# Chapter 3
# Vision-Based Mobile Robot Navigation Using Subgoals*

**Abstract.** This chapter discusses how a vision based robot navigation scheme can be developed, in a two-layered architecture, in collaboration with IR sensors. The algorithm employs a subgoal based scheme where the attempt is made to follow the shortest path to reach the final goal and also simultaneously achieve the desired obstacle avoidance. The algorithm operates in an iterative fashion with the objective of creating the next subgoal and navigating upto this point in a single iteration such that the final goal is reached in minimum number of iterations, as far as practicable.

## 3.1 Introduction

Recent advances in technologies in the area of robotics have made enormous contributions in many industrial and social domains. Nowadays numerous applications of robotic systems can be found in factory automation, surveillance systems, quality control systems, AGVs (autonomous guided vehicles), disaster fighting, medical assistance etc. More and more robotic applications are now aimed at improving our day-to-day lives, and robots can be seen more often than ever before performing various tasks in disguise [1]. For many such applications, autonomous mobility of robots is a mandatory key issue. Many modern robotic applications now employ computer vision as the primary sensing mechanism. As mentioned earlier in this book, vision system is considered as a passive sensor and possesses some fundamental advantages over the active sensors such as infrared, laser, and sonar sensors. Passive sensors such as cameras do not alter the environment by emitting lights or waves in the process of acquiring data, and also the obtained image contains more information (i.e. substantial, spatial and temporal information) than active sensors [2]. Vision is the sense that enables humans to extract relevant information about the physical world, and appropriately it is the sense that we, the humans, rely on most. Computer vision

* This chapter is adopted from Measurement, vol. 44, issue 4, May 2011, N. Nirmal Singh, Avishek Chatterjee, Amitava Chatterjee, and Anjan Rakshit, “A two-layered subgoal based mobile robot navigation algorithm with vision system and IR sensors,” pp. 620-641, © 2011, with permission from Elsevier.

techniques capable of extracting such information are continuously being developed and more and more real-time vision-based navigation systems for mobile robots are being implemented now.

Vision based Robot navigation is defined as the technique that guides a mobile robot to a desired destination, or along a desired path in an environment, by avoiding static (and may be dynamic) obstacles primarily using vision sensor [3], [4]. In this chapter, we describe the real-life implementation of a mobile robot navigation scheme, where vision sensing is employed as primary sensor for path planning and IR sensors are employed as secondary sensors for actual navigation of the mobile robot with obstacle avoidance capability in a static or dynamic indoor environment. As described previously, the popular choices for the creation of the environment maps can be grid-based [5, 6, 7], topological map [8, 9], hybrid map [10] etc. The mapless navigation systems are those that use no explicit representation at all of the space in which navigation is to take place and they rather resort to recognizing objects found in the environment or to tracking or avoiding those objects by generating motion commands based on visual observations [11, 12]. Several research works have so far been reported to acquire knowledge about the environment using camera(s) in stereo vision [13, 14], trinocular vision [15], omni-directional or panoramic vision [16, 17], and monocular vision [18, 19]. Each such solution in mobile robot navigation has its own advantages and disadvantages. In those situations where the knowledge of the map is available, an important problem in navigation is the path planning for intelligent control or guidance of the mobile robot. The popular general approaches for path planning can be based on roadmap, cell decomposition, potential field etc. [20]. They differ in how the connectivity graphs are constructed and their representations. Obviously, without any *a priori* knowledge of an environment, it is almost impossible to determine the true shortest path for navigation, among all possible paths. It is potentially possible to determine such paths by employing standard graph-search techniques, such as Dijkstra's algorithm [21] and A* algorithm [22].

As mentioned earlier, in this chapter we describe a goal driven approach for mobile robot navigation, using vision based sensing and IR sensor based navigation [28, 29]. This two-layer based approach attempts to determine the shortest path of navigation between the start point and the known goal point, given a static or dynamic environment, in presence of obstacles. In the first layer, vision acts as the primary sensing system to acquire image of the environment, for subsequent path planning. A series of image processing operations is performed on the acquired image and then a gradient descent based algorithm is employed to compute the shortest path between the present position of the robot and the goal, avoiding obstacles [26]. This shortest path is employed to generate a subgoal and this information is then locally utilized to navigate the robot, utilizing IR sensor based guidance. This second layer of IR sensor based robot navigation attempts to guide the robot to the subgoal, even if the environment changes dynamically. Once the robot reaches the subgoal, the two-layer based algorithm is again activated to generate a new subgoal and to navigate the robot till this new subgoal is reached. This process is repeated iteratively until the final goal is reached. This method simultaneously attempts to attain two objectives. Based on vision sensing, it attempts to implement a shortest path planning algorithm in a bid to reach the goal, avoiding obstacles, as fast as it can. Then, if the environment undergoes a change during navigation and obstacle information gets updated, then IR sensor based guidance equips the robot with the capability of handling the changed environment so that the robot can still navigate safely. The periodic usage of vision based updating of the environment, subsequent path planning and then IR based actual navigation helps to guide the robot to adapt its navigation temporally with dynamic variations in the environment and still attempt to reach the goal in shortest time, as quickly as practicable. This algorithm was implemented in our laboratory, for the KOALA robot [23], creating several real-life like environments. The results showed the usefulness of the proposed algorithm. The algorithm is described in detail in subsequent sections of this chapter.

## 3.2 The Hardware Setup

The KOALA robot was described in detail in the previous chapter. Still we recapitulate salient features of the KOALA robot to provide a brief introduction of the hardware setup utilized for this real-life implementation carried out. KOALA is a small (32 cm x 32 cm) six wheeled, differential drive vehicle manufactured by K-team, Switzerland [23]. The KOALA robot used in our laboratory is equipped with 16-proximity/ambient IR sensors, four ultrasonic sensors and wheel encoders. We have integrated two complete vision systems along with the KOALA robot in our Electrical Measurement and Instrumentation Laboratory, Electrical Engineering Department, Jadavpur University, Kolkata. The vision system is so developed that it can work either with a stereo vision system employing two cameras (as described in the previous chapter) or it can employ a single camera based system. The algorithm that we describe now is based on employing a single wireless camera for monocular vision. In KOALA, the hardware control is performed by an on- board microprocessor (Motorola 68331@ operating frequency 22MHz) [23]. Figure 3.1(a) shows a snapshot of the mobile robot with four ultrasonic sensors and the vision system, integrated in our laboratory, employing a single vision sensor. The ultrasonic sensors can detect obstacles over a wide range from 15 cm to 300 cm, and the IR sensors will provide a range of measurements from 5 cm to 20 cm. Our system utilizes single vision sensor comprising a JMK wireless camera (WS-309AS) with A/V receiver and a Frontech USB (TV Box) frame grabber, which is used for acquiring a running video stream. Figure 3.1(b) shows the vision system in schematic form. The entire system is developed with an objective of providing a low-cost solution which should prove attractive for the industrial community. This monocular vision system is developed with two degrees of freedom to provide pan control and tilt control. To add two degrees of freedom (DOFs) for this vision-system, a PIC (16F876A) microcontroller based system is developed in our laboratory for pan-control and tilt-control of the single-camera based robot system [24]. Here, the main onboard Motorola microcontroller acts as the master and the PIC microcontroller acts as a slave. The software, developed in interrupt driven mode, communicates with the mobile robot through the RS232C port. Figure 3.2 shows a snapshot of the user-interface developed in the PC side that can interact with the user. The main serial mode of communication is handled by passing ASCII message strings between the PC and the Motorola processor in the robot.

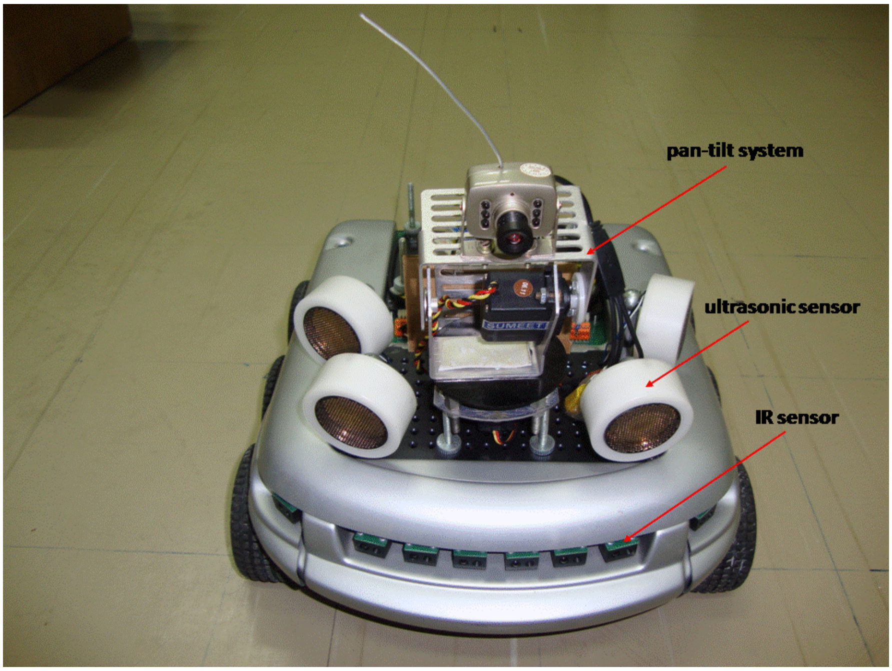


**Fig. 3.1(a).** The KOALA robot, equipped with sonar and IR sensors and integrated with a single camera based vision system

The RS 232C serial link set-up between the PC and the robot is always set at 8 bit data, 1 start bit, 2 stop bits and no parity mode. To give an example, the message string for RC servo action to provide pan or tilt control, sent from the PC end, comprise ‘Z’ as the identifier character, followed by <i>, which corresponds to the servomotor id for which position command is prepared (i = 1, 2), followed by the sign, given in <s>, for position command (‘+’ or ‘-’) and then the two digit actual position command, in degree (this can vary from -90$^0$ to +90$^0$). Every ASCII message string is terminated, as usual, by using carriage return, <CR>.

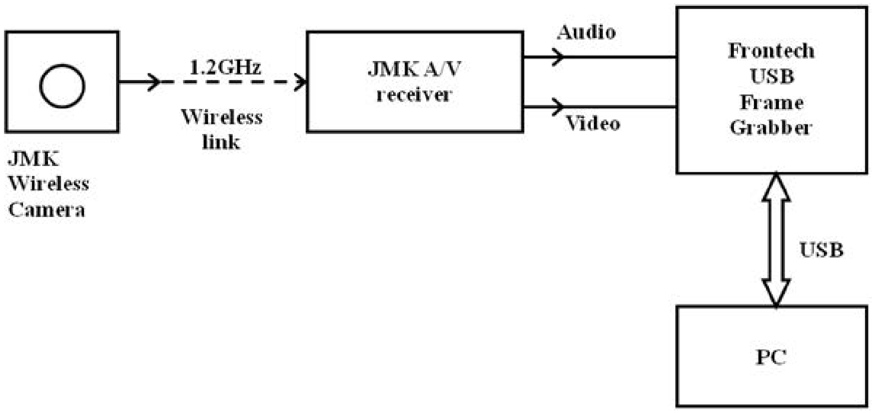
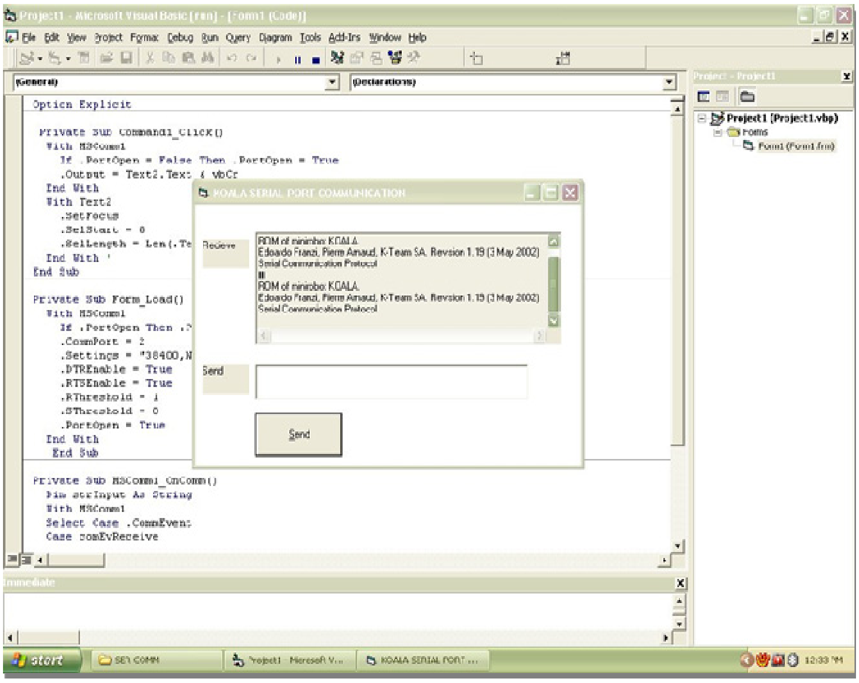


**Fig. 3.1(b).** The block diagram of the vision system



Project1 - Microsoft Visual Basic [run] - [Form1 (Code)]
File Edit View Project Format Debug Run Query Diagram Tools Add-Ins Window Help
(General)
(Declarations)
Project - Project1
Project1 (Project1.vbp)
Forms
Form1 (Form1.frm)
KOALA SERIAL PORT COMMUNICATION
Recieve
Send
Send
Immediate
start
SER COMM
Project1 - Microsoft V...
KOALA SERIAL PORT ...
12:33 PM

```
Option Explicit

 Private Sub Command1_Click()
  With MSComm1
    If .PortOpen = False Then .PortOpen = True
    .Output = Text2.Text & vbCr
  End With
  With Text2
    .SetFocus
    .SelStart = 0
    .SelLength = Len(.Te
  End With '
End Sub

Private Sub Form_Load()
  With MSComm1
    If .PortOpen Then .P
    .CommPort = 2
    .Settings = "38400,N
    .DTREnable = True
    .RTSEnable = True
    .RThreshold = 1
    .SThreshold = 0
    .PortOpen = True
  End With
   End Sub

Private Sub MSComm1_OnComm()
  Dim strInput As String
  With MSComm1
  Select Case .CommEvent
  Case comEvReceive
```

```
ROM of minirobo: KOALA
Edoardo Franzi, Pierre Arnaud, K-Team SA. Revision 1.19 (3 May 2002)
Serial Communication Protocol
II
ROM of minirobo: KOALA
Edoardo Franzi, Pierre Arnaud, K-Team SA. Revision 1.19 (3 May 2002)
Serial Communication Protocol
```

**Fig. 3.2.** Snapshot of the user-interface developed, that can interact with the user

## 3.3 A Two-Layer, Goal Oriented Navigation Scheme

Figure 3.3 shows the complete proposed navigation algorithm in a flow chart form. A wireless camera, as shown in Fig. 3.1(a), is used to capture a running video stream of the environment in front of the KOALA robot. An image frame can be acquired from this video stream for further processing at any point of time. This acquired image frame is first processed to make the image suitable for further processing, by employing a series of image processing operations like image filtering, edge detection and image segmentation. Then the shortest path

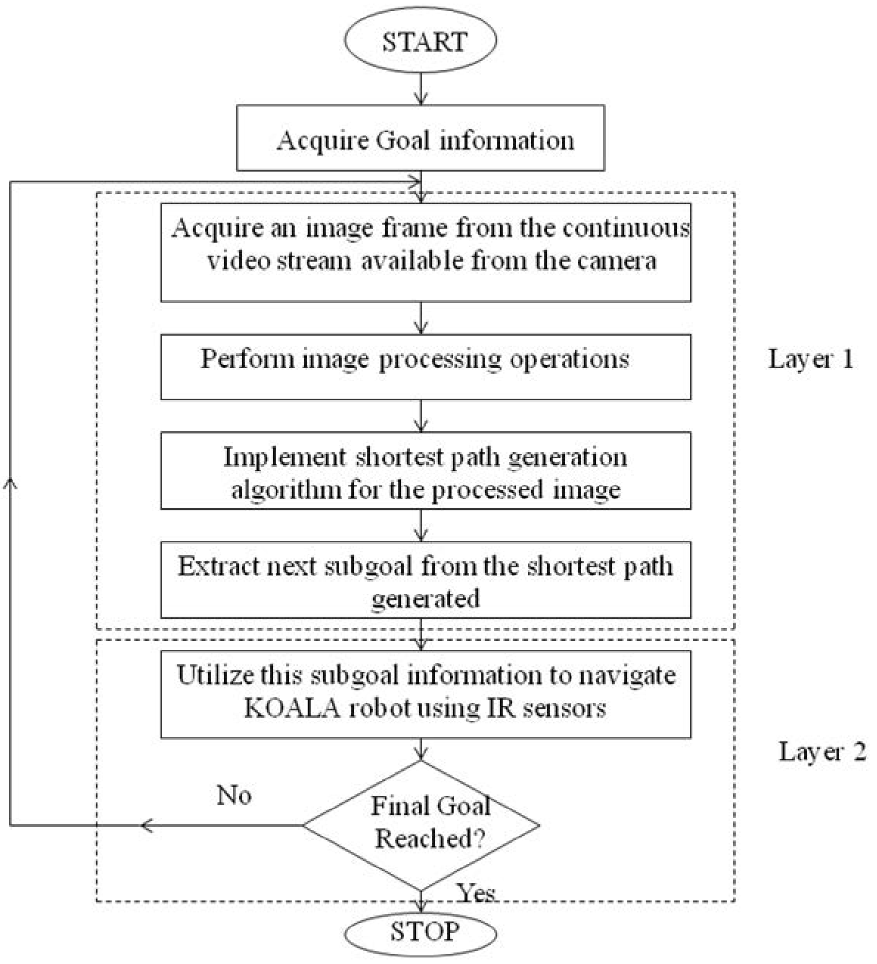

START
Acquire Goal information
Acquire an image frame from the continuous video stream available from the camera
Perform image processing operations
Implement shortest path generation algorithm for the processed image
Extract next subgoal from the shortest path generated
Layer 1
Utilize this subgoal information to navigate KOALA robot using IR sensors
Layer 2
No
Final Goal Reached?
Yes
STOP

**Fig. 3.3.** Flow chart for the proposed navigation algorithm

generation algorithm is employed for this processed image, using the goal information, available *a priori*. Next the shortest path generated is utilized to determine the next subgoal. This entire procedure constitutes layer 1 of the algorithm and is implemented in high level in a PC using Visual Basic (VB) platform. This subgoal information is next transferred to layer 2 where the KOALA robot is actually navigated towards the subgoal using obstacle avoidance capability so that the robot can be useful even in a dynamically changing environment. The navigation in layer 2 is performed using several IR sensors, connected at the front face and side faces of the KOALA robot. Once the subgoal is reached, the control is transferred back to layer 1 so that the next subgoal can be generated and actual navigation can be performed in layer 2. This process of local path planning, followed by actual navigation, is continued in an iterative fashion, until the final goal is reached. The algorithm in layer 2 for actual navigation is implemented by developing a C program whose cross-compiled version (a .s37 file) is downloaded in the Motorola processor of the KOALA robot. This .s37 program communicates with the VB program in the PC end, in the interrupt driven mode, in real life. The .s37 program generated from the C program written, is also equipped with the facility of providing support from VB based PC end for a pool of protocol commands for commanding the KOALA robot. These commands are originally only available for execution from a terminal emulator available with the KOALA robot package. We developed a system where all the KOALA robot protocol commands and our additional navigation algorithms are supported by the C program developed, so that the entire system can be completely controlled from the VB platform in the PC end.

## 3.4 Image Processing Based Exploration of the Environment in Layer 1

Image processing is a form of signal processing where the input signals are images and the output could be a transformed version of the input. The proposed system employs a map building method based on image segmentation, for vision based navigation for mobile robot in an indoor environment, with the assumption that the surface is uniform. The following steps are implemented as follows [27]:

*A. Acquire the image from the wireless camera*

The camera, mounted at the center of the pan-tilt system of the robot, keeps acquiring a running video stream of the environment ahead of it. From this video stream, an image frame can be acquired for further processing. Figure 3.4(a) shows such an acquired image.

*B. Employ low-pass filtering on the acquired image*

The acquired image is then low pass filtered to reduce noise. This causes a smoothing or blurring effect on the neighboring pixels. The system is developed using the popular arithmetic mean filter to perform low pass filtering. This arithmetic mean filter is utilized using a 5×5 matrix, centered on each pixel, whose intensity is computed as the average value of the pixels under the influence of the filter matrix.

*C. Detect edges in the filtered image by Canny edge detection*

An edge physically signifies a boundary between two regions with relatively distinct gray-level properties. The technique of edge-based segmentation signifies isolation of desired objects from a scene using different types of gradient operators. Edges of the image in our work are detected by using canny edge detection method. Figure 3.4(b) shows the edge image of the processed filtered version of the acquired image.

*D. Process the edge image to thicken and link the edges*

The edge image contains many small broken edges. To make any edge image a meaningful one, one needs to link nearby edges to bridge gaps and they can be thickened to make their presence distinct. Thickening can be performed by a morphological operation called dilation by a structuring element that is used to grow selected regions of foreground pixels in images. Dilation is normally applied to binary images, and it produces another binary image as output. This dilation operation "thickens" or "grows" objects in a binary image and the shape of thickening can be controlled by a suitable choice of the structuring element shape, used to perform dilation of the image. The concept of linking edges and thickening them by dilation in an edge image can also be performed by a suitable low pass filtering scheme with a suitable choice of the filter mask. This operation is carried out in this work by using geometric mean filtering. The geometric mean filter is member of a set of nonlinear mean filters, which are efficient in removing Gaussian type noise and preserving edge features than the arithmetic mean filter. Figure 3.4(c) shows the edge linked and thickened image.

*E. Perform region growing segmentation on the thickened edge image*

Once the thickening is done, the image is segregated into regions. To find the obstructed zone and unobstructed zone in the image, region growing based segmentation is performed on the thickened image. Region growing is a simple but efficient region-based image segmentation method and it is classified as one of the pixel-based image segmentation schemes which involves the selection of initial seed points. This approach to segmentation examines the neighboring pixels of the initial "seed points" and determines if the pixel should be added to the seed point or not. Region growing is done by examining properties of each such block created and merging them with adjacent blocks that satisfy some criteria (similar gray-level pixel values, texture etc). The seed point needed for performing region growing is chosen near the bottom center of the image. This point 'S' is shown in Fig. 3.4(c). Now the image is scanned along all the vertical lines from bottom to top. The point at which the floor area ends is regarded as the obstacle. All regions before the obstacles are free zone. All regions beyond the obstacles are termed the hidden zone. Figure 3.4(d) shows the unobstructed zone (free space) with green color and the hidden zone with yellow color. Next the obstructed zone is marked in red and Fig. 3.4(e) shows all these three regions. This entire process is continuous and the obstacle information gets continuously updated.

*F. Transform the region grown image to the floor region*

The entire grown up region updated with obstacle information is now transformed from image plane to floor region. In order to calculate a distance in the 3D coordinates using single camera, we assume that all the objects have contact at the bottom and interpret it in two dimensional coordinates. Figure 3.5 shows the relationship that, given the elevation of the camera and the elevation angle, how any point on the image plane can be directly mapped on the floor, relative to the position of the camera [25]. Here the robot/camera 3D coordinate frame is assumed with the corresponding notations shown in Fig. 3.5. This coordinate frame is assumed attached to present pose of the robot/camera, at any instant of time. This coordinate transformation mechanism allows one to determine the free points and the obstructed points in the world coordinate system (WCS) from the image acquired by the camera. Hence, with reference to Fig. 3.5, any point with coordinates ($u, v$) in the image plane can be transformed to the coordinates in the two dimensions $(x_c, y_c)$ in the robot/camera coordinate frame as:

$$x_c = \frac{uh}{(f \sin \theta_{EL} + v \cos \theta_{EL})}$$

$$y_c = \frac{h(v \sin \theta_{EL} - f \cos \theta_{EL})}{(v \cos \theta_{EL} + f \sin \theta_{EL})}$$

where

$h$ = height of the camera optical center from base plane

$f$ = focal length of the camera

$\theta_{EL}$ = elevation angle of the camera

$y_{c_{\min}} = A(\text{Dead zone})$

$y_{c_{\max}} = B$

$x_{c_{\min}} = \frac{C}{2}$

$x_{c_{\max}} = \frac{D}{2}$

Once this transformation is employed, one can obtain the actual position of a point $(x, y)$ on the floor, given this $(x_c, y_c)$ and the present pose of the robot$(x_R, y_R, \phi_R)$. Figure 3.4(f) shows the floor with obstacle information. The transformed floor region is in trapezoidal form. Then this floor plane image is copied to the 500 pixel x 500 pixel map which is 20m x 20m as a working space for the robot. Figure 3.6(a) shows a snapshot of the map created and Fig. 3.6(b) shows a snapshot with the floor image in the grid map. In Fig. 3.6(b), the trapezoidal floor region is shown in green color and the obstacle information is shown in red color. The above process of transformation is continuous even when the robot is in motion and it updates the new obstacle information in the map when it is in motion.

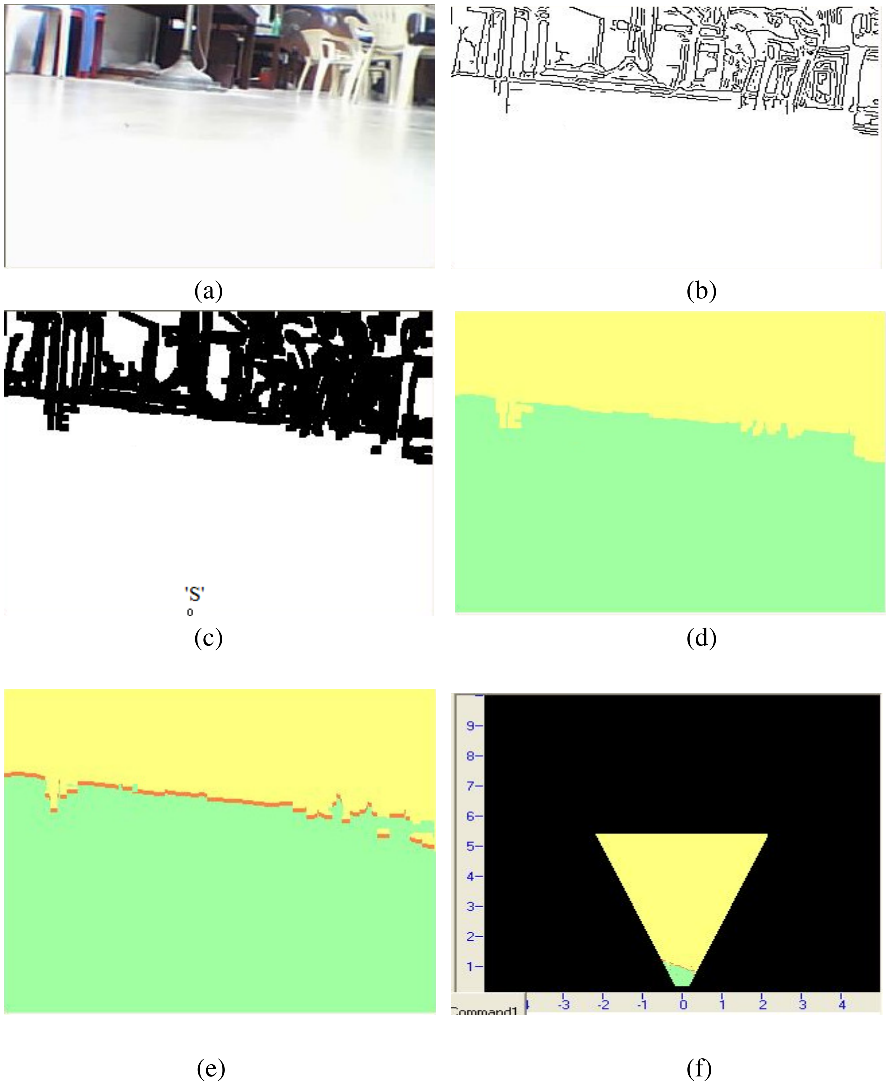


**Fig. 3.4.** (a) Image acquired by the wireless camera, (b) detected edge image, (c) thickened image, (d) region grown image, (e) image with the obstacle information, and (f) trapezoidal floor image

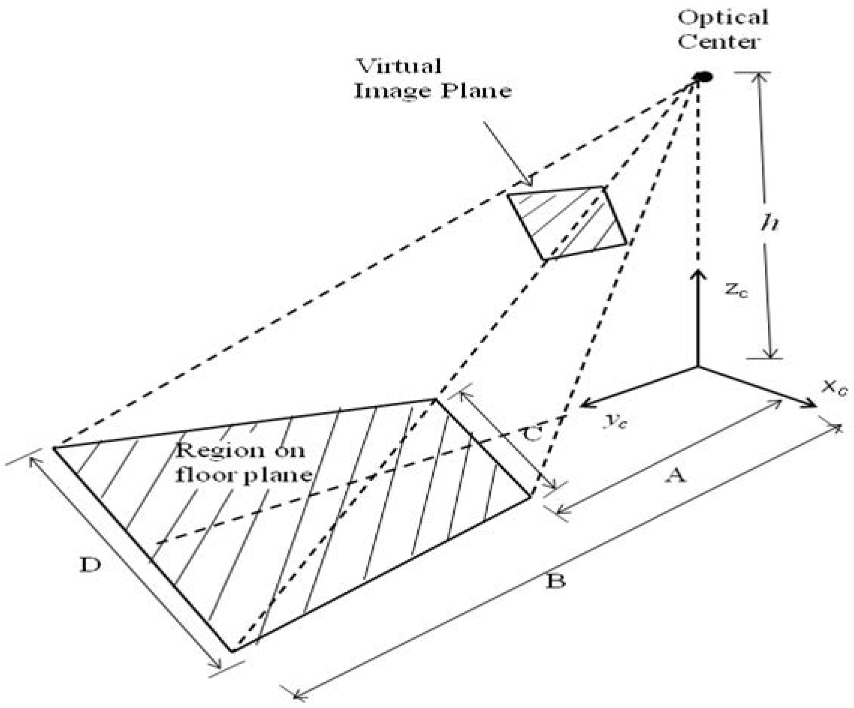


**Fig. 3.5.** Relationship between the image coordinate and the mobile robot coordinate

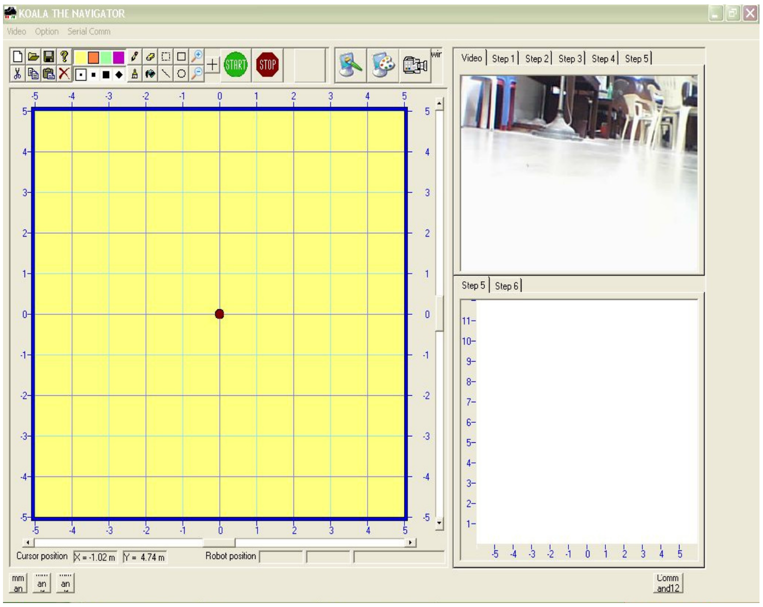


**Fig. 3.6(a).** A snapshot of the software developed for creating map

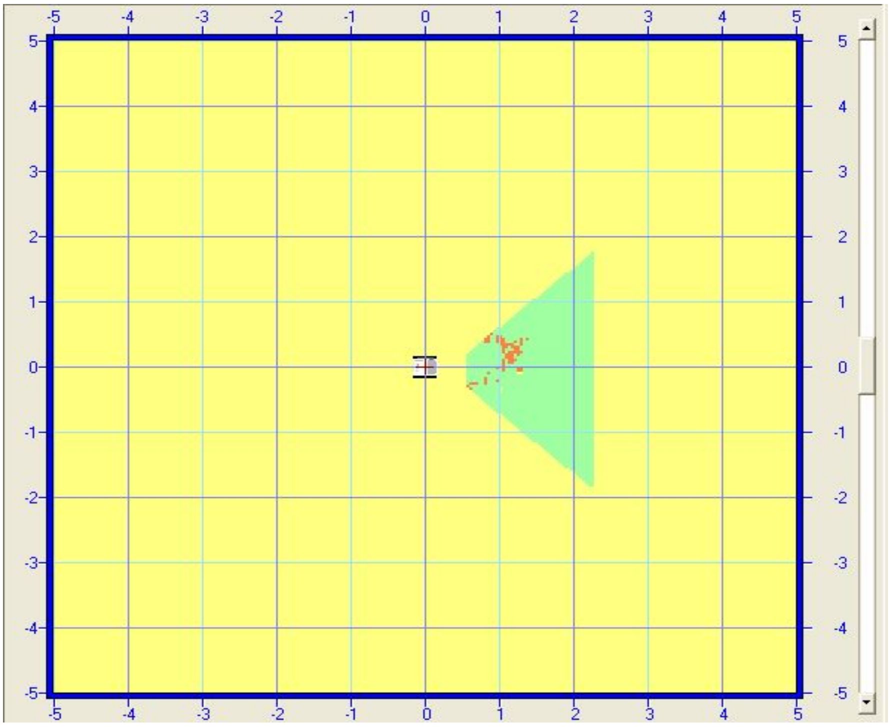


**Fig. 3.6(b).** A snapshot of the map updated with obstacle information

## 3.5 Shortest Path Computation and Subgoal Generation

In mobile robot navigation it should be an important objective to determine the optimum path between the present robot location and the goal point, so that the robot can reach the destination in minimum time, avoiding obstacles, as far as practicable. The present work employs a heuristic gradient based method which is based on grid-map for finding the shortest path [26]. Algorithm 3.1 shows this algorithm in detail. The initial and the final positions of the robot are known *a priori* with the obstacle information determined from the previous steps. Now the coordinates along the shortest path are determined by using steepest descent method. The steepest decent algorithm uses the gradient function to determine the direction in which a function is decreasing most rapidly. Each successive iteration of the algorithm moves along this direction for a specified step size and then recomputes the gradient to determine the new direction of travel. This heuristic approach employed here can be easily understood if a land is considered with known obstacles and the initial point and final point on it. The land surface is assumed frictionless such that, say, at the starting point, if we start pouring sand on the ground, it spreads towards all possible paths, similar to dispersion of a fluid in all possible directions. It is obvious that one cannot pass through the obstacles. In each iteration, we assume that a fixed amount of sand is poured and we let it spread. We set a time index to every point on the ground, equal to the iteration number, when the sand reaches a pre-assumed height. So the earlier the height is reached, the smaller is this index. Such a pre assumed height and a fixed amount of sand dispersed are chosen so as to avoid saturation in value within any finite considerable region. Hence a travel time matrix (**H**) can be calculated employing the finite element diffusion method and this **H** matrix is iteratively updated, until a termination criterion is met. At the end of this procedure, those entries in **H** which still contain zeros correspond to the obstacle cells. Next, the gradient descent based procedure is employed to determine the coordinates of the points on the shortest path by starting from the goal point and finally arriving at the present robot location. For this, the gradient matrices of **H** in *x*- and *y*-directions, i.e. $\nabla\mathbf{H}_x$ and $\nabla\mathbf{H}_y$, are calculated and based on them the new co-ordinates of the next point on the shortest path are computed, utilizing the last point obtained on the shortest path. The algorithm always proceeds backwards starting from the goal point. This method is an efficient one and it operates in an iterative fashion. Figure 3.7 shows a sample environment where the shortest path is computed between the initial and the goal point in the map.

```
BEGIN
1. Obtain the Occupancy grid matrix (M), the start point ($x\_start$, $y\_start$), and the goal point ($x\_goal$, $y\_goal$). $\mathbf{M}(i,j) = 0$ denotes a free cell and $\mathbf{M}(i,j) = 1$ denotes an obstacle cell.
2. Create diffusion matrix (W) and Travel Time Matrix (H) and make them of same size as M.    Initialize $\mathbf{W}_0 = \mathbf{H}_0 = \mathbf{0}$.
3. Set $\mathbf{W}_0(x\_start, y\_start) = 1$.
4. Set diffusion constant $(diffconst)$ and maximum number of iterations without updates $(no\_update\_max)$. Initialize number of iterations $(iter\_count)$ and number of iterations without updates $(no\_update\_iter\_count)$.
5. WHILE $(no\_update\_iter\_count < no\_update\_max)$
    5.1. $iter\_count = iter\_count + 1$.
    5.2. Diffuse cells downwards:
    $\mathbf{W}_{iter\_count}(i,j) = \mathbf{W}_{iter\_count}(i,j) + diffconst * \mathbf{W}_{iter\_count}(i+1,j)$
                    $i = 1,2,\ldots,(W\_ROWS-1); j = 1,2,\ldots,W\_COLS;$
    5.3. Diffuse cells upwards:
    $\mathbf{W}_{iter\_count}(i,j) = \mathbf{W}_{iter\_count}(i,j) + diffconst * \mathbf{W}_{iter\_count}(i-1,j)$
                    $i = 2,3,\ldots,W\_ROWS; j = 1,2,\ldots,W\_COLS;$
    5.4. Diffuse cells towards right:
    $\mathbf{W}_{iter\_count}(i,j) = \mathbf{W}_{iter\_count}(i,j) + diffconst * \mathbf{W}_{iter\_count}(i,j+1)$
                    $i = 1,2,\ldots,W\_ROWS; j = 1,2,\ldots,(W\_COLS-1);$
    5.5. Diffuse cells towards left:
    $\mathbf{W}_{iter\_count}(i,j) = \mathbf{W}_{iter\_count}(i,j) + diffconst * \mathbf{W}_{iter\_count}(i,j-1)$
                    $i = 1,2,\ldots,W\_ROWS; j = 2,3,\ldots,W\_COLS;$
    5.6. Make $\mathbf{W}_{iter\_count}(i,j) = 0$, if $\mathbf{M}(i,j) = 1$;
              $i = 1,2,\ldots,W\_ROWS; j = 1,2,\ldots,W\_COLS$;
    5.7. If any $\mathbf{W}_{iter\_count}(i,j)$ becomes greater than the height for the first time, then
         make corresponding $\mathbf{H}_{iter\_count}(i,j) = iter\_count$.
    5.8. Count $sum\_count_{iter\_count}$ as the sum of those entries in W matrix at present
         with value > 1.
    5.9. IF $\left| sum\_count_{iter\_count} - sum\_count_{(iter\_count-1)} \right| < 1$ THEN
           $no\_update\_iter\_count = no\_update\_iter\_count + 1$
         ENDIF
    ENDWHILE
6. All $\mathbf{H}(i, j)$ point still equal to zero are the obstacle points. Set these points to a
high value i.e. one more than their adjacent neighbor which one have the highest
value (steep gradient for obstacle occupied points).
7. Create shortest path coordinate vectors sh_path_co ord_row and
sh_path_co ord_col and initialize the first point:
sh_path_co ord_row $(1) = x\_goal$ ; sh_path_co ord_col $(1) = y\_goal$ .
Set $\mu$.
8. Compute gradient matrices of H matrix in $x$-direction ($\nabla\mathbf{H}_x$) and $y$- direction
($\nabla\mathbf{H}_y$).
9. $\nabla\mathbf{H}_x = -\nabla\mathbf{H}_x$; $\nabla\mathbf{H}_y = -\nabla\mathbf{H}_y$; $path\_index = 1$; $path\_flag = 1$;
10. WHILE ($path\_flag = 1$)
        10.1. Compute $del\_row$ by interpolation using the $\nabla\mathbf{H}_y$ matrix.
        10.2. Compute $del\_col$ by interpolation using the $\nabla\mathbf{H}_x$ matrix.
        10.3. Compute the coordinates of the next point on the shortest path:
        sh_path_coord_row$(path\_index + 1)$ = sh_path_coord_row$(path\_index)$ +
        $\mu * \frac{del\_row}{\sqrt{del\_row^2 + del\_col^2}}$
        sh_path_co ord_col $(path\_index + 1)$ = sh_path_co ord_col $(path\_index)$ +
        $\mu * \frac{del\_col}{\sqrt{del\_row^2 + del\_col^2}}$
        10.4. IF (initial point is reached) THEN
          $path\_flag = 0$;
          ENDIF
    ENDWHILE
11. Reverse vectors sh_path_coord_row and sh_path_coord_col.
END
```

**Algorithm 3.1.** The shortest path generation algorithm employing obstacle avoidance

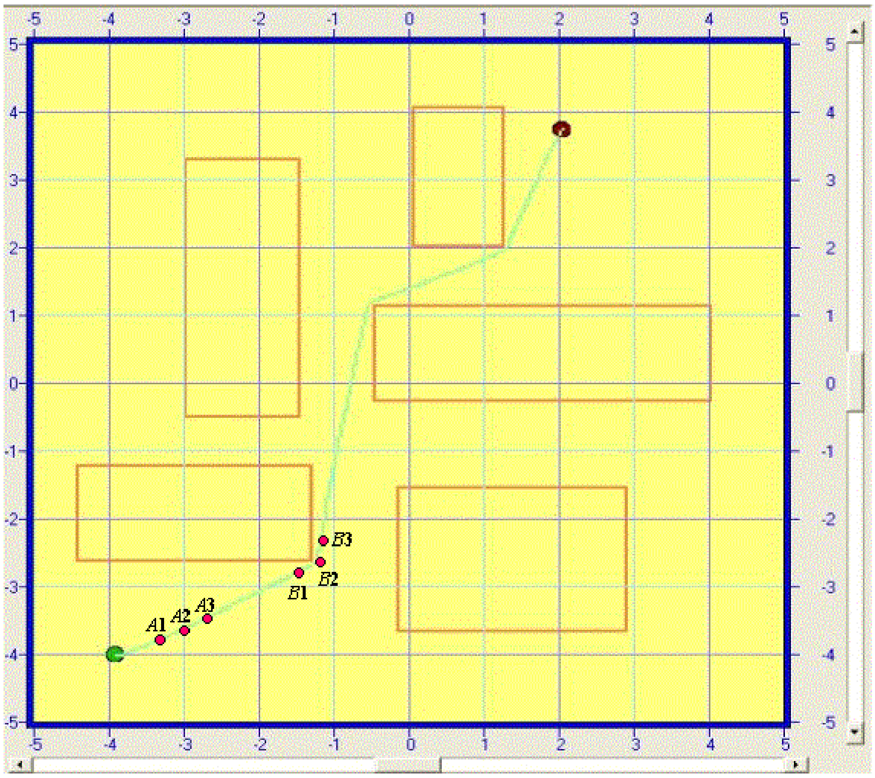
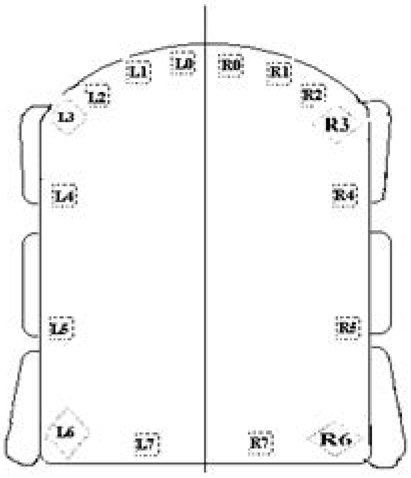


**Fig. 3.7.** A sample shortest path computed for an environment

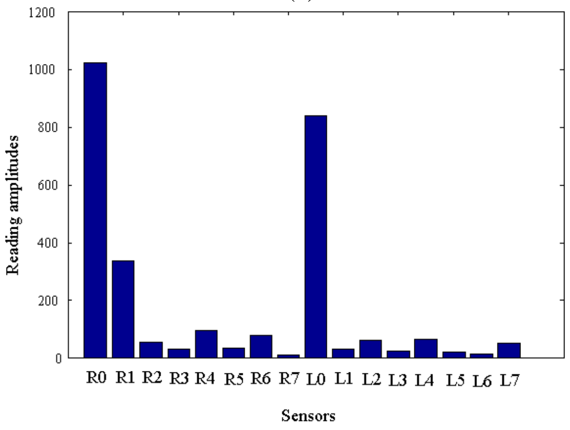
Once the shortest path is determined, we need to find the corner points nearer to an obstacle. To find the corner points, we take three consecutive points on the path and find the cosine of the angle between the two line segments joining the first two and last two points. If this value falls below a given threshold, then the middle of these three points is considered as a corner point, otherwise we move to the next subsequent point and again compute the cosine of the new angle. This process is continued until the suitable corner point is obtained. This corner point is stored as the next subgoal point for navigation. For example, in Fig. 3.7, when $A1$, $A2$, $A3$ are the three points under consideration, then the cosine of the angle between the line segments $\overline{A1A2}$ and $\overline{A2A3}$ is very high (above the chosen threshold). So $A2$ is not considered as a corner point. In this process, we keep moving forward, and when we reach the three consecutive points $B1$, $B2$, $B3$, the angle between the line segments $\overline{B1B2}$ and $\overline{B2B3}$ is large enough so that the cosine of the angle is below the chosen threshold. Then $B2$ is considered as a corner point.

## 3.6 IR Based Navigation in Layer 2

Once the subgoal point is determined, the control will be passed from layer 1 to layer 2. As soon as the new subgoal information is passed, the robot updates its present pose $(x_R, y_R, \phi_R)$, based on incremental wheel encoder information, and determines the new steering angle, based on its present pose and the subgoal information. Ideally this is the angle by which the robot should turn and proceed at a constant speed to reach the subgoal, in a static scenario. This is because the subgoal belongs to the set of points which were generated from the shortest path generation algorithm, employing obstacle avoidance. However, in a dynamic



(a)



(b)

**Fig. 3.8(a).** IR sensor arrangements of the KOALA robot [23]
**Fig. 3.8(b).** Measured values of the IR sensor readings, by placing a 1.5 cm wide obstacle in front of sensor (R0) at a distance of 10 cm.

scenario, after the last time the vision based mapping subroutine was activated, a new obstacle may have arrived or an old obstacle's position may have been changed. This may result in obstruction along the ideal path of travel between the robot and the subgoal. To cope with this dynamic environment, the navigation is guided by 16 IR sensors, mounted symmetrically along the periphery of the KOALA robot.

These IR sensors are densely populated in front and sparsely populated at the two sides of the robots. Figure 3.8(a) shows the sensor arrangement of the mobile robot and Fig. 3.8(b) shows a typical situation for the measured values of the sensors, by placing a 1.5 cm wide obstacle in front of the front sensor (R0) at a distance of 10 cm from the robot front face. For navigation, these 16 IR sensors scan the environment. Depending on these sensor readings, the system calculates the obstacle regions and free regions ahead of the robot. From these calculations the traversable area is determined. For determining the traversable area, separate thresholds are set for each of the 16 sensors, with the maximum priority given to the front sensors (R0 – R3, L0 – L3). For each sensor, if its reading exceeds its threshold, it means the direction ahead of it is obstructed, else the direction ahead is considered free for traversal. Now, depending on these readings, there can be traversable areas both to the left and to the right of the present pose of the robot. The decision of whether the robot should turn left or right is taken based on which direction will mean that the robot has to undertake the shorter detour with respect to its ideal direction of travel. Once the detour direction is determined, the speed of the robot is determined based on the IR sensor readings in that direction. When the robot travels a predetermined distance, the entire IR based scanning and determination of the new detour direction of traversal is reactivated and this procedure is continued until the robot reaches the subgoal or its closest vicinity. Then the robot stops and the control is transferred back to layer 1.

## 3.7 Real-Life Performance Evaluation

The performance evaluation has been carried out, for vision based navigation, in our laboratory, utilizing several environments. Here we present the results for four such experiments, two each in static and dynamic environments.

*Case Study – I*

The initial pose of the robot is (0, 0, 0) and the goal point is (2, 0). There lies an object between the robot and the goal position. It should be mentioned here that for the robot system which is equipped with a pan-tilt mechanism with its corresponding degrees of freedom, in this work, the pan angle and the tilt angle are suitably initialized for a particular environment and then they are kept fixed, for all subsequent experiments. Initially these two angles are so chosen for the robot system developed so that the monocular camera, in each frame, covers a reasonably large floor and environment area. The system is hence equipped with the flexibility where these angles can be suitably initialized depending on the environment where this navigation system is going to be implemented. Figure 3.9(a) shows the image frame acquired from the video stream of the camera

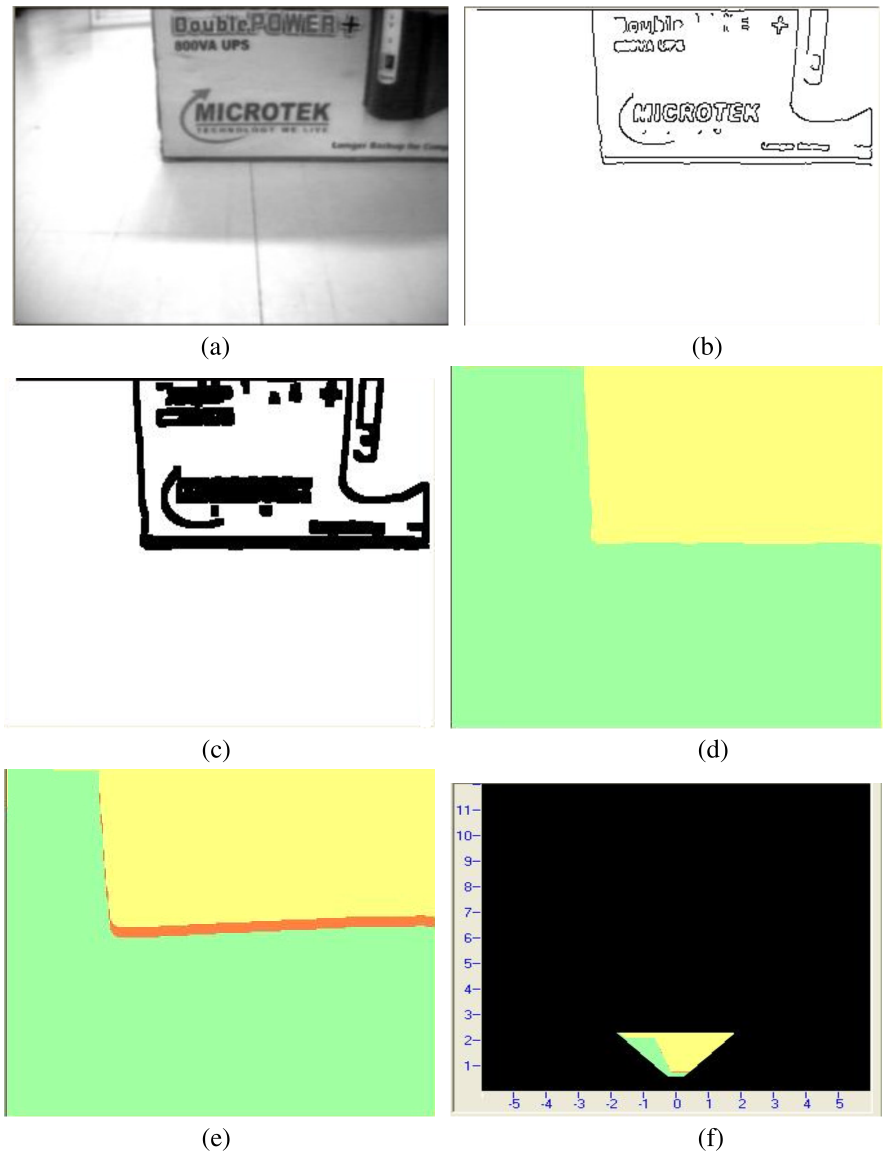


**Fig. 3.9.** (a): The image acquired, (b)-(f): sequence of image processing caried out in layer 1. (b): edge image; (c): thickened edge image; (d): region grown image; (e): image with free, obstacle, and hidden regions and (f): trapezoidal floor image.

and Figs. 3.9(b)–3.9(f) show the sequence of image processing steps, when the robot is in initial position. The edge of the face of the obstacle on the ground, viewed by the robot in front of it when the robot is at its initial pose, actually extends from (0.9, 0.3) to (0.9, -0.85). Figure 3.10 shows the snapshot of the grid

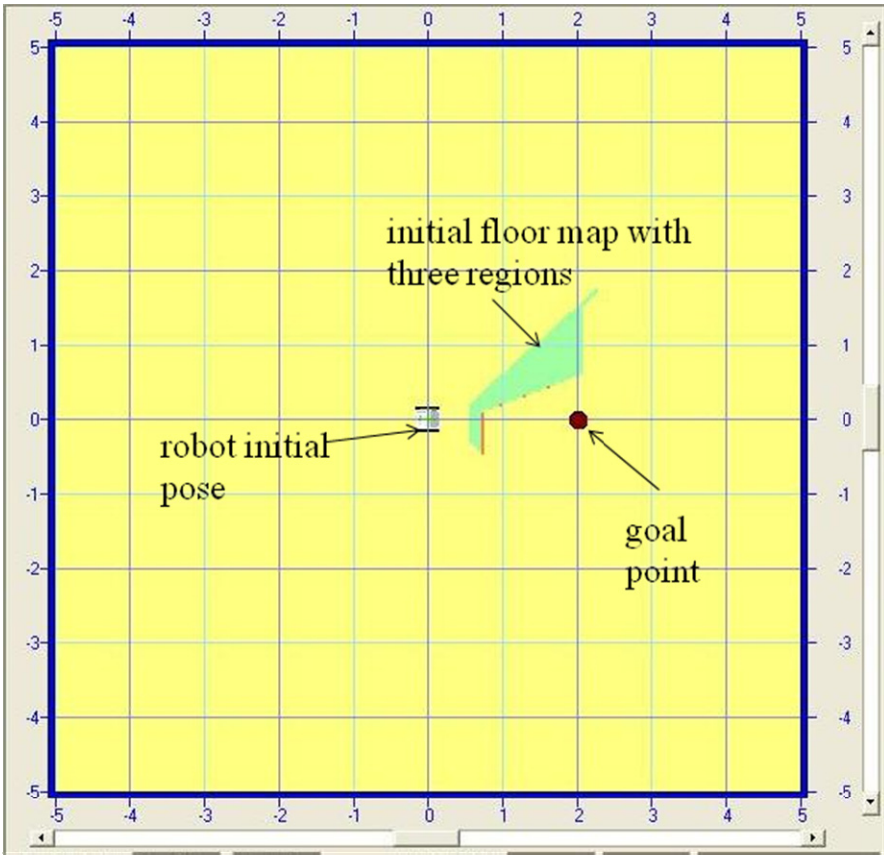


**Fig. 3.10.** The initial grid map

map with the obstacle information and free region. Here the shortest path is calculated and the layer 2 of the robot navigation algorithm is updated with the subgoal information. The algorithm calculates the subgoal 1 as (0.81, 0.33). When the control is transferred to layer 2, the robot navigates using IR sensor based guidance, upto subgoal 1. The robot actually stops at (0.819, 0.332) which has very small discrepancy with the calculated subgoal. Figure 3.11 shows the snapshot of the grid map when the robot reaches the first subgoal point. This grid map is developed when the control is transferred back to layer 1 and vision based processing is carried out once more. Figures 3.12(a)-3.12(d) show the results of image processing steps, when the robot is at the first subgoal point. These results are used for IR based navigation once more. This sequential process is continued to reach the final goal point. Figure 3.13 shows the grid map when the robot reached the destination. The robot finally stops at (1.963, 0.024) which is extremely close to the specified goal (2,0). Figure 3.14 shows the complete navigation path traversed by the robot, starting from the initial point and reaching the goal point, in presence of the obstacle, following the shortest possible path. Figure 3.15(a) shows the response of IR sensors on the right side of the robot (R0, R3) during navigation and fig. 15(b) shows the corresponding responses for the IR sensors on the left side of the robot (L0, L3). It can be seen that the reading of the R3 sensor reaches a high value when the robot is in the vicinity of the obstacle. As the robot crosses the obstacle and proceeds towards the goal point, the reading of the R3 sensor gradually decreases.

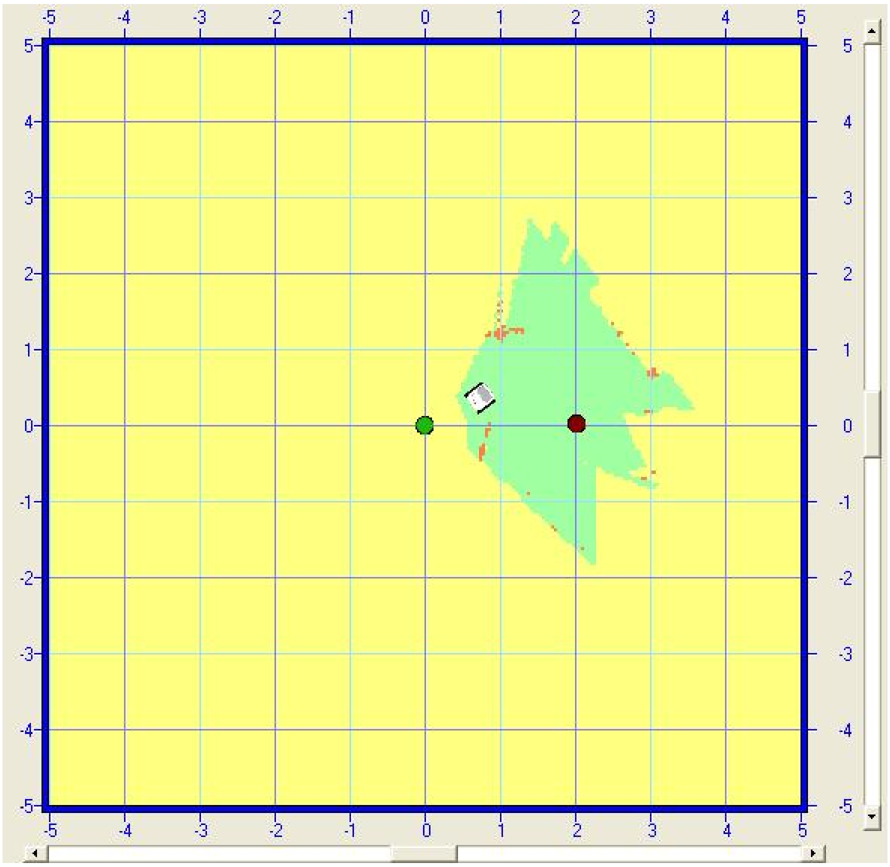


**Fig. 3.11.** The grid map, when the robot reaches the first subgoal

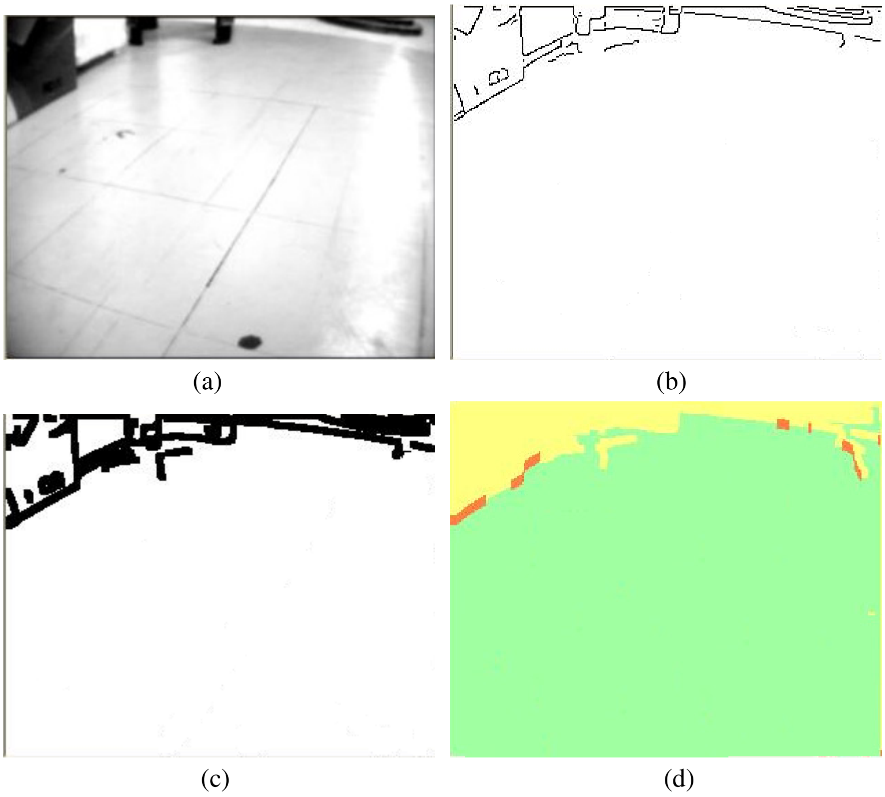


**Fig. 3.12.** (a)-(d). Results of image processing at subgoal 1. (a): the acquired image; (b): edge image; (c): thickened edge image, and (d): image with three distinct regions.

*Case Study – I1*

Here again the initial pose of the robot is (0, 0, 0) and the new goal point is (3, 0). Now two objects are introduced between the robot and the goal position. Figure 3.16(a) shows the image frame acquired at the initial position of the robot and Figs. 3.16(b)–3.16(f) show the results of subsequent image processing steps in layer 1. Figure 3.17 shows the snapshot of the initial grid map with the obstacle information and free region. Next the shortest path is calculated and the layer 2 of the robot navigation algorithm is implemented with this subgoal information. Figure 3.18 shows the snapshot of the grid map when the robot reaches the first subgoal point. When the robot reaches subgoal 1, the control is transferred back to layer 1. The system again performs the vision based processing, as shown in Fig. 3.19 and Fig. 3.20 shows the grid map when the robot reaches subgoal 2, using IR

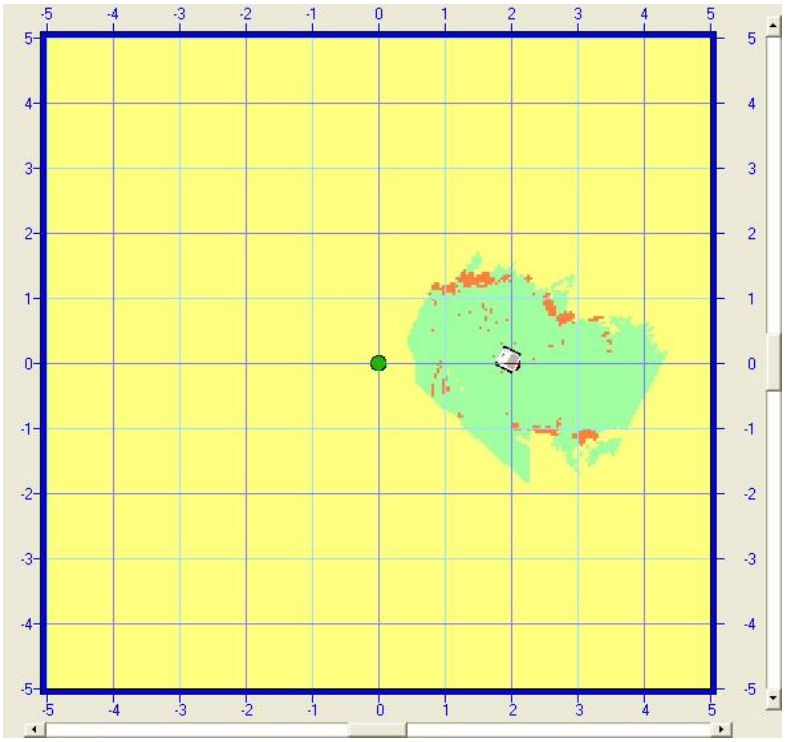


**Fig. 3.13.** The grid map, when the robot reaches the final goal point

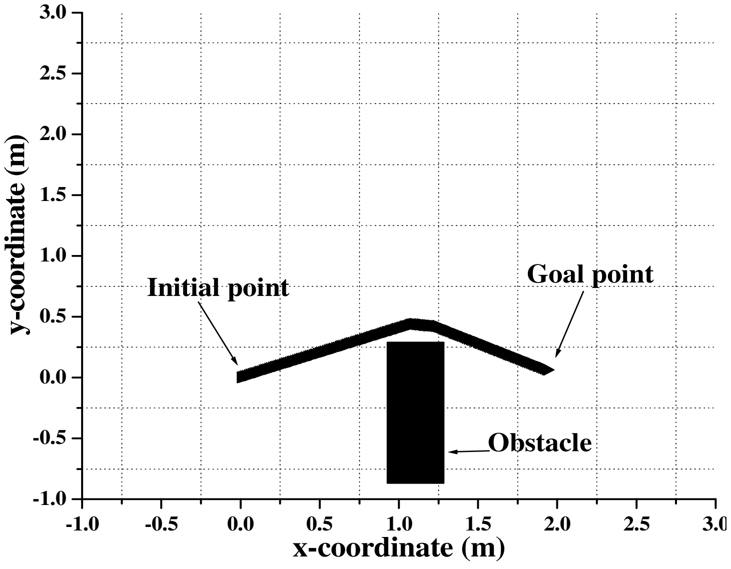
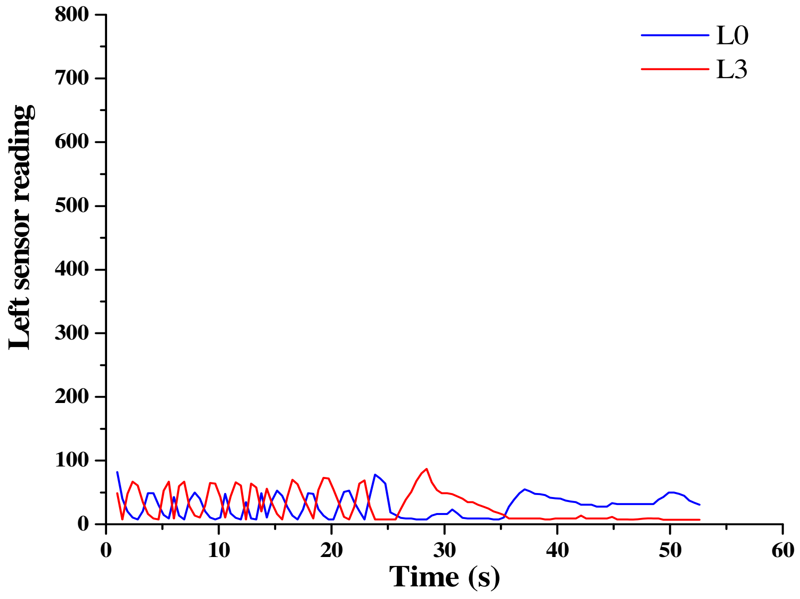
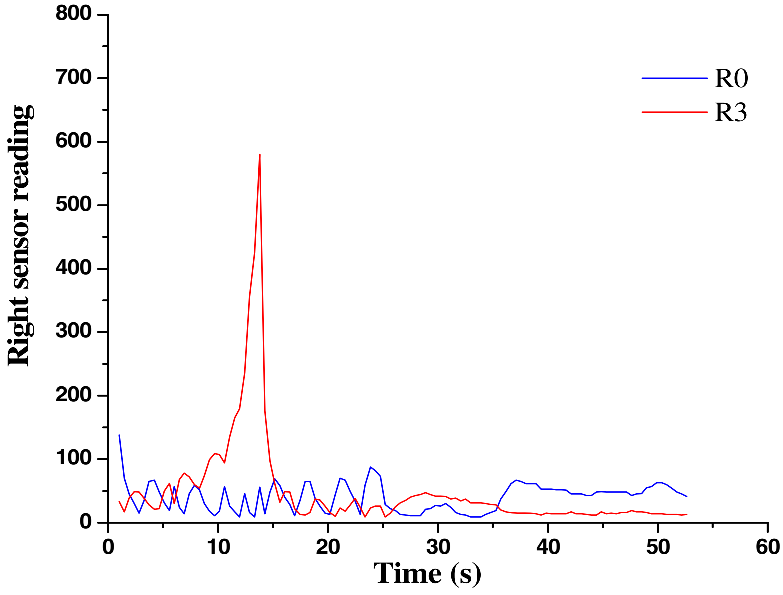


**Fig. 3.14.** The robot navigation path traversed



(a)



(b)

**Fig. 3.15.** Variation of (a) response of L0 & L3 IR sensors with time and (b) response of R0 & R3 IR sensors with time, for case study I

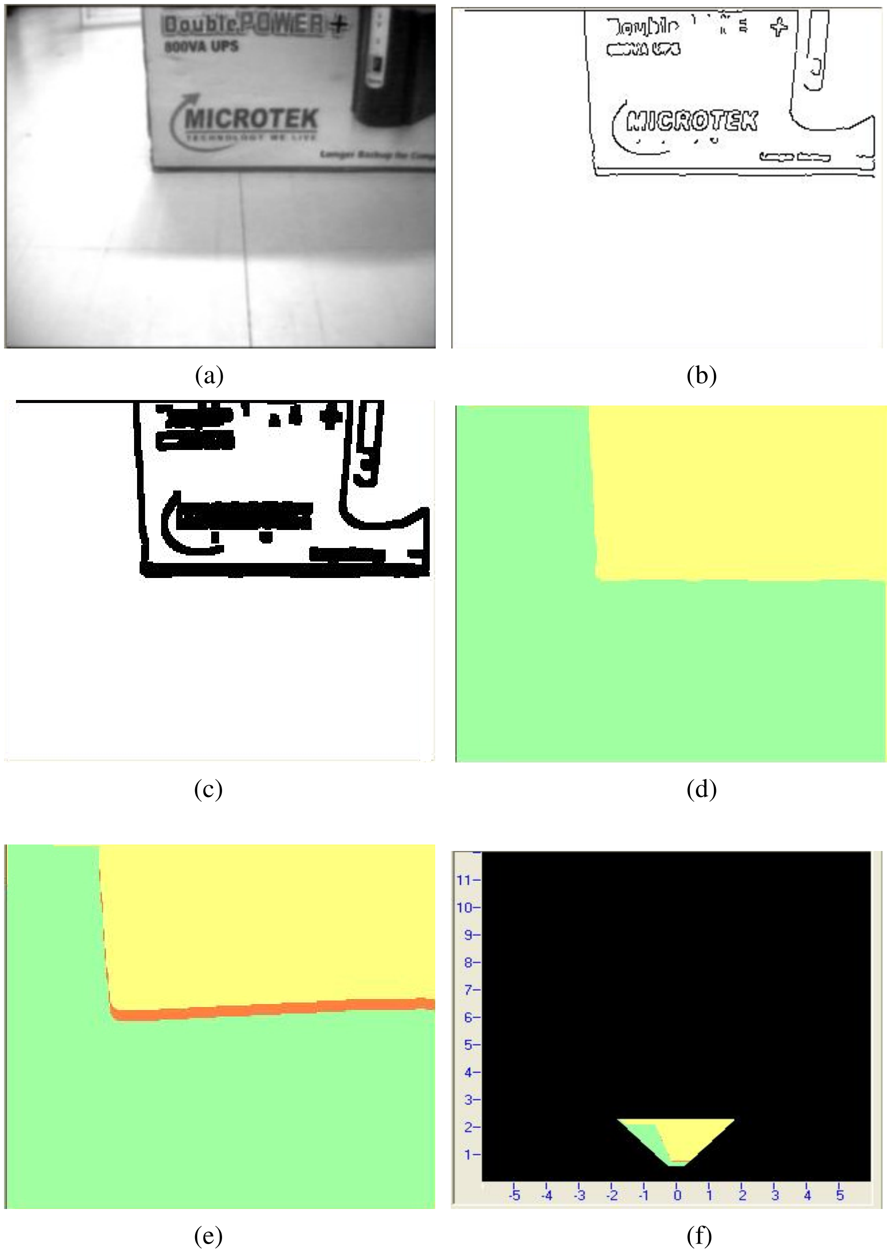


**Fig. 3.16.** (a): The image acquired, (b)-(f): results of image processing steps in layer 1. (b): edge image; (c): thickened edge image; (d): region grown image; (e): image with free, obstacle and hidden regions and (f): trapezoidal floor image.

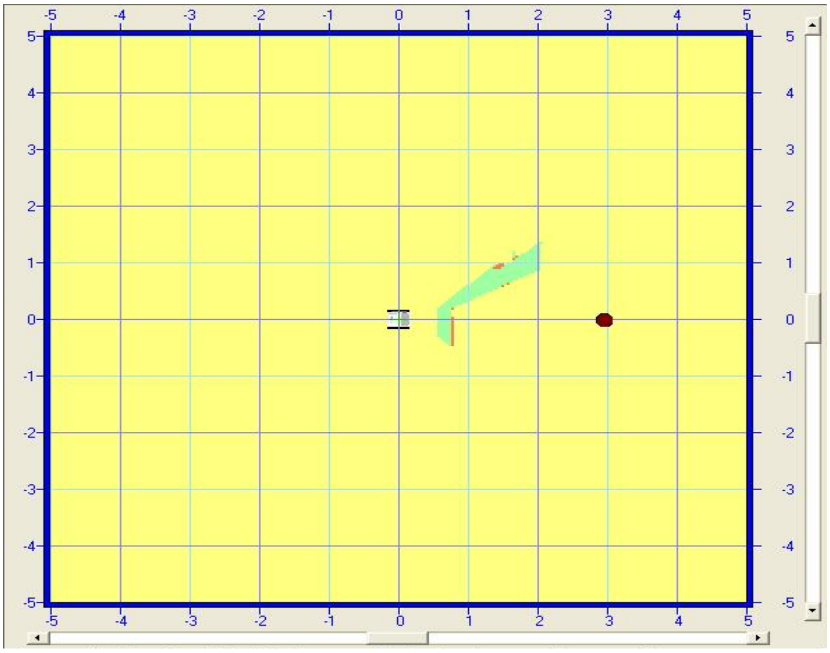


**Fig. 3.17.** The initial grid map

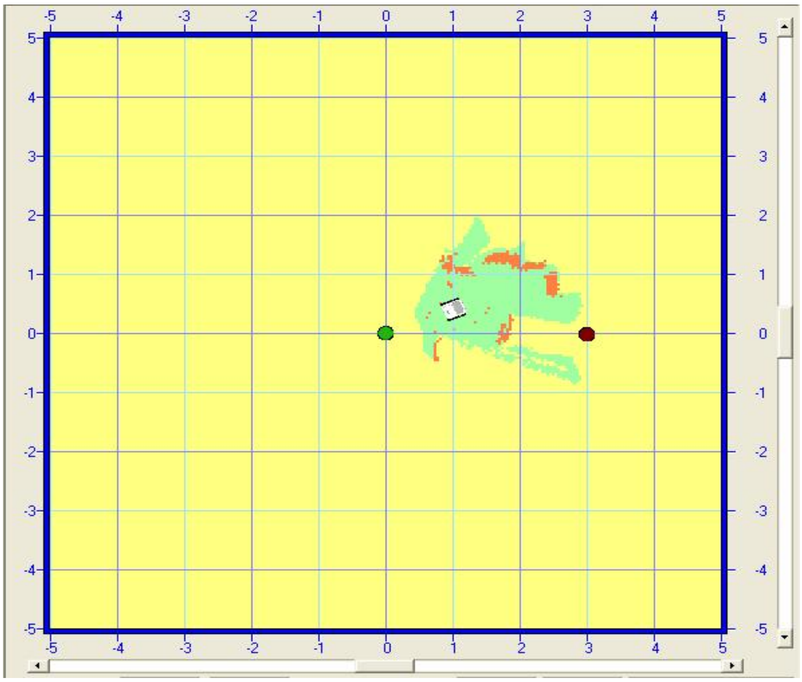


**Fig. 3.18.** The grid map, when the robot reaches the first subgoal

based navigation in layer 2. This iterative process is continued until the robot reaches the final goal. Figure 3.21 shows the grid map when the robot reaches the final goal point. Figure 3.22 shows the complete path of traversal of the robot for this static environment and shows that the robot reaches the goal satisfactorily. Figure 3.23 shows the variations of four IR sensors, R0, R3, L0, and L3, readings when the robot navigates towards its destination.

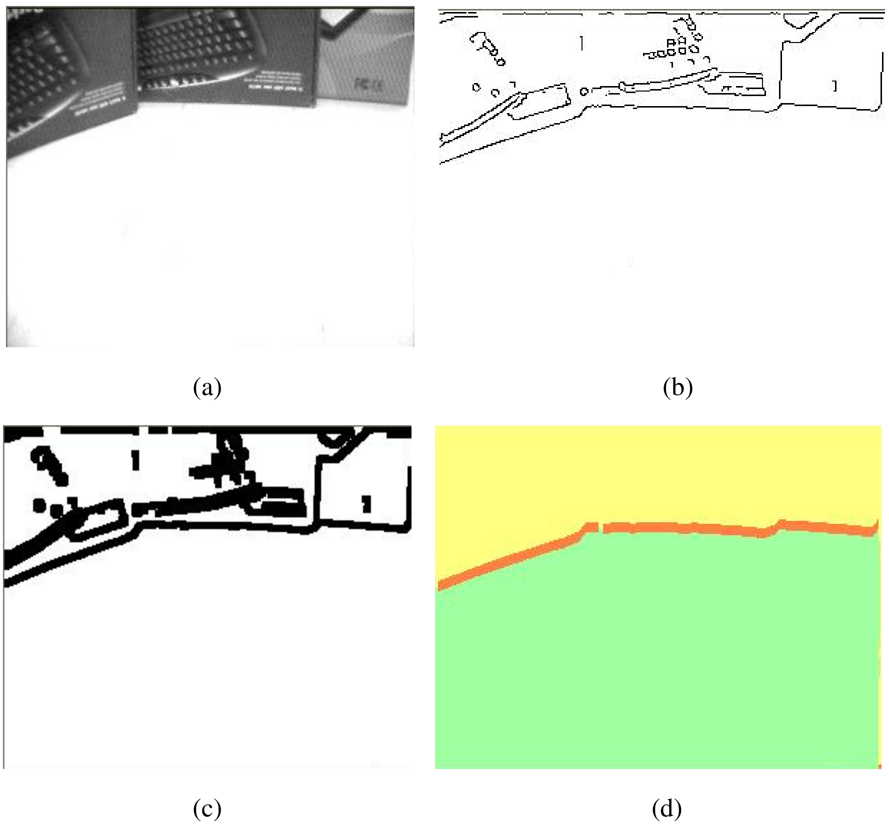


**Fig. 3.19 (a)-(d).** Results of image processing at subgoal 1. (a): the captured image; (b): edge image; (c): thickened edge image, and (d): image with three distinct regions.

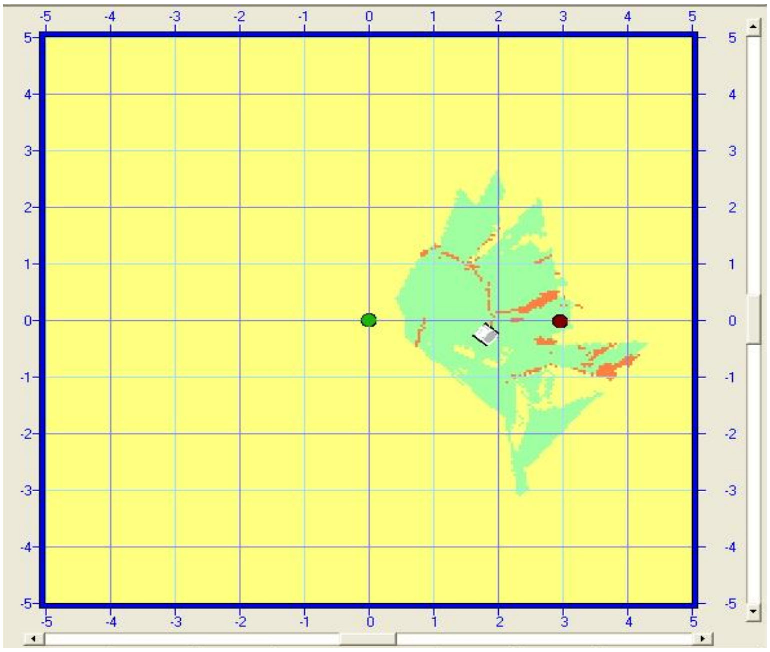


**Fig. 3.20.** The grid map, when the robot reaches the second subgoal

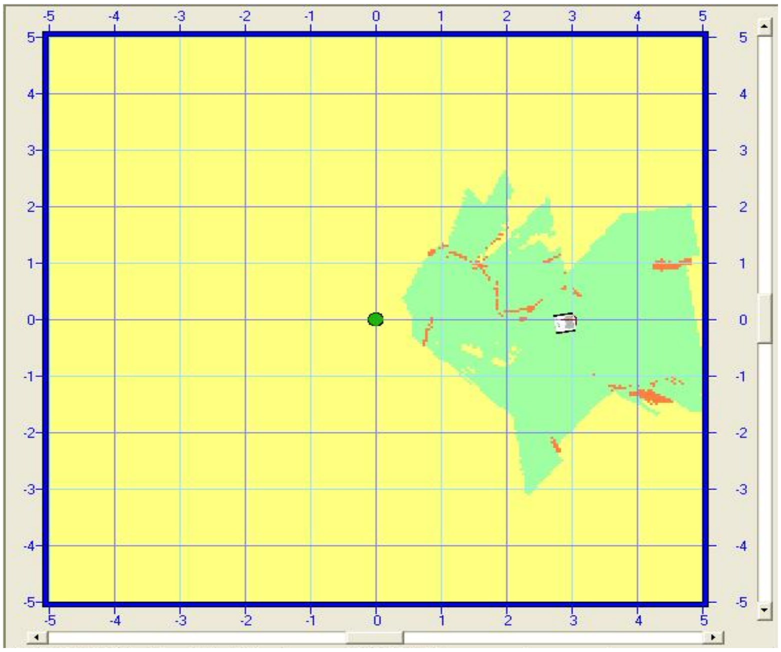


**Fig. 3.21.** The grid map, when the robot reaches the final goal point

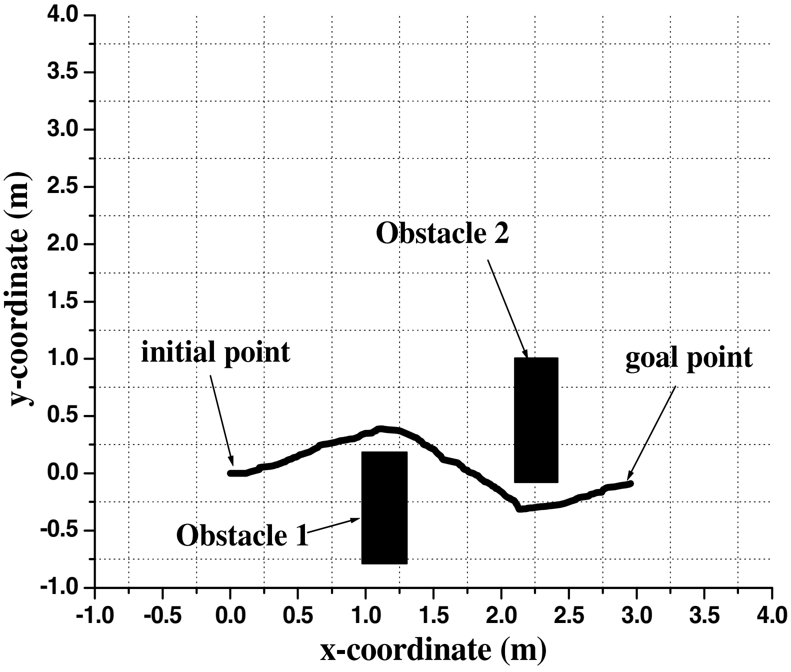
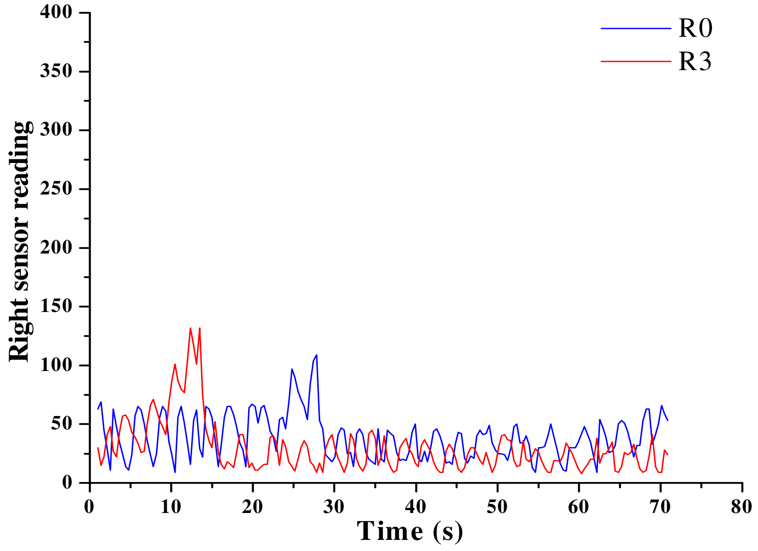


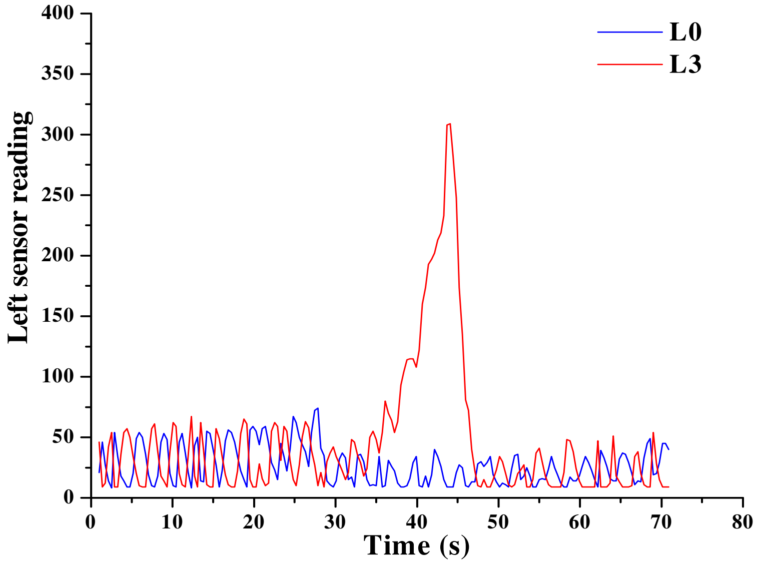
**Fig. 3.22.** The robot navigation path traversed, in case study I

*Case Study – III*

In the next two case studies, we demonstrate the utility of the proposed system in case of a dynamically changing environment. Here, for an environment similar to that considered in case study I, the robot starts from an initial pose (0, 0, 0), with a bid to reach the goal point (2,0), in presence of an obstacle between the robot and the goal position. However, after the robot starts its IR based navigation towards subgoal 1, determined using vision based image processing in layer 1 at the initial position of the robot, followed by the determination of the subgoal 1 utilizing the shortest path algorithm, the position of obstacle 1 is shifted. The new position of the obstacle is now shown in Fig. 3.24 where it is moved nearer to the robot and it is shifted towards the left of the robot, with reference to its initial pose. Because of this dynamic variation in the environment, the robot takes a detour towards its left but was still able to avoid the obstacle and reach its subgoal. The subsequent activations of the iterative algorithm show that the robot reaches its final goal almost perfectly, once more. Figure 3.24 shows this navigation of the robot in the dynamic environment. Figure 3.25(a) and Fig. 3.25(b) show the IR sensor readings, in front of the robot. It can be seen that the reading of R0 and L0 receive a sudden kick when the obstacle is moved in the dynamic environment.



(a)



(b)

**Fig. 3.23.** Variation of (a) response of R0 and R3 IR sensors and (b) response of L0 and L3 IR sensors during navigation, for case study II

Here it should be mentioned that if there arises an exceptional situation where the dynamically changing object arrives exactly on a subgoal, then, according to the algorithm, the IR-sensor based actual navigation guidance mechanism will ensure that the robot will stop at the shortest distance from the subgoal, satisfying obstacle avoidance or collision requirement.

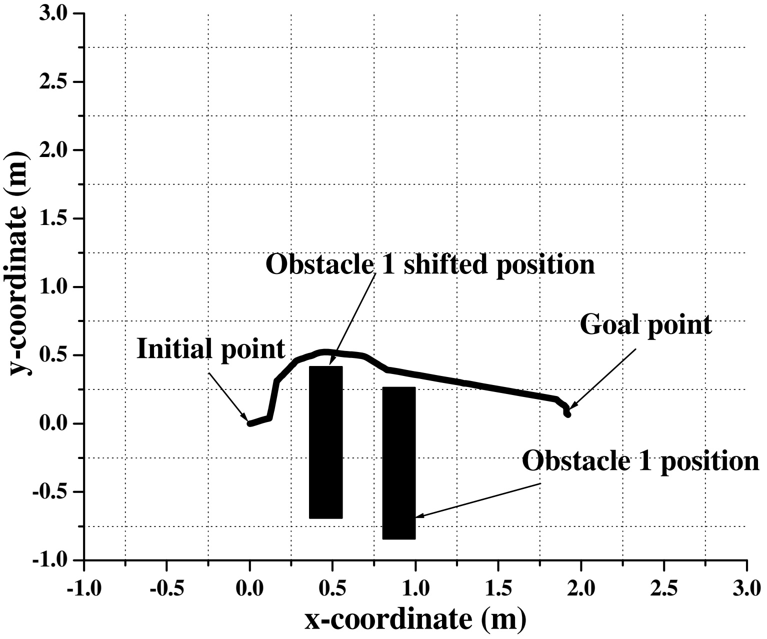
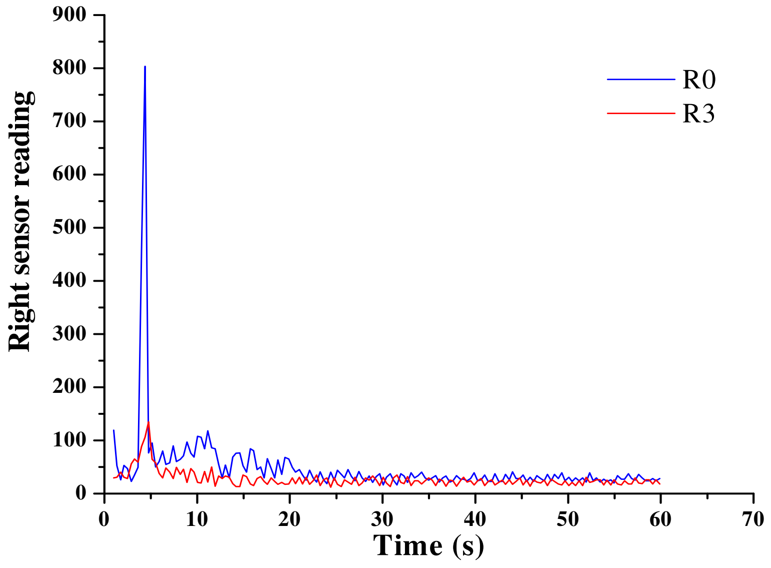


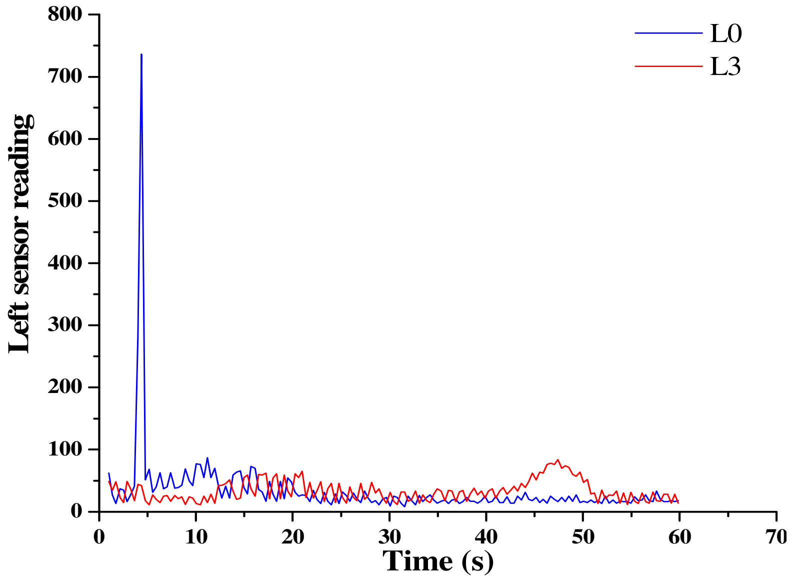
**Fig. 3.24.** The robot navigation path traversed in a dynamic environment

*Case Study – IV*

This situation is similar to case study II, but with both obstacles being made dynamic in nature. Here also, after the robot start traversing towards subgoal 1, avoiding obstacle 1 whose position was determined from the vision based image processing in layer 1, the position of the obstacle 1 was suddenly changed. It was brought closer to the robot and more towards its left, making partial dynamic blockage of the free region of traversal. Similarly, when the robot was attempting to traverse a shortest path avoiding obstacle 2, suddenly the position of the obstacle 2 was changed by bringing it closer to the robot. However the robot was able to undertake the required detour in its IR based navigation in each such situation and was able to reach the final goal satisfactorily, as shown in Fig. 3.26.



(a)



(b)

**Fig. 3.25.** Variation of (a) response of R0 and R3 IR sensors and (b) response of L0 and L3 IR sensors during navigation in the dynamic environment, for case study III

Figure 3.27(a) and 3.27(b) show the readings of the IR sensors R0, R3, L0, and L3. It can be seen that here also the readings of L0 and R0 receive two sudden kicks, when the two obstacle positions are changed.

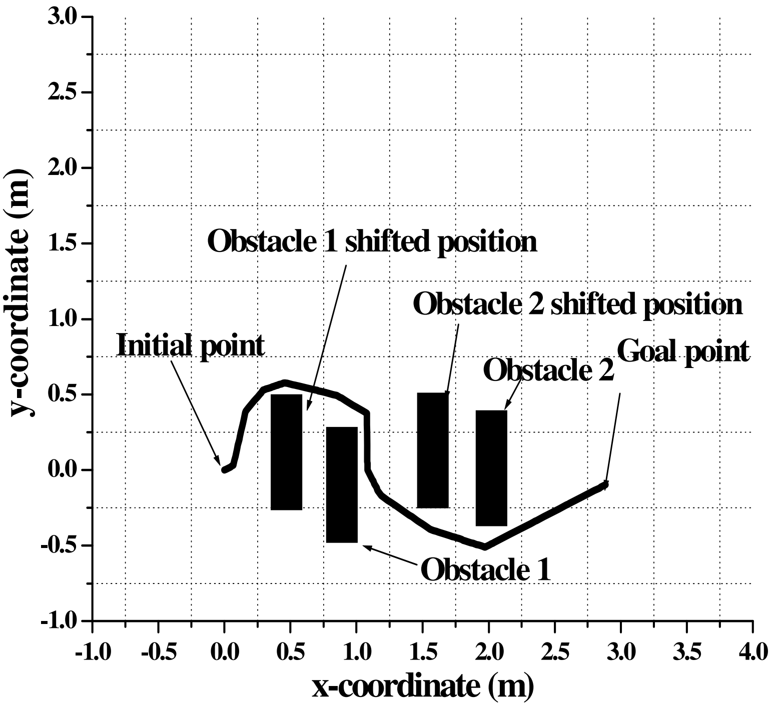
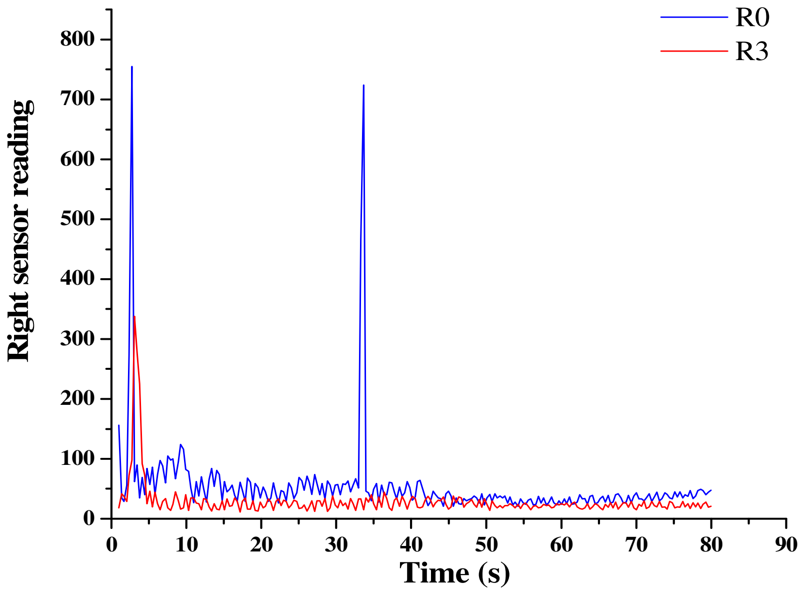
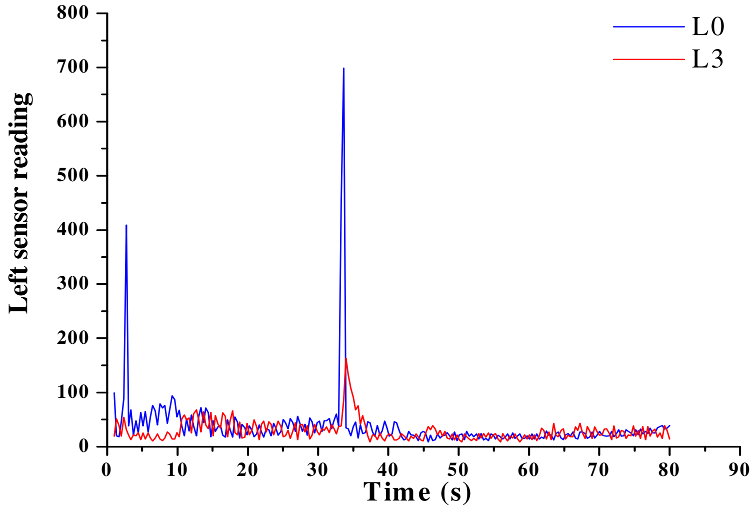


**Fig. 3.26.** The robot navigation path traversed in a dynamic environment



(a)



(b)

**Fig. 3.27.** Variation of (a) response of R0 and R3 IR sensors and (b) response of L0 and L3 IR sensors during navigation in the dynamic environment, for case study IV

## 3.8 Summary

In this chapter we described how a two-layered, goal oriented, vision based robot navigation scheme can be developed. The system employs vision based analysis of the environment in layer 1, which employs several image processing functions and a shortest path generation algorithm, to determine the next subgoal for navigation, with the objective of reaching the final destination as fast as possible, avoiding obstacles. This subgoal information is utilized by the robot in layer 2 to navigate in dynamic environments, utilizing a set of IR sensors, avoiding obstacles, to reach the subgoal or its closest vicinity. This two-layered algorithm is utilized iteratively to create the next subgoal and navigate upto it, so that the final goal is reached sufficiently quickly. This chapter has showed a successful implementation of how to hybridize the shortest path algorithm with camera based image processing to enhance the quality of vision based navigation of mobile robots in the real world, so that, the robot can reach its goal (known *a priori*), following the shortest practical path, avoiding obstacles. The robustness of the system is further ensured by the IR-sensor guided navigation, which helps the robot to adapt its navigation, based on any possible change in obstacle positions in a dynamic environment. This algorithm is implemented for several environments created for indoor navigation in our laboratory. It has been demonstrated that the KOALA robot could achieve its task, each time, satisfactorily, for both static environments and dynamic environments.

The developed programs comprise high-end programs developed in VB platform which communicate, in real-time, with the processor of the robot system, where cross-compiled versions of custom-designed C programs are downloaded. However, in the real implementation phase, the entire system is run from the high-end VB platform in a PC through a user-friendly GUI developed, so that it can be easily utilized by some common users.

For high illumination situations the algorithm is expected and has been demonstrated to provide satisfactory performance. However, for low illumination situations, the reflections of the obstacles on the floor may look dark enough (as is shown in the case of Figs. 3.4(a)-3.4(e)) so that the edge image may contain some edges corresponding to reflections on the floor. Hence these reflections may be interpreted as obstacle and this reduces the free zone computed. However, according to the algorithm, in these exceptional cases, the shortest path computed may be a little longer than the true shortest path but still safe and robust navigation of the robot avoiding obstacles towards the goal will be ensured.

The present system is developed where the robot pose in real environment is estimated by odometry using only incremental wheel encoder information. This suffices well for indoor applications with uniform floors, for which the system is primarily developed. The experiments conducted sufficiently demonstrate that the robot reaches the goal in real world, under these conditions, for a variety of environmental configurations. However, the accuracy of this system may suffer in outdoor environments due to problems like wheel slippage etc. One can undertake such future works into consideration which will attempt to adapt this system for outdoor environments too and this may be accomplished by additionally integrating e.g. extended Kalman filter based algorithms for robot localization, along with the current system developed.

**Acknowledgement.** The work described in this chapter was supported by University Grants Commission, India under Major Research Project Scheme (Grant No. 32-118/2006(SR)).

## References

[1] Chen, Z., Birchfield, S.T.: Qualitative vision-based mobile robot navigation. In: Proc. of the IEEE International Conference on Robotics and Automation (ICRA), Orlando, Florida (May 2006)

[2] Bertozzi, M., Broggi, A., Fascioli, A.: Vision-based intelligent vehicles: state of the art and perspectives. Robotics and Autonomous Systems 32, 1–16 (2000)

[3] Shin, D.H., Singh, S.: Path generation for robot vehicles using composite clothoid segments. The Robotics Institute, Internal Report CMU-RI-TR-90-31, Carnegie-Mellon University (1990)

[4] DeSouza, G.N., Kak, A.C.: Vision for mobile robot navigation: A Survey. IEEE Transactions on Pattern Analysis and Machine Intelligence 24(2), 237–267 (2002)

[5] Moravec, H.P.: Sensor fusion in certainty grids for mobile robots. Artificial Inteligence Magazine 9, 61–74 (1988)

[6] Elfes, A.: Sonar-based real-world mapping and navigation. IEEE Journal of Robotics and Automation 3(6), 249–265 (1987)

[7] Jensfelt, P., Christensen, H.I.: Pose tracking using laser scanning and minimalistic environmental models. IEEE Transactions on Robotics and Automation 17, 138–147 (2001)

[8] Yamauchi, B., Beer, R.: Spatial Learning for navigation in dynamic environments. IEEE Transactions on System, Man, and Cybernetics, Part B 26(3), 634–648 (1995)

[9] Pierce, D., Kuipers, B.: Learning to explore and build maps. In: Proc. of the Twelfth National Conference on Artificial Intelligence, Menlo Park, pp. 1264–1271. AAAI, AAAI Press/MIT Press (July 1994)

[10] Thrun, S.: Learning metric-topological maps for indoor mobile robot navigation. Artificial Intelligence 99, 21–71 (1998)

[11] Santos-Victor, J., Sandini, G., Curotto, F., Garibaldi, S.: Divergent stereo for robot navigation: Learning from Bees. In: Proc. IEEE Computer Society Conference on Computer Vision and Pattern Recognition, New York, pp. 434–439 (June 1993)

[12] Kim, D., Nevatia, R.: Recognition and localization of generic objects for indoor navigation using functionality. Image and Vision Computing 16(11), 729–743 (1998)

[13] Murray, D., Little, J.J.: Using real-time stereo vision for mobile robot navigation. Autonomous Robots 8, 161–171 (2000)

[14] Davison, A.J.: Mobile robot navigation using active vision. PhD thesis (1998)

[15] Ayache, N., Faugeras, O.D.: Maintaining representations of the environment of a mobile robot. IEEE Transactions on Robotics and Automation 5(6), 804–819 (1989)

[16] Fialaa, M., Basub, A.: Robot navigation using panoramic tracking. Pattern Recognition 37, 2195–2215 (2004)

[17] Gasper, J., Santos- Victor, J.: Vision-based navigation and environmental representations with an omnidirectional camera. IEEE Transactions on Robotics and Automation 16(6), 890–898 (2000)

[18] Ohya, A., Kosaka, A., Kak, A.: Vision-based navigation by a mobile robot with obstacle avoidance using single-camera vision and ultrasonic sensing. IEEE Transactions on Robotics and Automation 14(6), 969–978 (1998)

[19] Li, M.H., Hong, B.H., Cai, Z.S., Piao, S.H., Huang, Q.C.: Novel indoor mobile robot navigation using monocular vision. Engineering Applications of Artificial Intelligence, 1–18 (2007)

[20] Latombe, J.: Robot Motion Planning. Kulwer, Norwell (1991)

[21] Dijkstra, E.W.: A note on two problems in connection with graphs. Numerische Mathematik 1, 269–271 (1959)

[22] Nilsson, N.J.: Principles of Artificial Intelligence. Tioga Publishing Company (1980)

[23] KOALA User Manual, Version 2.0(silver edition), K-team S.A., Switzerland (2001)

[24] Singh, N.N., Chatterjee, A., Rakshit, A.: A PIC microcontroller-based system for real-life interfacing of external peripherals with a mobile robot. International Journal of Electronics 97(2), 139–161 (2010)

[25] Kim, P.G., Park, C.G., Jong, Y.H., Yun, J.H., Mo, E.J., Kim, C.S., Jie, M.S., Hwang, S.C., Lee, K.W.: Obstacle Avoidance of a Mobile Robot Using Vision System and Ultrasonic Sensor. In: Huang, D.-S., Heutte, L., Loog, M. (eds.) ICIC 2007. LNCS, vol. 4681, pp. 545–553. Springer, Heidelberg (2007), doi:10.1007/978-3-540-74171-8.

[26] `http://www.mathworks.com/matlabcentral/fileexchange/8625-shortest-path-with-obstacle-avoidance-ver-1-3`

[27] `http://atalasoft-imgx-controls-sdk.software.informer.com/`

[28] Nirmal Singh, N., Chatterjee, A., Chatterjee, A., Rakshit, A.: A two-layered subgoal based mobile robot navigation algorithm with vision system and IR sensors. Measurement 44(4), 620–641 (2011)

[29] Nirmal Singh, N.: Vision Based Autonomous Navigation of Mobile Robots. Ph.D. Thesis, Jadavpur University, Kolkata, India (2010)