1}(t) \ldots \mu_{e_m}(t))$ at an instant of time, where $m$ is the number of events,
- $\bar{f} : \overline{Q}\,[0, T] \times \overline{\Sigma}\,[0, T] \to \overline{Q}$ is the state transition function ,
- $\bar{h} : \overline{Q} \times \overline{\Sigma} \to \hat{\Sigma}$ is the output equation, where $\hat{\Sigma}$ is a set of detectable events, $\hat{\Sigma} \subseteq \overline{\Sigma}$
- $\bar{q}_0$ is the initial state vector showing the initial membership degrees of a system states.

In definition 2, the state transition function can be defined as a matrix called the state transition matrix whose components take values between zero and 1. Moreover, the number of the state transition matrices is equal to the number of events considered in a system.

## 3 Fault Detection and Identification (FDI) Method

The FDES approach is applied to detect and diagnose multiple faults in an unmanned small helicopter. The continuous-time 6 degrees of freedom linear model of a helicopter can be described by the following dynamical state equations [4], [14]:

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) + Du(t) \end{aligned} \tag{1}$$

The state variable vector $x$ (dimension 9) is defined as:

$$x = [u \quad v \quad w \quad p \quad q \quad r \quad \varphi \quad \theta \quad \psi]^T \tag{2}$$

where $u$, $v$ and $w$ represents the longitudinal, lateral and vertical velocity (ft/sec), respectively; $p$, $q$ and $r$ represents the roll, pitch and yaw rates (rad/sec) respectively; $\varphi, \theta$ represent the roll and pitch attitude (rad); and $\psi$ is the heading (rad). There are five control inputs i.e., $u = [u_1 \quad u_2 \quad u_3 \quad u_4 \quad u_5]^T$ where $u_1$ is the lateral stick (inch), $u_2$ is the longitudinal stick (inch), $u_3$ is the collective stick (inch), $u_4$ is the pedal position (inch) and $u_5$ is the horizontal tail incidence (degree).

### 3.1 Fuzzy FDI Method

In order to perform fuzzy FDI method in an efficient way we should take into account all possible fault types occurring in an unmanned small helicopter, which means that the database to be constructed will (hopefully) contain all possible fault types (i.e., those faults that will not prevent the helicopter to continue its execution). The proposed FDI method consists of 2 stages: off-line rule extraction and on-line fault detection and identification.

#### 3.1.1 Off-Line Rule Extraction

Fig. 1 shows off-line rule extraction procedure. The system's fault model (i.e., system model including faults) and healthy model is used together to generate residuals. The k-means algorithm is applied to classify the residuals (the system parameters or some variables related to stick or percentage actuator faults are changed in a simulation program within 1 second time interval).

Many simulations have been performed to decide the number of the class centers. After examining the relation between the obtained cost value and used class centers in the k-means classification algorithm, 101 centers (classes) are chosen. These centers are called sub-events and used as rules' antecedent parts. To fuzzify sub-events, triangular and trapezoidal membership functions (or others) are used. Rules' antecedent parts

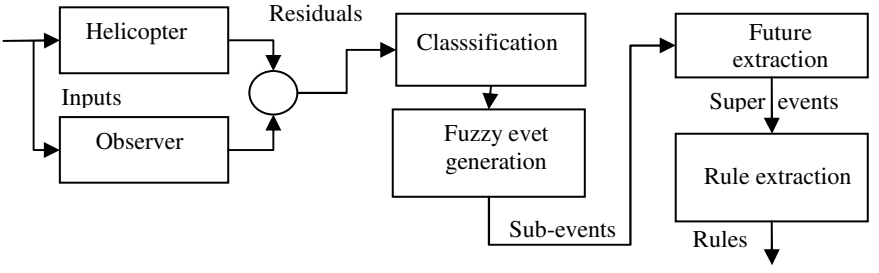



**Fig. 1.** Off-line rule extraction procedure

contain 20 sub-events (an event sequence). These event sequences are called feature vectors (super events). These super events are related to single, double and triple faults. The rule-base consists of 118 rules (the data base employed includes 118 different fault types). Super event table will not be presented here.

### 3.1.2 Off-Line Rule Extraction

Fig.2 shows on-line FDES based FDI procedure. There are 6 sensors that measure helicopter body speeds (*u, v*, and *w*) and angular velocities (*p, q*, and *r*) on real time. We used these sensor outputs to obtain residuals. In this study, two types of actuator faults called percentage and stuck actuator faults are created and used in the FDI algorithm. In our approach all events occur at the same time with different membership degrees. The sub-events membership degrees are calculated using Euclidean distance measure (the distance of measurement vectors to the predetermined class centers). Next, those, which are too small, are set to zero membership degree. The remaining sub-events are normalized among themselves. The super event generator labels residuals as sub-events and create a super event (an event sequence) by taking past 20 sub-events in time period. The membership degrees of super events are also calculated in this part. Super event membership degrees are calculated as follows: first, the last 20 (length is the same as super events) sub-event (taking place in the related super events) membership degrees are multiplied with each other. Next, those, which are too small, are set to zero membership degree. The remaining super events are normalized among themselves. The super events have a

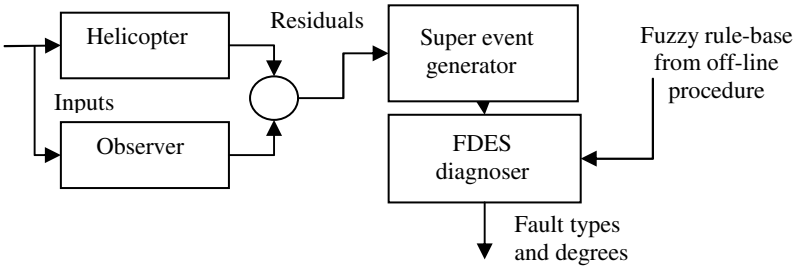



**Fig. 2.** On-line FDI procedure

dynamic structure; their membership degrees change at any time. These events are applied to the FDES diagnoser as inputs. The FDES diagnoser is constructed containing 12 fuzzy places (places show the system's faulty components related to 4 different parameters and actuators) based on rules derived using k means classification technique in off-line stage. They isolate faults and also give information about the percentage of the occurred fault types. The rule structure is given by

$$\mathbf{R^i}\textbf{: IF } e_t \ldots e_{t-N} \text{ is } A^i_1 \textbf{ THEN } (q_1 \text{ is } C^i_1, \ldots, q_m \text{ is } C^i_m), \text{ i=1,2,…,n} \quad (3)$$

Here, $N$ is the integer related to the model order and, $e_t \ldots e_{t-N}$ denote the past input vector related to the sub-events, $q_1 \ldots q_m$, show states $A^i_1$ and $C^i_1 \ldots C^i_m$ are linguistic values (labels) represented as fuzzy subsets of the respective universes of discourse. The diagnoser outputs are degrees of failure (fault percentage) related to the faulty components. This is accomplished by using the COA defuzzification.

## 4 Simulation Results

There are many system parameters in a helicopter but we dealt with only four of them. These parameters are numbered as 1, 2, 3 and 4. The parameters 1 and 2 are related to the helicopter wings and 3 and 4 are related to the helicopter main rotor and tail rotor blade. Parameter faults are restricted to percentage faults.

There are four actuators in the helicopter. All of the actuators are taken into account for fault diagnosis. There are two types of faults in the actuators. The first one is percentage fault and the other is stuck fault. Both of these fault types are considered for fault diagnosis.

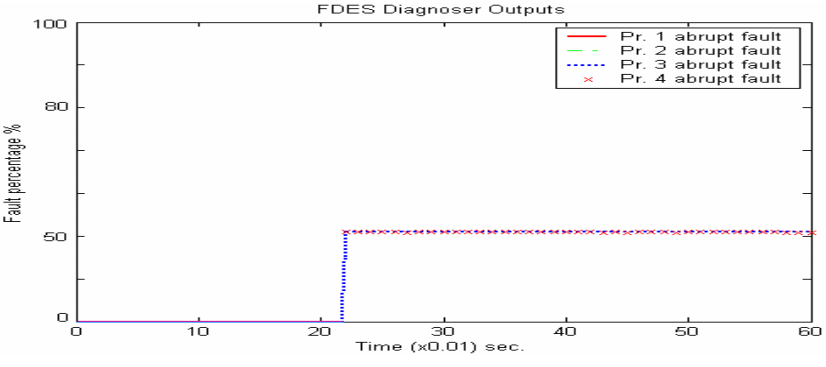



**Fig. 3.** Results obtained for the fault scenario I

The overall FDES based FDD method is tested on two different scenarios using a simulation program. In the first scenario, we created abrupt faults. There is no fault between 0 and 0.02 second. 50 % abrupt faults in parameters 3 and 4 (i.e paramaters related to the helicopter main rotor blades) are created at time 0.02 second. Figure 3 shows simulation result obtained by using the FDES diagnoser.

In the second scenario, again multiple faults are created. There is no fault between 0 and 0.02 seconds. 50 % abrupt faults in parameters 3 and 4, and 50 % stuck fault in actuator 1 (i.e., actuator begins to work normally, at the instant *t*; it sticks and remains at working condition of time *t*) are created at time 0.02 second. Figure 4 shows simulation result obtained by using the FDES diagnoser.

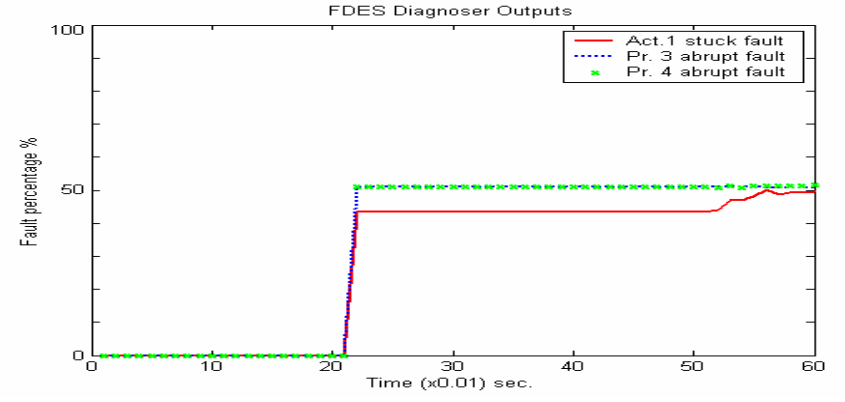



**Fig. 4.** Results obtained for the fault scenario II

## 5 Conclusion and Future Work

In this paper, a fuzzy rule based FDES approach to solve FDI problem is introduced and used in a logical way towards distinguishing multiple faults in an unmanned small helicopter. The fuzzy rule-base employed is constructed using event-fault relations. The fuzzy events are obtained using k-means clustering algorithm. Actually, the dynamic aspects of the fault diagnosis depends on the definition of "fault" events; if those events are based on time histories, then the resultant FDES is automatically a dynamical system. We are planning to tune the membership function parameters using a performance function with a genetic algorithm to improve FDI algorithm performance (fault isolation capability). Additionally, more advanced or complicated event definitions are being investigated to distinguish faults better. We believe that the approach introduced in this study will show researchers a new way to cope with the FDI problem in complex systems.

## References

1. Willsky, A.S.: A survey of design methods for failure detection systems. Automatica, vol.12. (1976) 601-611
2. Isermann, R.: Supervision, fault-detection and fault diagnosis methods-an introduction. Control Eng. Practice, vol. 5 no. 5 (1997) 639– 652
3. Ayoubi, M., Isermann, R.: Neuro-fuzzy systems for diagnosis. Fuzzy Sets and Systems, 89 (1997) 289-307

4. Karasu, Ç.: Small-sized unmanned model helicopter guidance and control. M.Sc. thesis, Middle East Technical University Ankara Turkey November (2004)
5. Vinay, B., et al.: Diagnosis of helicopter gearboxes using structure-based networks. Proc.of the American Control Conference, Seattle Washington June (1995) 1623-1627
6. Constantino, R., et al.: Failure detection and identification and fault tolerant control using the IMM-KF with applications to the Eagle-Eye UAV. Proceedings of the 37th IEEE Conference on Decision & Control, Tampa Florida USA December (1998) 4208-4213
7. Sampath, M., et al: Failure diagnosis using discrete event models. IEEE Trans. on CST., vol.4. no.2. (1996) 105-124
8. Prock, J.: A new technique for fault detection using Petri nets. Automatica, vol. 27. no. 2. (1991) 239-245
9. Zadeh, L.A.: Fuzzy sets. Inform. Control, vol. 8. (1965) 338-353
10. Santos E.S.: Max-min automata, Inform. Control, vol.13. (1968) 363-377
11. Passino, K.M., Yurkovich, S.: Fuzzy Control. Addison Wesley Longman Inc, (1998)
12. Lin, F., Ying H.: Modelling and control of fuzzy discrete event systems. IEEE Trans. on Syst. Man and Cybernetics, vol. 32. no. 4. August (2002) 408-415
13. Wang, W.: Identification of gear mesh signals by Kurtosis maximisation and its application to CH46 helicopter gearbox data. Proceeding of the 11th IEEE signal processing workshop on statistical signal processing, 6-8 August (2001) 369-372
14. Yu, X.H.: Actuator fault compensation for a helicopter model. Proceedings of IEEE Conference on Control Applications, vol. 1. 23-25 June (2003) 1372-1374
15. Amulya, K., et al.: Hybrid reasoning for prognostic learning in CBM sytems. IEEE Aerospace Conference Proceedings, vol. 6. 10-17 March (2001) 2957-2969
16. Cassandrass, C.G., Lafortune, S.: Introduction to Discrete Event Systems. Boston MA: Kluwer (1999)
17. Isermann, R., Ballě, P.: Trends in the application of model-based fault detection and diagnosis of technical processes. Control Eng. Practice, vol. 5. no. 5. (1997) 709–719
18. Fuessel, D., Isermann, R.: Hierarchical motor diagnosis utilizing structural knowledge and self-learning neuro-fuzzy scheme. IEEE Trans. on Industrial Electronics, vol. 47. no. 5. October (2000)
19. Isermann, R.: On fuzzy logic applications for automatic control, supervision and fault diagnosis, IEEE Trans. on Systems, Man and Cybernetics – Part A: Systems and Humans, vol. 28. no. 2. March (1998) 221-235
20. Wen, F., Deb S.: Signal processing and fault detection with application to CH-46 helicopter data. IEEE Aerospace Conference Proceedings, vol. 6. 18-25 March (2000) 15-26
21. Dubois, D., Prade, H.: Fuzzy set-based methods in instance-based reasoning. IEEE Trans. on Fuzzy Systems, vol. 10. no. 3. June (2002) 322-332
22. Cayrac, D., Dubois, D., Prade, H.: Handling uncertainty with possibility theory and fuzzy sets in a satellite fault diagnosis application. IEEE Trans. on Fuzzy Systems, vol. 4. no. 3. August (1996) 251-269
23. Ulieru, M.: Fuzzy logic in diagnostic decision: Probabilistic networks. Ph.D. dissertation, Technical University of Darmstadt Germany (1996)
24. Wang, H., et al.: A framework of fuzzy diagnosis. IEEE Trans. on Knowledge and Data Eng., vol. 16. no. 12. December (2004) 1571-1582
25. Ying, H., et al.: A fuzzy discrete event system for HIV/AIDS treatment planning. IEEE International Conf. on Fuzzy Systems, vol. 1. 25-29 Budapest Hungary July (2004) 197-202

# A Hybrid Neuro-Fuzzy Controller for Brushless DC Motors

Muammer Gökbulut, Beşir Dandil, and Cafer Bal

Firat U., Faculty of Technical Edu., Dep. of Electr. and Computer Science Elazig/Turkey
mgokbulut@firat.edu.tr, bdandil@firat.edu.tr, cbal@firat.edu.tr

**Abstract.** In this paper, a hybrid neuro-fuzzy controller (NFC) is presented for the speed control of brushless DC motors to improve the control performance of the drive under transient and steady state conditions. In the hybrid control system, proportional-derivative (PD) type neuro-fuzzy controller (NFC) is the main tracking controller, and an integral compensator is proposed to compensate the steady state errors. A simple and smooth activation mechanism described for integral compensator modifies the control law adaptively. The presented BLDC drive has fast tracking capability, less steady state error and robust to load disturbance, and do not need complicated control method. Experimental results showing the effectiveness of the proposed control system are presented.

## 1 Introduction

Brushless DC (BLDC) motors are widely used in most servo applications in robotics, dynamic actuation, machine tools and positioning devices, due to their favorable electrical and mechanical characteristics, high torque to volume ratio, high efficiency and low moment of inertia [1-3]. High accuracy is not usually imperative for most electrical drives, however, in high performance drive applications, a desirable control performance must be provided even when the parameters of the motor and loads are varying during the motion. Conventional constant gain controllers used in the high performance variable speed drives become poor when the load is nonlinear and, parameter variations and uncertainties exist. Therefore, control strategy of high performance electrical drives must be adaptive and robust. As a result, interest in developing adaptive control systems for electrical drives has increased considerably with in the last decade and several adaptive control schemes for brushless DC motors are proposed based on linear model [4,5].

During the past decade, a neural network-based fuzzy system called Adaptive Neuro-Fuzzy Inference System (ANFIS) has become attractive in the control of nonlinear systems [6-8]. In the field of electrical drives, fuzzy-neural network control is applied to induction motors [9] and brushless DC motors in [10], and used in [11] to update the control gain of the sliding mode position controller for an induction motor drive. Fuzzy-neural network controller is augmented with an IP controller [12], PD controller [13] and an adaptive controller [14]. Furthermore, recurrent fuzzy-neural network controller is applied to a dynamical system in [15] and a linear

F.A. Savacı (Ed.): TAINN 2005, LNAI 3949, pp. 125 – 132, 2006.

induction motor drive in [16]. In [17], PD and PI type fuzzy controllers for the direct torque control of induction motor drives are presented.

In this study, a hybrid neuro-fuzzy controller is proposed for a high performance BLDC drive. In the hybrid control system, proportional-derivative (PD) type neuro-fuzzy controller (NFC) is the main tracking controller, which is used to mimic the perfect control law, and an integral compensator is proposed to compensate the steady state errors. A simple and smooth activation mechanism described for integral compensator modifies the control law adaptively. The backpropagation algorithm is used for the training of the NFC in the direct adaptive control scheme and then, the trained NFC is used in experiments. Speed control performance of the proposed controller is evaluated under parameter and load variations of the motor using the experimental setup including PC and DS-1104 signal processing kit.

## 2 A Hybrid Neuro-Fuzzy Controller for BLDC Drives

A brushless DC machine is basically synchronous machine with a permanent magnet in the rotor circuit. The armature windings, which are mounted on stator, are electronically switched according to position of the rotor [1-3]. The state space model of a BLDC motor referred to the rotor rotating reference frame is given in Eqn.(1).

$$\frac{d}{dt}\begin{bmatrix} i_q \\ i_d \\ \omega \\ \theta \end{bmatrix} = \begin{bmatrix} -\frac{R}{L} & -\omega & -\frac{\lambda}{L} & 0 \\ -\omega & -\frac{R}{L} & 0 & 0 \\ \frac{3P^2\lambda}{8J} & 0 & -\frac{B}{J} & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} i_q \\ i_d \\ \omega \\ \theta \end{bmatrix} + \begin{bmatrix} \frac{1}{L} & 0 & 0 \\ 0 & \frac{1}{L} & 0 \\ 0 & 0 & -\frac{P}{2J} \\ 0 & 0 & 0 \end{bmatrix} \begin{bmatrix} v_q \\ v_d \\ T_L \end{bmatrix} \tag{1}$$

Where, $i_d$ and $i_q$ is the direct and quadrature components of the stator current, R is the stator resistance, L is the inductance, $\lambda$ is the magnitude of the flux linkage established by the rotor magnet and ω is the electrical rotor speed, $T_L$ is the load torque (Nm) and, J and B are inertia and friction of the motor respectively. The model is based on the assumptions that the air-gap is uniform, rotor induced current are neglected due to the high resistivity of magnet and stainless steel, and the motor is supplied by a three phase sinusoidal input source. Furthermore, BLDC motor has a surface mounted permanent magnet, and thus the d-axis inductance is assumed to be equal to the q-axis inductance. Stator currents are decomposed into the flux and torque components which can be controlled independently for the vector control (or field oriented control) of electrical drives. The output of the speed controller is the quadratic current component $i_q^*$ (torque current), and direct current component $i_d^*$ (exciting or flux current) is set to zero to avoid demagnetization of the permanent magnet on the rotor.

Block diagram of the BLDC drive is shown in Fig.(1), which consists of a BLDC motor, current controllers, PWM modulator, voltage source inverter, and the speed controller. The speed control algorithm is realized in a PC including dSPACE-1104 signal processor. Current control loops, space vector PWM modulation and coordinate transformation are also implemented by the same processor.

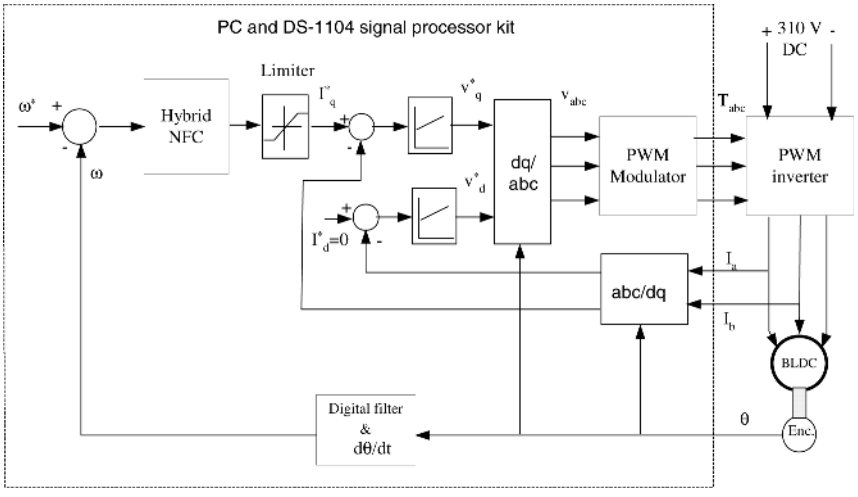



**Fig. 1.** Block diagram of the BLDC drive

*The Hybrid Neuro-Fuzzy Controller*

Using the vector control technique and assuming ideal current control, i.e., $i_q^*=i_q$ and $i_d^*=i_d$, the simplified block diagram of the BLDC drive system including the hybrid NFC can be represented as shown in Fig.(2), where, $T_e$ is the electric torque, $K_T$ is the torque constant, $i_q$ is the torque current, and plant transfer function is $G_p(s)=1/(Js+B)$.

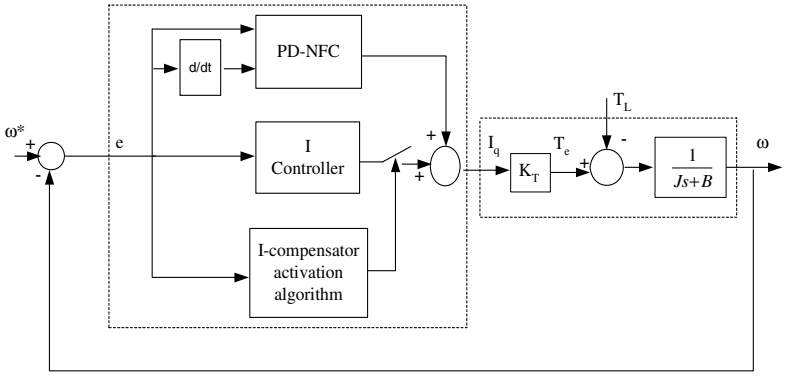



**Fig. 2.** The hybrid neuro-fuzzy controller for the BLDC drive

Neuro-fuzzy controller based on Mamdani fuzzy model is adopted for the speed control of brushless DC motors in this study. PD type NFC has favorable transient response characteristics; however, significant steady state error occurs when the load torque is applied since it has no integral mechanism. Steady state errors can be reduced by long training period for the NFC; however, this may result in high gains

which cause noise problems in the control system. Drawbacks of the PD-NFC can be overcome with the PI-NFC which includes an integral compensator at the output. Steady state response of the PI-NFC is acceptable; however, tuning of the integral gain has a considerable effect on the control performance such as overshoot and settling time. The hybrid NFC is proposed in this paper to cope with these issues, which consists of proportional-derivative (PD) type neuro-fuzzy controller, and an integral compensator activated in steady state region to compensate the steady state errors. The inputs are the speed tracking error $e$ and the change of error $\Delta e$ for the PD-NFC and error $e$ for the I-compensator. The output is the sum of the outputs of two sub-controllers as $i_q = i_{qNFC} + i_{qI}$.

A simple activation mechanism depending on the tracking error is provided. The activation mechanism modifies the control law adaptively and thereby achieving high performance control for both transient and steady state. The activation mechanism is as follows:

$$\begin{cases} PD-NFC \quad is\ activated \quad if \quad |e| > \omega_{thr} \\ PD-NFC\ \ plus\ \ I-compenasator\ is\ activated \quad if \quad |e| \le \omega_{thr} \end{cases}$$

The threshold value of the speed is chosen 10% of the reference speed for this study. The proposed controller is implemented experimentally under various load conditions.

## 3 Architecture of Neuro-Fuzzy Controllers

A four-layer neuro-fuzzy system based on Mamdani fuzzy model is adopted to implement the neuro-fuzzy controller in this study. For the Mamdani NFC a common rule set with two fuzzy if-then rules is the following:

*If* $x_1$ *is* $A_1$ *and* $x_2$ *is* $B_1$, *then* $\omega_1$

*If* $x_1$ *is* $A_2$ *and* $x_2$ *is* $B_2$, *then* $\omega_2$

Input nodes bypass input signals to the next layer, i.e., $y_i^1 = net_i^1$, where $net_i^1$ is the *i*th input to node of layer 1, which are speed error $x_1 = e$, and the change of the error $x_2 = \Delta e$.

The second layer calculates the degree of membership functions for the input values. The Gaussian activation function is utilized to represent the membership functions. The weights between the input and membership layer are assumed to unity. The output of this layer is expressed as,

$$net_j^2 = -\frac{(x_i - m_{ij})^2}{2(\sigma_{ij})^2}, \quad y_j^2 = \exp(net_j^2) \tag{2}$$

where, $\sigma_{ij}$ and $m_{ij}$ are the standard deviation and mean of the Gaussian function in the *j*th term of the *i*th input node. The third layer of the NFC includes the fuzzy rule base and the nodes in this layer represented by $\Pi$ determine the fuzzy rules. Each

node takes two inputs, one from the membership value for the speed error, and the other from the membership value for the change in speed error. For the *k*th rule node,

$$net_k^3 = \prod_j w_{jk}^3 x_j^3 \ , \ y_k^3 = net_k^3 \tag{3}$$

where, $x_j^3$ represents the *j*th input to the node of rule layer, and $w_{jk}^3$ is assumed to be unity. The output layer acts a defuzifier. The single node in this layer collects all incoming signals from the rule layer to obtain the final results

$$net_o^4 = \sum_k w_{ko}^4 y_k^3 \tag{4}$$

where the links weights $w_{ko}^4$ represent the output action of the *k*th rule. The output of the system using the central de-fuzzification for the Mamdani fuzzy model is given as

$$y_o^4 = \frac{net_o^4}{\sum_k y_k^3} \tag{5}$$

The neuro-fuzzy controller can adjust the fuzzy control rules by modifying the output weights. The parameters of the membership functions and the output weights of NFC controller are modified using the backpropagation algorithm to minimize the performance index E

$$E = 0.5.e^2 \tag{6}$$

where the speed tracking error is $e = \omega_* - \omega$. $\omega_*$ is the reference speed and $\omega$ is the actual shaft speed. The parameters $w_{ko}$ in Egn.(4) can be modified as

$$w_{ko} = w_{ko} - \eta \frac{\partial E}{\partial w_{ko}} \tag{7}$$

where the $\eta$ *is* the learning rate. The gradient of the performance index can be derived as follows:

$$\frac{\partial E}{\partial w_{ko}} = -e.\mathrm{sgn}(\frac{\Delta \omega}{\Delta y_o^4}) \frac{\partial y_o^4}{\partial net_o^4} \frac{\partial net_o^4}{\partial w_{ko}} = \delta \frac{y_k^3}{\sum_k y_k^3} \tag{8}$$

where, $\partial \omega / \partial y_o^4$ should be calculated using the motor dynamics. Using equations from (2) to (5), the gradient of the performance index for the parameters of the membership functions can be derived as

$$\frac{\partial E}{\partial m_{ij}} = -e.\mathrm{sgn}( \frac{\Delta \omega}{\Delta y_o^4} ) \frac{1}{\sum_k y_k^3} w_{ko}^4 \frac{x_i - m_{ij}}{(\sigma_{ij})^2} y_j^2 \tag{9}$$

$$\frac{\partial E}{\partial \sigma_{ij}} = -e.\mathrm{sgn}( \frac{\Delta \omega}{\Delta y_o^4} ) \frac{1}{\sum_k y_k^3} w_{ko}^4 \frac{(x_i - m_{ij})^2}{(\sigma_{ij})^3} y_j^2 \tag{10}$$

## 4 Experimental Results

The neuro-fuzzy control system proposed in this study is implemented using the dSPACE-DS1104 signal processor control kit under the various load conditions and mechanical parameter variations of the motor. dSPACE-DS1104 control card allows user to construct the system in MATLAB/Simulink and then to convert the model files to real-time codes using the Real-Time Workshop of the MATLAB/Simulink and Real-Time Interface (RTI) of the dSPACE-DS1104 control card. Neuro-fuzzy controllers are trained using the simulation model which is verified by the experimental data obtained from the motor, and then the trained neuro-fuzzy controllers are used in experiments. Real time values of the physical systems' variables can be assigned to the user defined variables using the dSAPCE-Control Desk Developer (CDD) software. Thus the graphical user interface can be designed by the user, to observe the real time values of the variables or to change the input variables such as reference speed.

Some experimental results are provided to demonstrate the effectiveness of the proposed hybrid neuro-fuzzy controller and the control performance is tested under different operating conditions of the motor. To test the loads characteristic of the controller, a DC generator which produces torque proportional to the speed is coupled to the motor. Results are obtained by the designed graphical user interface and can be saved as data files for further analysis. Due to the physical limitations of the hardware, a limiter is added to limit the torque current command (control effort) in experiments. Some selected results are given in Fig.(3)-(5).

First, inertia and friction of the motor are increased approximately thirteen and six times of the nominal inertia and friction. The experimental results given in Fig.(3) displays the motor speed and torque current for the hybrid NFC. As shown in Fig.3.a, an excellent control performance is obtained from the hybrid NFC which provides no overshoot, no steady state error and fast response. Fig. 3.b shows that the current regulation performance of the current controller is acceptable and, the noise in torque currents is less than the derivative based controllers.

In the second experiment, the controllers are tested under increased inertia and friction with the speed dependent load produced by the DC generator. The maximum

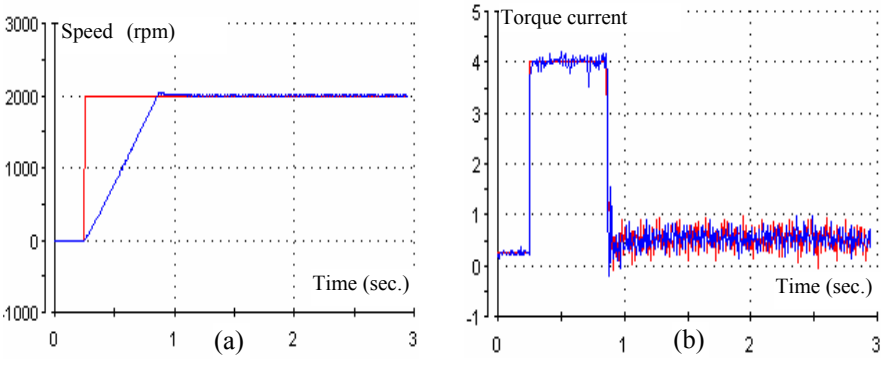



**Fig. 3.** Experimental results of the hybrid NFC for increased inertia and friction. (a) rotor speed, (b) torque current.

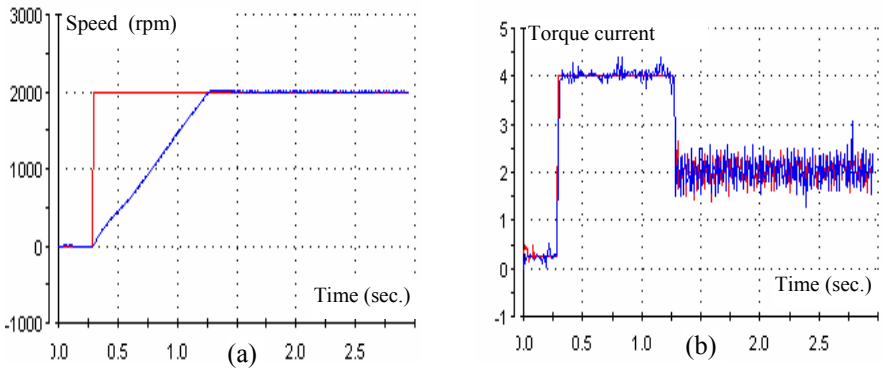



**Fig. 4.** Experimental results of the hybrid NFC for the 0.9 p.u. load conditions with increased inertia and friction. (a) rotor speed, (b) torque current.

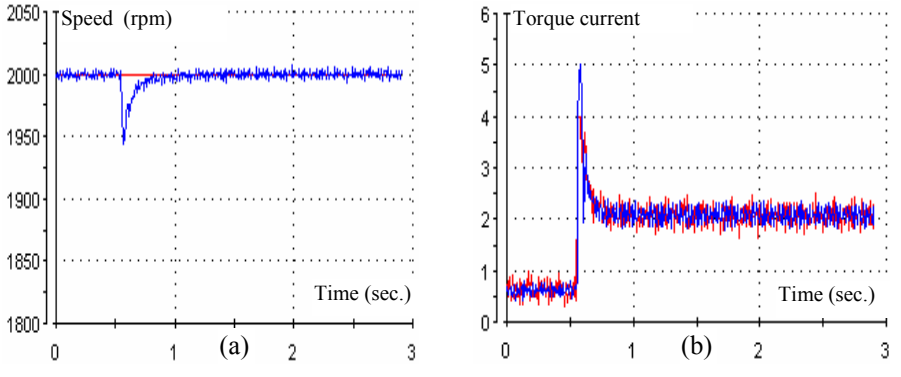



**Fig. 5.** Experimental results of the hybrid NFC for the 0.9 p.u. step load disturbance with increased inertia and friction (a) rotor speed, (b) torque current

value of the load is 90% of the nominal value, which is equivalent to torque current of 2 A. The speed tracking response and the associated torque current are shown in Fig.(4.a) and (b) respectively. As the system is loaded, rise time is increased compared to Fig. (3). An excellent control performance is obtained again, which provides no overshoot and no steady state error.

Finally, Fig.(5.a) shows the speed tracking performance of the controller when 90% load disturbances is applied and, Fig.(5.b) shows the associated torque current. For clarity, the speed response is zoomed. It can be seen from the figures that load disturbance is compensated in a short time.

## 5 Conclusions

In this paper, a hybrid neuro-fuzzy controller is proposed for a BLDC drive to improve the control performance of the drive system at transient and steady state

conditions. From the experimental results, favorable control characteristics are obtained using the proposed hybrid neuro-fuzzy controller which is an alternative approach for neuro-fuzzy control. The effectiveness of the proposed control system is shown experimentally under the parameter and load variations.

## References

1. Rahman, M.A., Zhow, P.: Analysis of brushless permanent magnet synchronous motors. IEEE Transactions on Industrial Electronics, Vol. 43. (1996), 256-267.
2. Boldea, I., Nasar, S.A.: Vector Control of AC Drives, CRC Pres, New York, (1992).
3. M. Gokbulut, Adaptive control of brushless DC motors using neural networks, PhD Thesis, Erciyes University Kayseri, (1998).
4. El-Sharkawi, M. A.: Development and implementation of high performance variable structure tracking control for brushless motors. IEEE Trans. on Energy Conversion, Vol. 6. (1991), 114-119.
5. El-Samahy, A.A., El-Sharkawi, M. A. and Sharaf, S.M.: Adaptive multi-layer self-tuning tracking control for DC brushless motors. IEEE Trans. on Energy Conversion, Vol. 9. (1994), 311-316
6. Da, F. and Song, W.: Fuzzy neural networks for direct adaptive control. IEEE Transactions on Industrial Electronics, Vol. 50. (2003), 507-513
7. Lazerini, B., Reyneri, L.M., Chiaberge, M. : A neuro-fuzzy Approach to hybrid intelligent control. IEEE Transactions on Industry Applications, Vol. 35. (1999), 413-425.
8. Chen, Y.C. and Teng, C.C.: A model reference control structure using a fuzzy neural network. Fuzzy Sets and Systems, Vol. 73. (1995), 291-312.
9. Dandil, B.: Robust speed control of induction motors using neuro-fuzzy controllers, PhD Thesis. Firat University Elazig, (2004)
10. Rubai, A., Ricketts, D. and Kanham, D. : Development and implementation of an adaptive fuzzy-neural network controller for brushless drivers, IEEE Transaction on Industry Applications, Vol. 38. (2002), 441-447.
11. Wai, R.J. and Lin, F.J.: Fuzzy neural network sliding mode position controller for induction motor drive, IEE Proc. Electrical Power Appl. Vol. 146. (1999), 297-308.
12. Lin, F.J., Wai, R.J., Chen, H.P.: A PM synchronous servo motor drive with an on-line trained fuzzy neural network controller, IEEE Transactions on Energy Conversion, Vol. 13. (1998), 319-325
13. Er, M J. and Gao, Y.: Robust adaptive control of robot manipulators using generalized fuzzy neural networks, IEEE Transactions on Industrial Electronics, Vol. 50. (2003). 620-628
14. Lin, F.J. and Wai, R.J.: Adaptive fuzzy-neural network control for induction spindle motor drive, IEEE Trans. on Energy Conversion, Vol. 17. (2002),507-513
15. Lee, C.H. and Teng, C.C.: Identification and control of dynamic systems using recurrent fuzzy neural Networks. IEEE Transactions on Fuzzy System. Vol. 8. (2000) 349-366
16. Lin, F. J. and Wai, R. J.: Hybrid control using recurrent fuzzy neural networks for linear induction motor servo drive. IEEE Transactions on Fuzzy system. Vol. 9. (2001) 102-115
17. Lai, Y.S. and Lin, J.C.: New hybrid fuzzy controller for direct torque control induction motor drives. IEEE Transactions on Power Electronics. Vol. 18. (2003) 1211-1219

# Can a Fuzzy Rule Look for a Needle in a Haystack?

Akira Imada

Brest State Technical University
Moskowskaja 267, 224017 Brest, Republic of Belarus
akira@bsty.by

**Abstract.** This paper reports a snapshot of our on-going experiments in which a common target we call *a-tiny-island-in-a-huge-lake* is explored with different methods ranging from a data-mining technique to an artificial immune system. Our implicit interest is a network intrusion detection, and we assume data floating in the *huge lake* are normal while ones found on the *tiny island* are abnormal. Our goal here is twofold. One is to know (i) *whether or not it is possible to train a system using just normal data alone.* The other is to study (ii) *a limit of the size of the detectable area*, when we decrease the size of the island eventually shrinking to zero, equivalently so-called *a-needle-in-a-haystack* which is still an open and worth while tuckling problem. To learn these two issues, a fuzzy rule extraction system – one with fixed triangle/trapezoid membership functions for our first goal, and for the second goal with Gaussian membership functions whose shape is adaptively determined for the second goal, are exploited in this paper.

## 1 Introduction

> *A sultan has granted a commoner a chance to marry one of his 100 daughters by presenting the daughters one at a time letting him know her dowry that had been defined previously. The commoner must immediately decide whether to accept or reject her and he is not allowed to return to an already rejected daughter. The sultan will allow the marriage only if the commoner picks the daughter with the highest dowry. — "Sultan's Dowry Problem"* [1]

In real world, we have many problems in which it is easy to access to any one of the many candidate solutions which could be the true solution but most likely not, which we don't know in advance.

The ultimate extreme is called *a-needle-in-a-haystack* problem. The needle originally proposed by Hinton & Nowlan [1] was exactly the one configuration

[1] According to the author(s) of the web-page of Cunningham & Cunningham, Inc. (http://c2.com) the problem was probably first stated in Martin Gardner's Mathematical Recreations column in the February 1960 issue of The Scientific American. To explore the problem more in detail, see, e.g., http://mathworld.wolfram.com. We thank Mariusz Rybnik at University Paris XII for suggesting that the problem is reminiscent of our context.

F.A. Savacı (Ed.): TAINN 2005, LNAI 3949, pp. 133–140, 2006.

of 20 binary bits. In other words, the search space is made up of $2^{20}$ points and only one point is the target. No information such as how close is a currently searching point to the needle.

Yet another problem, *a-tiny-flat-island-in-a-huge-lake* — this is a problem we came across when we had explored a fitness landscape defined on all the possible synaptic weight values of a fully-connected spiking neurons to give them a function of associative memory [2]. To simplify it we formalized the problem in more general form as follows.

**Testfunction 1 (A tiny flat island in a huge lake).** [2] *Find an algorithm to locate a point in the region A all of whose coordinates are in* $[-a, a]$ $(a < 1)$ *in an universe of the n-dimensional hypercube all of whose coordinate* $x_i$ *lie in* $[-1, 1]$ $(i = 1, c, n)$.

Our implicit interest is a network intrusion detection where we usually do not know what does an illegal transaction pattern look like until it completes the intrusion when actually it is too late. Hence, our interest is to train the intrusion detection system only using legal patterns. From this context, we assume data floating in the *lake* are normal while those found on the *island* are abnormal.

In this paper, we approach the problem from this view point. That is, we take it, more in general, just a pattern classification problem, but under the constraint that we have two classes one of which includes an extremely few patterns while the other includes an almost infinite number of patterns. Or, we might as well take it a task of discrimination of a few of non-self cells as anomaly patterns from enormous amount of self cells which represent normal patterns.

We have so far exploited the following lately reported approaches: (i) *artificial immune system* approach, especially a *negative selection* algorithm in which constant or variable sized hyper-sphere detectors detect non-self cells (see, e.g. [3]); (ii) *immuno-fuzzy* approach where a set of fuzzy rules is designed to cover non-self region (see, e.g. [4]); (iii) *evolutionary computation* approach where a set of detectors randomly created at the beginning eventually evolves to detect non-self; and so on.

In this paper, we study *a fuzzy rule extraction using a neural network* proposed by Castellano et al. [5]. The system they proposed were neatly described in the paper, and it seems to be very sound and efficient, except for the way in that their normal/abnormal data are used to train and test the system. They employed an *Iris-flower-database* in a popular public domain. The database contains three different classes of iris family and one class is assumed to be self whilst the other two are assumed to be non-self. The training samples are chosen at random from these two classes and train the system. Then system is tested using the rest of the data in the database. The result was successful. We, however, doubt the real applicability of the idea of using artificial data set in such a way, at least in a context of intrusion detection. This is principally because of the following

[2] It is not necessarily to be said for the top of the island to be "flat", but the originally this was a test-bed for evolutionary computations, and the fitness of the island region is one and zero in a lake region, that is why.

two reasons: (i) we don't know what does a non-self datum look like until it completes its intrusion successfully; and (ii) the number of non-self (anomaly) data available is extremely fewer than the number of self (normal) data.

Hence our current interest is also two-fold. Firstly, (i) training should be made only by self data; and secondly, (ii) the non-self region should be tiny. We explore these two points using two different fuzzy models.

## 2 Methods

Two fuzzy models are as follows.

### 2.1 Preliminary Experiment — Immuno-Fuzzy Model

A set of fuzzy rules is used to cover the non-self patterns. As already mentioned, self/non-self sells are represented by $n$-dimensional real valued vectors each of whose coordinate lies in $[-1, 1]$. That is, the self/non-self space is $[-1, 1]^n$, and a self/non-self pattern is represented by a vector $(x_1, \cdots, x_n)$ where $x_i \in [-1, 1]$. Then a fuzzy rule to detect non-self patterns is

$$\text{If } x_1 \text{ is } T_1, \cdots, \text{ and } x_n \text{ is } T_n \text{ then } \mathbf{x} \text{ is non-self}$$

where $T_i$ is a fuzzy linguistic terms which is either of

$$\{\text{Low, Low-Middle, Middle, Middle-High, or High}\}.$$

Each of $T_i$ maps the $x_i$ to a real value between 0 and 1, expressing the degree to how it is likely to the linguistic value. This is calculated by a membership function, which is defined here using fixed shaped triangular and trapezoidal fuzzy membership functions.

Then a genetic algorithm evolves these fuzzy rules, with chromosomes being $(T_1, \cdots, T_n)$, starting with those chromosomes randomly created. To be more specific, our chromosome is made up of $n$ integer genes whose value is chosen from $\{0, 1, 2, 3, 4\}$. The fitness of a rule is evaluated by applying the rule to all the self patterns $\mathbf{x} = (x_1, \cdots, x_n)$, one by one, and calculated as

$$\text{fitness}(R) = 1 - \max_{\mathbf{x} \in Self} \{ \min_{i=1,\cdots,n} \{\mu_{T_i}(x_i)\}\}$$

which implies how the rule covers the non-self space.

### 2.2 A Fuzzy Neural Network Approach

The goal is to classify the data taken from the $n$-dimensional data-set into either of the pre-defined $m$ classes. For the purpose, Castellano et al. [5] used the inference mechanism of the zero-order Takagi-Sugeno fuzzy model; then realized the idea by a fuzzy neural network model. To train the fuzzy neuronal network, they employed a combination of (i) *a competitive learning* to determine the architecture of the fuzzy neural network at first and (ii) *a gradient descent*

*learning* to optimize the synaptic weights afterwards. We, on the other hand, employ an evolutionary computation technique to train the network, since we already know the optimal network structure under our current interest, and as such, our concern is just to obtain the solution of weight configuration of the fuzzy neural network.

In the following three sub-subsections, Takagi-Sugano fuzzy model, a realization of the model by fuzzy neural network, and how we optimize the weight of the fuzzy neural network by an evolutionary computation are described more in detail.

**Takagi-Sugeno Model.** Though Castellano et al. [5] stated the method very clearly in their paper, let us briefly describe it with an intention of making this paper self-contained. Takagi-Sugeno fuzzy inference model is made up of a set of $H$ rules, such as

$$R_k\text{: IF } (x_1 \text{ is } A_1^k) \text{ and } \cdots \text{ and } (x_n \text{ is } A_n^k)$$
$$\text{THEN } (y_1 \text{ is } \nu_{k1}) \text{ and } \cdots \text{ and } (y_m \text{ is } \nu_{km})$$

where $R_k$ is the $k$-th rule ($k = 1, \cdots H$), $x_i$ denotes the $i$-th variable of the input data ($i = 1, \cdots, n$), $y_j$ denotes the $j$-th output variable ($j = 1, \cdots, m$), $A_i^k$ denotes a fuzzy set which is usually expressed by a linguistic term such as *"Medium-Large"* but here expressed by a shape of membership function defined one by one on the corresponding input variable, and $\nu_{kj}$ denotes a fuzzy singleton each defined on the output variables indicating the likeliness of how the input belongs to the $j$-th class according to the $k$-th rule.

$A_i^k$ is defined by Gaussian membership functions

$$\mu_{ik}(x_i) = \exp\{-(x_i - w_{ik})^2/\sigma_{ik}^2\}.$$

Then defuzzification for an input $\mathbf{x}^0 = (x_1^0, \cdots, x_n^0)$ is via the equation:

$$y_j^0 = \{\sum_{k=1}^{H}(\mu_k(\mathbf{x}^0) \cdot \nu_{kj})\}/\sum_{k=1}^{H}\mu_k(\mathbf{x}^0)$$

where

$$\mu_k(\mathbf{x}^0) = \prod_{i=1}^{n}\mu_{ik}(x_i^0)$$

is the results of application of the Larsen product operator.

In other words, the procedure of inference is as follows. When an input $\mathbf{x} = (x_1, \cdots x_n)$ is given, each of the $H$ rules evaluates the $\mathbf{x}$ and output the likeliness of the class, from one class to the next, to which $\mathbf{x}$ belongs to. The evaluation by $k$-th rule of $x_i$ is by the corresponding membership function $\mu_{ik}(x_i)$ which is specified by giving two parameters $w_{ik}$ and $\sigma_{ik}$ so that it returns a value ranging from 0 to 1. See, e.g., Fig. 1 where the $i$-th coordinate of the input $\mathbf{x}$ is evaluated by $A_i^k$, the $i$-th antecedent of the *IF* part of the Rule$_k$, which is represented by a

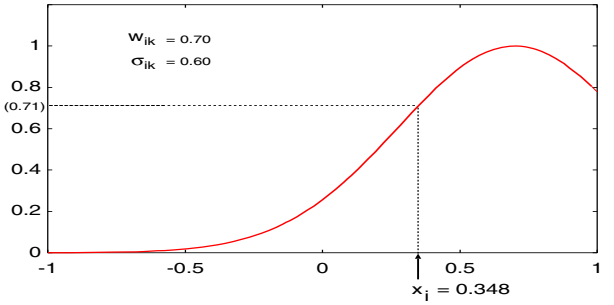



**Fig. 1.** A fictitious sketch of an evaluation of $x_i$, the $i$-th entry of the input $\mathbf{x}$, by the $i$-th antecedent part of the $k$-th rule $A_i^k$

membership function not by a usual linguistic term like *"Small"*. The returned membership value in this example in the figure is 0.71, suggesting, say, "The likeliness of if the variable is "Medium Large" is 0.71."

Using those $n$ values of $\mu_{ik}(x_i)$, each of the $H$ rules calculates $\mu_k(\mathbf{x})$, and finally these $H$ values are combined to calculate $m$ values of $y_j$, the resultant defuzzified value for each of the $m$ classes.

**Fuzzy Neural Network Implementation.** The procedure described in the previous sub-subsection can be realized when we assume a neural network architecture such as depicted in Fig. 2. The 1st layer is made up of $n$ input neurons. The 2nd layer is made up of $H$ groups of a neuronal structure each contains $n$ neurons where the $i$-th neuron of the $k$-th group has a connection to the $i$-th neuron in the 1st layer with a synaptic connection which has a pair of weights $(w_{ik}, \sigma_{ik})$. Then $k$-th group in the second layer calculates the value $\mu_k(\mathbf{x})$ from the values which are received from each of the $n$ neurons in the first layer. The 3rd layer is made up of $m$ neurons each of which collects the $H$ values from the output of the second layer, that is $j$-th neuron of the 3rd layer receives the value from $k$-th output in the second layer with the synapse which has the weight $\nu_{kj}$.

**How It Learns?** Castellano et al. [5] used (i) a competitive learning to determine how many rules are needed under initial weights created at random. Then, in order to optimize the initial random weight configuration, they use (ii) a gradient method performing the steepest descent on a surface in the weight space employing the same training data, that is, supervised learning.

Here, on the other hand, we use a simple genetic algorithm, since our target space is specific enough to know the network structure in advance, i.e., only unique rule is necessary. Our concern, therefore, is just obtaining the solution of weight configuration of the network. That is to say, all we want to know is a set of parameters $w_{ik}$, $\sigma_{ik}$ and $\nu_{kj}$ $(i = 1, \cdots n)$, $(k = 1, \cdots H)$, $(j = 1, \cdots m)$ where $n$ is the dimension of data, $H$ is the number of rules, and $m$ is the number of outputs. Hence our chromosome has those $n \times H \times m$ genes. Starting with a population of chromosomes whose genes are randomly created, they evolve under *simple truncate selection* where higher fitness chromosome are chosen, with

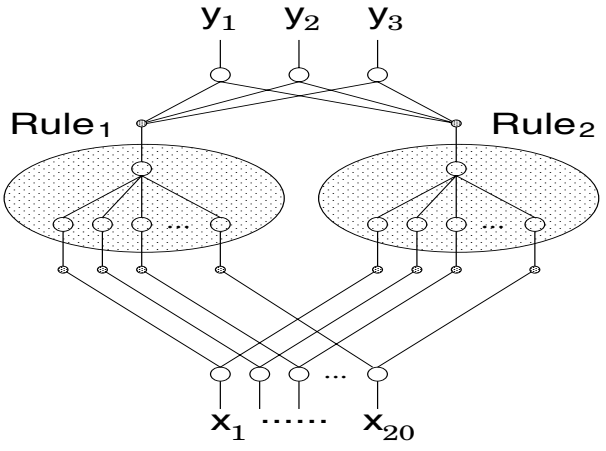



**Fig. 2.** Architecture of the proposed fuzzy neural network which infers how an input $\mathbf{x} = (x_1, \cdots x_n)$ is likely to belong to the $j$-th class by generating outputs $y_j$ each of which reflect the degree of the likeliness. In this example, a 20-dimension data input will be inferred to which of the 3 classes the input belongs by using 2 rules.

*uniform crossover* and occasional *mutation* by replacing some of a few genes with randomly created other parameters, expecting higher fitness chromosomes will be emerged. These settings are determined by trials and errors experimentally.

## 3 Experiment

An experiment was carried out in the 20-dimensional space. Our assumption is normal data exist in the *lake* region while abnormal data in the *island* region. We control the size of the island by changing the parameter value $a$. Furthermore, it is easy to guess that only one inference rule is enough to classify an input into either of the two classes. The architecture of the fuzzy network is, therefore, 20 input nodes, 1 rules, and 2 output nodes.

## 4 Results and Discussion

Though our experiments have sometimes reversed our expectations depending on parameter setting, we are obtaining a series of successful results.

**Where Is the Tiny Island?** In the preliminary experiment using a five fixed shaped triangular and trapezoidal fuzzy membership functions, evolution converges to the chromosome

$$\{M, M, M, \cdots, M\}$$

which implies

IF $x_1$ is Middle, and $\cdots$, and $x_{20}$ is Middle THEN no-self.

However, this holds only on the condition that the island is fairly large.

**How Tiny Island Can Be Detected?** In Fig. 3, we showed an example of obtained membership function corresponding to one antecedent of the rule (Left), as well as one of the output singletons obtained in the same experiment (Right). Training Samples are from the assumed legal data exist in the *lake* region to identify the illegal data exists in the *tiny island* region defined as $a = 0.1$. In the figure, although only one example of membership function is shown out of 20 others, the other 19 membership factions are more or less similar to the one shown in the figure. This suggest that

$R_1$: IF all of $x_n$ is near the origin THEN ($y_1$ is HIGH) and ($y_2$ is LAW).

Namely, the input belongs to the abnormal class.

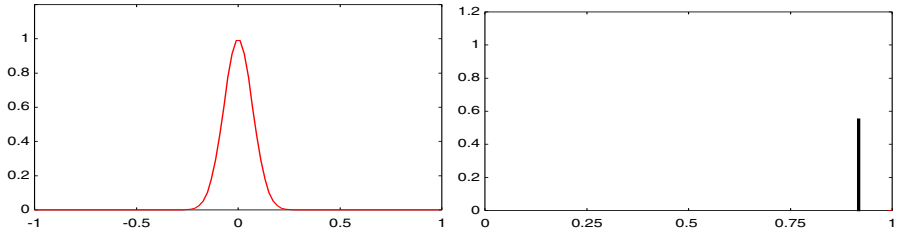



**Fig. 3.** An example of experimental result of a membership function of one antecedent membership function of a rule (Left), and one of the two output singletons of the same experiment (Right). Training Samples are from the assumed legal data exist in the *lake*, while the illegal data is assumed to be in the *tiny island* defined as $a = 0.1$.

## 5 Summary

In this paper, we have reported our on-going investigations, that is, how already proposed methods work on a special situation of what we call *a-tiny-island-in-a-huge-lake*. When we increase the difficulty of the problem by making the size of the island shrink to zero, it will become what they call *a-needle-in-a-haystack* [1]. As far as we know, this issue has resisted to be fully solved and still remains open. Though our results so far has not been matured yet, we hope a lot of experiments await our exploration which might result in useful observations in considering how we design a network intrusion detection system.

## References

1. G. E. Hinton and S. J. Nowlan (1987) *How Learning can Guide Evolution.* Complex Systems, 1, pp. 495–502.
2. A. Imada (2004) "How a Peak on a Completely-flatland-elsewhere can be Searched for? — A Fitness Landscape of Associative Memory by Spiking Neurons." Proceedings of Advanced Computer Systems (ACS) and Computer Information Systems and Industrial Management Applications (CISIM), Vol.2, pp. 171–150.

3. Zhou Ji and D. Dasgupata (2004) “Augmented Negative Selection Algorithm with Variable-Coverage Detectors.” Proceedings of the Congress on Evolutionary Computation. pp. 1081–1088.
4. J. Gomez, F. Gonzalez, and D. Dasgupta (2003) ”An Immuno-Fuzzy Approach to Anomaly Detection” proceedings of the 12th IEEE International Conference on Fuzzy Systems, Vol. 2, pp. 1219-1224.
5. G. Castellano and A. M. Fanelli(2000) ”Fuzzy Inference and Rule Extraction using a Neural Network.” Neural Network World Journal Vol. 3, pp. 361–371.

# Protein Solvent Accessibility Prediction Using Support Vector Machines and Sequence Conservations

Hasan Oğul[1] and Erkan Ü. Mumcuoğlu[2]

[1] Department of Computer Engineering, Başkent University, 06530, Ankara, Turkey
hogul@baskent.edu.tr
[2] Information Systems and Health Informatics, Informatics Institute, Middle East Technical University, 06531, Ankara, Turkey
mumcuoglu@ii.metu.edu.tr

**Abstract.** A two-stage method is developed for the single sequence prediction of protein solvent accessibility from solely its amino acid sequence. The first stage classifies each residue in a protein sequence as exposed or buried using support vector machine (SVM). The features used in the SVM are physico-chemical properties of the amino acid to be predicted as well as the information coming from its neighboring residues. The SVM-based predictions are refined using pairwise conservative patterns, called maximal unique matches (MUMs). The MUMs are identified by an efficient data structure called suffix tree. The baseline predictions, SVM-based predictions and MUM-based refinements are tested on a nonredundant protein data set and ~73% prediction accuracy is achieved for a solvent accessibility threshold that provides an evenly distribution between buried and exposed classes. The results demonstrate that the new method achieves slightly better accuracy than recent methods using single sequence prediction.

## 1 Introduction

The difficulties in the determination of protein structure and function have been led to an increase on the demand for computational tools for protein analysis. Since there are many parameters which play role in forming protein conformations, high-resolution analysis tools are required to identify different kind of features of proteins. One of those features is the solvent accessibility of the residues in a protein. A protein is composed from a chain of amino acid residues and the solvent accessibility of a residue is described in terms of the degree of its interaction with the water molecules. This interaction degree is inferred from the accessible surface area of the residue in the protein polypeptide chain. The residues with an interaction level of lower than a specified threshold are called as *buried* and the others are called as *exposed*. Thus, the solvent accessibility prediction problem turns out to be a binary classification problem in which each residue of a protein is categorized as buried (negative class) or exposed (positive class).

Various methods have been proposed for the prediction of solvent accessibility from the primary sequence of proteins [3]. One group of methods is based on the

F.A. Savacı (Ed.): TAINN 2005, LNAI 3949, pp. 141 – 148, 2006.

single sequence prediction of the solvent accessibility from local amino acid compositions. Single sequence methods identify local statistics from amino acid sequences and predict the solvent accessibility using different classification schemes, such as neural networks [1,11], Bayesian statistics [13], multiple linear regression [8] or support vector machines [17]. Richardson and Barlow has provided a baseline method which uses only the statistics inferred from the tendency of each amino acid to be buried or exposed [12]. The single sequence prediction accuracy is about 71% and this can be increased using multiple sequence information in the data set. Multiple sequence predictions use evolutionary information inferred from the profiles constructed by multiple sequence alignments. Multiple sequence methods increase the prediction accuracy up to about 79% [3]. However, using multiple alignments is computationally inefficient and it is not always guaranteed that informative profiles could be constructed in the given dataset.

In this work, a two-stage method is developed for the single sequence prediction of solvent accessibility and ~73% accuracy is achieved with an accessibility threshold of 22.2% on a nonredundant data set of 420 proteins. The first stage uses support vector machines to predict the two-class solvent accessibility using the residue features such as hydropathy scale and residue mass, as well as the neighborhood information from the left and right side of an amino acid. The second stage searches the maximal and unique amino acid sequence matches between the target protein and the other proteins in the data set. The SVM-based predictions are refined using the conservations over the maximal unique matches (MUMs).

## 2 Methods

Baseline, SVM-based and MUM-based methods are developed for the prediction of solvent accessibility. The methods are applied individually and as an ensemble to test their performance over the protein data set.

### 2.1 Baseline Predictions

The baseline predictions can be obtained using the solvent accessibility statistics of each amino acid in the selected data set. Solvent accessibility values are taken from DSSP database [7] and the statistics are extracted from the training set. DSSP gives a solvent accessibility value between 0 and 9 for each residue of the proteins such that the value of 0 refers to a completely buried (0%) residue, 1 refers to a solvent accessibility of (0-11.1]%, 2 refers to (11.1-22.2]% and so on. The tendency of an amino acid to be buried is determined simply by comparing the counts of buried and exposed occurrences of that amino acid in the training set. If the number of buried occurrences is higher than exposed ones, this amino acid is predicted as buried for all test cases. Otherwise, it is marked as exposed. According to the statistics, V, I, L, F, M, W, C are buried and G, A, P, S, T, N, Q, Y, H, D, E, K, R are exposed amino acids with an accessibility threshold of 22.2%. For 0% threshold, all are marked as buried, whereas only G is exposed for 55.5% threshold.

### 2.2 SVM-Based Prediction

Support vector machine (SVM) is a binary classifier which works based on the structural risk minimization principle [15]. An SVM non-linearly maps its n-dimensional input space into a high dimensional feature space. In this high dimensional feature space a linear classifier is constructed. Because of its power and robustness in performing binary classification and real valued function approximation tasks, it has been more popular in recent years and applied on many problems in computational biology [9,16,17]. In many applications, it has been shown that SVM is consistently superior to other supervised machine learning methods, such as Bayesian classifiers or neural networks.

To train the classifier which makes a separation between the buried and exposed classes (exposed used for positive and buried for negative classes), we used the feature vectors which represent the physicochemical properties of the amino acid, the binary code of the amino acid itself and the properties of left and right neighbors of the center amino acid to be predicted. In the previous work of Yuan et al [17], only amino acid codes were used for feature vectorization. The properties used in our vectorization step are listed in Table 1, where the hydropathy scales (free energy changes for transfer from oil to water for each amino acid) are taken from the Horton's book [6] and the relative residue mass values are given by Li and Pan [8]. For each 3-length amino acid string, a feature vector with a length of 66 (3x22) is used. 20 of the vector elements represent one of the 20 different amino acids and 2 elements are physicochemical properties explained before.

**Table 1.** Chemical and physical properties of amino acids used in the feature representations

| Amino acid | Hydropathy Scale | Relative residue mass (W as 1.0) |
|---|---|---|
| **G** | **0.67** | **0.00076** |
| **A** | **1.0** | **0.115** |
| **V** | **2.3** | **0.33** |
| **I** | **3.1** | **0.13** |
| **L** | **2.2** | **0.13** |
| **F** | **2.5** | **0.7** |
| **P** | **-0.29** | **0.323** |
| **M** | **1.1** | **0.577** |
| **W** | **0.0** | **1.0** |
| **S** | **-1.1** | **0.238** |
| **T** | **-0.75** | **0.346** |
| **N** | **-2.7** | **0.446** |
| **Q** | **-2.9** | **0.55** |
| **Y** | **0.08** | **0.82** |
| **H** | **-1.7** | **0.63** |
| **D** | **-3.0** | **0.446** |
| **E** | **-2.6** | **0.55** |
| **K** | **-4.6** | **0.48** |
| **R** | **-7.5** | **0.777** |
| **C** | **0.17** | **0.36** |

We used the SVM-*Gist* software implemented by Noble and Pavlidis (www.cs.columbia.edu/compbio/svm) in our tests. In the G*ist* software, a kernel function acts as the similarity score between the pairs of input vectors. The base kernel is normalized in order to make that each vector has a length of 1 in the feature space, that is,

$$K(X,Y) = \frac{X.Y}{\sqrt{(X.X)(Y.Y)}}$$

where $X$ and $Y$ are the input vectors, $K(.,.)$ is the kernel function, and *"."* denotes the dot product.

To tune an SVM, the most significant parameter needed is the kernel function. We use radial basis function, which can be expressed with a modified kernel function $K'(X,Y)$, as follows:

$$K'(X,Y) = e^{-\frac{K(X,X)-2K(X,Y)+K(Y,Y)}{2\sigma^2}} + 1$$

where the width $\sigma$ is the median Euclidean distance from any positive training example to the nearest negative example. Since the separating hyperplane of SVM is required to pass from the origin, the constant 1 is added to the kernel so that the data goes away from the origin.

### 2.3 MUM-Based Refinement

In spite of the fact that the data set we used is composed of non-homolog or remote homolog proteins, they may still share some conservative patterns between them. If these kinds of sequence conservations refer also to the conservations in solvent accessibility, we can use the statistics inferred from them to refine the incorrectly identified residues.

We use the maximal unique match definition, which is originally described in MUMer [4] to accelerate the alignment of long DNA sequences, to define the local conservations between two proteins. A maximal unique match between two sequences can be defined as the substring that appears only once in both sequences and not contained in any longer such substrings. Because of their uniqueness, maximal unique matches are important local similarities and give important clues about the structural conservations between two proteins.

Since we have already trained 3-letter strings in SVM applications, here, we extract only the matches longer than 5 amino acids and calculate the solvent accessibility statistics obtained from the middle-point of each maximal unique match. Among all maximal unique matches extracted from the dataset, the percentage of correctly identified residues is 79.4%. For any homolog data set, this percentage promises good prediction accuracy for solvent accessibility. However, in our data set, containing less or no homology, the number of maximal unique matches is relatively low. Therefore, the information gathered from the maximal unique matches can only be used for the refinement of the predictions made by other methods.

*Suffix Trees*

To find the maximal unique matches, we used a special data structure called suffix tree. A suffix tree is a compacted tree that stores all suffixes of a given text string. (An example suffix tree is shown in Figure 1.). It is a powerful and versatile data structure which finds application in many string processing algorithms, such as string matching, text compression and analyzing genetic sequences [5].

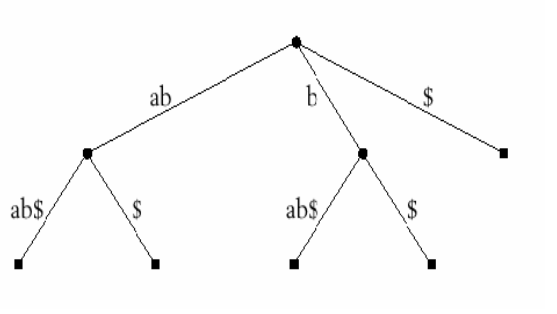



**Fig. 1.** Suffix tree of "abab$"

Let $A$ be string of $n$ characters, $A=s_1s_2\ldots s_n$, from an ordered alphabet $\Sigma$, except $s_n$. Let $\$$ be a special character, matching no character in $\Sigma$, and $s_n$ be $\$$. The suffix tree $T$ of $A$ is a tree with $n$ leaves such that;

- Each path from the root to a leaf of $T$ represents a different suffix of $A$.
- Each edge of $T$ represents a non-empty string of $A$.
- Each non-leaf node of $T$, except the root, must have at least two children.
- Substrings represented by two sibling edges must begin with different characters.

There are many algorithms for the construction of a suffix tree. We used Ukkonen's linear-time construction algorithm [14] in our implementation.

*Finding Maximal Unique Matches*

To find the maximal unique matches between any two sequences, first, a generalized suffix tree is constructed for the sequences. This is simply done by concatenating two sequences with a dummy character (not contained in the alphabet) between them and constructing a suffix tree for newly created sequence. In our representation, a maximal unique match is a maximal pair in the concatenated sequence one of which appears before the dummy character and the other appears after that. The algorithm to find maximal pairs is given by Gusfield [5]. We used a variation of this algorithm considering the fact that each of the pair should appear in different sequences. The details of the algorithm can be found in the previous study of Oğul and Erciyes [10].

In MUM-based refinement stage, each protein in the test set is searched for the maximal unique matches with all other proteins in a pairwise fashion. The solvent accessibility of the residue appearing in the middle of a maximal unique match is determined by a simple voting scheme.

## 3 Experiment and Results

The baseline predictions, SVM-based predictions, and MUM-based refinements are applied into data set for a solvent accessibility threshold of 22.2%. The data set contains 420 proteins which have no pair with a sequence similarity above 25%. In SVM-based prediction stage, 15 proteins which are randomly selected from the dataset are used for training the SVM. Total number of training examples is 3067, where 1564 of them are exposed and 1503 of them are buried with 22.2% threshold. The remaining proteins are used for the tests. The proteins in the training and test sets are given in the Table 2 with their Protein Data Bank [2] identification numbers.

**Table 2.** Data set

| | |
|---|---|
| **Training Set** | 1acx, 1amp, 1aya, 1ctf, 1hmp, 1hmy, 1hnf, 1hor, 5lyz, 6cpa, 6dfr, 6tmn, 7rsa, 9api, 9wga, |
| **Test Set** | 154l, 1aaz, 1add, 1ade, 1ahb, 1alk, 1amg, 1aor, 1aoz, 1asw, 1atp, 1avh, 1azu, 1bam, 1bbp, 1bcx, 1bdo, 1bds, 1bet, 1bfg, 1bmv, 1bnc, 1bov, 1bph, 1brs, 1bsd, 1cbg, 1cbh, 1cc5, 1cdl, 1cdt, 1cei, 1cel, 1cem, 1ceo, 1cew, 1cfb, 1cfr, 1cgu, 1chb, 1chd, 1chk, 1chm, 1cks, 1clc, 1cns, 1coi, 1col, 1com, 1cpc, 1cpn, 1cqa, 1crn, 1cse, 1csm, 1cth, 1ctn, 1ctm, 1ctn, 1ctu, 1cxs, 1cyx, 1daa, 1dar, 1del, 1dfj, 1dfn, 1dih, 1dik, 1din, 1dkz, 1dlc, 1dnp, 1dpg, 1dsb, 1dts, 1dup, 1dyn, 1eca, 1ece, 1ecl, 1ecp, 1edd, 1edm, 1edn, 1eft, 1efu, 1epb, 1ese, 1esl, 1etu, 1euu, 1fba, 1fbl, 1fc2, 1fdl, 1fdt, 1fdx, 1fin, 1fjm, 1fkf, 1fnd, 1fua, 1fuq, 1fxi, 1gal, 1gcb, 1gcm, 1gd1, 1gdj, 1gep, 1gfl, 1ghs, 1gky, 1gln, 1gmp, 1gnd, 1gog, 1gp1, 1gp2, 1gpc, 1gpm, 1grj, 1gtm, 1gtq, 1gym, 1han, 1hip, 1hcg, 1hcr, 1hiw, 1hjr, 1hpl, 1hsl, 1htr, 1hup, 1hvq, 1hxn, 1hyp, 1il8, 1ilk, 1inp, 1irk, 1isa, 1isu, 1jud, 1kin, 1knb, 1kpt, 1krc, 1kte, 1ktq, 1kuh, 1l58, 1lap, 1lat, 1lba, 1lbu, 1leh, 1lib, 1lis, 1lki, 1lpb, 1lpe, 1mai, 1mas, 1mct, 1mda, 1mdt, 1mjc, 1mla, 1mmo, 1mns, 1mof, 1mrr, 1mrt, 1msp, 1nal, 1nar, 1nba, 1ncg, 1ndh, 1nfp, 1nga, 1nlk, 1nol, 1nox, 1noz, 1oac, 1onr, 1otg, 1ovb, 1ovo, 1oxy, 1oyc, 1paz, 1pbp, 1pbw, 1pda, 1pdn, 1pdo, 1pga, 1pht, 1pii, 1pky, 1pmi, 1pnm, 1pnt, 1poc, 1pow, 1ppi, 1ppt, 1ptr, 1ptx, 1pyp, 1pyt, 1qbb, 1qrd, 1r09, 1rbp, 1rec, 1reg, 1req, 1rhd, 1rhg, 1rie, 1ris, 1rld, 1rlr, 1rpo, 1rsy, 1rvv, 1s01, 1scu, 1sei, 1ses, 1sfe, 1sft, 1sh1, 1smn, 1smp, 1spb, 1sra, 1srj, 1stf, 1stm, 1svb, 1tab, 1taq, 1tcb, 1tcr, 1tfr, 1tht, 1thx, 1tie, 1tif, 1tig, 1tii, 1tml, 1tnd, 1tnf, 1tpl, 1trb, 1trh, 1trk, 1tsp, 1tss, 1tul, 1tup, 1ubd, 1ubq, 1udh, 1umu, 1vca, 1vcc, 1vhh, 1vhr, 1vid, 1vjs, 1vmo, 1vnc, 1vok, 1vpt, 1wap, 1wfb, 1whi, 1wsy, 1xva, 1ypt, 1yrn, 1znb, 1zym, 256b, 2aai, 2aat, 2abk, 2adm, 2afn, 2ak3, 2alp, 2asr, 2bat, 2blt, 2bop, 2cab, 2ccy, 2cmd, 2cpo, 2cyp, 2dkb, 2dln, 2dnj, 2ebn, 2end, 2erl, 2fox, 2fxb, 2gbp, 2gcr, 2gls, 2gn5, 2gsq, 2hft, 2hhm, 2hip, 2hmz, 2hpr, 2i1b, 2ltn, 2mev, 2mhu, 2mlt, 2mta, 2nad, 2npx, 2olb, 2pab, 2pgd, 2phh, 2phy, 2pol, 2reb, 2rsl, 2rsp, 2scp, 2sil, 2sns, 2sod, 2spt, 2stv, 2tgi, 2tgp, 2tmd, 2tmv, 2trt, 2tsc, 2utg, 2wrp, 2yhx, 3ait, 3b5c, 3bcl, 3blm, 3cd4, 3chy, 3cla, 3cln, 3cox, 3eca, 3gap, 3hmg, 3icb, 3ink, 3mdd, 3pgk, 3pgm, 3pmg, 3rnt, 3tim, 4bp2, 4cpa, 4fis, 4gr1, 4pfk, 4rhv, 4rxn, 4sdh, 4sgb, 4ts1, 4xia, 5cyt, 5er2, 5ldh, 5sic, 6acn, 6cpp, 6cts, 6hir, 6rlx, 7cat, 7icd, 821p, 8adh, 9ins, 9pap |

All resulting predictions are compared with the actual values of solvent accessibilities obtained from DSSP database. The accuracy is defined as the percentage of number of correctly identified residues among all residues. The experimental results are given in Table 3 with varying threshold values.

As we can see from the table, SVM-based predictions give an improvement of 2.8% over baseline predictions for a 22.2% threshold, which is the case of evenly

distribution of buried and exposed classes. When applied on the baseline predictions, MUM-based refinement improves the baseline accuracy by 0.5%. The MUM-based refinement improves the SVM-based predictions by 0.4%. Overall improvement achieved by the combination of SVM-based and MUM-based predictions over the baseline accuracy is 3.2%. The methods are also tested for 0% and 55.5% thresholds and the results are given in the Table 3. For those threshold values, same improvement can not be achieved with SVM. This is probably due to the fact that the positive and negative examples are not evenly distributed for those threshold values.

**Table 3.** Results showing the accuracies achieved with different solvent accessibility thresholds

| Threshold / Method | 0% | 22.2% | 55.5% |
|---|---|---|---|
| **Baseline** | 75.3% | 69.5% | 79.6% |
| **Previous SVM method (Yuan et al, 2002)** | 70.9% | 71.4% | 78.7% |
| **New SVM method with extended features** | 71.6% | 72.3% | 79.1% |
| **MUM-refinement over baseline** | **75.7%** | 70.0% | **79.8%** |
| **MUM-refinement over new SVM** | 72.3% | **72.7%** | 79.3% |

Since there is no common benchmarking set for the solvent accessibility prediction, a direct comparison with the previous methods that used different data sets is not valid. According to the recent review of Chen et al [3], which reports a baseline prediction accuracy of 69.6% in their data set, the accuracies achieved with the tested methods are 71.5% for decision tree model and 71.2% for Bayesian statistics with a 20% threshold. We could make a fair comparison only with the baseline method and the previous SVM method of Yuan et al [17] with the same threshold over the same experimental setup (Table 3). Comparing with these results, our methods achieve slightly better accuracy.

## 4 Conclusion

Protein solvent accessibility is an important property for the annotation of newly extracted protein sequences. We introduce a new computational method for the prediction of solvent accessibility using solely the sequence information and report the results of the tests performed on a non-redundant protein set. The new method uses an improved SVM approach with extended features for the prediction of the accessibilities and refines the SVM predictions along with the pairwise conservations, i.e. maximal unique matches, between the sequences. The main reason for the improvement in SVM predictions is the incorporation of new physicochemical features of the protein residues in the vectorization phase. Although the maximal unique match refinement does not make a significant improvement on the accuracy, it promises good results when the sufficient number of homologs is found. Whenever the larger datasets are made available, this scheme can be used to obtain better refinements.

## References


1. Ahmad S., Gromiha M.M.: NETASA: neural network based prediction of solvent accessibility. Bioinformatics 18 (2002) 819-824.
2. Berman H.M., Westbrook J., Feng Z., Gilliland G., Bhat T.N., Weissig H., Shindyalov I.N., Bourne P.E.: The Protein Data Bank. Nucleic Acids Research 28 (2000) 235-242.
3. Chen H., Zhou H., Hu X., Yoo I.: Classification comparison of prediction of solvent accessibility from protein sequences. 2nd Asia-Pacific Bioinformatics Conference, Dunedin, New Zelland (2004).
4. Delcher A., Kasif S., Fleishmann R., Peterson J., White O., Salzberg S.: Alignment of whole genomes. Nucleic Acids Research 27 (1999) 2369-2376.
5. Gusfield D.: Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology. Cambridge University Press, (1997).
6. Horton H.B., Moran L.A., Ochs R.S., Rawn J.D., Scrimgeour K.G.: Principles of Biochemistry. Prentice Hall, (2002).
7. Kabsch W., Sander C.: Dictionary of protein secondary structure: pattern recognition of hydrogen bonded and geometrical features. Biopolymers 22 (1983) 2577-637.
8. Li X., Pan X-M.: New method for accurate prediction of solvent accessibility from protein sequence. Proteins 42 (2001) 1-5.
9. Liao L., Noble W.S.: Combining pairwise sequence similarity and support vector machines for remote homology detection. Proc. 6th. Int. Conf. on Computational Molecular Biology, (2002) 225-232 .
10. Oğul H., Erciyes K.: Identifying all local and global alignments between two DNA sequences. Proc. 17th Int. Sym. on Computer and Information Sciences, (2001) 468-475.
11. Rost B., Sander C.: Conservation and prediction of solvent accessibility in protein families. Proteins 20 (1994) 216-226.
12. Richardson C.J., Barlow D.J.: The bottom line for prediction of residue solvent accessibility. Protein Engineering 12 (1999) 1051-1054.
13. Thompson M.J., Goldstein R.A.: Predicting solvent accessibility: higher accuracy using Bayesian statistics and optimized residue substitution classes. Proteins 25 (1996) 38-47.
14. Ukkonen E.: On-line construction of suffix-trees. Algorithmica 14 (1995) 249-60.
15. Vapnik V.: The nature of statistical learning theory. Spring-Verlag, New York (1995).
16. Ward J., McGuffin L. C., Buxton B. F., Jones D. T.: Secondary structure prediction with support vector machines. Bioinformatics 19 (2003) 1650-55.
17. Yuan Z., Burrage K., Mattick J.: Prediction of protein solvent accessibility using support vector machines. Proteins 48 (2002) 566-570.

# Instrument Independent Musical Genre Classification Using Random 3000 ms Segment

Ali Cenk Gedik[1] and Adil Alpkocak[2]

[1] Dokuz Eylul University
Department of Musicology
Narlıdere, 35320 İzmir, Turkey
cenk.gedik@emo.org.tr
[2] Dokuz Eylul University
Department of Computer Engineering
Buca, 35160 İzmir, Turkey
alpkocak@cs.deu.edu.tr

**Abstract.** This paper presents a new approach to musical genre classification by using randomly selected maximum 3000 ms long segment of each recording. Recordings are classified into three root genres as jazz, classical and pop. A *k*-Nearest Neighbor (*k*-NN) classifier with pitch, harmony and rhythm features, independent from instruments, and a training set of 180 MIDI recordings is used for classification. The classifier is evaluated both quantitatively and cognitively using 45 MIDI recordings from outside the training set and considered for specificity, selectivity and accuracy measures. Experiments have demonstrated the good accuracy of the classifier and its high potential of use.

## 1 Introduction

Music classification bears an inherent ambiguity in naming similar recordings both for our daily life and musicological researches. And genre classification can be said to be one of the most problematic area, although it is also the most popular and pragmatic way of describing music. This ambiguity arises from different taxonomies used by various record companies, shops, Billboard and Top 50 lists, musical web sites and online stores, radios, TVs and even by musicologists.

Besides cognitive and cultural issues studied so far in musicology, automatic genre classification adds some other new problematic issues. These problems can be roughly formulated as follows which are discussed by Aucouturier [1] and Downie [2]:

- Representing music and genre information: which features should be selected and how to represent them.
- Classification: which classifier to use, linear or non-linear and the criterion of classifying a musical piece as false or right.
- Incomparability of researches: there is no standardized data collection to work on.

In automatic genre classification area, Tzanetakis[3] steps forward with a success rate of 61% for 10 genres among audio data classifications where acoustic features

F.A. Savacı (Ed.): TAINN 2005, LNAI 3949, pp. 149 – 157, 2006.

such as FFT Coefficients, Cepstrum and Mel Cepstrum Coefficients (MFCC) are used to extract low level timbre, pitch and beat information. In comparison to audio classification there is much less study made using symbolic data, where timbre, pitch and beat information are readily found and used to extract high level features. Recently McKay [4] has achieved worthy results with 950 MIDI recordings, using 109 features for 3 root and 9 leaf genres with 98% success rate for root and 90% for leaf genres.

In classifications based on time limitation; Lippens [5] used central 30000ms (30 sec.) of audio records, achieving success rate of 59% within 5 genres, Meng [6] used again 30000 ms in the same way to show improvement of classification by integration of short-time features. And also Tzanetakis [7] presents an experiment, to show the effect of duration on classification by increasing the first *n* ms of recordings up to 300000 ms(300 sec.) with a accuracy not exceeding 48% for the definite first 300000 ms. Although it is not a genre classification, Schreirer [8] used randomly selected 5000 ms(5 sec.) segments of records in a study based on perceptional complexity of short time listening process of human and compared them with a computational model.

Human, with a moderate musical background, can classify recordings into ten genres by listening to a part of 3000 ms(3 sec.) only [9]. This duration is very close to the human's ability of selecting a music channel from radio by a rapid tuning. However, there is a serious gap to reach the level of this ability for machines.

Therefore, we intend to fill the gap and propose a new method to classify music much closer to human behavior. So this study presents instrument independent musical genre classification by using randomly selected 3000 ms segment of each recording, with features enabling cognitive considerations of results. Recordings are classified to three root genres, jazz, western classical and pop music by a *k*-NN classifier and only three features, harmony, pitch and rhythm are used as feature set. Furthermore, although it is one of the most important feature in both audio and symbolic genre classification, knowledge of instrument is not used,. All the codes written are developed under MatLab and MidiToolBox [10] is used to process MIDI recordings.

Remainder of the paper is organized as follows: the next section describes the details of how we represent musical data, the features used in classification and how these features are extracted from musical content. Section 3 presents the details of training and classification experiments and results are discussed in section 4. Finally, section 5 concludes the paper, discusses the findings we obtained and gives a look to the future studies on this subject.

## 2 Feature Selection and Extraction

Even they occur at different levels of a classification, representing musical information and extracting features are closely related. And while extracting high-level semantic features from audio data is still developing, studies using symbolic data, where low level features are directly accessible, have not even yet solved the problem of extracting high level features. Anyway high level feature extraction is only one step toward solution, as the answer lies in the scope of cognitive sciences. Furthermore, a comparison of classification performance using pitch histograms of

audio and symbolic (MIDI) musical data demonstrates that there is a discernible distant against audio [7]. Symbolic musical data also provides a more robust platform to apply cognitive processes to computational applications, as it is well demonstrated by Temperley [11] in his comprehensive work.

However in both representations of classification, cognitive evaluations of false and true classified samples are not considered so far. So in this study for a cognitive discussion of results symbolic data MIDI is preferred which enables extracting high level semantic features also.

And as an optimal feature set, three features which are thought to be intrinsic to music genres, *harmony*, *pitch* and *rhythm* are selected and defined below, based on a preference rule system.

**Harmony Feature:** As it is not possible for every record of 3000 ms to perceive the chord progression depending on recording's tempo, here concept of harmony is used in its literal meaning. That is to say harmony is considered with concepts of consonance and dissonance which are also useful tools in analyzing music history.

While jazz can be said to be more dissonant than Classical Period of Western music and pop, Romantic Period and beyond becomes increasingly dissonant reaching to its climax with the occurrence of 12-tone and electronic music. And chord structure of classical four part music is formed of triads [12] while $7^{th}$ chord, also $9,11,13^{th}$, are characteristics of jazz which separates it from Classical Period. And although it is not as discriminative as in jazz, pop music involves $7^{th}$ chord structure as well. So, harmony feature, $H$, is described as follows:

$$H = (w_{d*}\, h_d + w_{7*}\, h_7 + w_{c*}\, h_c) / n \tag{1}$$

$h_d$ denotes number of dissonant events and defined as number of at least two notes sounding vertically within the interval of minor second, $h_7$ denotes again number of major and minor seventh interval by the same way, both expressing dissonance. $h_c$ denotes number of occurrence of $7^{th}$ chord and finally $n$ is the number of notes which normalizes the feature between records. The weights $w_{d.}$, $w_7$ and $w_c$ are found empirically, in turns, as 20, 5 and 10. The definitions of chord and dissonant interval are crucial here; they are taken into account as not only the note events started at the same time, but also the events even starting at different moments that form a vertical group , based on Gestalt principles.

**Pitch Feature:** Jazz music, in addition to major and minor, uses other scales such as Dorian, Phrygian. On the other hand western classical music and pop music is highly constructed on major-minor tonality. So pitch feature trying to capture this concept is used to discriminate classical and pop music from jazz. However pop should be discriminated from classical as well. So bass accompaniment feature is patched to pitch feature, but again this feature is differentiated according to the way the bass accompaniment is used, to differentiate jazz and pop music.

$$P = (w_{p*} p + w_{b*} b) / n. \tag{2}$$

To analyze pitch events based on above approach, $p$ is calculated as the number of neighbor notes occurring in pitch class distribution which is thought to be implying jazz scales or say it horizontal dissonance. For bass accompaniment shown as $b$ in

eq.(2) vertical note events -chords- but now for the ones starting at the same time are found and cancelled, as bass accompaniment is thought to be somewhere out of chords.

After these steps, number of bass events is calculated according to conditions of being below fifth C and difference of average pitch value and its standard deviation. If one bass event is found in all segments, $b$ is assigned as 1/5. Finally for $p$ equals to zero, weight of bass, $w_b$ is assigned as 2.5, otherwise it is assigned as 5 and for $b$ equals to zero, weight of $p$, $w_p$ is assigned as 2.5, and otherwise it is assigned as 5. This reverse relation is used to differentiate jazz from pop, as classical music is thought to be having no bass accompaniments or dissonant pitch relations.

**Rhythm Feature:** This feature tries to capture the variety of note events occurring at different times for each segment of recordings. By this approach polyrhythmic or irregular beat structure of jazz is considered to differentiate it from classical and pop which have more regular beat structures. First, Inter Onset Interval (IOI) of each note event is quantized and each IOI is rounded to a value between 1 and 10 for each measure. Second, a set of integer numbers between 1 and 10 is obtained from each segment according to involving these numbers, so beat values. And this set is checked for the elements of irregular beat set IB={1,4,6,7,9} which are thought to be denoting irregular beats. For each correspondence that element is added to rhythm feature set R and finally n(R), number of elements of set R, determines the value of rhythm feature.

### 2.1 Feature Extraction

Before feature extraction, a maximum of 3000 ms segment is randomly extracted from each recording. As a result of this process, recordings with varying durations between 1000 ms and 3000 ms segments are obtained which implies the random rapid tuning of radio channels. Secondly, drum channels are omitted. Finally, the features defined above are extracted from each recording. After normalizing these features, the training set became ready to be used in training.

## 3 Training

One of the main reasons of ambiguity in genre classification is the non-linear relation of genres. So, *k*-Nearest Neighbor is preferred as a classifier which also enables to analyze the results in contrary to other alternative non-linear classifiers such as Neural Network which has a black-box nature. Non-linearity of three genres can be easily seen in Fig. 1 showing jazz, classical and pop training samples in 3-D feature space at the training level of classification. Although classical samples are tied together forming a group at the down left side of Fig. 1, if carefully observed it can be seen that there is a respectively much smaller region of classic genre at the right hand side of this corner. This non-linearity is more explicit for jazz and especially for pop.

**Training Set:** 60 samples for each genre, totally 180 samples are selected as follows:

- Jazz samples range from swing era to bebop, post-bop, cool and modal examples including also Latin jazz examples. Only free-jazz is excluded as it is better to classify in modern music genre.

- Classical samples range from Baroque to Romantic period and also works of composers just before modern period are added.
- May be the most difficult selection process was for pop music. To reflect the same variety of examples of jazz and classical music, pop samples are selected to range from 60's cult names such as Simon & Garfunkel and Beatles to 70's pops like James Brown, Boney-M and from 80s Madonna. Briefly also rock, hard rock, rap and hip-hop examples are also included as coming closer to today's pop examples.

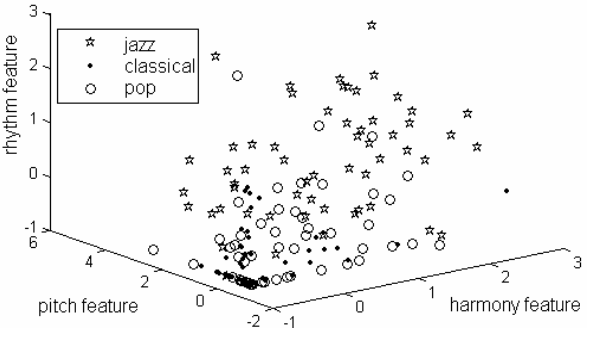



**Fig. 1.** Representation of genres, each of 60 samples as training set in 3-D feature space

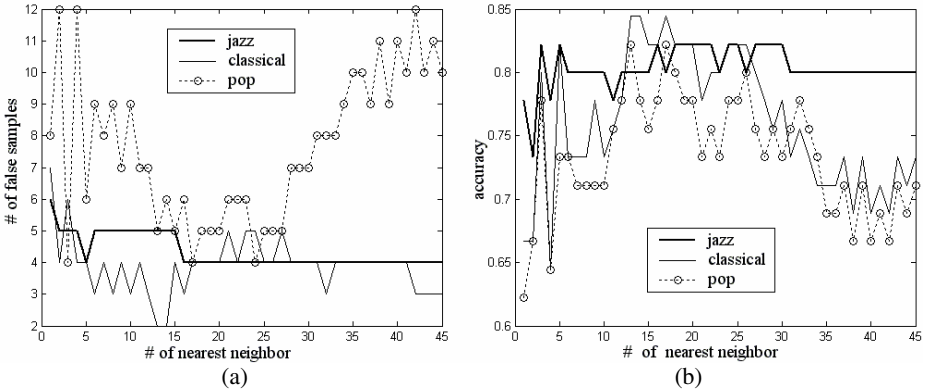
Fig.1 also demonstrates the discrimination power of each feature for each genre of 60 samples. While classical samples lie at the bottom of Fig.1, jazz samples try to go far away and pop samples oscillates between. Each feature's discriminative power reveals itself comparatively as most of the jazz and pop samples and some classical samples get different values for each. And this shows that for example if a jazz sample's harmony feature is not discriminative then its pitch or rhythm feature discriminates it. The same conditions also hold for pop and classical vice versa.

Some considerations about training set in 3-D feature space can be stated; for example random 3000 ms segment of "Django", a MIDI record of Modern Jazz Quartet, fell inside classical region, as they are a mainstream jazz orchestra, both the record and its random segment reflects the Baroque or Bach style interpretation of this famous jazz song. More or less the same conditions are also held for classical and pop samples. Even we avoid discussing the details of all experimentation results; the training set involves meaningful sub regions at least for classical music. So while Baroque lies in the deepest left corner of Fig.1 Classical, Romantic and pre-modern period samples diverge from that group.

After labeling all training samples with normalized feature sets, Euclidean distance is used as a metric to find the $k$-nearest neighbors of each sample. Instead of giving weights to features in measuring distance, weights are assigned in feature extraction process as $w_h=w_p=3$ for harmony and pitch, and $w_r=1$ for rhythm feature which are found experimentally. To determine the optimum $k$ value for training, leave-one-out $k$-fold cross validation is used and found as 13.

## 4 Evaluation

To evaluate our classifier and to ensure the comparability of our classification results, while considering the same variety of training set, we used a test collection of 45 samples, 15 for each genre, from the collection proposed by Corry McKay for the contest held in International Symposium of Music Information Retrieval [13]. So features are extracted from test samples and classified as described in section 2 and 3. The results are evaluated both for 13-NN and *k*-NN with varying *k* value.




(a) (b)

**Fig. 2.** Experimentation of *k* values vs. (a) false sample number, (b) classification accuracy

Fig.2a demonstrates the number of false classifications of each test genre samples with varying nearest neighbor numbers between 1 and 35. It is a well-known character of *k*-NN classifier that it is noisy for too small and too large values of *k* where it is optimal for 12<*k*<27 and this is consistent with our 13-NN classifier.

On the other hand as false jazz and classical samples are more stable, we can see the unstable character of pop music quite clear from Fig. 2a as it is in Fig. 1.While Fig. 2b shows the accuracy of classification for varying *k* values between 1 and 35, Table 1. explicitly shows the evaluation criteria and results obtained for *13*-NN genre classifier. Although jazz's evaluation values for *k*=13 is the smallest, Fig. 2b shows that jazz is more accurate for overall *k* values then classical and pop music.

**Table 1.** Classification results for *13*-NN

| | True | | False | | Sensitivity | Selectivity | Accuracy |
|---|---|---|---|---|---|---|---|
| | + | - | + | - | | | |
| **Jazz** | 10 | 26 | 5 | 4 | 0.71 | 0.84 | 0.80 |
| **Classical** | 13 | 25 | 2 | 5 | 0.72 | 0.93 | 0.84 |
| **Pop** | 10 | 27 | 5 | 3 | 0.77 | 0.84 | 0.82 |

**Cognitive Evaluations:** Even all results are evaluated with excerpts and compared with our results; we have limited the discussion in details with only two false

classified jazz samples due to space limitations, where original labels [13] of recordings are given in parenthesis.

*"Afro-Blues"- John Coltrane (Cool-Jazz) is false classified as pop:* Random segment of piece hit to a transition passage, including only a regular chord progression with little rhythm and pitch feature values. As a result of low tempo, no bass note and 1000ms segment, only harmony feature yield it to fell in pop region for all k, but could be true classified if chord progression feature would have been used.

*"Climax Rag"-Joseph Lamb and "The Entertainer"-Scott Joplin (Ragtime) are both classified as classical for all k:* These two famous ragtime pieces and their random segments sound like classical as they belong to first period of jazz. Even the latter has reasonable amount of harmony, pitch and rhythm feature values, it fell into at most near to the Romantic Period classical pieces. As the former has no 7th chord, bass note and only have little rhythm feature value, as a result of hitting to almost a merely melodic transition; it is hard to differentiate it from classical music.

As a result, false classifications are due to: firstly dependent to both low tempo and short duration (e.g. converging to 1000 ms). Secondly, random segments hitting to transition parts (e.g. false classified classical segments hit to chromatic movements, as the case for Scarlatti in training set). On the other hand, pre-modern classical test samples such as Debussy oscillates between jazz and classical, so this demonstrates the classical music boundaries of our classifier. But most of the false classified pop samples except very short duration or less note event problems could be true classified if other features such as chord progression would have been added. By the way, it is natural for pop to oscillate between jazz and classical genres as the sources of pop music are considered.

## 5 Conclusion

In this paper we presented a musical genre classification approach by a *k*-NN classifier. Our proposed new approach can be summarized as follows: First, recordings are represented with a randomly selected of 1000 - 3000 ms segment. Second, knowledge of instrument is not taken into consideration. Third, only three features harmony, pitch and rhythm are used. Last, our classifier forms a basement where cognitive discussions can be built upon. Classifier is trained with 180 samples and the optimum *k* value is found as 13. The system is evaluated quantitatively using 45 MIDI recordings from outside the training database and accuracy of 82% in average. However, if cognitive considerations are taken into account, such as the reasons of false and true classifications, overall accuracy of our classifier would be much better.

These results can be read as a part of discussion foregoing in automatic genre classification. One of the main questions is "Is timbre information enough for genre classification?" As our study is instrument independent the answer could be that it would improve the results if it is used as an additional feature, on the other hand it would mislead the classifier also. But the arguments such as "musical surface" (implying timbre) is enough and in 3000 ms human could not perceive high level

semantic features, are arguable according to the quality of 3000 ms segments and the results we obtained.

The discussion about false classified samples, tries to give another answer to genre classification problem such as "what is intrinsic to genres cognitively?" The answer lies in the selection and definition of musical features as described in section 2. But for a more meaningful and applicable results, experiments should be compared with tests of human having moderate musical knowledge. However as this study is concerned with excerpts as the first step, we planned to cover this gap in future works. Even though we have considered here three root genres only, our definition of features made it enable to improve the classifier with additional genres and sub-genres in future.

It is well known that *k*-NN gives more reasonable results for large data set while demonstrating its lazy learning character more explicitly. In future work we also intend to increase the data set and use methods trying to overcome this lazy learning process such as edited *k*-NN. As our data set is small respectively we did not need such methods, on the other hand *k*-NN classifier worked well with this amount of data set, as well.

Finally, the hopeful results we obtained shows that our motivation of modeling human behavior while selecting a music channel from radio by a rapid tuning, seems to be applicable in near future as radios automatically selecting channels from the varying radio broadcasting according to the musical genres set by the users.

## Acknowledgments

Thanks to Cihan Işıkhan for his idea of using $7^{th}$ chord in harmony feature and Yetkin Özer for his reviews about the very draft form of this paper. Both being from DEU Department of Musicology and others from Department of Computer Engineering, the collaborative sprit of our new but having a 2 year informal past Group, DEMIR (DEU Music Information Retrieval) 'lighted the fire'.

## References

1. Aucouturier, J. J., and F. Pachet, Representing Musical Genre: A State of the Art. Journal of New Music Research 32 (1): 1–12, 2003.
2. Downie, J. S. "Toward the Scientific Evaluation of Music Information Retrieval Systems", Proceedings of the Fourth International Conference on Music Information Retrieval,Baltimore, MD, pp. 25-32, 2003.
3. Tzanetakis, G., G. Essl, and P. Cook., Automatic Musical Genre Classification of Audio Signals, Proceedings of the International Symposium on Music Information Retrieval. 205-10, 2001.
4. McKay, C., Fujinaga, I., Automatic Genre Classification Using Large High-Level Musical Feature Sets, 5th International Conference on Music Information Retrieval,2004
5. Lippens, S., Martens, J., Mulder T., Tzanetakis G., A Comparison of Human and Automatic Musical Genre Classification, In Proc. IEEE Int. Conf. on Audio, Speech and Signal Processing Montreal, Canada, 2004.

6. Meng, A., Ahrendt, P., Larsen J., Improving Music Genre Classification by Short-Time Feature Integration, IEEE Int. Conf. on Audio, Speech and Signal Processing Philadelphia, PA, USA, 2005.
7. Tzanetakis, G., Ermolinskyi, A., Cook, P., Pitch Histograms in Audio and Symbolic Music Information Retrieval, Proc. 3rd ISMIR: 31-38, 2002.
8. Scheirer E. D., Watson R. B., Vercoe B. L.., On the Perceived Complexity of Short Musical Segments. Proc. International Conference on Music Perception and Cognition, Keele, UK, 2000.
9. Perrott, D., and R. O. Gjerdingen, Scanning the Dial: An Exploration of Factors in the Identification of Musical Style. Research Notes. Department of Music, Northwestern University, Illinois, USA, 1999.
10. Eerola, T. & Toiviainen, P. (2004). MIDI Toolbox: MATLAB. Tools for Music Research. University of Jyväskylä: Kopijyvä, Jyväskylä, Finland
11. Temperley, D., The Cognition of Basic Musical Structures. Cambridge, MA: MIT Press, 2001.
12. Pachet, F., Surprising Harmonies, International Journal on Computing Anticipatory Systems, 1999
13. Retrieved April 1 2005, http://www.ismir.net/

# Unsupervised Image Segmentation Using Markov Random Fields

Abdulkadir Şengür[1], İbrahim Türkoğlu[1], and M. Cevdet İnce[2]

[1] Fırat University, Department of Electronic and Computer Science, 23119, Elazı
ksengur@firat.edu.tr, iturkoglu@firat.edu.tr
[2] Fırat University, Department of Electric-Electronic Engineering, 23119, Elazı
mcince@firat.edu.tr

**Abstract.** In this study, we carried out an unsupervised gray level image segmentation based on Markov Random Fields (MRF) model. First, we use the *Expectation Maximization* (EM) algorithm to estimate the distribution of the input image and the number of the components is automatically determined by the *Minimum Message Length* (MML) algorithm. Then the segmentation is done by the Iterated Conditional Modes (ICM) algorithm. For testing the segmentation performance, we use both artificial images and real images. The experimental results are satisfactory.

## 1 Introduction

Segmentation is an important process in digital image processing which has found extensive applications in several areas. It aims to find the homogeneous regions for labeling objects and background. In other words, image segmentation is the process of grouping pixels of a given image into regions with respect to certain features and with semantic content [1]. Numerous segmentation methods have been proposed in the research literature. Thresholding is the most popular approach [2-3]. Clustering methods [4], region growing and splitting methods [5] and multi resolution [6] techniques are the other proposed approaches.

In this study, we use Markov Random Fields (MRF) for segmenting gray level images without supervision. With a seminal paper by *Geman and Geman* [7], MRF has been introduced to image processing and the computer vision community. MRF is a powerful tool to model the joint probability distribution of the image pixels in the terms of local spatial interactions [8]. *Besag* [9] proposed a method, called the Iterated Conditional Modes (ICM), which uses the local MRF's for segmentation of the true image from the noisy one. Several last decades, MRF is used for color image segmentation [10]. *Kato et al.* used MRF for segmenting the color images [11]. In ref. 11, a MRF based segmentation schema, which can perform better solutions even noisy images, is proposed. The algorithm only needs the number of classes as a priori. Several researchers also use the MRF for refining the coarse segmentation [12].

In this paper, we use MRF for segmentation of the gray level images when there is no prior information about the model parameters. Many methods, assume that the number of classes is known in advance. Here, we use Expectation Maximization (EM) algorithm for parameter estimation [13]. We also use Minimum Message

F.A. Savacı (Ed.): TAINN 2005, LNAI 3949, pp. 158 – 167, 2006.

Length (MML) for estimating the number of classes [14]. The experimental results are satisfactory. The rest of the paper is organized as follow: in section 2, we review the MRF and ICM algorithms briefly. In section 3, EM and MML algorithms are given. In section 4 we discuss experimental result. In section 5, finally we conclude the paper.

## 2 MRF for Image Segmentation

This section introduces the general framework to MRF image analysis and gives a brief overview of the MRF theory. MRF is n-dimensional random process defined on a discrete lattice. Usually the lattice is a regular 2-dimensional grid in the plane [7]. A random field can be considered as a MRF, if its probability distribution at any site depends only upon its neighborhood [8]. According to the Cliff-Hammersley theorem, any MRF can be described by a probability distribution of the Gibbs form:

$$p(x) = \frac{1}{Z} e^{(-U(x))} \tag{1}$$

Where $x$ is the random field, $Z$ is the normalization constant and the energy function $U(x)$ is defined as;

$$U(x) = \sum_{c \in C} V_c(x) \tag{2}$$

| | | | | |
|---|---|---|---|---|
| 5 | 4 | 3 | 4 | 5 |
| 4 | 2 | 1 | 2 | 4 |
| 3 | 1 | t | 1 | 3 |
| 4 | 2 | 1 | 2 | 4 |
| 5 | 4 | 3 | 4 | 5 |

(a)

| | | | | |
|---|---|---|---|---|
| t:-11 | t:-7 | t:-6 | t:+8 | t:+12 |
| t:-9 | t:-3 | t:-2 | t:+4 | t:+10 |
| t:-5 | t:-1 | t | t:+1 | t:+5 |
| t:-10 | t:-4 | t:+2 | t:+3 | t:+9 |
| t:-12 | t:-8 | t:+6 | t:+7 | t:+11 |

(b)

**Fig. 1.** Definition of Neighborhoods and Relative Neighborhoods

Where $V_c(x)$ is the potential function. We assume that the image is defined on an $MxN$ rectangular lattice $L = \{(i, j), 1 \le i \le M, 1 \le j \le N\}$ and c is a set of pixels, called a clique that consists of either a single pixel or a group of pixels. Figure 1-(a), demonstrates the first-order spatial neighbors of a site $t$ as 1, second order neighbor as 2 so on and figure 1-(b) provides a convenient labeling for neighbors of each pixel. The image observed is denoted by the $MN$-vector random variable $Y$ and is obtained by adding a noise process to the true image. Therefore, the density model for Y given the true image is;

$$f(y \mid X = x) = \prod_{t=1}^{MN} f_t(y_t \mid x_t) \tag{3}$$

Note that $f_t(.|x_t)$ is the conditional density function for $Y_t$, the gray level at pixel *t*. We take $f_t(y_t|x_t)$ to be the Gaussian density function with mean $\mu_{x_t}$ and standard deviation $\sigma_{x_t}$. The standard deviation depends on the both noise and the gray level variation of each image label. The a posteriori probability mass function for the pixel labels *X*, given the observed image *Y=y* also has the form of a Gibbs random fields respect to a neighborhood system cliques.

$$P(X = x \mid Y = y) = \frac{e^{-U(x|y)}}{Z} \tag{4}$$

Where Z is the normalizing constant and the energy function is as follows;

$$U(x \mid y) = \sum_{t=1}^{MN} [\frac{1}{2}\ln(\sigma_{x_t}^2) + \frac{(y_t - \mu_{x_t})^2}{2\sigma_{x_t}^2} + \sum_{r=1}^{c} [\theta_r, J(x_t, x_{t:+r})]] \tag{5}$$

Where $J(a, b) = -1$ if $a=b$, $0$ if $a \neq b$ and $c=2$ for first-order neighbor model. $[\theta_1,...,\theta_c]$ are the clique parameters. The local properties of an MRF can be derived from Gibbs random fields. Let $X_{\partial t}$ be a random variable presenting the gray level of neighbor of pixel *t* denoted by $[x_{t:+r}\ x_{t:-r}]$ for *r* from *1* to *c*. The conditional probability of $X_t$ can be written as [15];

$$P(X_t = x_t \mid X_{\partial t} = x_{\partial t}, Y = y) = \frac{e^{-U_t(x_t, x_{\partial t}|y)}}{Z} \tag{6}$$

and

$$U(x_t, x_{\partial t} \mid y) = \sum_{t=1}^{MN} [\frac{1}{2}\ln(\sigma_{x_t}^2) + \frac{(y_t - \mu_{x_t})^2}{2\sigma_{x_t}^2} + \sum_{r=1}^{c} \theta_r, [J(x_t, x_{t:+r}), J(x_t, x_{t:-r})] \tag{7}$$

Now the segmentation problem is considered as observing y and estimating the labels in the true image. The Maximum A-Posteriori (MAP) estimate is the vector $x'$ which maximizes $P(X=x \mid Y=y)$ with respect to true image *x*.

### 2.1 Iterated Conditional Modes (ICM)

ICM is an optimization method. Besag [8] proposed the ICM method as a computationally feasible alternative to MAP. In ICM, all sites are visited iteratively without restriction where the label that yields the maximum a posterior probability accepted as the estimate for the site. It is motivated for reducing the computational time produced by using the stochastic techniques such as Gibbs sampler. The ICM method can be summarized by the following equation where the label of the pixel *t*, given the observed image y and the current estimates $x_{\partial t}$ of the labels of all pixels in the neighborhood of pixel *t*.

$$P(X_t = x_t \mid y, X_{S|r} = x_{S|r}) = f_t(y_t \mid x_t) P(X_t = x_t \mid X_{\partial t} = x_{\partial t}) \tag{8}$$

Maximizing the conditional probability in eq. (8) is equivalent to minimizing the energy function which is given in eq. (7). The ICM algorithm can be represented as follows;

**Step 1:** Initialize $x'$ by maximizing $f_t(y_t \mid x_t)$ for all pixels.

**Step 2:** For t=1 to MN, update $x_t^{'}$ to the value of $x_t$ which maximizes energy function in eq. (7).
**Step 3:** Go to the step 2 for N times.

## 3 Parameter Estimation with EM Algorithm

Our goal is to segment the observed image using an unsupervised classification algorithm. For estimating the probability distributions of the labels in the observed image, we need to estimate the mean $\mu_{x_t}$ and the variance $\sigma_{x_t}^2$ of the each class label. There is no prior information so we can not use maximum likelihood approach for estimating the parameters of the probability distributions of the each class. In statistics, this problem is called as the incomplete data problem [16]. EM algorithm, which has been proposed by Dempster et al., aims to find these parameters [13]. EM algorithm consists of an E-step and an M-step and it starts with initial values $p_m^0, \mu_m^0$ *and* $\sigma_m^0$ for the parameters and iteratively performs these two steps until convergence. Suppose that $^t$ denotes the estimation of obtained after the *t* th iteration of the algorithm. Then at the *(t+1)* th iteration the E-step computes the expected complete log-likelihood function;

$$Q(\theta, \theta^t) = \sum_{k=1}^{K} \sum_{m=1}^{M} \{\log \alpha_m \, p(x_k \mid \theta_m)\} P(m \mid x_k; \theta^t) \tag{9}$$

where $P(m \mid x_k; \theta^t)$ is a posterior probability and it is computed as follows;

$$P(m \mid x_k; \theta^t) = \frac{\alpha_k^t \, p(x_k \mid \theta_m^t)}{\sum_{l=1}^{M} \alpha_l^t \, p(x_k \mid \theta_l^t)} \tag{10}$$

The M-step finds the *(t+1)* estimation $^{t+1}$ of by maximizing Q ( , $^t$)

$$\alpha_m^{t+1} = \frac{1}{K} \sum_{k=1}^{K} P(m \mid x_k; \theta^t) \tag{11}$$

$$\mu_m^{t+1} = \frac{\sum_{k=1}^{K} x_k P(m \mid x_k; \theta^t)}{\sum_{k=1}^{K} P(m \mid x_k; \theta^t)} \tag{12}$$

$$\sigma_m^{t+1} = \sqrt{\frac{1}{D}\frac{\sum_{k=1}^{K} P(m \mid x_k;\theta^t) \parallel x_k - \mu_m^{t+1} \parallel^2}{\sum_{k=1}^{K} P(m \mid x_k;\theta^t)}} \tag{13}$$

### 3.1 MML Algorithm

Main issue of using MRF model in image segmentation is the difficulty to estimate the number of components. Many methods, such as the method which Kato et al. proposed, assume the number of components is determined by the users. In this study we use the MML [14] criterion for overcoming this situation. For an M-components mixture model, the number of $M$ can be detected where the $\arg\min L(\theta, y)$. The function of L is defined as follows;

$$L(\theta, y) = \frac{N}{2}\sum_{m=1}^{M}\log(\frac{n\alpha_m}{12}) + \frac{M}{2}\log(\frac{n}{12}) + \frac{M(N+1)}{2} - \sum_{k=1}^{K}\log\sum_{m=1}^{M}\alpha_m p(x_k \mid \theta_m) \tag{14}$$

where N is the number of parameters specifying each component, n is the number of pixels and M is the number of components. and are the estimated parameters from the EM algorithm. In this study N is chosen 2.

## 4 Experimental Study and Results

In this study, an unsupervised gray level image segmentation schema is proposed based on ICM algorithm. The parameters of the components are estimated by using the EM algorithm and the number of the components is determined with MML criterion so the overall process is unsupervised. For testing the performance of the unsupervised segmentation algorithm, we generate several artificial gray level images of size 128x128 on MATLAB environment. These images can be seen at figure 2. The images are generated as they constitute of three, four, five and six classes respectively. Before running the ICM algorithm for segmentation, the unknown parameters such as number of classes and mean and variance of the components are obtained. Firstly, we assume that the number of components $M$ is a fixed number such as 2. According to this assumption we employ EM algorithm thus the component parameters can be determined. The initial values for EM algorithm are selected randomly. These values are then used for calculation the L value in MML algorithm. Then we increase the M and we run the aforesaid process again. In this study the maximum value of the M is restricted 7. The component parameters of the artificial images, which are estimated, are represented in the following tables. Table 1 shows the estimated parameters of artificial image which is labeled as (a) in figure 2. The actual parameter values for the each component is $\mu_{c1}$= 10 $_{c1}$=5, $\mu_{c2}$= 100 $_{c2}$=25 and $\mu_{c3}$= 180 $_{c3}$=45.

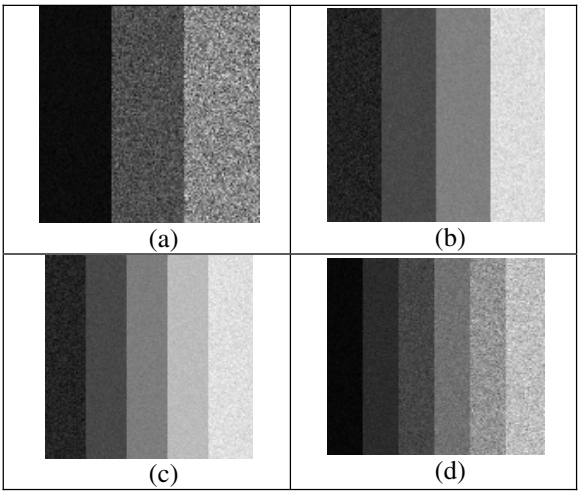



**Fig. 2.** Artificial images (a) 3 classes (b) 4 classes (c) 5 classes (d) 6 classes

**Table 1.** L values for image (a)

| Number of component | $L(\theta, y)$ |
|---|---|
| M=2 | 8.6140e+004 |
| M=3 | 8.5493e+004 |
| M=4 | 8.5505e+004 |
| M=5 | 8.5531e+004 |
| M=6 | 8.5533e+004 |
| M=7 | 8.5540e+004 |

Table 2 shows the estimated parameters of artificial image which is labeled as (b) in figure 2. The actual parameter values for the each component is $\mu_{c1}$= 20 $_{c1}$=10, $\mu_{c2}$= 60 $_{c2}$=5, $\mu_{c3}$= 120 $_{c3}$=10 and $\mu_{c4}$= 220 $_{c4}$=5.

**Table 2.** L values for image (b)

| Number of component | $L(\theta, y)$ |
|---|---|
| M=2 | 9.1350e+004 |
| M=3 | 8.1330e+004 |
| M=4 | 7.7953e+004 |
| M=5 | 7.7966e+004 |
| M=6 | 7.7981e+004 |
| M=7 | 7.7995e+004 |

Table 3 shows the estimated parameters of artificial image which is labeled as (c) in figure 2. The actual parameter values for the each component is $\mu_{c1}$=20 $_{c1}$=5, $\mu_{c2}$=60 $_{c2}$=10, $\mu_{c3}$=120 $_{c3}$=5, $\mu_{c4}$=190 $_{c4}$=5 and $\mu_{c5}$=230 $_{c5}$=10.

**Table 3.** L values for image (c)

| Number of Component | $L(\theta, y)$ |
|---|---|
| M=2 | 9.0908e+004 |
| M=3 | 8.6061e+004 |
| M=4 | 8.3381e+004 |
| M=5 | 8.0629e+004 |
| M=6 | 8.0642e+004 |
| M=7 | 8.0651e+004 |

**Table 4.** L values for image (d)

| Number of Component | $L(\theta, y)$ |
|---|---|
| M=2 | 9.2028e+004 |
| M=3 | 9.1309e+004 |
| M=4 | 9.1191e+004 |
| M=5 | 9.1005e+004 |
| M=6 | 8.7400e+004 |
| M=7 | 8.7414e+004 |

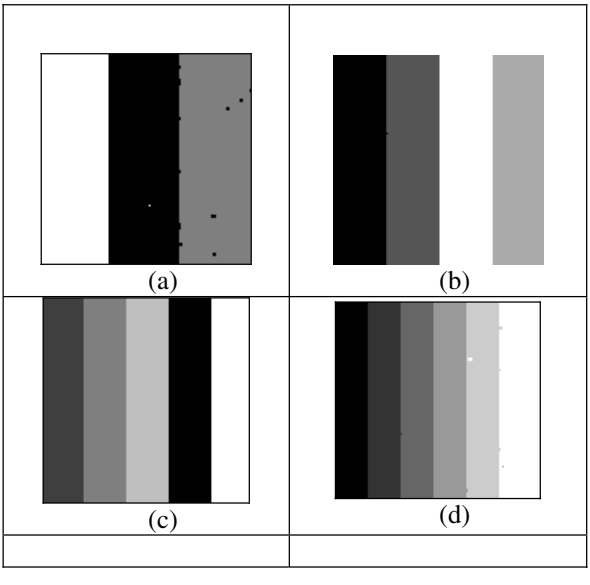



**Fig. 3.** ICM segmentation results

And finally, Table 4 shows the estimated parameters of artificial image which is labeled as (d) in figure 2. The actual parameter values for the each component is $\mu_{c1}$= 20 $_{c1}$=3, $\mu_{c2}$= 60 $_{c2}$=5, $\mu_{c3}$= 90 $_{c3}$=10, $\mu_{c4}$= 140 $_{c4}$=10, $\mu_{c5}$= 180 $_{c5}$=20 and $\mu_{c6}$= 220 $_{c6}$=20.

While using the ICM algorithm for segmentation of the gray level images, we use first-order spatial neighbors and we choose $\Theta_r$ values as 2. The ICM segmentation results can be seen in figure 3.

**Table 5.** L values for the real image

| Number of Component | $L(\theta, y)$ |
|---|---|
| M=2 | 3.1833e+004 |
| M=3 | 3.1711e+004 |
| M=4 | 3.1725e+004 |
| M=5 | 3.1739e+004 |
| M=6 | 3.1754e+004 |
| M=7 | 3.1791e+004 |

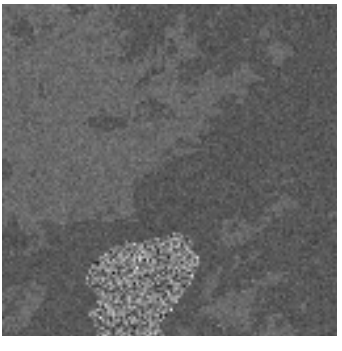


**Fig. 4.** A real world image

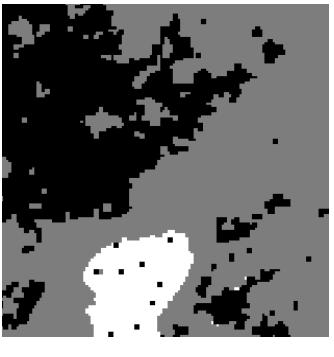


**Fig. 5.** Segmentation using first-order spatial neighbor

In Fig. 4, we use a real image for testing the algorithm. The image is obtained from the web site of Lund University [17]. The segmentation result of the image is seen at figure 5. In figure 6, the segmentation result of the real image is shown by using the first and the second order spatial neighbors.

**Fig. 6.** Segmentation using first and second order spatial neighbor

## 5 Conclusion

In this paper, we have examined an unsupervised gray image segmentation algorithm. The segmentation model is defined in a MRF framework. The examined algorithm is fully unsupervised because the number of the component is determined by the algorithm. No user information is needed. To estimate the component parameters, we use an iterative algorithm. EM and MML algorithms are employed for obtaining the crucial values. Then we use ICM algorithm for completing the segmentation procedure. The algorithm has been tested on a variety of artificial and real images and results are very satisfactory.

## References

1. Gonzalez R. C., Woods R. E.: Digital image processing, Prentice Hall, 2002.
2. Sahoo P. K., Soltani S., Wong A. and Chen Y.: A survey of thresholding techniques, Computer Vision Graphics Image Processing, vol. 41, pp. 233-260, 1988.
3. engür A., Türko lu . and nce M. C.: Performance Comparison of Thresholding Algorithms on Uneven Illuminated Image, Asian Journal of Information technology, 3 (10), 956-959, 2004.
4. Tsao E.C.K., Bezdek J. C. and Pal N.R.: Fuzzy Kohonen clustering networks, Patt. Recog. vol. 27, pp. 757-764, 1994.
5. Adams R., Bischof L.: Seeded region growing, IEEE Trans. on PAMI, pp. 641-647, 1994.
6. Kurugollu F., Sankur B. and Harmanci A. E.: Color image segmentation using histogram multithresholding and fusion, Image and Vision Computing, vol. 19, pp. 915-928, 2001.
7. Geman S., Geman D.: Stochastic relaxation, Gibbs distributions, and Bayesian restoration of images, IEEE Trans. PAMI, vol. 6, pp. 721-741, 1984.
8. Deng H., Clausi D. A.: Unsupervised image segmentation using a simple MRF model with a new implementation schema, Patt. Recog. Vol. 37, pp. 23223-2335, 2004.

9. Besag J.: On the statistical analysis of dirty pictures, J. Roy. Statist Soc. Ser. 48, pp. 259-302, 1986.
10. Kato Z., Zerubia J., Berthod M.: Unsupervised parallel image classification using Markovian models, Patt. Recog., vol. 32, pp. 591-604, 1999.
11. Kato Z., Pong T.C. and Lee J.: Color image segmentation and parameter estimation in a markovian framework, Patt. Recog. Lett. Vol. 22, pp. 309-321, 2001.
12. Yang X. and Liu J.: Unsupervised texture segmentation with one-step mean shift and boundary Markov random fields, Pattern Recognition Letters, Vo. 22,pp. 1073-1081, 2001.
13. Dempster A., Laird N.,and Rubin D.: Maximum likelihood estimation from incomplete data via the EM algorithm, J. Royal Statistical Soc. B. vol. 39 pp. 1-38, 1977.
14. Wallace C., Dowe D.: Minimum Message Length and Kolmogorov complexity, The computer J., vol. 42, pp. 270-283, 1999.
15. Dubes R. C., Jain A. K., Nadabar S.G. and Chen C.C.: MRF model based algorithms for image segmentation, IEEE, 1990.
16. McLanchlan G., Krishnan T.: The EM algorithm and extensions, New York, John Wiley and Sons, 1997.
17. http://www.maths.lth.se/

# Modeling Interestingness of Streaming Classification Rules as a Classification Problem

Tolga Aydın and Halil Altay Güvenir

Department of Computer Engineering, Bilkent University,
06800 Ankara, Turkey
{atolga, guvenir}@cs.bilkent.edu.tr

**Abstract.** Inducing classification rules on domains from which information is gathered at regular periods lead the number of such classification rules to be generally so huge that selection of interesting ones among all discovered rules becomes an important task. At each period, using the newly gathered information from the domain, the new classification rules are induced. Therefore, these rules stream through time and are so called streaming classification rules. In this paper, an interactive classification rules' interestingness learning algorithm (ICRIL) is developed to automatically label the classification rules either as "interesting" or "uninteresting" with limited user interaction. In our study, VFFP (Voting Fuzzified Feature Projections), a feature projection based incremental classification algorithm, is also developed in the framework of ICRIL. The concept description learned by the VFFP is the interestingness concept of streaming classification rules.

## 1 Introduction

Data mining is the efficient discovery of patterns, as opposed to data itself, in large databases. Patterns in the data can be represented in many different forms, including classification rules, association rules, clusters, sequential patterns, time series, contingency tables, and others. However, for example, inducing classification rules on domains from which information is gathered at regular periods lead the number of such classification rules to be generally so huge that selection of interesting ones among all discovered rules becomes an important task. At each period, using the newly gathered information from the domain, the new classification rules are induced. Therefore, these rules stream through time and are so called streaming classification rules.

In this paper, an interactive classification rules' interestingness-learning algorithm (ICRIL) is developed to automatically label the classification rules either as "interesting" or "uninteresting" with limited user interaction. In our study, VFFP (Voting Fuzzified Feature Projections), a feature projection based incremental classification learning algorithm, is also developed in the framework of ICRIL. The concept description learned by the VFFP is the interestingness concept of streaming classification rules. Being specific to our concerns, VFFP takes the rule interestingness factors as features and is used to learn the rule interestingness concept and to classify the newly learned classification rules.

F.A. Savacı (Ed.): TAINN 2005, LNAI 3949, pp. 168 – 176, 2006.

Section 2 describes the interestingness issue of rules. Section 3 is devoted to the knowledge representation used in this study. Section 4 and 5 are related to the training and classifying phases of the VFFP algorithm. ICRIL is explained in Section 6. Giving the experimental results in Section 7, we conclude in Section 8.

## 2 Interestingness Issue of Rules

There are factors contributing to the interestingness of a discovered rule such as coverage, confidence, completeness, actionability and unexpectedness. The first three factors are objective, actionability is subjective and unexpectedness is regarded both as subjective [2, 3, 4] and objective [5, 6]. Objective interestingness factors can be measured independently of the user and domain knowledge. But, subjective ones are user and/or domain knowledge dependent.

In this paper, different from the existing approaches in the literature, we learn the interestingness concept of the classification rules (rather than giving the interestingness concept as an input) and develop ICRIL algorithm in this respect. ICRIL tries to automatically label the rules with sufficient certainty factor. If it fails, user is requested to label the rule himself. Each rule labeled either as "interesting" or "uninteresting" by the user is treated as a training instance. And some interestingness factors that have the capability to determine the interestingness of rules are regarded as features, and interestingness labels ("interesting" or "uninteresting") of the rules are regarded as the classes of the training instances. Any classification algorithm can be used in the training and the querying phases. However, we also developed VFFP in the framework of ICRIL. VFFP is an incremental feature projection based classification algorithm. It represents each different nominal feature value as a point in the associated feature projection. In the case of numeric features, it fuzzifies the feature and always uses three linguistic terms: low, medium and high. The shape of the membership function is given in Figure 1. The user supplies parameters p1, p2, p3 and p4 for each different feature.

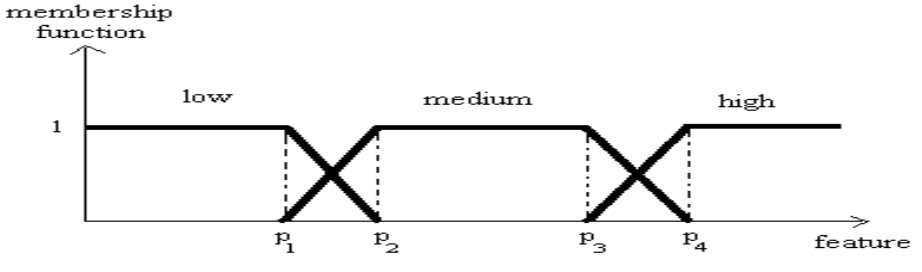



**Fig. 1.** Shape of the membership functions used for numeric features

## 3 Knowledge Representation

We think of a domain from which information is gathered at regular periods. For each period *p*, classification rules are induced from the gathered information and these rules' interestingness labeling seems to be an important problem. This labeling

problem is modeled as a new classification problem and a *rule set* is produced for these rules. Each instance of the rule set is represented by a vector whose components are the interestingness factors having the potential to determine the interestingness of the corresponding rule and the interestingness label of the same rule. Rules used in this study are probabilistic and have the following general structure:

If ($A_1$ *op* $value_1$) AND ($A_2$ *op* $value_2$) AND …AND ($A_n$ *op* $value_n$) THEN
($Class_1$: $vote_1$, $Class_2$: $vote_2$,…,$Class_k$: $vote_k$)

$A_i$'s are the features, $Class_i$'s are the classes and $op \in \{=, \leq, \geq\}$.

The instances of the rule set have either "interesting" or "uninteresting" as the interestingness label, and have the interestingness factors shown in Table 1. In this new classification problem, these factors are treated as determining features, and interestingness label is treated as the target feature (class) of the rule set.

**Table 1.** Features of the rule set

| Feature | Short description and/or formula |
|---|---|
| Major Class | $Class_i$ that takes the highest vote |
| Major Class Frequency | Ratio of the instances having $Class_i$ as the class label in the data set |
| Rule Size | Number of conditions in the antecedent part of the rule |
| Confidence with respect to Major Class | $\|Antecedent \,\&\, Class_i\| \,/\, \|Antecedent\|$ |
| Coverage | $\|Antecedent\| \,/\, \|N\|$ |
| Completeness with respect to Major Class | $\|Antecedent \,\&\, Class_i\| \,/\, \|Class_i\|$ |
| Standard Deviation of Class Votes | Standard deviation of the votes of the classes |
| Decisive | True if Std.Dev.of Class.Votes > $s_{min}$ |

Each feature carries information of a specific property of the corresponding rule. For instance, letting $Class_i$ to take the highest vote makes it the *Major Class* of that rule. If we shorten the representation of any rule as "If *Antecedent* THEN $Class_i$" and assume the data set to consist of *N* instances, we can define *Confidence*, *Coverage* and *Completeness* as in Table 1. Furthermore, a rule is decisive if the standard deviation of the votes is greater than $s_{min}$, whose definition is given as follows:

$$s_{min} = \frac{1}{(Class\ Count - 1)\sqrt{Class\ Count}} \tag{1}$$

## 4 Training in the VFFP Algorithm

VFFP (Voting Fuzzified Feature Projections) is a feature projection based classification algorithm developed in this study. It is used to learn the rule interestingness concept and to classify the unlabeled rules in the context of modeling rule interestingness problem as a new classification problem.

The training phase, given in Figure 4, is achieved incrementally. On a nominal feature, concept description is shown as the set of points along with the numbers of

instances of each class falling into those points. On the other hand, on a numeric feature, concept description is shown as three linguistic terms (low, medium and high) along with the numbers of instances of each class falling into those linguistic terms. The user gives the parameters of the membership functions of each numeric feature as inputs. Training can better be explained by looking at the sample data set in Figure 2, and the associated learned concept description in Figure 3.

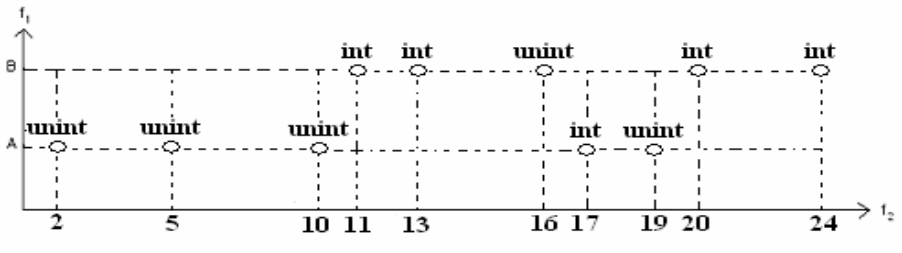



**Fig. 2**. Sample data set

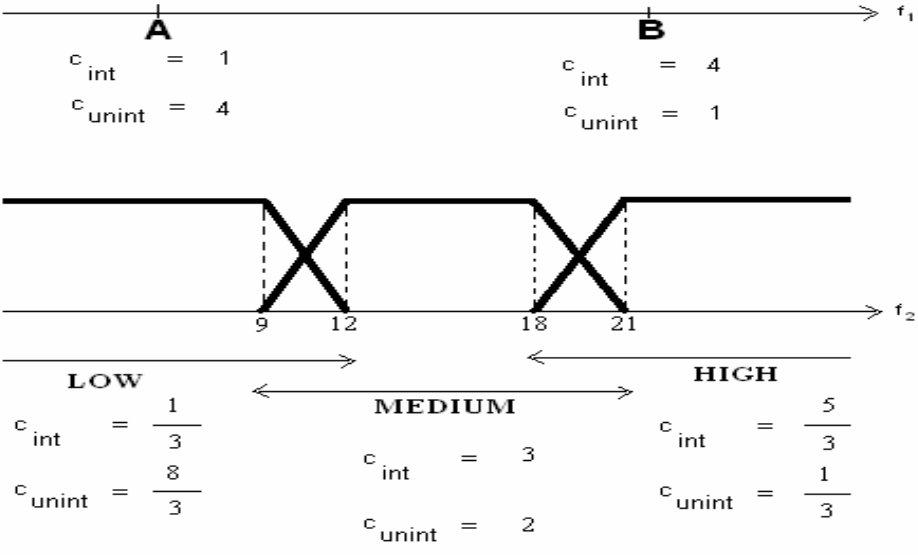



**Fig. 3.** Concept description learned for the sample data set

The example data set consists of 10 training instances, having nominal $f_1$ and numeric $f_2$ features. $f_1$ takes two values: 'A' and 'B', whereas $f_2$ takes some integer values. There are two possible classes: "int" and "unint". $f_2$ is assumed to have the following given parameter values: $p_1 = 9$, $p_2 = 12$, $p_3 = 18$ and $p_4 = 21$.

```
VFFP_train (t)      /* t: newly added training instance */
   let s be the class of t
   let others be the remaining classes
   class_count[s]++

   for each feature f

      if f is nominal
         p = find_point(f,t_f)
         if such a p exists
            point_class_count [f,p,s] ++
         else  /* add new point for f */
            add a new p' point
            point_class_count [f,p',s] = 1
            point_class_count [f,p',others] = 0

      else if f is numeric
        if membership value of t_f = 1
           Let l be the linguistic term that t_f falls in
           linguistic_term_class_count [f,l,s] ++
        else
           Let t_f be between parameters p_left and p_right
           Let left be the lefthandside linguistic term
           Let right be the righthandside linguistic
           term
           linguistic_term_class_count [f,left,s] +=
           (p_right - t_f) / (p_right - p_left)
           linguistic_term_class_count [f,right,s] +=
           (t_f - p_left) / (p_right - p_left)

              On numeric features (∀f, l, c)
return           linguistic_term_class_count [f, l, c]
              On nominal features (∀f, p, c)
                 point_class_count[f, p, c]
```

**Fig. 4.** Incremental train in VFFP

In Figure 4 for a nominal feature *f*, *find_point* ($f$, $t_f$) searches $t_f$, the new training instance's value at feature *f*, in the *f* projection. If $t_f$ is found at a point *p*, then *point_class_count* [*f*, *p*, *s*] is incremented, assuming that the training instance is of class *s*. If $t_f$ is not found, then a new point *p'* is constructed and *point_class_count* [*f*, *p'*, *class*] is initialized to 1 for *class* = *s*, and to 0 for *class* = *others*.

For a numeric feature *f*, if the membership value of $t_f$ is 1 then the new training instance falls in a linguistic term *l* with full membership. That is, $t_f$ lies in one of the following three intervals: [0, $p_1$], [$p_2$, $p_3$] or [$p_4$, ∞). As a consequence, *linguistic_term_class_count* [*f*, *l*, *s*] is incremented. If the membership value of $t_f$ is not 1, then the new training instance falls in the region shared by the linguistic terms *left* and *right*. That is, $t_f$ lies in either of the two intervals ($p_1$, $p_2$) or ($p_3$, $p_4$). As a consequence, *linguistic_term_class_count* [*f*, *left*, *s*] and *linguistic_term_class_count* [*f*, *right*, *s*] are increased by amounts inverse proportional to the distance between $t_f$ and the shared parameters $p_{left}$ or $p_{right}$. It is apparent that the total increase of class counts, after arrival of a new training instance, is always 1.

## 5 Classification in the VFFP Algorithm

Classification in VFFP is shown in Figure 5. The query instance is projected on all features, and each feature gives normalized votes for the query instance. Normalization ensures each feature to have equal power in classification.

The classification starts by giving zero votes to classes on each feature projection. For a nominal feature $f$, *find_point* ($f$, $q_f$) searchs whether $q_f$ exists in the $f$ projection. If $q_f$ is found at a point $p$, feature $f$ gives votes for each class $c$ as given in Equation 2, and then these votes are normalized to ensure equal voting power among features.

$$feature\_vote\ [f, c] = \frac{point\_class\_count\ [f, p, c\ ]}{class\_count\ [\ c\ ]} \tag{2}$$

In Equation 2, the number of class $c$ instances on point $p$ of feature projection $f$ is divided by the total number of class $c$ instances to avoid favoring major classes. For a numeric feature $f$, each class gets the vote given in Equation 3 given that the query instance falls in a linguistic term $l$ with full membership. If the query instance falls in the region shared by the linguistic terms *left* and *right* ($q_f$ lies in either of the two intervals ($p_1$, $p_2$) or ($p_3$, $p_4$)), each class gets the vote given in Equation 6. We note that *left_vote* [$f$, $c$] and *right_vote* [$f$, $c$] are increased by amounts inverse proportional to the distance between $q_f$ and the shared parameters $p_{left}$ or $p_{right}$ of *left* and *right* linguistic terms. In both cases, votes of classes are again normalized.

$$feature\_vote\ [f, c] = \frac{linguistic\_term\_class\_count[\ f, l, c\ ]}{class\_count\ [\ c\ ]} \tag{3}$$

$$left\_vote\ [f, c] = \frac{linguistic\_term\_class\_count[\ f, left, c\ ]}{class\_count\ [\ c\ ]} \tag{4}$$

$$right\_vote\ [f, c] = \frac{linguistic\_term\_class\_count[\ f, right, c\ ]}{class\_count\ [\ c\ ]} \tag{5}$$

$$\begin{aligned} feature\_vote\ [f, c] = {} & left\_vote\ [f, c] * ((p_{right} - q_f) / (p_{right} - p_{left}))+ \\ & right\_vote\ [f, c] * ((q_f - p_{left}) / (p_{right} - p_{left})) \end{aligned} \tag{6}$$

Final vote for any class $c$ is the sum of all votes given by the features. If there exists a class $i$ that uniquely gets the highest vote, then it is predicted to be the class of the query instance. The certainty factor of the classification is computed as follows:

$$C_f = \frac{final\ vote\ [i]}{\sum_{c=1}^{\#Classes} final\ vote\ [c]} \tag{7}$$

```
VFFP_query(q)          /* q: query instance*/
   feature_vote[f,c] = 0 (∀f, c)
   for each feature f
      if f is nominal
         p = find_point(f,q_f)
         if such a p exists
            for each class c
               Equation 2
            normalize_feature_votes (f)

      else if f is numeric
         if membership value of t_f = 1
            Let l be the linguistic term that q_f falls in
            for each class c
               Equation 3
            normalize_feature_votes (f)
         else
            Let q_f be between parameters p_left and p_right
            Let left be the lefthandside linguistic term
            Let right be the righthandside linguistic term
            for each class c
               Equation 4
               Equation 5
               Equation 6
            normalize_feature_votes (f)

   for each class c
      final_vote [c] = Σ_{f=1}^{#Features} feature_vote [f, c]

   if min_{c=1}^{#Classes} final_vote[c] < final_vote [k] = max_{c=1}^{#Classes} final_vote[c]
      classify q as "k" with a certainty factor C_f
      return C_f
   else return -1
```

**Fig. 5.** Classification in VFFP

## 6 ICRIL Algorithm

ICRIL takes two input parameters: $R_p$ (The set of streaming classification rules of period $p$ and $MinC_t$ (Minimum Certainty Threshold)). It tries to classify the rules in $R_p$. If $C_f \geq MinC_t$ for a query rule $r$, this rule is inserted into the successfully classified rules set ($R_s$). Otherwise, two situations are possible: either the concept description is not able to classify $r$ ($C_f$ = -1), or the concept description's classification (prediction of $r$'s interestingness label) is not of sufficient strength. If $C_f < MinC_t$, rule $r$ is presented, along with its interestingness factor values such as *Coverage*, *Rule Size*, *Decisive* etc., to the user for classification. This rule is then inserted into the training rule set $R_t$ and the concept description is reconstructed incrementally.

All the rules in $R_p$ are labeled either automatically by the classification algorithm, or manually by the user. User participation leads learning process to be interactive. When the number of instances in the training rule set increases, the concept description learned tends to be more powerful and reliable. ICRIL executes on classification rules of all the periods and finally concludes by presenting the labeled

rules in $R_s$. VFFP, the classification algorithm used in ICRIL, is more predictive than VFP, the classification algorithm used in IRIL [7]. For numeric features, VFP learns gaussian probability distribution functions for each class $c$, where as VFFP learns linguistic term class counts for each class $c$. The user gives the parameters of the membership functions of each numeric feature as inputs in the case of VFFP. However, there is no such a situation in VFP.

```
ICRIL (R_p, MinC_t)
   if p is the 1st period          //Warm-up Period
      R_t ← ∅,    R_s ← ∅
      for each rule r ∈ R_p
               ask the user to classify r
               set C_f of this classification to 1
               insert r into R_t
               VFFP_train (r)
   else
      for each rule r ∈ R_p
         C_f ←VFFP_query (r)
            if C_f < MinC_t
               ask the user to classify r
               set C_f of this classification to 1
               insert r into R_t
               VFFP_train (r) //Update Concept Description
            else
               insert r into R_s
   return rules in R_s
```

**Fig. 6.** ICRIL algorithm

## 7 Experimental Results

ICRIL was tested to classify 1555 streaming classification rules induced from a financial distress domain between years 1989 and 1998. Each year has its own data and classification rules induced by using a benefit maximizing rule learner proposed in [1]. The data set of the financial distress domain is a comprehensive set consisting of 25632 data instances and 164 determining features (159 numeric, 5 nominal). There are two classes: "Succeed" and "Fail". The data set includes some financial information about 3000 companies collected during 13 years and the class feature states whether the company succeeded for the following three years. Domain expert previously labeled all the 1555 induced rules by an automated process to make accuracy measurement possible. Rules of the first year are selected as the warm-up rules to construct the initial concept description.

**Table 2.** Results for ICRIL

| | $MinC_t$ 51% | $MinC_t$ 53% | $MinC_t$ 55% | $MinC_t$ 57% |
|---|---|---|---|---|
| Number of rules | 1555 | 1555 | 1555 | 1555 |
| Number of rules classified automatically with high certainty | 1359 | 1294 | 1202 | 1048 |
| User participation | 13% | 17% | 23% | 33% |
| Overall Accuracy | 90% | 93% | 94% | 97% |

**Table 3.** Results for IRIL

| | $MinC_t$ 51% | $MinC_t$ 53% | $MinC_t$ 55% | $MinC_t$ 57% |
|---|---|---|---|---|
| Number of rules | 1555 | 1555 | 1555 | 1555 |
| Number of rules classified automatically with high certainty | 1344 | 1286 | 1196 | 1096 |
| User participation | 13% | 17% | 23% | 29% |
| Overall Accuracy | 80% | 82% | 86% | 88% |

In Table 2, results for $MinC_t = 51\%$ show that 1359 rules are classified automatically with $C_f > MinC_t$. User participation is 13% in the classification process. In the classification process, it is always desired that rules are classified automatically, and user participation is low.

The accuracy values generally increase in proportion to the $MinC_t$. Because higher the $MinC_t$, higher the user participation is. And higher user participation leads to learn a more powerful and predictive concept description. ICRIL achieves better accuracy values than IRIL, whose results are shown in Table 3.

## 8 Conclusion

ICRIL feature projection based, interactive classification rules' interestingness learning algorithm was developed and gave promising experimental results on streaming classification rules induced on a financial distress domain, when compared to our previous study in [7]. VFFP, the concept description learner developed in the course of ICRIL, makes use of less but more meaningful interestingness factors when compared to our previous study in [7].

## References

1. Güvenir, H.A., "Benefit Maximization in Classification on Feature Projections" *Proceedings of the 3rd IASTED International Conference on Artificial Intelligence and Applications* (AIA'03), 2003, 424-429.
2. Klemettinen, M., Mannila, H., Ronkainen, P., Toivonen, H., and Verkamo, A.I., "Finding interesting rules from large sets of discovered association rules" *Proceedings of the 3rd Int. Conf. on Information and Knowledge Management*, 1994, 401-407.
3. Liu, B., Hsu, W., and Chen, S., "Using general impressions to analyze discovered classification rules" *Proceedings of the 3rd Int. Conf. on KDD*, 1997, 31-36.
4. Liu, B., and Hsu, W., "Post-analysis of learned rules", *AAAI*, 1996, 828-834.
5. Hussain, F., Liu, H., Suzuki, E., and Lu, H., "Exception rule mining with a relative interestingness measure" *Proceedings of the Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 2000, 86-97.
6. Dong, G., and Li, J., "Interestingness of discovered association rules in terms of neighborhood-based unexpectedness" *Proceedings of the 2nd Pacific-Asia Conference on Knowledge Discovery and Data Mining*, 1998, 72-86.
7. Aydın, T., and Güvenir, H.A., "Learning Interestingness of Streaming Classification Rules" *Proceedings of 19th International Symposium on Computer and Information Sciences* (ISCIS 2004), Antalya, Turkey (Oct. 27-29, 2004), LNCS 3280, Springer-Verlag, Cevdet Aykanat, Tugrul Dayar and Ibrahim Korpeoglu (Eds.), 62-71.

# Refining the Progressive Multiple Sequence Alignment Score Using Genetic Algorithms

Halit Ergezer[1] and Kemal Leblebicioğlu[2]

[1] Department of Computer Engineering Başkent University,
06530 Ankara, Turkey
ergezer@baskent.edu.tr
[2] Department of Electrical and Electronics Engineering,
Middle East Technical University
Ankara, Turkey
kleb@metu.edu.tr

**Abstract.** Given a set of $N$ ($N$>2) sequences, the Multiple Sequence Alignment (MSA) problem is to align these $N$ sequences, possibly with gaps, that bring out the best score due to a given scoring criterion between characters. Multiple sequence alignment is one of the basic tools for interpreting the information obtained from bioinformatics studies. Dynamic Programming (DP) gives the optimal alignment of the two sequences for the given scoring scheme. But, in the case of multiple sequence alignment it requires enormous time and space to obtain the optimal alignment. The time and space requirement increases exponentially with the number of sequences. There are two basic classes of solutions except the DP method: progressive methods and iterative methods. In this study, we try to refine the alignment score obtained by using the progressive method due to given scoring criterion by using an iterative method. As an iterative method genetic algorithm (GA) has been used. The sum-of-pairs (SP) scoring system is used as our target of optimization. There are fifteen operators defined to refine the alignment quality by combining and mutating the alignments in the alignment population. The results show that the novel operators, *sliding-window*, *local-alignment*, which have not been used up to now, increase the score of the progressive alignment by amount of % 2.

## 1 Introduction

MSA is the most useful tool for interpreting the information obtained from bioinformatics studies. It used for classifying proteins, predicting the secondary and tertiary structure of the proteins, to help database search, which used for the demonstration of homology between new sequences and existing sequences, to construct the phylogenetic trees, and molecular modeling. DP gives the optimal alignment of the two sequences for the given scoring scheme. But in the case of multiple sequence alignment it requires enormous time and space to obtain the optimal alignment. The time and space requirement increases exponentially with the number of sequences. There are two basic solutions except the exact method, progressive methods and iterative methods. In progressive methods, pairwise alignments are used to obtain a multiple alignment and the order of the pairwise alignments is important. The iterative methods try

F.A. Savacı (Ed.): TAINN 2005, LNAI 3949, pp. 177 – 184, 2006.

to optimize the objective score iteratively. The brief definition of the pairwise alignment and multiple alignment will be given in the next section and then the progressive alignment method that uses the pairwise alignment for multiple alignment will be described. The particular iterative method for refining the alignment will be described. In the refining step we will briefly explain the operators defined. The *sliding-window* operators and the *local-alignment* operators are the contributions of this research and very useful operators. They alone increase the score value up to % 2 when we apply only these operators.

### 1.1 Pairwise Alignment

Given two sequences S and T defined on the same alphabet Σ. An alignment *A* maps S and T into sequences $S^*$ and $T^*$, respectively. The characters of the sequences $S^*$ and $T^*$ are in the alphabet $\Sigma^* = \Sigma \cup \{\text{'-'}\}$ where '-' denotes the gap symbol. The sequences $S^*$ and $T^*$ must satisfy two conditions;

1. $|S^*| = |T^*| = l$
2. The removal of all spaces from $S^*$ and $T^*$, leaves S and T, respectively,

where |.| denotes the length of the sequence. The score of the alignment is given by

$$\text{score}(S^*, T^*) = \sum_{i=1}^{l} \sigma(S^*[i], T^*[i])$$

where S*[i] denotes the $i^{th}$ character of the sequence S, and $\sigma(x,y)$ denotes the score (or penalty) of aligning the characters *x* and *y* which are in $\Sigma^*$.

### 1.2 Multiple Sequence Alignment (MSA)

In a similar way, we can define the multiple sequence alignment problem. Given *N* sequences (*N*>2) $S_1, S_2, \ldots, S_N$ defined on the alphabet Σ. An alignment *A* maps $S_1, S_2, \ldots, S_N$ into $S_1^*, S_2^*, \ldots, S_N^*$ sequences, respectively. The characters of the sequences $S_1^*, S_2^*, \ldots, S_N^*$ are in the alphabet $\Sigma^* = \Sigma \cup \{\text{'-'}\}$ where '-' denotes the gap symbol. The sequences $S_1^*, S_2^*, \ldots, S_N^*$ must satisfy the conditions;

1. $|S_1^*| = |S_2^*|, \ldots, |S_N^*| = l$
2. The removal of all spaces from $S_i^*$, $(1 \le i \le N)$, leaves $S_i$, respectively.

The sum-of-pairs (SP) scoring system is used as our target of optimization. This scoring system has been chosen since it conforms to accelerated computation, reasonably reflects correct alignments and easily incorporates proper gap penalties. Total score is calculated by the following;

$$\text{SCORE}(A) = \sum_{i=1}^{N} \sum_{\substack{j=1 \\ i \neq j}}^{N} score(S_i, S_j)$$

In the refining step to accelerate the score calculation, we have used the column histograms. That is, we count each character in the column and then using these values scores are calculated.

### 1.3 Optimal Pairwise Alignment

Given sequences S and T defined on the same alphabet we can find the optimal pairwise alignment, due to given scoring function, using dynamic programming (DP). DP depends upon the representation of a multistage decision problem as a sequence of single-stage problems; that is, one casts an n-variable problem into a sequence of n single-variable problems, which are solved successively. The recursive definition of the DP is given by

$$V(i,j) = \max\{\ V(i-1,j-1) + \sigma(S[i],T[j]),\\ V(i-1,j) + \sigma(S[i],-),\\ V(i,j-1) + \sigma(-,T[j])\}$$

with the base conditions;

$$\mathrm{V(i,0)} = \sum_{k=1}^{i} \sigma(S[k],-)\ ,\ \ \mathrm{V(0,j)} = \sum_{k=1}^{j} \sigma(-,T[k])$$

In the figure 1, score matrix, traceback, and the optimal alignment of the two sequences are shown. The memory requirement of this algorithm is O(*n.m*), where *n* and *m* is the lengths of the S and T, respectively.

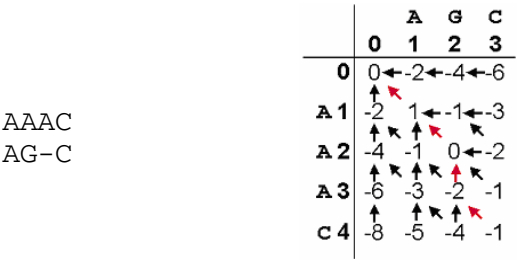



**Fig. 1.** Score matrix, traceback, and optimal alignment

As we can see from this requirement the memory requirement of the optimal multiple alignment increases exponentially with the number of sequences. For example, the memory requirement for the optimal alignment of 4 sequences of length 1000 is 2TB.

## 2 Progressive Alignment

The most natural application of the pairwise alignment method to *N*-dimensional problem would be the progressive method. In this method order of the sequences is very important. The center-star [2], [3] method has been applied to obtain as a progressive method.

### 2.1 Center-Star Algorithm

This approximation algorithm is as follows. The input is a set $\Omega$ of *N* sequences. First find $S_1 \in \Omega$ that minimizes

$$\sum_{S \in \Omega - \{S_1\}} D(S_1, S)$$

where $D(S_1,S)$ is the value of the global alignment of $S_1$ and S sequences.

This can be done by running the DP algorithm of 1.3 on each of $\binom{N}{2}$ pairs of sequences in $\Omega$. Call the remaining sequences in $\Omega$ $S_2,\ldots S_N$. Add these sequences $S_2,\ldots S_N$ one at a time to a multiple alignment that initially consists only of $S_1$ as follows.

Suppose $S_1, S_2,\ldots S_N$ are already aligned as $S_1^*, S_2^*,\ldots S_{i-1}^*$. To add $S_i$, run the DP on $S_i$ and $S1^*$ to produce $S_1^{**}$ and $S_i^*$. Adjust $S_2^*,\ldots S_{i-1}^*$ by adding spaces to these columns where spaces where added to get $S_1^{**}$ from $S_1^*$. Replace $S_1^*$ from $S_1^{**}$. This algorithm runs in time $O(N^2 \cdot m^2)$ [2], [3] when given N sequences each of length at most m.

## 3 Refining Alignment Using Genetic Algorithms

Generally, there are three methods for MSA; exact solution, progressive methods, and iterative methods. The main disadvantage of progressive methods is the failure to correct for errors introduced in an early phase of the procedure. Iterative methods try to optimize the score of the alignment, which is initially randomly aligned or obtained by using a progressive method. The major drawback of the iterative methods is that the time requirement of the methods may be greater than exact solution approaches. To avoid this, we use the difference between two successive scores. If the change in the score is below % 0.01, then we stop the iterations.

### 3.1 Initial Population Generation

The members of the population are the alignments of the given sequences, which are obtained by using the center-star method. By mutating, inserting/deleting gap(s), changing gap positions in the result of the center-star algorithm initial alignment population is generated. The important point here is that the length constraint is not applied to each member of population in the same way; %10 of the population have been generated using the group alignment algorithm with the same order obtained in the center star algorithm.

### 3.2 Operators

We have used additional operators in addition to the operators that have been used in early studies [9], [10]. The *sliding-window* and *local-alignment* operators will be described briefly.

#### 3.2.1 The Sliding-Window Operators

This proposed operator is the new algorithm for (MSA). It is based on the local alignment of the sequences according to the column histogram. We select two points

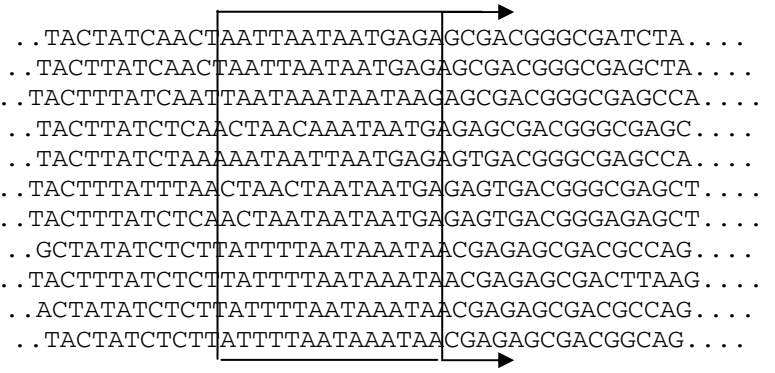



**Fig. 2.** The sliding window operator

that are aligned well, and apply this operator between these two points. In figure 2, we show the sequences and the window slid over sequences.

The first step of the algorithm starts with finding the histogram of the characters in the first column of the windowed sub-sequence. This can be accomplished with time complexity $O$ (k). Then, the histogram values are sorted in ascending order; time complexity of this step is $O$ (|Σ| log (|Σ|)), where |Σ| denotes the number of characters in the alphabet under consideration. At the end of the first step a "match" decision is made for the character that has the greatest repeating value. If there is more than one greatest repeating value we select one of them to decide on the "match" for this character. After deciding on the match case, next we must decide on the actions for other characters in the column under consideration. Since there are two possible actions for each character in the column under consideration ("mismatch" or "gap") the maximum number of possible actions in this step is $2^{|\Sigma|-1}$ if the number of sequences to be aligned is greater than or equal to the number of characters in the alphabet. This maximum number occurs when column includes the all the characters in the alphabet. There is an assumption that the decision for the same characters in current column is the same. If the movement of a character at one sequence to the right increases the score then it is obvious that, for the same column, moving this character(s) in other sequences will increase the score. Then we calculate the score of the current column for each possible case and construct the first level of the tree shown in figure 3. For each possible action, we also compute the relative positions of the each sequence. That is, if we decide on the "gap" for one character than the position for the sequence(s) which includes that character at the current column must be increased by one. Using this relative positions we compute the possible actions for the second column in the window in the same way and construct the next levels of the tree shown in figure 3. In this figure left-most branch represents the "mismatch " decision for all other characters, and right-most branch represents the movement of all characters to the right by one position. There are four different types of sliding-window operator each having a different length.

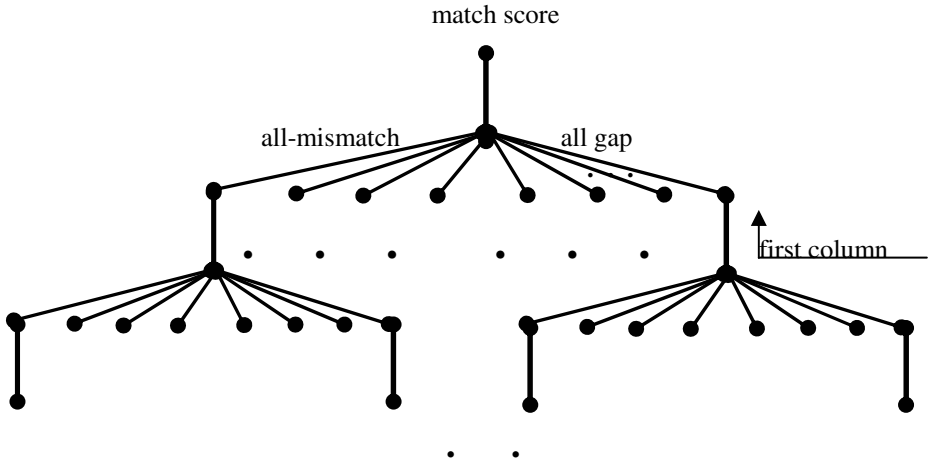



**Fig. 3.** Decision Tree, each line represents scores and positions of the decision

### 3.2.2 Local Alignment Operator

This operator has been used to align the regions that have many mismatches and gaps as follows. It finds the points for each sequence in the alignment and taking these local regions. There is a constant window size for each sequence and a margin value. After taking the sub-sequences, we remove the gaps and then align these sub-sequences using the center star method. Then, aligned sequences are replaced by the originals. This operator is able to correct the errors in the early phases of the algorithms.

### 3.2.3 Local Shuffling Operators

The local shuffling operator is the operator that changes the gap position to obtain better score value as illustrated below.

```
...                                ...
CACCC-TAA------------              CACCC-TAA------------
GGCCCATA-G---ATT-T---  →           GGCCCATA-G---ATT---T-
TA--CAT--GTTTATTGAAT-              TA--CAT--GTTTATTGAAT-
```

In these operators the number of gaps can be selected. In the example above, the number of gaps to be changed is two.

### 3.2.4 Crossover Operators

There are two different crossover operators: single point crossover and multi (two-point) crossover. The single point crossover operator randomly selects a point on the first sequence and counts the characters from the left for each sequence to obtain the same number of the characters (except gap symbol) in the left for all sequences. Any two members of the population are mated as shown in below.


**Fig. 4.** Single Point Crossover

The two-point crossover operates like single point crossover except it selects two randomly crossover points.

## 4 Results

In Table 1, we give the refinement percentages of our contributions, and overall refinement percentage of the alignment score obtained using progressive method.

**Table 1.** Results: The refinement percentage of progressive alignment scores

| Number of Sequences | Lengths of sequences | % Refinement of *sliding-window* operator | % Refinement of *local-alignment* operator | % Overall Refinement |
|---|---|---|---|---|
| 23 | 430-443 | 0.61 | 0.25 | 3.21 |
| 84 | 1168-1417 | 1.70 | 0.32 | 7.52 |
| 72 | 2851-2884 | 2.03 | 0.89 | 9.81 |
| 29 | 9033-9111 | 1.58 | 1.27 | 6.27 |
| 420 | 532 | 1.32 | 0.28 | 6.56 |

## 5 Conclusions

We can conclude that to minimize the effects of drawbacks of progressive methods and iterative methods, it is useful to use both classes of methods in a hybrid manner. Since in general progressive methods fail to correct errors arising in the early phases of the procedure and the time requirement of iterative methods may be greater than the time required to obtain the optimal solution using DP, we have used these two

methods together. The results given in this paper show that the disadvantages of these methods may be eliminated with such an approach. To obtain better scores, new operators should be defined or the starting alignment should be obtained using another progressive method.

## References

1. B. Morgenstern, A.W.M. Dress and T. Werner: Multiple DNA and protein sequence alignment based on segment-to-segment comparison, Proc. N&l *Acad. Sci. USA 93,* (1996). 12098-12103
2. D. Gusfield: Efficient methods for multiple sequence alignment with guaranteed error bounds. *Bulletin of Mathematical Biology*, (1993) 55:141–154
3. D.Gusfield: Algorithms on Strings, Trees, and Sequences, Computer Science and Computational Biology, Cambridge University Press, Cambridge (1997)
4. L. Wang and T. Jiang: On the complexity of multiple sequence alignment. *Journal of Computational Biology*, (1994) 1:337–348
5. M.S. Waterman: Introduction to Computational Biology: Maps, Sequences, and Genomes, Chapman & Hall, London. (1995)
6. S. Altschul, D. Lipman: Trees, stars, and multiple sequence alignment, SIAM J. Appl. Math. 49 (1989) 197-209.
7. S.B. Needleman and C.D. Wunsch: A general method applicable to the search for similarities in the amino acid sequence of two proteins, J. Mol. Biol. 48, (1970) 443-453.
8. Notredame,C. and Higgins,D.G.: SAGA: Sequence Alignment by Genetic Algorithm. *Nucleic Acids Res.*, **24**, (1996) 1515–1524.
9. Notredame,C., O'Brien,E.A. and Higgins,D.G.: RAGA: RNA sequence alignment by genetic algorithm. *Nucleic Acids Res.*, **25**, (1997) 4570–4580.

# An Evolutionary Local Search Algorithm for the Satisfiability Problem

Levent Aksoy and Ece Olcay Gunes

Istanbul Technical University Faculty of Electrical and Electronics Engineering Department of Electronics and Communication Engineering 34469 Maslak Istanbul Turkey
levent@ehb.itu.edu.tr, ece.gunes@itu.edu.tr

**Abstract.** Satisfiability problem is an NP-complete problem that finds itself or its variants in many combinatorial problems. There exist many complete algorithms that give successful results on hard problems, but they may be time-consuming because of their branch and bound structures. In this manner, many successful incomplete algorithms are introduced. In this paper, the improvement of incomplete algorithms is of interest and it is shown that the incomplete algorithms can be more efficient if they are equipped with the problem specific knowledge, goal-oriented operators, and knowledge-based methods. In this aspect, an evolutionary local search algorithm is implemented, tested on a randomly generated benchmark that includes test instances with different sizes, and compared with prominent incomplete algorithms. Also, effects of goal-oriented genetic operators and knowledge-based methods used in the evolutionary local search algorithm are examined by making comparisons with blind operators and random methods.

## 1 Introduction

The satisfiability (SAT) problem is a paradigmatic NP-complete problem shown by Cook [1] and can be stated as to determine an assignment of variables given in a conjunctive normal form of a Boolean formula that makes the formula true. The Boolean formula consists of clauses that are in a disjunction of literals where a literal is a Boolean variable or its negation. Many NP-complete problems in Electronic Design Automation (EDA) like VLSI design, circuit synthesis [2], computer aided design, and test pattern generation [3] can be represented by a Boolean formula in a conjunctive normal form and formulated as an SAT problem. So, finding a method for an SAT algorithm by which the algorithm finds solutions for hard problems more quickly and efficiently can be accepted as an improvement. Also, the development of efficient methods for the SAT problem can be exploited for solving combinatorial optimization problems.

Existing methods for the SAT problem can be classified into two categories: complete and incomplete methods. Complete methods [4,5] find a solution if it exists or prove that the formula is unsatisfiable considering exponential worst-case complexity. Incomplete methods [6-12] can find a solution quickly if the formula is satisfiable but cannot show that the formula is unsatisfiable without using any additional techniques and may not find a solution when the formula is satisfiable.

F.A. Savacı (Ed.): TAINN 2005, LNAI 3949, pp. 185 – 193, 2006.

Incomplete methods are based on heuristic algorithms such as local search [6,7], simulated annealing [8], tabu search [9], and evolutionary algorithms [10,11,12]. Incomplete methods or hybrid of these methods are applied to the SAT problem successfully and promising results have been obtained.

Rest of the paper is organized as follows. In Section 2, an Evolutionary Local Search Algorithm called ELSA that includes goal-oriented and knowledge-based operators and uses the problem specific knowledge is introduced briefly. Afterwards, the properties of ELSA and other successful incomplete algorithms are described. In Section 4, test instances are introduced and experimental results of the incomplete algorithms on these test instances are given. Effects of methods and genetic operators used in ELSA are examined in Section 5 and finally conclusion is given.

## 2 Evolutionary Local Search Algorithm (ELSA)

ELSA is an evolutionary algorithm with a local search method designed for the SAT problem. ELSA consists of two main parts: local search and evolutionary algorithm. The local search focuses the search on promising solutions, as the evolutionary algorithm accomplishes the tasks of finding more promising solution candidates (selection and crossover) and avoiding from local maximum points (mutation).

In both the evolutionary algorithm and the local search part, an individual (solution candidate) in the population is represented in a bit string where each variable is associated with one bit. The fitness of an individual is determined as the number of clauses that are satisfied by the individual. The aim of ELSA is to find an assignment of variables that satisfies all (maximum) clauses in the given formula. So, the SAT problem is considered as a maximization problem.

Initial population is formed by the knowledge that exists in the given formula. Firstly, for each variable, the number of clauses that are satisfied by both 0 and 1 values of a variable is found. Then, the probability of assigning 1 value to each variable is determined with the ratio between the number of satisfied clauses by the 1 value of the variable and the number of satisfied clauses with both 0 and 1 values of the variable. In construction phase of the initial population, a random number between 0 and 1 is generated and the value of each variable is determined as 1, if the generated random number is smaller than or equal to its probability of assigning value 1, otherwise its value is determined as 0. By doing so, the initial population is aimed to converge to local/global maximum points in the search space more quickly.

In the evolutionary algorithm part, a mating pool is formed by individuals selected from the population with a tournament selection between two individuals according to their fitness values. The crossover operator is applied on two individuals selected randomly from the mating pool. In crossover, initially, variables in unsatisfied clauses are found for each parent. Then, the uniform crossover operator is applied on these variables in each parent. As a result, two offspring that share the knowledge necessary for their maximum fitness values are generated. So, searching in the non-solution areas near neighbourhoods determined by each parent is avoided by applying the crossover operator only on variables in unsatisfied clauses.

A local search method is incorporated into the evolutionary algorithm to improve individuals. Local search in ELSA is based on a greedy local search method called GSAT [6]. When an individual with its fitness value is taken as an input to the local

search, initially its fitness value is determined as a current fitness value and variables in unsatisfied clauses are found. Then, for each variable in unsatisfied clauses fitness values are evaluated, when values of these variables are flipped separately. The variables that yield the largest increase according to the current fitness value is found, one of these variables is chosen randomly, its value after flip is accepted, and its fitness value is determined as the current fitness value. Such moves taken in the search space are determined as uphill moves. If the largest fitness value after the values of variables in unsatisfied clauses are flipped separately is equal to the current fitness value, then sideway moves are taken. The number of sideway moves to be taken in local search is accepted as the number of variables and whenever there is an uphill move, the counter held for sideway moves is reset. If the limit number of sideway moves is reached, or the largest fitness value after the values of variables in unsatisfied clauses are flipped separately is smaller than the current fitness value (downhill moves), local search is aborted and returned to the evolutionary algorithm with the individual that has the current fitness value, otherwise local search is continued with the individual that has the current fitness value.

In the evolutionary algorithm part, an individual is determined as a local maximum point, if its fitness value is not increased by the local search applied after the crossover operation. To avoid from this local maximum point, bitwise mutation operator is applied on the individual. By doing so, escaping from local maximum points is aimed.

The procedure of ELSA expressed in pseudocode is given below.

Procedure of ELSA

```
Initialize population
Apply local search
if a solution is found then return "a satisfying
assignment is found"
while (stopping criteria is not reached & no satisfying
assignment found) do
  Select parents
  Recombine pairs of parents
  Apply local search
  if the offspring is determined as a local maximum
  point then
    Mutate offspring
    Apply local search
  end if
  Replace parents by offspring
end do
return "no satisfying assignment found"
```

## 3 Properties of Incomplete Algorithms

There are many algorithms designed for the SAT problem that give successful results and have great influence on other efficient algorithms. In this section, prominent incomplete algorithms that have similar structure with ELSA are described and

properties of these algorithms and ELSA are given. The experimental results given in Section 4 are based on the properties of incomplete algorithms given in this section.

WSAT [7] is the most popular local search method implemented for the SAT problem. In WSAT, initially, a solution candidate is generated randomly and then, to improve a solution candidate, an unsatisfied clause is selected randomly among the unsatisfied clauses and a variable in the selected clause is chosen with a heuristic. There are six proposed heuristics for the selection of a variable. One of these heuristics used in WSAT for the comparison with other incomplete algorithms is the heuristic BEST. According to this heuristic, the selection of a variable is accomplished by selecting a random variable with probability P, otherwise selecting the variable that yields the fewest number of unsatisfied clauses when its value is flipped. The probability P called noise is taken as 0.5.

FlipGA [10] is an evolutionary local search algorithm that generates offspring by blind genetic operators and subsequent improvements by means of local search. FlipGA employs population size 10, proportional parent selection, and a generational replacement scheme. Uniform crossover is always applied and a mutation operator is used with probability 0.9. Mutation operator flips each value of a variable with probability 0.5. A flip heuristic is applied to each individual after performing crossover and mutation by means of local search. The flip heuristic takes a solution candidate as an input and yields a solution candidate that cannot be improved by flipping any entry. It starts with a random permutation of variables. Then, scans the variables in order; each variable's value is flipped and its gain (that is the number of clauses that are satisfied after the flip minus the number of clauses that are satisfied before the flip) is determined. If the gain is greater than or equal to the zero, then the flip is accepted. The process is repeated if the fitness value of the obtained individual has been increased with respect to the previous scan of the variables.

ASAP [11] is a variant of FlipGA and is obtained from FlipGA by considering only one individual, (1+1) replacement, and an adaptive mechanism to control diversification in the search path. Mutation operator is always applied and flips the value of each variable with its probability value that is between 0 and 0.5, and adapted during execution. Next, flip heuristic as in FlipGA is applied to improve the individual. Moreover, the adaptive mechanism is based on tabu search that is employed for prohibiting the flip of some variables and controlling the mutation rate of each variable.

ELSA as described in Section 2 uses generational without elitism replacement method. The population size is determined to be 10. Crossover operator is always applied. In crossover, values of variables in unsatisfied clauses of each parent are flipped with probability 0.8, if they differ in both parents. Bitwise mutation operator is applied on every gene of an offspring with probability 0.5.

## 4 Experimental Results

The incomplete algorithms with their properties described in Section 3 are tested on a randomly generated benchmark given in [12]. These test instances that are forced to be satisfiable are determined as hard instances with the number of clauses ($m$) and variables ($n$) ratio ($m/n$) 4.3. The test instances are given in Table 1.

**Table 1.** Test instances

| Suite | Problem Size ($n$) | Test Instances for each $n$ | Total Test Instances |
|---|---|---|---|
| A | 30,40,50,100 | 3 | 12 |
| B | 50,75,100 | 50 | 150 |
| C | 20,40,60,80,100 | 100 | 500 |

Only ELSA is implemented among the incomplete algorithms described in Section 3 and experimental results of other algorithms are taken from [12]. Experimental results for WSAT are based on 10 runs for each test instance. FlipGA is run 50 times on Suite A and B, and 5 times on Suite C test instances. ASAP is run 50 times on Suite A, 10 times on Suite B, and 5 times on Suite C test instances. ELSA has the same run times as determined for ASAP. All the algorithms are terminated if a solution is found or the limit of 300000 bit flips is reached. The number of bit flips includes the flips made in crossover, mutation, and local search. Also, all the algorithms are evaluated with Success Rate (SR), successful runs where a solution is found over total runs, and Average Flip to a Solution (AFS), average number of bit flips made in successful runs. Experimental results are given in Table 2, 3, and 4 for Suite A, B, and C respectively.

**Table 2.** Experimental results for Suite A

| Algorithm | $n = 30$ | | $n = 40$ | | $n = 50$ | | $n = 100$ | |
|---|---|---|---|---|---|---|---|---|
| | SR | AFS | SR | AFS | SR | AFS | SR | AFS |
| WSAT | 1.00 | 1631 | 1.00 | 3742 | 1.00 | 15384 | 0.80 | 19680 |
| FlipGA | 1.00 | 25490 | 1.00 | 17693 | 1.00 | 127900 | 0.87 | 116653 |
| ASAP | 1.00 | 9550 | 1.00 | 8760 | 1.00 | 68483 | 1.00 | 52276 |
| ELSA | 1.00 | 1367 | 1.00 | 1045 | 1.00 | 3572 | 0.92 | 35869 |

**Table 3.** Experimental results for Suite B

| Algorithm | $n = 50$ | | $n = 75$ | | $n = 100$ | |
|---|---|---|---|---|---|---|
| | SR | AFS | SR | AFS | SR | AFS |
| WSAT | 0.95 | 16603 | 0.84 | 33722 | 0.60 | 23853 |
| FlipGA | 1.00 | 103800 | 0.82 | 29818 | 0.57 | 20675 |
| ASAP | 1.00 | 61186 | 0.87 | 39659 | 0.59 | 43601 |
| ELSA | 1.00 | 6016 | 0.95 | 42488 | 0.63 | 57603 |

**Table 4.** Experimental results for Suite C

| Algorithm | $n = 20$ | | $n = 40$ | | $n = 60$ | | $n = 80$ | | $n = 100$ | |
|---|---|---|---|---|---|---|---|---|---|---|
| | SR | AFS | SR | AFS | SR | AFS | SR | AFS | SR | AFS |
| WSAT | 1.00 | 334 | 1.00 | 5472 | 0.94 | 20999 | 0.72 | 30168 | 0.63 | 21331 |
| FlipGA | 1.00 | 1073 | 1.00 | 14320 | 1.00 | 127520 | 0.73 | 29957 | 0.62 | 20319 |
| ASAP | 1.00 | 648 | 1.00 | 16644 | 1.00 | 184419 | 0.72 | 45942 | 0.61 | 34548 |
| ELSA | 1.00 | 88 | 1.00 | 1687 | 1.00 | 12360 | 0.86 | 48273 | 0.70 | 53726 |

Experimental results show that the evolutionary algorithms with local search methods except ELSA are competitive with the prominent local search method and ELSA has better performance than WSAT on these test instances. Among the evolutionary algorithms with local search, ELSA finds solutions with higher success rates (except Suite A, $n = 100$) and less number of flips according to the evolutionary algorithms that use blind genetic operators and random methods.

## 5 Effects of Methods and Genetic Operators in ELSA

In this section, methods and genetic operators used in ELSA are examined on Suite A test instances with the same termination conditions and the number of run given in Section 4 to demonstrate the performance of ELSA on different situations.

One of the main parts of ELSA is the local search. In [13], local search method used in ELSA and FlipGA are examined, tested on the test instances given in Table 1, and it is seen that local search method used in ELSA gives more promising results on these test instances. Also, these local search methods are incorporated into a simple genetic algorithm where solutions are obtained with higher success rates and less number of flips with a simple genetic algorithm including the local search method used in ELSA. Additionally, the local search methods are examined on the test instances where the number of variables is greater than 100 and it is seen that the flip heuristic becomes competitive with the local search method used in ELSA.

In ELSA, taking sideway moves during the local search plays a great role in finding a solution more quickly. So, initially the local search with sideway moves is compared with the local search without sideway moves. Experimental results are given in Table 5.

**Table 5.** Effect of sideway moves in local search

| Local Search | $n = 30$ | | $n = 40$ | | $n = 50$ | | $n = 100$ | |
|---|---|---|---|---|---|---|---|---|
| | SR | AFS | SR | AFS | SR | AFS | SR | AFS |
| Without sideway moves | 1.00 | 2102 | 1.00 | 3129 | 1.00 | 12256 | 0.75 | 29462 |
| With $n$ sideway moves | 1.00 | 1367 | 1.00 | 1045 | 1.00 | 3572 | 0.92 | 35870 |
| With $5n$ sideway moves | 1.00 | 1836 | 1.00 | 1293 | 1.00 | 5562 | 0.77 | 33329 |

As can be seen from the results, sideway moves in the local search that walk on the plateaus and explore the search space more deeply are necessary to find a solution more quickly. Besides, because the search on the plateaus may yield an improvement in the fitness value, determining the number of sideway moves to be taken in the local search or when to give up searching is important not to make unnecessary flips. Local maxima are detected when all one-step transitions from the current individual reduce the fitness function. However, there are a number of individuals that have equal fitness value and are interconnected by one-step transitions. The plateau is a local maximum, if none of the individuals can be improved by single-step transitions. So, in ELSA, the number of sideway moves to be taken in the local search is determined as the number of variables as given in [14]. Experimental results with $5n$ sideway

moves taken in the local search are given in Table 5 and it can be seen that unnecessary flips that effect SR and AFS are made, when the number of sideway moves is increased.

After an individual is defined as a local maximum in the evolutionary part of ELSA, the mutation operator is applied as a restart. Because the probability of flipping the value of a variable is important, two additional 0.1 and 0.9 probability values are taken as mutation rates and tested on Suite A. Experimental results are given in Table 6.

**Table 6.** Effect of mutation rate in ELSA

| Mutation Rate | $n = 30$ | | $n = 40$ | | $n = 50$ | | $n = 100$ | |
|---|---|---|---|---|---|---|---|---|
| | SR | AFS | SR | AFS | SR | AFS | SR | AFS |
| 0.1 | 1.00 | 12179 | 0.93 | 13769 | 0.77 | 17393 | 0.63 | 23269 |
| 0.5 | 1.00 | 1367 | 1.00 | 1045 | 1.00 | 3572 | 0.92 | 35870 |
| 0.9 | 1.00 | 3807 | 1.00 | 8240 | 1.00 | 11372 | 0.95 | 35507 |

As expected from 0.1 probability value that is not a real restart and only effects on a few variables, very poor results are obtained in means of SR and AFS with respect to other probability values. With the mutation rate 0.9, many genes of an individual are flipped and as a result, a quite different individual is obtained with ignoring its previous knowledge in reaching a solution. So, many flips are required to find a solution. However, higher mutation rates are needed to explore the search space in a wide range, when the search space is too large ($n = 100$) to explore. With the mutation rate 0.5 used in ELSA, an individual that is different, but also carries its former characteristics is created and a solution is obtained with less number of flips.

In ELSA, the crossover operator is applied on individuals to share their knowledge on variables in unsatisfied clauses, to create more promising solution candidates, and to reach a solution quickly. It is similar to a uniform crossover expect the variables applied on. The goal-oriented uniform crossover is compared with a blind uniform crossover and experimental results are given in Table 7.

**Table 7.** Effect of crossover operator in ELSA

| Crossover Operator | $n = 30$ | | $n = 40$ | | $n = 50$ | | $n = 100$ | |
|---|---|---|---|---|---|---|---|---|
| | SR | AFS | SR | AFS | SR | AFS | SR | AFS |
| Blind uniform | 1.00 | 1682 | 1.00 | 968 | 1.00 | 4111 | 0.87 | 24882 |
| Goal-oriented uniform | 1.00 | 1367 | 1.00 | 1045 | 1.00 | 3572 | 0.92 | 35870 |

It is seen that the goal oriented uniform crossover gives better results ($n = 30, 50, 100$) than the blind uniform crossover, because it focuses on the necessary areas where a solution is found and avoids taking unnecessary moves.

Initialization of the population is accomplished by creating an individual where the value of each variable yields the largest number of satisfied clauses. This method is similar to a heuristic in a complete algorithm that selects the nodes and edges in the

construction of a search tree so that the largest number of clauses is satisfied. The initialization method used in ELSA is compared with the random initialization method. Experimental results are given in Table 8.

**Table 8.** Effect of initialization method in ELSA

| Initialization Method | $n = 30$ | | $n = 40$ | | $n = 50$ | | $n = 100$ | |
|---|---|---|---|---|---|---|---|---|
| | SR | AFS | SR | AFS | SR | AFS | SR | AFS |
| Random | 1.00 | 1213 | 1.00 | 740 | 1.00 | 4267 | 0.88 | 33558 |
| Problem specific | 1.00 | 1367 | 1.00 | 1045 | 1.00 | 3572 | 0.92 | 35870 |

Although the local search method applied after the initialization of the population can decrease the efficiency of any method used in the construction of the population, the initialization with the problem specific knowledge is seemed to be effective ($n$ = 50, 100) and it is seen that this method orientates the individuals to local/global maximum points in the search space.

Population size is expected to be very small according to the standard evolutionary algorithms, when a local search method and a mutation operator are applied. Experiments with the population size 4 and 20 are carried out on Suite A, and results are given in Table 9.

**Table 9.** Effect of population size in ELSA

| Population Size | $n = 30$ | | $n = 40$ | | $n = 50$ | | $n = 100$ | |
|---|---|---|---|---|---|---|---|---|
| | SR | AFS | SR | AFS | SR | AFS | SR | AFS |
| 4 | 1.00 | 1516 | 1.00 | 986 | 1.00 | 3213 | 0.89 | 36798 |
| 10 | 1.00 | 1367 | 1.00 | 1045 | 1.00 | 3572 | 0.92 | 35870 |
| 20 | 1.00 | 1062 | 1.00 | 849 | 1.00 | 5021 | 0.93 | 38213 |

When the population size is 4, the same results in means of AFS as the results with population size 10 are obtained, and solutions with higher success rates and less number of flips ($n$ = 30, 40, and 100) are obtained when the population size is 20, because it offers diversity in the population and reveals the efficiency of crossover.

As a result, it can be seen that ELSA is more effective with goal-oriented operators and knowledge-based methods according to blind operators and random methods. Also, it is believed that ELSA can be improved with the adaptive mechanism and tabu search as given in ASAP and these topics are determined as future work on the development of ELSA.

## 6 Conclusion

It is shown that evolutionary local search algorithms are competitive with local search algorithms and they can be improved using knowledge-based methods and goal-oriented genetic operators. Searching in non-solution areas, making so many

unnecessary flips, and spending so much time can be prevented and a solution can be found with higher success rates and less number of flips by using these kinds of methods and genetic operators.

## References


1. Cook, S.: The Complexity of Theorem-Proving Procedures. Proceedings of Third Annual ACM Symposium on Theory of Computing, ACM, New York (1971) 151-158
2. Brayton, K. R., Sangiovanni-Vincentelli, L. A., McMullen, T. C., Hachtel, D. G.: Logic Minimization Algorithms for VLSI Minimization. Kluwer, Boston (1985)
3. Larrabee, T.: Efficient Generation of Test Patterns Using Boolean Satisfiability. IEEE Transactions on Computer-Aided Design, Vol. 11, No. 1. (1992) 4-15
4. Davis, M., Putnam, H.: A Computing Procedure for Quantification Theory. Journal of ACM, Vol. 7. (1960) 201-215
5. Marques-Silva, J. P., Sakallah, K. A.: GRASP: A Search Algorithm for Propositional Satisfiability. IEEE Transactions on Computers, Vol. 48, No. 5. (1999) 506-521
6. Selman, B., Levesque, H., Mitchell, D.: A New Method for Solving Hard Satisfiability Problems. Proceedings of Tenth National Conference on Artificial Intelligence, AAAI Press, California (1992) 440-446
7. Selman, B., Kautz, H., Cohen, B.: Noise Strategies for Improving Local Search. Proceedings of Twelfth National Conference on Artificial Intelligence, AAAI Press, California (1994) 337-343
8. Spears, W.M.: Simulated Annealing for Hard Satisfiability Problems. DIMACS Series in Discrete Mathematics and Theoretical Computer Science, Vol. 26. (1996) 533-558
9. Mazure, B., Sais, L., Gregoire, E.: Tabu search for SAT. Proceedings of the 14th National Conference on Artificial Intelligence and 9th Innovative Applications of Artificial Intelligence Conference (1997) 281-285
10. Marchiori, E., Rossi, C.: A Flipping Genetic Algorithm for Hard 3-SAT Problems. Proceedings of Genetic and Evolutionary Conference, Morgan Kauffman, California (1999) 393-400
11. Rossi, C., Marchiori, E., Kok, J.: An Adaptive Evolutionary Algorithm for the Satisfiability Problem. Proceedings of ACM Symposium on Applied Computing, ACM, New York (2000) 463-469
12. Gottlieb, J., Marchiori, E., Rossi, C.: Evolutionary Algorithms for the Satisfiability Problem. Evolutionary Computation, Vol. 10, No. 1. (2002) 35-50
13. Aksoy, L., Tekin, O.A.: Hybridization of Local Search Algorithms with a Simple Genetic Algorithm for the Satisfiability Problem. Proceedings of International Symposium on Innovations in Intelligent Systems and Applications, (2005) 235-238
14. Hampson, S., Kibler, D.: Large Plateaus and Plateau Search in Boolean Satisfiability Problems: When to Give Up Searching and Start Again. DIMACS Series in Discrete Mathematics and Theoretical Computer Science, Vol. 26. (1996) 437-455

# HIS: Hierarchical Solver for Over-Constrained Satisfaction Problems

Zerrin Yumak and Tatyana Yakhno

Department of Computer Engineering, Dokuz Eylul University, Izmir, Turkey
{zyumak, yakhno}@cs.deu.edu.tr

**Abstract.** Constraint programming is an approach for solving mostly combinatorial problems by declaratively describing the problem and using special solving algorithms. Restrictions called constraints are stated over the problem variables that reduces the values each variable can take. In some cases it is not possible to satisfy all constraints or the user can state some preferences on them. One of the techniques to solve this kind of problems is modelling and solving the problem as a constraint hierarchy. This paper describes a hierarchical constraint solver HIS, which is developed in C++ using ILOG Solver as an ordinary constraint solver. HIS is based on refining algorithm. As an application a layout problem is considered.

## 1 Introduction

For the last decade Constraint Programming (CP) has become extremely popular area because of its potential to solve hard real life problems [9]. The main idea of CP is to formulate a problem declaratively, that is to specify variables and constraints over them and consequently find solutions satisfying all these constraints. One of the techniques to find a solution is Constraint Satisfaction (CS), which deals with problems defined over finite domains. Let us notice that most of industrial applications in planning and scheduling use finite domains [3].

However, in these applications it often appears that it is not possible to satisfy all the constraints because of inconsistency. Such problems are called over-constrained and several approaches were proposed to deal with these kind of problems. Among them are Partial Constraint Satisfaction (PCS) and Constraint Hierarchy (CH) [2].

PCS involves finding values for a subset of the variables that satisfy a subset of the constraints. By weakening some of the constraints, additional acceptable value combinations are permitted. Freuder & Wallace [8] formalize a notion of a partial constraint satisfaction problem by means of metrics over a set of CSPs.

CH is another approach of handling over-constrained problems which is first introduced in [4]. The constraint is weakened explicitly by specifying its strength or preference. It allows one to specify not only constraints that are required to hold (so-called hard constraints) but also weaker constraints (so called soft constraints). Many algorithms are introduced to solve CHs. In fact, all algorithms

F.A. Savacı (Ed.): TAINN 2005, LNAI 3949, pp. 194–202, 2006.

belong to one of two main groups of algorithms: refining algorithm or local propagation algorithm [1]. DeltaStar [7] is the most popular from refining algorithms. There are several local propagation algorithms such as DeltaBlue [6], SkyBlue [10], Ultraviolet [5].

As far as programming tools concern, most of the constraint programming languages are extended from logic programming paradigm because in both approaches problems are formulated very close to their declarative semantics. CLP(R), CLP(F), PROLOG III and CHIP are successful examples of such languages [3]. However, this brings some disadvantages, in particular it is hard to integrate new components implemented with some widely used programming languages such as C, C++, Java etc. Besides that hierarchy of constraints is not included in any leading constraint packages like ILOG Solver, SICStus Prolog, or Eclipse [1].

One main goal in this paper is to deal with over-constrained problems and develop a finite domain CH solver. Our solver is based on refining algorithm. Another goal is to implement our solver in a more widely used programming language. To satisfy this goal, ILOG Solver is used as a general constraint solver and C++ is chosen as an implementation language. ILOG Solver provides general constraint solving mechanisms and C++ is used to make some extensions for realizing the CH framework.

The paper is organized as follows. In section 2 we recall the basic notions of constraint satisfaction and constraint hierarchy framework. Section 3 describes the general structure of our finite domain CH solver, HIS. After that we formulate a layout problem and give some implementation details about HIS using this example (such as construction of hierarchy tree and how comparator algorithms work on this tree). At the end of this section the results to the example problem are stated. Section 4 mentions the conclusion and future work of this research.

## 2 Main Definitions

Definitions given in this section are consistent with definitions from [4,11].

**Definition 1.** *A constraint satisfaction problem(CSP) is defined as a triple (V,D,C), where*

- $V = \{v_1, \ldots, v_n\}$ *is a finite set of variables,*
- $D = \{D_1, \ldots, D_n\}$ *is a set of domains. Each domain is a finite set of possible values for the corresponding variable,*
- $C = \{c_1, \ldots, c_n\}$ *is a set of constraints, restricting the values that the variables can simultaneously take.*

**Definition 2.** *A valuation* $\theta = \{\langle v_1, d_1\rangle, \ldots, \langle v_n, d_n\rangle\}$ *for a set of variables* $\{v_1, \ldots, v_n\}$ *assigns value* $d_i \epsilon D(v_i)$ *to each* $v_i$.

A solution to a CSP is a valuation, where for each variable a value from domain is assigned in such a way that every constraint is satisfied. A problem is over-constrained when there is no valuation that satisfies all constraints. Such

problem arises frequently when real world applications are modelled as CSPs. For this category of problems we need to relax some of the constraints in order to obtain a solution. Therefore, a new type of constraints, called a soft constraint is proposed to model the over-constrained problems. A soft constraint can be seen as a preferential constraint whose satisfaction is not required but preferred. A constraint that strictly requires to be satisfied is called a hard constraint.

A labelled constraint $c^s$ is a constraint $c$ with a strength $s\epsilon\{1,\ldots,k\}$. The strengths are totally ordered. Constraints with strength $s = 0$ are hard constraints and those with strength $1 \leq s \leq k$ are soft constraints. The larger the strength, the weaker the constraint is. In addition each labelled constraint can be associated with a weight $w$ (for use with global comparators).

**Definition 3.** *A constraint hierarchy $H$ is a finite set of labelled constraints. Symbol $H_i$ denotes a set of labelled constraints with $s = i$. $H_0$ denotes the set of hard constraints which must be satisfied. $H_1,\ldots,H_k$ denotes the sets of soft constraints that can be violated.*

**Definition 4.** *(Solution to Constraint Hierarchy) The set of solutions $S_0$ contains those valuations that satisfy hard constraints. A partial ordering predicate "better" over solutions describes the relative solution quality and is called a comparator. The solution set $S$ contains only those valuations that are optimal with respect to predicate "better".*

$$S_0 = \{\theta \mid \forall c\epsilon H_0, c\theta holds\}$$

$$S = \{\theta \mid \theta\epsilon S_0 \wedge \forall\delta\epsilon S_0\ \neg better(\delta,\theta,H)\}$$

Comparators are based on error functions. An error function $e(c\theta)$ is a measure of how well a constraint $c$ is satisfied by valuation $\theta$. The error function returns non-negative real number and must satisfy the property $e(c\theta) = 0$, if and only if $c\theta$ holds.

Comparators can be classified as to whether they compare the valuations by considering one constraint at a time or whether they take some aggregate measure of how well the constraints are satisfied at a given level. Comparators that compare solutions constraint-by-constraint are local or regional, those that use an aggregate measure are global. The well-known comparator types in theory are locally-better(l-b), regionally-better(r-b), weighted-sum-better(w-s-b), worst-case-better(w-c-b), least-squares-better(l-s-b) and unsatisfied-count-better(u-c-b) [11].

For example, let us consider two valuations $\theta$ and $\delta$. We say that $\theta$ is locally-better than $\delta$, if $\theta$ does exactly as well as $\delta$ for all constraint at levels $1\ldots k-1$, and at level $k$, $\theta$ must do at least as well as $\delta$ for all constraints, and strictly better for at least one. In some cases, the levels may be incomparable by locally-better comparator through some level $k-1$. Valuations that are incomparable through level $k-1$ can still be compared at level $k$ according to regionally-better comparator.

For a global comparator, the errors for all constraints until level $k$ are aggregated by a specific combining function $g$. A valuation $\theta$ is globally-better than

another valuation $\delta$, if for each level through some level $k-1$, the combined error $g(H_i\theta)$ of the constraints after applying $\theta$ is equal to that of applying $\delta$, and at the level $k$ it is strictly less ($1 \leq i \leq k$). These combined error functions are calculated as the sum of the weighted errors or the maximum of the weighted errors or the sum of the squares of the weighted errors.

## 3 HIS: Solver for Soft Constraints

### 3.1 General Structure

HIS is a hierarchical constraint solver designed for solving over-constrained problems in which problem variables have finite domains. The hierarchy solving algorithm of HIS is based on the refining algorithm.

HIS consists in three main parts, HIS user methods, hierarchy tree and comparators all of which rise on top of ILOG Solver (Fig.1). HIS user methods let the user to make calls to underlying methods of the hierarchy tree and comparators. In addition, all components of HIS use data structures and methods provided by ILOG Solver. Each node of the hierarchy tree is a constraint with a label and a weight. The main idea behind constructing a hierarchy tree is to obtain the subsets of constraints that contain as much constraints as possible that are satisfied together. Comparators component is integrated with the hierarchy tree since comparator algorithms traverse the hierarchy tree in some way to discard the paths that are worse than any other path.

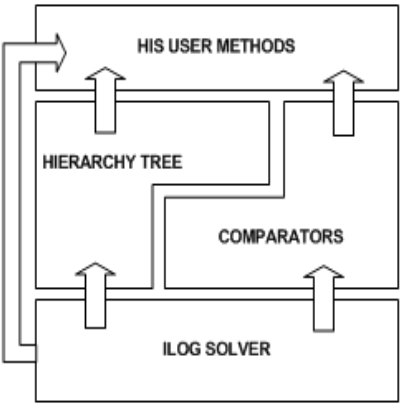



**Fig. 1.** General structure of HIS

### 3.2 Example

We formulate a layout problem which briefly is specified as follows. The problem is to assign different areas from Fig.2 to different rooms such as bedrooms, living rooms, kitchens etc. according to specific designer rules. We use two type of designer rules for this problem: adjacency and size. Designer states some

preferences such as "Kitchen is adjacent to dining room is more preferable to me than bedroom is adjacent to bathroom". Areas that share the same boundary are adjacent (e.g area0 and area1) in the layout and area0 and area4 are larger than the other areas.

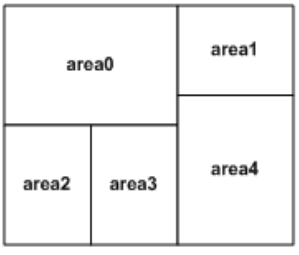



**Fig. 2.** Example layout problem

Table 1 shows the problem variables, domains of the variables in addition with the constraints imposed on these variables. There are five variables in our problem: $\{Kitchen, Bathroom, Livingroom, Bedroom, Diningroom\}$. The domain of the variables are the areas from the layout: $\{area0, area1, area2, area3, area4\}$. Constraints are shown in a level by level structure and with their weights. At level $H_0$, there are two hard constraints, *alldifferent* and *distribute*. *Alldifferent* constraint restricts the problem variables not to take same values from the domain, since the same area in the layout cannot be assigned to different rooms. In addition, we require that all of the areas in the plan must be assigned. This is done by the *distribute* constraint. The remaining constraints are soft constraints.

**Table 1.** Constraint hierarchy for layout problem

| |
|---|
| $V = \{K,\ Bt,\ L,\ B,\ D\}$ |
| $D = \{a0,\ a1,\ a2,\ a3,\ a4\}$ |
| $H = \{H_0,\ H_1,\ H_2\ , H_3\}$ |
| $H_0 = \{c_1^0 : alldiff(V),\ c_1^0 : dist(1, V, D)\}$ |
| $H_1 = \{c_1^1 : adj(K, D), c_2^1 : larger(D, K)\}\ \ w_{c_1^1} = 6,\ w_{c_2^1} = 4$ |
| $H_2 = \{c_1^2 : larger(L, Bt),\ c_2^2 : larger(B, Bt)\}\ \ w_{c_1^2} = 5,\ w_{c_2^2} = 3$ |
| $H_3 = \{c_1^3 : larger(D, Bt),\ c_2^3 : larger(L, K),\ c_3^3 : larger(B, K),\ c_4^3 : adj(B, Bt)\}$ |
| $w_{c_1^3} = 8, w_{c_2^3} = 7,\ w_{c_3^3} = 6,\ w_{c_4^3} = 5$ |

First, hard constraints are solved with ILOG Solver and if they are all satisfied they are added to the hierarchy tree. Soft constraints are inserted into the tree one by one after they are sorted. The one with lower strength and higher weight takes the first place in the sorted list. Each new constraint is tried to be solved with the constraints on each path of the tree. In the example, $alldiff$, $dist$, $adj(K, D)$, $larger(D, K)$ and $larger(L, Bt)$ constraints are successfully solved together. But when $larger(B, Bt)$ is added, the problem becomes over-constrained. So, $larger(B, Bt)$ is added to the hierarchy tree as the second

child of $larger(D, K)$. The algorithm below for constructing the hierarchy tree is based on the idea of continuously satisfying the constraints in the tree with the new constraint starting from the root node and finding a place in the tree to insert the new constraint.

```
cur_node = last_hard(htree) %last hard constraint added to hierarchy
function add_softs(cur_node,new_cons)
    if satisfy_with(cur_node,new_cons) %solve with ILOG Solver
        foreach child of cur_node
            add_softs(child,new_cons)
        end
        %add new constraint as next child of current node
        add(cur_node,new_cons)
    end
end
```

Our hierarchy tree algorithm produces a large tree for this problem. Without considering the unnecessary paths in the tree, we have two main paths (Fig.3).

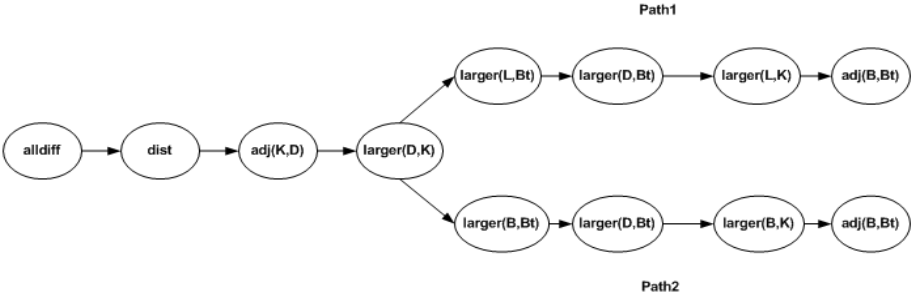



**Fig. 3.** Hierarchy tree of layout problem

### 3.3 Comparators

All of the comparator algorithms in HIS are based on the idea of comparing the paths in the hierarchy tree and discarding the ones that are detected to be worse than any other path. Valuations satisfying the constraints on the same path are already equal to each other because they satisfy the same subset of constraints. Valuations satisfying the constraints on different paths of the tree are compared according to a comparator. Not only the first path found to be better according to a comparator is returned. All the paths that are detected to be equal or incomparable are obtained.

Let us first consider the locally-better comparator. The comparison is done constraint by constraint instead of a global measure. Error values are required to make comparisons between valuations. HIS uses trivial error function, where $e(c\theta)$ returns a boolean result. For example, $\theta_1$ from Table 2 has the following error values for the constraints at level $H_2$ :

$$e(c_1^2\theta_1) = 0 \wedge e(c_2^2\theta_1) = 1$$

The error values for the valuations belonging to path 1 are equal to each other since they satisfy the same subset of constraints. It is the same for the valuations belonging to path 2. When we compare a valuation from path 1 with a valuation from path 2, locally-better algorithm is used. Initially they are compared at level $H_1$. They both satisfy the same subset of constraints from this level. So we continue to compare at level $H_2$. It is recognized that two paths are incomparable at this level and none of the paths is discarded. We can not go on the comparison for the next level in this case. However, if we use regionally-better comparator, we still have a chance to discard one of the paths after comparisons at the next levels. Unfortunately, the paths are also incomparable at level $H_3$. Both locally-better and regionally-better comparators do not discard any paths.

**Table 2.** Valuations satisfying the constraints on path1 and path2

| Solution to Path 1 |
|---|
| $\theta_1 = \{\langle K, a1\rangle, \langle Bt, a2\rangle, \langle L, a0\rangle, \langle B, a3\rangle, \langle D, a4\rangle\}$ |
| $\theta_2 = \{\langle K, a1\rangle, \langle Bt, a2\rangle, \langle L, a4\rangle, \langle B, a3\rangle, \langle D, a0\rangle\}$ |
| $\theta_3 = \{\langle K, a2\rangle, \langle Bt, a1\rangle, \langle L, a3\rangle, \langle B, a4\rangle, \langle D, a0\rangle\}$ |
| $\theta_4 = \{\langle K, a2\rangle, \langle Bt, a3\rangle, \langle L, a1\rangle, \langle B, a4\rangle, \langle D, a0\rangle\}$ |
| Solution to Path 2 |
| $\theta_5 = \{\langle K, a1\rangle, \langle Bt, a2\rangle, \langle L, a3\rangle, \langle B, a0\rangle, \langle D, a4\rangle\}$ |
| $\theta_6 = \{\langle K, a1\rangle, \langle Bt, a3\rangle, \langle L, a2\rangle, \langle B, a4\rangle, \langle D, a0\rangle\}$ |
| $\theta_7 = \{\langle K, a2\rangle, \langle Bt, a1\rangle, \langle L, a3\rangle, \langle B, a4\rangle, \langle D, a0\rangle\}$ |
| $\theta_8 = \{\langle K, a2\rangle, \langle Bt, a3\rangle, \langle L, a1\rangle, \langle B, a4\rangle, \langle D, a0\rangle\}$ |
| $\theta_9 = \{\langle K, a3\rangle, \langle Bt, a1\rangle, \langle L, a2\rangle, \langle B, a0\rangle, \langle D, a4\rangle\}$ |
| $\theta_{10} = \{\langle K, a3\rangle, \langle Bt, a2\rangle, \langle L, a1\rangle, \langle B, a0\rangle, \langle D, a4\rangle\}$ |

For global comparators, nothing much changes for our comparison algorithm except the calculations of error values. Let us consider weighted-sum-better comparator. Every path is again compared with the others starting from the strongest level and paths worse than some other path is discarded. If there only one path remains, the valuations satisfying the constraints on this path become our solution. If there are more than one path after the comparison at the strongest level, we continue to compare for the next level to discard other possible paths. When comparing two paths, total weight of unsatisfied constraints on each path is calculated and the one with higher value is discarded. Error values returned from the weighted-sum combining function can be seen in Table 3.

**Table 3.** Combining error values for w-s-b

| $\theta$ | $g(H_1\theta)$ | $g(H_2\theta)$ | $g(H_3\theta)$ |
|---|---|---|---|
| $\theta_1$ | 0 | 3 | 6 |
| $\theta_5$ | 0 | 5 | 7 |

**Table 4.** Accepted paths for comparators

| Comparators | Path 1 | Path 2 |
|---|---|---|
| l-b | Accept | Accept |
| r-b | Accept | Accept |
| w-s-b | Accept | Discard |
| w-c-b | Accept | Discard |
| l-s-b | Accept | Discard |
| u-c-b | Accept | Accept |

For the rest of the global comparators, the only thing that changes in our comparator algorithm is the type of the combining function. Worst-case function find the maximum weight of unsatisfied constraints on each path and discard the one with higher value. Least-squares gives the same result with weighted-sum, because error function only returns boolean values. Unsatisfied-count-better function finds the total number of unsatisfied constraints on each path. The path with bigger value is discarded. Accepted paths for each comparator is shown in Table 4.

## 4 Conclusion and Future Work

In this paper, we describe a solver, namely HIS, that solves over-constrained satisfaction problems. Our solver is based on the CH framework and uses the refining algorithm. It can also find solutions to over-constained problems according to arbitrary comparators. In addition, it is implemented in a more widely used programming language, C++, instead of a declarative language to make the integration with new components easier.

HIS solver is tested for the extended layout problem modelling a two floor building. The problem contains 9 rooms (9 variables) and in the layout there are 9 areas (each variable takes a value from a domain of size 9). Three new designer rules $onFirstFloor$, $onSecondFloor$ and $onTopOf$ are used in addition to $adj$ and $larger$. Total number of constraints are 18. As a future work, we need to optimize our hierarchy tree construction algorithm since the number of nodes in the tree increases dramatically as the problem size increases. In addition, we will extend the layout problem to let the user specify different designer rules through a friendly user interface.

## References

1. Bartak, R., A Theoretical Framework for Constraint Hierarchy Solvers, Proceedings of the 15th European Conference on Artificial Intelligence, IOS Press, Amsterdam, 2002, 146–150.
2. Bartak, R.,Modeling Soft Constraints: A Survey, Charles University, Prague, 2002.
3. Bistarelli, S., Codognet, P., Hui, H.K.C., Lee, J.H.M., Solving Finite Domain Constraint Hierarchies by Local Consistency and Tree Search, Proceedings of the 9th International Conference Principles and Practice of Constraint Programming, Ireland, 2003.

4. Borning, A., Duisberg, R., Freeman-Benson, B., Kramer, A., Woolf, M., Constraint Hierarchies, Proceedings of the ACM Conference on Object Oriented Programming Systems, Languages and Applications, 1987, 48–60.
5. Borning, A., Freeman-Benson, B., Ultraviolet: A Constraint Satisfaction Algorithm for Interactive Graphics, Constraints, 1998, 3(1), 9–32.
6. Freeman-Benson, B., Maloney J., Borning, A., An Incremental Constraint Solver, Communications of the ACM, 1990, 33(1), 54–63.
7. Freeman-Benson, B.,Wilson, M., Borning, Alan., DeltaStar: A General Algorithm for Incremental Constraint Satisfaction of Constraint Hierarchies, 11.th Annual IEEE Phoenix Conference on Computers and Communications, 1992, 561-568.
8. Freuder, E.C., Wallace R.J., Partial Constraint Satisfaction, Artificial Intelligence, 1992, 58, 21–70.
9. Marriott, K., Stuckey, P.J., Programming with Constraints: An Introduction, MIT Press, 1998.
10. Sannella, M., The SkyBlue Constraint Solver and Its Applications, Proceedings of the First Workshop on Principles and Practice of Constraint Programming, MIT Press, 1994.
11. Wilson, M., Borning A., Hierarchical Constraint Logic Programming, The Journal of Logic Programming Special Issue on Constraint Logic Programming, 1993, 16, 227–318.

# Elevator Group Control by Using Talented Algorithm

Ulvi Dagdelen[1], Aytekin Bagis[1], and Dervis Karaboga[2]

[1] Erciyes University, Dept. of Electronic Eng., 38039, Kayseri, Turkey
{Dagdelen, Bagis}@erciyes.edu.tr
[2] Erciyes University, Dept. of Computer Eng., 38039, Kayseri, Turkey
Karaboga@erciyes.edu.tr

**Abstract.** A simple and efficient control approach based on an operation strategy with a talented algorithm is presented for the elevator group control. In order to analyze the performance of the presented system, the control method was evaluated by considering different performance characteristics. The results of the method were compared with the results of the area weight algorithm. The results obtained from the simulations indicated that the presented approach exhibits high performance over the area weight algorithm with the minimum time consumption results.

## 1 Introduction

Elevator group control systems (EGCS) are control systems that manage to assign a service car of multiple elevators in a building in order to efficiently transport the passengers waiting in a hall. The main aim of the optimal elevator group control is to obtain the highest handling capacity, the shortest waiting and/or traveling time of passengers. The performance of the group control system is measured by several evaluation criteria such as the average waiting time of passengers, the percentage of passengers waiting more time of 60 sec and power consumption. Since the controller considers many different cases, the EGCS is a difficult and very complex control problem. In the EGCS, there are many uncertain factors such as number of passengers, hall calls, and car calls in any time. Furthermore, it must be possible for a skilled system operator to change the control strategy.

Some methods used to achieve the elevator group control have been presented in [1-8]. Kim et al. presented the design of a fuzzy elevator group control system based on the classification of the passenger traffic and system manager's requirements [1]. Ishikawa et al. proposed a group control system which addresses riding time and waiting time [2]. Tobita et al. employed a parameter tuning method for an elevator group control system using a genetic algorithm [3]. Another group control approach with floor attribute control method was presented by Fujino et al. [4]. Cho et al. described a control procedure with accurate estimation of hall call waiting times [5]. On the other hand, Ho and Fu defined a dynamic scheduling approach for the group control [6]. In [7], Gudwin et al. proposed a fuzzy group controller with linear context adaptation. Crites and Barto developed an elevator group control using multiple reinforcement learning agents [8].

F.A. Savacı (Ed.): TAINN 2005, LNAI 3949, pp. 203 – 210, 2006.

In this paper, a more simple and efficient control approach is presented for the elevator group control. The method is based on an operation strategy with a talented algorithm. In the operation strategy, the area weight algorithm is employed by the talented algorithm. The EGCS is introduced in the second section. The proposed control problem is defined in Section 3. In the Section 4, the operation approach and simulation results are presented and the results are discussed.

## 2 Elevator Group Control System (EGCS)

There are two types of calls in the elevator group control system (Fig.1). The hall call is given through buttons on the hall of the building, and the car call is given in the elevator by passengers. An EGCS has a pair of hall call buttons on each floor, one for up hall call and the other for down hall call. If a passenger presses a hall call button, an elevator is selected by the group control system for the passenger. The selected elevator moves to the floor where the hall call occurred.

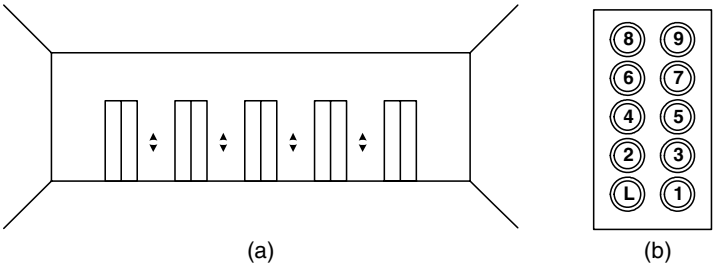



**Fig. 1.** (a) Hall call. (b) Car call.

Fig. 2 shows the structure of an EGCS. The EGCS consists of the passenger generation, group controller, elevator controller and monitoring units.

**(a) Passenger Generation:** In this module, there are some information about building configuration like number of floor and height of floors, number and capacity of cars, elevator configuration like door time and building traffic. Traffic model and profile is formed in this module. There are three kinds of passenger profile in the elevator systems. These are;

*i) The passengers calling upward:* These are the passengers going from the entrance hall to upwards. They make up-call.
*ii) The passengers calling downward: These are the passengers making calls from the upper halls to the entrance hall. They make down-call.*
*iii) The passengers calling to interfloor: These are the passengers making calls from the intermediate halls to the other intermediate halls. They make either up- or down-call.*

**(b) Group Controller:** In this module, hall and car information of calls are kept. The arrangement of the passengers is realized in this module by using the information about the cars. There are two task lists for hall calls and car calls that contain the

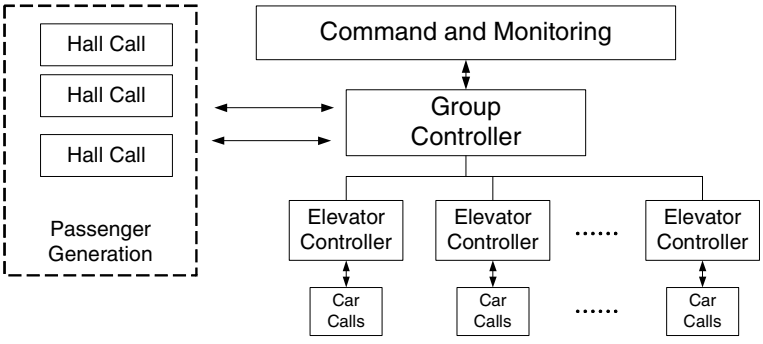



**Fig. 2.** Elevator group controller architecture

passengers getting on and already existing in the elevator, respectively. Stop and way information for the cars is determined by using this task list. When the car is charged for a given way, it does not give any response to requests in the reverse direction.

**(c) Elevator Controller:** In this module, the car will move after the current hall and target hall for car are compared. There is some information in this module such as the passengers in the car, target floor and the number of the passengers carried on a round trip.

**(d) Monitoring:** In this module, the data obtained from the evaluation criteria are computed and graphically reflected on the monitor.

The elevator group control system has to consider many factors about both the current and future states of the elevator system. In this step, we know the current data such as the position of each elevator and the hall call's and car call's allocation states, but do not know the information such as the number of passengers where a hall call happened. Furthermore, the information about the future hall calls and car calls are uncertain. Therefore, it is difficult to select an appropriate elevator when a call has happened. The selection of the elevator is made in order to minimize the average waiting time of passengers, the long wait probability and service time. This selection method of the appropriate elevator is called the hall call assignment method. In this method, an evaluation function is used to achieve the above multiple objectives. The function is evaluated for each elevator and the elevator with the smallest function value is selected. Let $\phi(\mathrm{k})$ be the evaluating function for the $k^{th}$ elevator, and then this function can be represented with the following formula [9]:

$$\phi(k) = T_{AVR} - \alpha T_{\alpha}(k), \qquad \text{k=1, 2, 3…..... N} \tag{1}$$

Here, $T_{AVR}(k)$ is the estimated arrival time of the $k^{th}$ elevator, which is the waiting time of the passenger when the $k^{th}$ elevator is assigned for the new call hall and N is the total number of elevators. $T_{AVR}(k)$ is calculated by the following formula:

$$T_{AVR}(k) = \sum_{stop} T_{stop}(k) + \sum_{drive} T_{drive}(k) \tag{2}$$

$$T_{stop}(k) = T_{speed\_down} + T_{get\_on/off}(k) + T_{speed\_up} \tag{3}$$

In the above formula, the path of the elevator is divided into stop and drive. Stop term means floors where hall calls and car calls are assigned, and drive term means floors where there are no calls near the floor. α and $T_{\alpha}(k)$ are described as the area weight and the area value, respectively. Area weight is a weighting factor for area value and it affects the performance of EGCS by the patterns of passenger traffic. The value of α also affects the possibility that the elevator close to the relevant floor can be selected. Therefore, α is an important parameter in the hall call assignment method.

The value of the $T_{\alpha}(k)$ is determined by the positions of assigned calls for each elevator. If an elevator is assigned to serve a call on a floor, the area of the elevator on the floor is defined. In general, the area is defined in the form of a triangle or a trapezoid as shown in Fig. 3(a). In Fig. 3(b), the trapezoidal area of elevator k on the floor n is given. The area value $T_{\alpha}(k)$ is defined for the floors where a call (hall and car) has happened. It can be seen that the $T_{\alpha}(k) = 1$ for the floors n, n+1, n-1; $T_{\alpha}(k) = 0.5$ for floors n+2, n-2; and $T_{\alpha}(k) = 0$ for the others.

If a new hall call is generated on the floor where the k[th] elevator is going to stop, the value of $\alpha T_{\alpha}(k)$ is subtracted from the evaluating function value of the k[th] elevator.

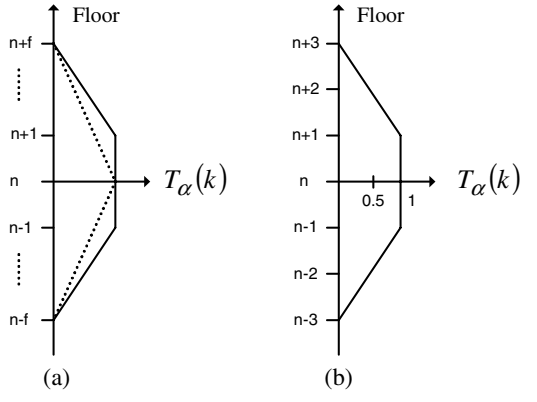



**Fig. 3.** Example of the area value

## 3 Proposed Talented Control Algorithm

Many evaluation criteria can be used to estimate the performance of the EGCS. In this paper, the following criteria are used;

i) Average waiting time (AWT) is the time until the service elevator arrives at the floor after a passenger presses a hall call button. AWT is the average of all waiting times in a unit time.
ii) Average journey time (AJT) is the total time a passenger spends within an elevator system first by waiting and then by riding inside a car. AJT is the average of all journey times in a unit time.
iii) Long waiting percent (LWP) is the percentage of the passengers who wait more then 60 sec in a unit time.

The flowchart of the general area weighting algorithm is given in Fig.4. In this flowchart, proposed talented control algorithm is also represented as the dotted area.

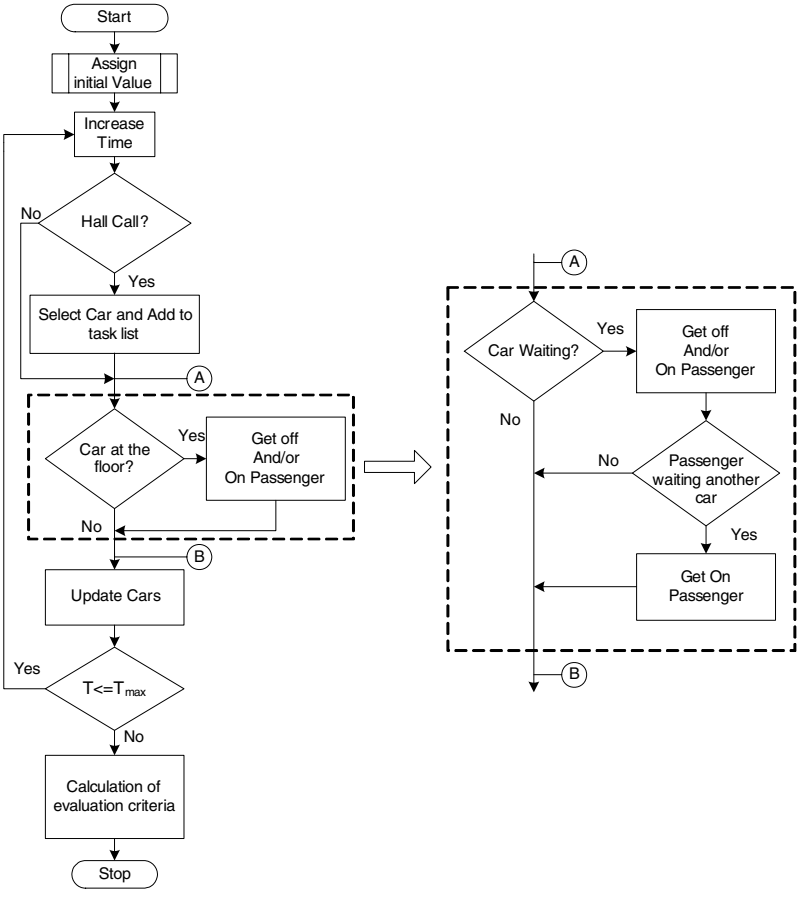



**Fig. 4.** Flowchart of simulation program

In the area weighting algorithm, firstly, the main parameters such as the number of floors, in the building, the number of elevator cars, speed and capacity of elevator cars, time that elevator car stops at floor to load/unload users are preserved by the

program. The situation of the passengers is randomly generated. For the each elevator car, a task list is obtained, and this list is updated in the predetermined time intervals. The performance of the algorithm is computed by using the evaluation criteria at end of this process. The main feature of the proposed talented algorithm is with the task lists of the elevator cars. When a car stops at the any floor, its task list is compared with the task lists of other cars at different floors. If other cars have a task at this floor, these tasks are undertaken by this car at this floor. Thus, the time consumption is significantly decreased by the proposed algorithm.

## 4 Simulation Results

The conditions for simulation are given Table 1. As shown in this table, simulation time is selected as 60 min (1 hour, $T_{max}$). The number of floors and elevator cars used in the building are 16 and 6, respectively. In the simulation study, the person number of 50 to 200 is used for 5 minutes time interval and it is also assumed that the capacity of the each elevator is 20 people.

**Table 1.** Simulation conditions

| | |
|---|---|
| **Number of floors** | 16 floors |
| **Number of elevator / Capacity** | 6 cars / 20 persons |
| **Number of person** | 50 to 200 persons per 5 minutes |
| **Elevator speed** | 2 seconds per floor |
| **Stopping time** | 14 seconds |
| **Out traffic** | 1st to other floors : 25% |
| | Others to 1st floor : 50% |
| | Others to other floors : 25% |

**Table 2.** Average waiting time

| **Passenger (5min/men)** | **Average Waiting Time (sec) (AWT)** | | | | |
|---|---|---|---|---|---|
| | **Area Weight Algorithm [9]** | | | | **Proposed Talented** |
| | **$\alpha = 0$** | **$\alpha = 15$** | **$\alpha = 30$** | **$\alpha = 40$** | |
| **50** | 30,09 | 28,38 | 27,28 | 27,13 | 24,69 |
| **75** | 36,31 | 34,23 | 31,41 | 30,96 | 26,84 |
| **100** | 43,44 | 41,10 | 37,04 | 34,65 | 29,70 |
| **125** | 50,62 | 47,76 | 42,14 | 39,93 | 32,85 |
| **150** | 57,90 | 54,87 | 48,25 | 45,02 | 36,52 |
| **175** | 69,09 | 67,15 | 55,50 | 53,46 | 40,56 |
| **200** | 91,09 | 83,58 | 74,70 | 67,25 | 45,03 |

Tables 2-4 show the simulation results obtained for evaluation criteria given above. In these tables, the area weight algorithm and the proposed talented algorithm are compared for different performance indexes such as average waiting time (AWT), average journey time (AJT), long weighting percent (LWP). Since the value of $\alpha$

significantly affects the performance of area weight, the simulation results were repeated for different values of α (0, 15, 30 and 40).

**Table 3.** Average journey time

| Passenger (5min/men) | Average Journey Time (sec) (AJT) | | | | |
|---|---|---|---|---|---|
| | Area Weight Algorithm [9] | | | | Proposed Talented |
| | α = 0 | α = 15 | α = 30 | α = 40 | |
| **50** | 64,48 | 63,46 | 62,94 | 62,75 | 58,22 |
| **75** | 75,55 | 73,91 | 71,39 | 71,27 | 63,82 |
| **100** | 88,35 | 86,85 | 83,19 | 80,66 | 71,26 |
| **125** | 101,94 | 99,55 | 93,74 | 91,50 | 79,29 |
| **150** | 114,77 | 111,89 | 105,34 | 102,29 | 87,95 |
| **175** | 130,56 | 129,09 | 117,48 | 115,56 | 97,21 |
| **200** | 156,17 | 149,15 | 140,50 | 133,06 | 105,67 |

**Table 4.** Long waiting percent

| Passenger (5min/men) | Long Waiting Percent (%) (LWP) | | | | |
|---|---|---|---|---|---|
| | Area Weight Algorithm [9] | | | | Proposed Talented |
| | α = 0 | α = 15 | α = 30 | α = 40 | |
| **50** | 15,22 | 13,69 | 12,03 | 11,40 | 9,56 |
| **75** | 20,93 | 18,58 | 15,03 | 14,00 | 11,37 |
| **100** | 26,12 | 23,68 | 19,03 | 16,30 | 13,89 |
| **125** | 31,11 | 28,32 | 22,55 | 19,90 | 16,82 |
| **150** | 35,17 | 32,24 | 26,14 | 22,50 | 20,87 |
| **175** | 38,12 | 35,68 | 29,25 | 26,00 | 24,93 |
| **200** | 40,56 | 37,29 | 32,64 | 29,60 | 28,89 |

From these tables, the following conclusions can be drawn:

(1) The average waiting time for the passengers is the most crucial factor in the elevator group control. The waiting time occurred when the proposed control algorithm is used is quite shorter than that of the area weight algorithm (Table 2). As the number of passengers increases, it is seen that the waiting time significantly decreases when the results of both algorithms are compared. For example, when the number of passengers is 200 AWT of talented algorithm is 45.03 sec. For the same condition, AWT of area weight algorithm is 67.25 sec (α = 40).

(2) From Table 3, it can be clearly seen that the average journey time for the talented algorithm is also shorter than that of the area weight algorithm. For example, the difference between the average journey times of the talented algorithm and the area weight algorithm for 200 person is 50.50 sec and 27.39 sec for α = 0 and α = 40, respectively.

(3) As apparently seen from Table 4, the long waiting percent of the passengers for the elevator cars is considerably decreased in the proposed algorithm.

## 5 Conclusion

In this paper, a simple and efficient control approach based on a talented algorithm is presented for elevator group control system. When it is compared with the area weighting algorithm, it is verified that the proposed control algorithm is a very effective and reliable control method for any condition of the passenger. The presented algorithm is also capable to adopt for different process parameters such as the number of elevator car, capacities of the cars, floor numbers, different traffic situations, and it is a distinctive advantage of the algorithm.

## References

1. Kim, C.B., Seong, K.A., Kwang, H.L., Kim, J.O.: Design and Implementations of a Fuzzy Elevator Group Control System. IEEE Trans. on Systems Man and Cybernetics-Part A 28(3) (1998) 277–287
2. Ishikawa, T., Miyauchi, A., Kaneko, M.: Supervisory Control for Elevator Group by Fuzzy Expert System Which Also Addresses Traveling Time. IEEE International Conference on Industrial Technology Proceeding (2000) 87–94
3. Tobita, T., Fujino, A., Segawa, K., Yoneda, K., Ichikawa, Y.: A Parameter Tuning Method for an Elevator Group Control System Using a Genetic Algorithm. Electrical Engineering in Japan 124(1) (1998) 55–64
4. Fujino, A., Tobita, T., Segawa, K., Yoneda, K., Togawa, A.: An Elevator Group Control System with Floor–Attribute Control Method and System Optimization Using Genetic Algorithm. IEEE Trans. On Industrial Electronics 44(4) (1997) 546–552
5. Cho, Y.C., Gagov, Z., Kwon, W.H.: Elevator Group Control with Accurate Estimation of Hall Call Waiting Times. IEEE Proceeding of Te International Conference on Robotics&Automation (1999) 447–452
6. Ho, Y.W, Fu, L.C.: Dynamic Scheduling Approach The Group Control of Elevator Systems with Learning Ability. IEEE Proceeding of The International Conference on Robotics&Automation (2000) 2410–2415
7. Gudwin, R., Gomide, F., Netto, M.A.: A Fuzzy Group Controller with Linear Context Adaptation. Fuzzy Systems Proceedings (1998) 481–486
8. Crites, R.H., Barto, A.G.: Elevator Group Control Using Multiple Reinforcement Learning Agents. Machine Learning 33 (1998) 235–262
9. Kim, C.B., Seong, K.A., Kwang, H.L., Kim, J.O., Lim, Y.B.: A Fuzzy Approach to Elevator Group Control System. IEEE Trans. on Systems Man and Cybernetics 25(6) (1995) 985–990

# A Fault Tolerant System Using Collaborative Agents

Sebnem Bora

Department of Computer Engineering
Ege University
Izmir, 35100, Turkey
bora@staff.ege.edu.tr

**Abstract.** Replication of data or processes is an effective way to provide enhanced performance, high availability and fault tolerance in distributed systems. For instance, in systems based on the client-server model, a server may serve many clients and because of heavy loads, the server cannot respond to the requests on time. In such a case, replicating data or servers may improve performance. Moreover, data and processes can be replicated to protect against failures. However, this is a very complex procedure. In this paper, I propose a method, to make systems fault tolerant based on replication, by way of exploiting the use of collaborative agents. This method is also used to improve fault tolerance in multi-agent systems.

## 1 Introduction

Replication is a key technique to achieve fault tolerance in distributed and dynamic environments by masking errors in the replicated component. A distributed application is a set of cooperating objects, where an object could be a virtual node, a process, a variable, an object as in object-oriented programming, or even an agent in multi-agent systems. When an object is replicated, the application has several identical copies of the object (called replicas). When a failure occurs on a replica, the failure is masked by its other replicas, therefore availability is ensured in spite of the failure.

Replication mechanisms have been successfully applied for distributed applications. However, replication to be used in the application is decided by the programmer before the application starts. Since the environmental context related to distributed systems is very dynamic, the environment changes and needs are considered while selecting a replication approach, and then a suitable approach is applied adaptively to the system for fault tolerance. Moreover, since the available resources are not infinite, the number of replicas must be dynamically updated.

There are several existing distributed systems that are designed to exhibit some form of fault tolerant behavior, or provide the tools for the development of fault tolerant applications. ISIS [1] was the first system to provide a reliable broadcast service with multiple ordering primitives. The successor of ISIS, Horus [2] provides a wide variety of group communication primitives that can be selectively added according to the application needs. The Advanced Automation System (AAS) [3] is a distributed real-time system that integrates all the services of the US air traffic control network. The high availability requirements were fulfilled by introducing both hardware and

F.A. Savacı (Ed.): TAINN 2005, LNAI 3949, pp. 211 – 218, 2006.

software redundancy. Critical services are replicated using either the active or the passive approach, according to the application semantics and the hardware configuration. Consul [4] is a collection of protocols that provides low-level mechanisms required for the development of fault tolerant distributed applications according to the State Machine approach [5]. Psync is a reliable broadcast protocol that maintains the causal order of messages exchanged between group members and supports Consul. The Manetho protocol [6] is a combination of a rollback-recovery protocol and a process replication protocol based on the primary backup approach. Simple ad hoc broadcast and membership algorithms are used for this purpose. Manetho performs very well in environments with low failure probability. It introduces minimal delays due to replication. However, the price for this is a considerable overhead in the control information. There are also some other works done to design adaptive fault tolerance systems. AFTM [7] is an adaptive fault tolerant middleware, which uses a CORBA-compliant object request broker. In AFTM the most suitable fault tolerant and resource allocation scheme is selected dynamically through user requests and the parameters in its three databases. Its services provide automatic reconfiguration of the groups, transparently masking faults from the users. AQuA[8] is an operating system architecture which provides adaptive fault tolerance for distributed applications. AQuA allows application developers to specify the desired levels of dependability at compilation time in accordance with the availability of resources and occurence of the faults. AQuA uses QuO to specify QoS requirements at application level, and the Proteus dependability manager to configure the system in response to faults and availability requirements.

All related works mentioned above, provide either static fault tolerance or adaptive fault tolerance to the distributed systems. The main difference presented work in this paper from those works is that the components of the infrastructure that provides fault tolerance to a system are implemented by using software agents. Since software agents are autonomous processes capable of interacting with its environment, they are suitable to be parts of the middleware architecture that provides adaptive fault tolerance to distributed systems. Moreover, the same middleware architecture can be used to provide fault tolerance to multi-agent systems due to the dynamic and distributed nature of multi-agent system.

This paper is organized as follows: Section 2 addresses the problems related to the fault tolerant systems. Section 3 presents a fault tolerant building system based on agent technology in order to overcome the problems mentioned in Section 2. Section 4 gives the implementation details of a fault tolerant middleware architecture. Section 5 gives a summary of the paper.

## 2 Problem Description

In replication based approaches for achieving fault tolerance, there must be multiple copies of the same object (replicas). In this paper, this model is applied in the client-server environment and multi-agent system environment, in which case an object is a server or an agent, respectively. All replicas of an object (a server or an agent) run concurrently, possibly in different environments. It is obvious that the true cost of a single object is that of the object and its replicas plus overhead to coordinate their activities. Moreover, a fixed number of resources in a system are reserved to provide redundancy for performing a certain task. However, usage of a fixed number of

resources in a system can be expensive. Varying the replication degree, which is number of replicas in a fault tolerant system, can reduce the cost caused by the replication. Adaptive fault tolerance enables a system to change its replication degree in accordance with its environment and consistent with a user defined value. Moreover, it also provides applying adaptively replication approaches while considering the environment changes and needs.

In this paper, I propose to exploit the use of software agents to implement a fault tolerant system, in which replication mechanism is adaptively applied and the number of replicas is dynamically updated, in order to overcome the problems mentioned above.

## 3 A Fault Tolerant System Using Agents

Replicating objects is the only way to achieve fault tolerance in distributed systems. The idea is very simple. If one replicated object fails, there will be another replica to take over for it. However, this comes at the price of complexity and system overhead. Keeping all of the replicas updated requires extra communication and extra processing.

There are two approaches for replication: the primary-backup approach (also called passive replication) and the active replication approach. In the active and passive replication, there are extra copies of an object (called replicas). In active replication, each replica processes the requests and synchronizes its internal state with every other replica. If a primary replica fails, any replica can become a primary replica. In primary-backup approach, there are extra copies (replicas) of an object. However, primary one responds to the requests. Primary periodically updates replicas' states. If primary fails, one replica can be elected as a primary replica. The replication service of a fault tolerant system can use active and passive replication approaches to replicate servers/agents. Each technique has its own merits. There is a tradeoff between both approaches in terms of recovery speed and overhead [9].

In order to implement replica coordination in both forms of replication, replicas must communicate with each other. Protocols like TCP support point-to-point communication. With these types of protocols, more than two replicas cannot communicate with each other. Group communication provides multi-point-to-multi-point communication by organizing replicas in groups and group membership service [10]. Group membership service uses the failure detector to reach a decision about the group's membership.

The services used in fault tolerant system such as group communication service, membership service, replication service can be implemented by software agents. Software agents are playing an increasingly important role in distributed and dynamic environment.

A software agent is an autonomous process capable of interacting with its environment, initiating actions that affect its environment, and exchanging information with users and other agents. Here I present an agent based technique for the development of fault tolerant applications such as replicated servers and agents. This technique uses a system-centric approach in which operations related to solving the problems introduced by the fault tolerance and recovery is performed by separate agents. It dynamically adapts the number of replicas in the replication group and the replication strategy to meet the environmental requirements.

In this present work, in groups having replicated servers/agents, operations related to all replication management, membership management, multicasting and dynamic configuration of the group are performed by the agents. Components of the proposed fault tolerant system are replicas of a server or an agent, a replication manager, a mobile agent, a sequencer agent and a failure detector. All components of the system are agents except server replicas, programmed by using Grasshopper Agent Platform.

Grasshopper is a mobile agent platform that is built on top of a distributed processing environment. In this way, an integration of the traditional client/server paradigm and mobile agent technology can be achieved. Mobile agent technology is a new area in the field of distributed applications. Mobile agents are software components, which are able to migrate actively from one physical network location to another. By moving to locations where required information or logic is hosted, mobile agents are able to take advantage of local communication instead of interacting remotely via the network. For example, an agent needs to gather information from different databases. It visits the database hosts one after the other, accessing the databases, and filtering the information locally before migrating to the next host. Two different kinds of migration can be separated: Strong migration in which agent migrates together with its whole execution state and weak migration in which an agent just maintains its data state when traveling from one location to another. The Java programming language does not capture the execution state of a process or thread. The only possibility to achieve this is to modify the Java Virtual Machine. However, strong migration can be simulated by using weak migration. Grasshopper agents use weak migration for traveling from one agency to another. Parts of the agent's data state make an agent to continue its task processing after a migration instead of restarting its task from the beginning. Agent's task can be separated into different execution blocks containing a set of operations to be performed at each location [11]. In the next section, the implementation of components of a fault tolerant system by using agents will be presented.

## 4 Implementation of Fault Tolerant System Using Agents

The proposed system includes several components. Main component of the system is the replication manager. The replication manager is in charge of replication management. It decides the number of replicas and replication strategy to be applied. Moreover, deciding where, when, and by whom copies are to be placed are among its jobs. When it is created, it also creates other agents of the proposed fault tolerant system. Mobile agent helps the replication manager in placing the replicas when the replication manager decides its necessity. In order to realize that, it sends the mobile agent to a node (a processor connected to network) in the region where replication is required in order to respond to more requests or improve fault tolerance. Each new replica becomes a member of the replication group. Information related to the members of the group is hold in a member vector.

In the present work, mobile agent helps the replication manager to replicate new replicas. Therefore, it goes to the Grasshopper address given by the replication manager, places the new replica which its code is embedded into mobile agent, and activates the replica. Thus, a new member (replica) joins the group. The different behaviors of the mobile agent are realized by means of the agent's data state that is

represented with an integer variable called state. When the state is equal to zero, mobile agent waits for a move command and a location to migrate from the replication manager. If the replication manager commands the mobile agent move to a new location, the mobile agent sets the state to one and puts the replica code and necessary codes, needed for the operation of the replica and the communication, into streams. Thereafter, it migrates to the location whose address is given by the replication manager. After arrival of the mobile agent to the next location, mobile agent's new state is equal to one. In this state, the codes migrated with mobile agent are placed in a predefined directory and executed, and the mobile agent's state is then set to zero. Since all these operations are executed in an infinite loop, mobile agent continues its task from the initial state.

A replica is placed and activated by the mobile agent in the location whose address is given by replication manager as mentioned above. When a replica receives a request, before processing the request it sends the request to the sequencer agent, which assigns a number to the requests. Thereafter the request that is numbered by the sequencer agent is sent to the all member replicas or only the primary replica based on the replication strategy. In case of passive replication, a response is sent back to the client after the request is received and processed by the primary replica.

In case of active replication, the request is received and processed by each replica. However, each replica is required to be a state machine and runs on a nonfaulty processor for active replication scheme. The state of a replica is a deterministic function of their initial states and the sequence of operations that it applies to them. Each replica starts in the same initial state and executes the same requests in the same order. Therefore, all replicas do the same thing and produce the same output. However, the agents' behaviors are not deterministic due to their multithreaded natures. Therefore, I preferred applying semi-active replication strategy instead of active replication strategy to the replica group besides the passive replication strategy, since I aimed to improve fault tolerance in both distributed systems and multi-agent systems. In semi-active replication strategy, replicas are organized in a group, and all replicas execute incoming requests. One replica is designated as the primary replica and responsible from providing responses.

As mentioned above, the requests received from clients/agents are sent to the sequencer agent by means of the Grasshopper agent platform communication service. Moreover, all communication between replicas and fault tolerance middleware components achieved by Grasshopper agent platform communication service. The Grasshopper Communication service is an essential part of each core agency and allows location-transparent interactions between agents, agencies, and non-agent-based entities. Remote interactions are generally achieved by means of specific protocols such as the Internet Inter-ORB Protocol (IIOP), Java's Remote Method Invocation (RMI), and plain socket connections. The communication service supports synchronous and asynchronous communication, multicast communication, as well as dynamic method invocation. In order to use the Grasshopper communication service for sending the request to the sequencer agent, the server just creates a proxy of the sequencer agent. However, Grasshopper communication service does not support ordering schemes for group communication service. Therefore, a sequencer agent is implemented to provide total ordering scheme to communication service in replica groups.

In order to implement this scheme, the sequencer agent assigns totally ordered identifiers to requests so that each server in the group makes the same decision based on these identifiers. Whenever a sequencer agent receives a request to be enumerated, the sequencer assigns a group specific ascending number to that request and sends it to the replica members, whose addresses are hold in membership vector, by using reliable multiple unicast. In order to achieve reliable multiple unicast, a server must implement a server interface that defines those methods that have to be made accessible via the Grasshopper communication service. The sequencer agent contacts each server interface to send the request back to each server. The request is hold in a queue until all servers in the group receive it. If any server finds out that it has lost a request, it will then get it from the sequencer.

In order to send each enumerated request or state to the group, group communication incorporates membership service which holds the list of member replicas. Group membership service incorporates failure detectors which are basic building block for fault tolerant systems. Many fault tolerant algorithms have been proposed based on unreliable failure detectors. A failure detector maintains a list of processes that it currently suspects to have crashed. If a failure detector is unreliable, it can mistakenly add a running process to the list. However, if the failure detector then decides that the process has not crashed and continues running, it removes the same process from the list.

The failure detector concerned here is an unreliable failure detector. It multicasts periodically the numbered heartbeatings to the replicas. Upon reception of the number, each replica sends back the same number to the failure detector. If the number sent back by any replica is not received during timeout or the number received by the sequencer is not the same number sent by the sequencer that replica is added to a suspect list. In the case of reoccurrence of timeout or disparity, the replica is removed from the group.

The components of the system mentioned above are basic building blocks of the fault tolerant system using agents. In those kinds of systems, it is necessary that communication within the group must be highly reliable. Multiple unicast is obviously not efficient since it wastes network bandwidth. However, if the number of replicas is not large, achieving reliable communication through multiple unicast is a simple solution. In order to achieve reliable communication in a very large replica group, hierarchical approach is adopted. In hierarchical approach, the replica group is partitioned into a number of subgroups. Each subgroup has own its fault tolerant building system as shown in Fig. 1. Whenever a replica in any subgroup receives a request, it handles the request in the subgroup as mentioned above and the response is sent to the client/agent. However, in order to provide consistency between the replicas in other subgroups, while the sequencer agent are sending the numbered request to the replicas in the current subgroup, it also sends the request to other sequencers for delivering the request to the other replicas. The replicas in the other subgroups are then forced to take the same decision. This guarantees that the state evolution in all replicas is the same. However, the subgroup, which receives the request, replies the client/agent. By the way of hierarchical approach, the scalability problem can be solved. In addition to scalability, the fault tolerant building system mentioned above can be made fault tolerant against its own failures.

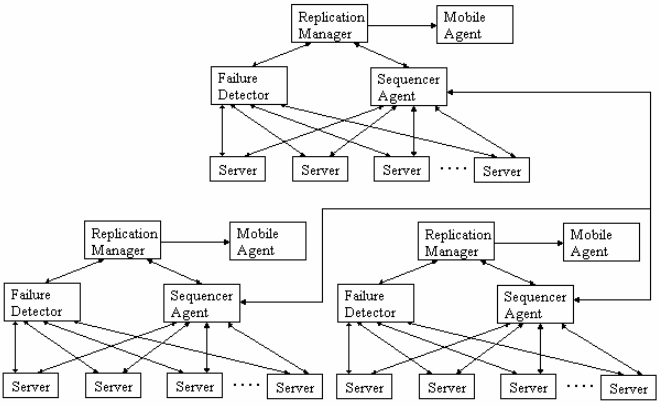



**Fig. 1.** A scalable fault tolerant system using agents

The presented fault tolerant middleware architecture is implemented to develop fault tolerant distributed systems. The same middleware architecture is used to develop fault tolerant multi-agent system (MAS). All components of fault tolerant middleware architecture are the same in case of using it in multi-agent systems. However, in case of fault tolerance in MAS, the agents will be replicated instead of services.

## 5 Summary

In this paper, we presented an approach to build fault tolerant distributed and multi-agent systems by exploiting the use of collaborative agents. The proposed system is based on classical fault tolerant mechanisms and collaboration of the agents that form a part of a multiagent system. Since the system is assumed to be in a dynamic environment, the replication strategy and the number of replicas will be adapted to meet the requirements of the environment by using collaboration of the agents. In the near future, the advisor module which makes system-wide decisions based on the system information and application requests, and determines a strategy for handling faults, is intended to be implemented. Since the agents are autonomous, intelligent and social software components, they can make such decisions in order to apply adaptive fault tolerance policy in replica groups.

The system components described above have been developed by using Grasshopper mobile agent platform. Although the project is still in a prototype development stage, first results are encouraging and indicating that efficient replication is sustainable using this model.

## References

1. The ISIS Project. http://www.cs.cornell.edu/Info/Projects/Isis/
2. The Horus Project. http://www.cs.cornell.edu/Info/Projects/HORUS/

3. Cristian, F. et al.: Fault-Tolerance in the Advanced Automation System. In 20th International Conference on Fault-Tolerant Computing, Newcastle upon Tyne, England (1990)
4. Mishra, S.: Consul: A Communication Substrate for Fault-Tolerant Distributed Programs. PhD thesis, Dept. of Computer Science, Univ. of Arizona,Tuscon, Arizona (1992)
5. Schneider, F.: Implementing Fault-Tolerant Services Using the State Machine Approach: A Tutorial. ACM Computing Surveys 22(4), pages 299-319. (1990)
6. Elmootazbellah, N. et al.: Replicated Distributed Processes. In Proceedings of the Twenty-Second International Symposium on Fault Tolerant Computing (FTCS-22) pages 18-27, (1992)
7. Shokri, E. et al.: An Approach for Adaptive Fault-Tolerance in Object-Oriented Open Distributed Systems. Workshop on Object-Oriented Reliable Distributed Systems (1997)
8. Ren, J., Cukier, M., Rubel, P., Sanders, W., Karr, D.: Building Dependable Distributed Applications using AQuA. In Proceeding of the 4th IEEE International Symp. On High Assurance Systems Engineering, pages 189-196, (1999)
9. Tanenbaum, A. S. and van Steen, M.: Distributed Systems: Principles and Paradigms, Pren- tice-Hall (2002)
10. Chockler, G. V., Keidar, I. and Vitenberg, R.: Group Communication Specifications: A Comprehensive Study. ACM Computing Surveys 33(4), pages 1-43, (2001)
11. Grasshopper Programmer's Guide. http://www.grasshopper.de.

# 3-D Object Recognition Using 2-D Poses Processed by CNNs and a GRNN

Övünç Polat and Vedat Tavşanoğlu

Electronics and Communications Engineering Department
Yıldız Technical University
Besiktas, Istanbul 34349, Turkey
{opolat, tavsanav}@yildiz.edu.tr

**Abstract.** This paper presents a novel approach to automatically recognize objects. The system used is a new model that contains two blocks; one for extracting direction and pixel features from object images using Cellular Neural Networks (CNN), and the other for classification of objects using a General Regression Neural Network (GRNN). A data set consisting of different properties of 10 different objects is prepared by CNN.

## 1 Introduction

Among the artificial neural network(ANN) approaches developed for pixel-based and feature-based object recognition are feed-forward, Hopfield, and fuzzy ANNs. In general, the types of features that are used for object recognition differ from the features used by the neural-based segmentation approaches [1].

In this work, object recognition is achieved by comparing extracted features of the object with the features available in a reference set. To this end, each object is placed on a turntable which is rotated through 360 degrees and poses are taken with a fixed camera. Images of the objects are taken at pose intervals of 5 degrees. This corresponds to 71 poses per object, in total 710 poses for 10 objects. By using a CNN, a feature space consisting of poses of the image taken from 71 directions is constituted and this procedure is repeated for all object images. Apart of the constituted feature vectors are used to simulate the GRNN, and the remaining are used as test patterns.

### 1.1 The Cellular Neural Network (CNN)

The CNN used in this paper is defined by the following equations:

$$\dot{x}_{ij} = -x_{ij} + \sum_{C(k,l)\in S_r(i,j)} A(i,j;k,l)\,y_{kl} + \sum_{C(k,l)\in S_r(i,j)} B(i,j;k,l)\,u_{kl} + z_{ij} \tag{1}$$

$$y_{ij} = f(x_{ij}) = \frac{1}{2}\left(|x_{ij}+1| - |x_{ij}-1|\right) \tag{2}$$

F.A. Savacı (Ed.): TAINN 2005, LNAI 3949, pp. 219 – 226, 2006.

where:

$x_{ij}$ is the state of cell C(i,j)

$u_{ij}$ is the input of cell C(i,j)

$y_{ij}$ is the output

A is the feedback template
B is the control or input template
z is the threshold of cell C(i,j)

In this paper the A and B templates used are of dimension 3x3 [2] and here eqn.(1) is solved using the rastering algorithm [3].

In the literature there are a number of papers that used feature extraction for character recognition. In [4] CNN-Gabor filters were used to extract orientation information from hand-written characters. In [5] various templates were used and the decision was taken on a basis of how many pixels remain black after processing and also the position of these black pixels were taken into account.

### 1.2 Artificial Neural Network

In this work, both multilayer perceptron(MLP) and radial basis function neural networks (RBFNN) are employed. Among all results, the best results are obtained by using the general regression neural network(GRNN) which is a kind of RBFNN. The GRNN model developed by Specht [6] is a powerful regression tool with a dynamic network structure whose network training speed is extremely fast. Due to the simplicity of the network structure and its implementation, it has been widely applied to a variety of fields including image processing. Specht [7] addressed the basic concept of inclusion of clustering techniques in the GRNN model.

## 2 The Realization of the System for Object Recognition

Fig. 1 shows the system for object recognition using a CNN and a GRNN. The system used is a new model that contains two blocks: one for extracting direction and pixel features from object images using a CNN, and the other for classification of objects using a GRNN.

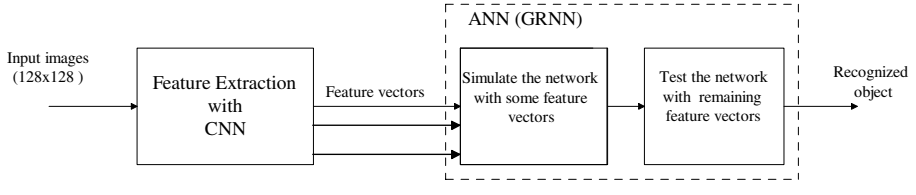



**Fig. 1.** System for object recognition using a CNN and a GRNN

### 2.1 Feature Extraction from the Object Images

In this work, a CNN is used for feature extraction from the input images obtained from the object to be recognized. Edge detection templates are applied to the input images which are converted to gray level and the number of pixels between the levels 0.4-0.8 of the image obtained is determined as a feature. Then the "Horizontal skeleton from the right" template is applied and this time the number of pixels equal to 1 and between 0.4-0.8 are determined as two different features. Also "Horizontal skeleton from the left", "Vertical skeleton from the bottom" and "Vertical skeleton from the top" templates are applied to the edge detected image and the number of pixels between 0.4-0.8 are determined as three different features. Thus, a feature vector consisting of six different features of the input images is obtained. Now the application of horizontal and vertical templates yields the components of the object images at that direction. The templates used for feature extraction from the input images are given as follows :

$$A=\begin{bmatrix}0&0&0\\0&1&0\\0&0&0\end{bmatrix}, B=\begin{bmatrix}-1&-1&-1\\-1&8&-1\\-1&-1&-1\end{bmatrix}, \text{z=-1,} \quad \text{(Edge detection)}$$

$$A=\begin{bmatrix}0&0&0\\0&3&0\\0&0&0\end{bmatrix}, B=\begin{bmatrix}0.5&0&0.125\\-0.5&0.5&-0.5\\0.5&0&0.125\end{bmatrix}, \text{z=-1} \quad \text{(Horizontal skeleton from the left)}$$

$$A=\begin{bmatrix}0&0&0\\0&3&0\\0&0&0\end{bmatrix}, B=\begin{bmatrix}0.125&0&0.5\\-0.5&0.5&-0.5\\0.125&0&0.5.\end{bmatrix}, \text{z=-1} \quad \text{(Horizontal Skeleton from the right)}$$

$$A=\begin{bmatrix}0&0&0\\0&3&0\\0&0&0\end{bmatrix}, B=\begin{bmatrix}0.5&-0.5&0.5\\0&0.5&0\\0.125&-0.5&0.125\end{bmatrix}, \text{z=-1} \quad \text{(Vertical skeleton from the top)}$$

$$A=\begin{bmatrix}0&0&0\\0&3&0\\0&0&0\end{bmatrix}, B=\begin{bmatrix}0.125&-0.5&0.125\\0&0.5&0\\0.5&-0.5&0.5\end{bmatrix}, \text{z=-1} \quad \text{(Vertical skeleton from the bottom)}$$

### 2.2 Recognition with GRNN

The feature vector, obtained by extracting features from ten object images by CNNs, is used to simulate and test the GRNN. In the simulation of GRNN, different number of poses from the original data set with different rotation intervals are selected as the reference poses. The remaining images in the data set are used to test the network. This procedure is shown in Figure 2.

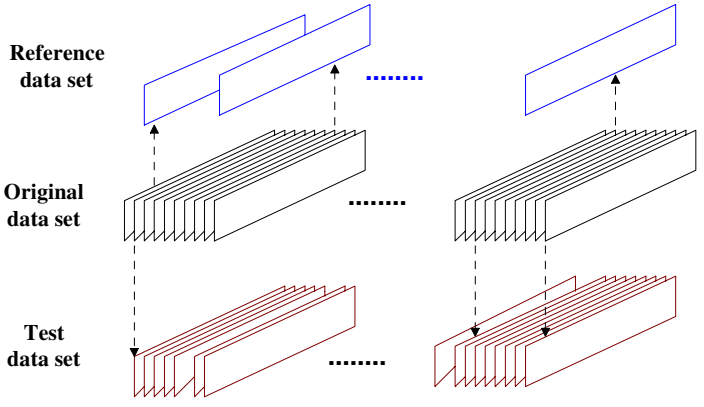



**Fig. 2** Grouping of the database

## 3 Simulation and Results

For the experiments we are currently using images of the Columbia image database. Columbia Object Image Library (COIL-100) is a database of color images of 100 objects. We selected the 10 objects from the data set shown in Figure 3. The objects

**Fig. 3.** 10 objects used the simulate recognition system

**Fig. 4.** The image sequence of object-6 in database with rotations $0^o$ to $25^o$

were placed on a motorized turntable against a black background. The turntable was rotated through 360 degrees to vary object pose with respect to a fixed color camera. Images of the objects were taken at pose intervals of 5 degrees. This corresponds to 72 poses per object. The images were size normalized.[8]. We take image sequences of 71 images of each of the selected objects. Figure 4 shows the frames with rotations $0^o$ to $25^o$ in the object-6 data set from COIL100.

The images obtained by applying "edge detection", "horizontal skeleton from the right" and "vertical skeleton from the bottom" templates to the pose of the object taken at $0^o$ rotation is given in Figure 5.

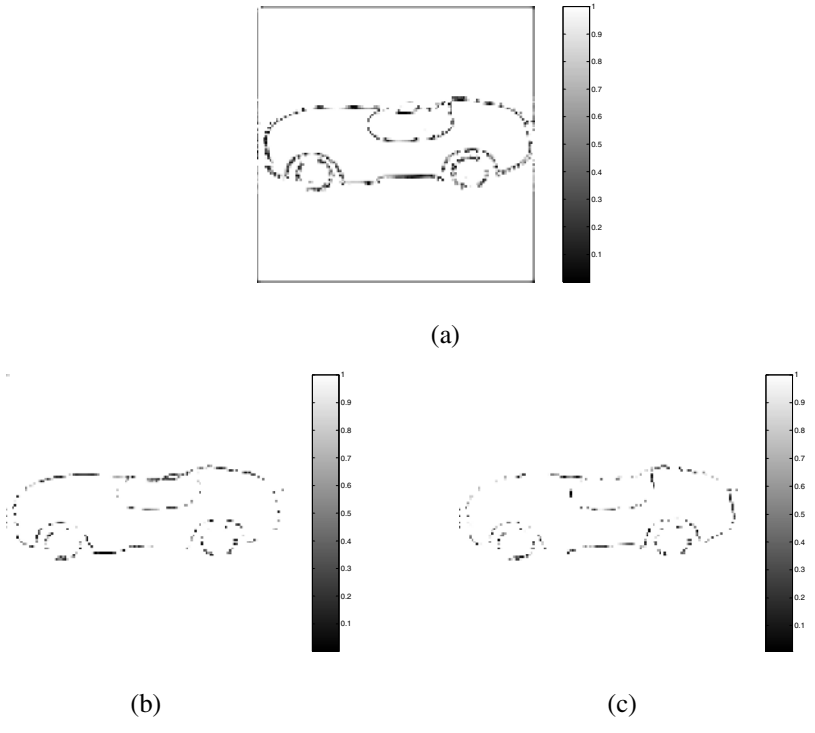



**Fig. 5.** Extracting features from object-6 with CNN a) Edge detection, b) Horizontal Skeleton from the right, c)Vertical skeleton from the bottom

As described above, different number of poses from the original data set with different rotation intervals are selected as reference poses. In this step, a code for every object is given to the GRNN target values. In Table 1, 10,18 and 35 poses from the original data set with $35^o$, $20^o$ and $10^o$ rotation intervals, respectively are selected as the reference poses. The remaining images in the data set are used to test the network.

In Table 2, the accuracy of the test results obtained by using 35 poses taken with $10^o$ rotation intervals for the ten objects considered are given.

**Table 1.** Number of reference images and corresponding recognition rates

| Number of reference images | Recognition rate of the test set |
|---|---|
| 10 poses from original data set with 35° rotation intervals are selected as the reference poses. The remaining 61 images in the data set are used to test the network. | % 82,13 |
| 18 poses from original data set with 20° rotation intervals are selected as the reference poses. The remaining 53 images in the data set are used to test the network. | % 86,42 |
| 35 poses from original data set with 10° rotation intervals are selected as the reference poses. The remaining 36 images in the data set are used to test the network. | % 89,17 |

**Table 2.** Recognition rates of objects

| | Obj1 | Obj2 | Obj3 | Obj4 | Obj5 | Obj6 | Obj7 | Obj8 | Obj9 | Obj10 | Average |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Recog. rate | % 97,2 | % 100 | % 97,2 | % 100 | %80,6 | %72,2 | % 100 | % 55,5 | % 97,2 | % 91,6 | % 89,17 |

As can be seen from Table 2, recognition rate is low only for the ones that are similar. Because of the similarities between object 6 and 8, recognition rate for these objects decrease with the decrease in the number of reference poses. In order to increase the recognition rate, new features are added to the feature vector and the GRNN is again simulated and tested. In the processing with CNN, gray level images are converted to binary images and edge detection process is carried out.

**Table 3.** Number of reference images and corresponding recognition rates after adding new features

| Number of reference images | Recognition rate of the test set |
|---|---|
| 10 poses from original data set with 35° rotation intervals are selected as the reference poses. The remaining 61 images in the data set are used to test the network. | % 87,21 |
| 18 poses from original data set with 20° rotation intervals are selected as the reference poses. The remaining 53 images in the data set are used to test the network. | % 91,32 |
| 35 poses from original data set with 10° rotation intervals are selected as the reference poses. The remaining 36 images in the data set are used to test the network. | % 94,17 |

Then the "Horizontal skeleton from the right" and "Vertical skeleton from the bottom" templates are applied to the edge detected images and this time the number of pixels between 0.1-0.3 are determined as two different features. Thus, feature vector consisting eight different features of the input images is obtained. The test results of the simulated network with new features vector is given in Table 3.

Also in Table 4 the accuracy of the test results obtained by using 35 poses taken with $10^o$ rotation intervals for the ten objects considered are given.

**Table 4.** Recognition rates of objects after adding new features

| | Obj1 | Obj2 | Obj3 | Obj4 | Obj5 | Obj6 | Obj7 | Obj8 | Obj9 | Obj10 | Average |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Recog. rate | %97,2 | %100 | %100 | %100 | %100 | % 80,56 | %100 | % 75 | %91,67 | % 97,2 | % 94,17 |

As can be seen from Tables 3 and 4 the recognition rate in the test process increases after adding new features to the features vector. Also recognition rate in the test process varies with different spread(*width of the kernel in GRNN)* values. Depending on the number of reference poses, spread value can be determined in order to increase recognition rate.

## 4 Conclusion

In this work, a model is designed for 3D object recognition using 2D poses processed by CNNs where the recognition is achieved by a GRNN. The application is carried out for 10 objects and high recognition rate is obtained. When the number of objects is increased or similar objects are chosen for recognition, extraction of diagonal or convex components with CNNs procedure or choosing pixels between different intervals as input can help to increase the recognition rate.

## References

1. M. Egmont-Petersen, D. de Ridder, H. Handels. : Image processing with neural networks - a review. The Journal of the Pattern Recognition Society, Vol. 35, No. 10, pp. 2279-2301, (2002).
2. Chua, L.O., Roska,T.: Cellular Neural Networks & Visual Computing. Cambridge University Press (2002).
3. Saatci,E., Tavsanoglu,V. : On the optimal choice of integration time-step for raster simulation of a CNN for gray level image processing. Circuits and Systems, ISCAS (2002). IEEE International Symposium on, Volume:1, Pages:I-625 - I-628, (26-29 May 2002).
4. Saatci, E., Tavsanoglu, V.: Multiscale handwritten character recognition using CNN image filters. Neural Networks, 2002. IJCNN '02. Proceedings of the 2002 International Joint Conference on , Volume: 3 , Pages:2044 – 2048,( 12-17 May 2002).
5. Salermo, M., Sargeni, F., Bonaiuto, V., Favero, F.M. : Multifont character recognition by 9/spl times/9 DPCNN board. Circuits and Systems, 1997. Proceedings of the 40th Midwest Symposium on , Volume: 2 , Pages:1338 – 1341,( 3-6 Aug. 1997).

6. D. F. Specht : A general regression neural network. IEEE Trans. Neural Networks, vol. 2, pp. 568–576, (Nov. 1991).
7. D. F. Specht,: Enhancements to probabilistic neural network. in Proc.Int. Joint Conf. Neural Network, vol. 1, pp. 761–768, (1991).
8. Sameer A. Nene , Shree K. Nayar , Hiroshi Murase : Columbia Object Image Library (COIL-100) . Technical Report No. CUCS-006-96, Department of Computer Science Columbia University New York, N.Y. 10027.

# Author Index