J Intell Robot Syst (2009) 56:319–343
DOI 10.1007/s10846-009-9319-9

Autonomous Vehicle Parking Using Hybrid Artificial Intelligent Approach

Chen-Kui Lee · Chun-Liang Lin · Bing-Min Shiu

Received: 20 September 2008 / Accepted: 11 February 2009 / Published online: 6 March 2009

Abstract This paper devotes to design and implement a hybrid artificial intelligent control scheme for a car-like vehicle to perform the task of optimal parking. The parallel parking control scheme addresses three issues: trajectory planner, decisional kernel, and trajectory tracking control. Design of the control scheme consists of several techniques: genetic algorithm, Petri net, and fuzzy logic control. The genetic algorithm is used to determine the feasible parking locations. The Petri net is used to replace the traditional decision flow chart and plan alternative parking routes especially in global space. The parking routine can be re-performed if the initially assigned route is interfered or when the targeted parking space has been occupied. The fuzzy logic controller is used to drive the vehicle along with the optimal parking route. The proposed scheme is put into several scenarios to test and verify its applicability and to manifest its distinguished features.

Keywords Optimization · Vehicle parking · Genetic algorithm · Petri net · Fuzzy control

1 Introduction

The study of motion autonomy for various vehicles has been a focus of interest for researchers. The state of art regarding this topic involves the application of diversified methods combined with the traditional and hierarchical schemes. Parking a car is usually a problem of great complexity to beginners. Not to mention only

C.-K. Lee · C.-L. Lin (✉) · B.-M. Shiu
Department of Electrical Engineering, National Chung Hsing University,
Taichung, Taiwan 402, R.O.C.
e-mail: chunlin@dragon.nchu.edu.tw

B.-M. Shiu
e-mail: 49264032@ee.nchu.edu.tw

beginners, parking a car in a crowded parking lot is also difficult to experienced drivers. Therefore, how to develop automated parking strategies for the vehicle-like mechanism has attracted researchers in recent decades.

Nonholonomic systems include mobile robots, automobiles, tractor-trailers, orbiting satellites, space-based robot manipulators, and many more. Generally, nonholonomic conditions arise in wheeled vehicles under the no-slipping constraint and two rigid bodies in rolling contact. A classical approach for solving the motion problem is to apply Dubins' curves [1] or Reed and Sheep's curves [2]. Lafferiere and Sussmann [3] presented the first general plan for vehicles based on a constructive proof of controllability, whereas Murry and Sastry [4] approached the problem by steering a nonholonomic system by means of sinusoid paths between arbitrary points. However, neither one of the papers mentioned above addressed the issue of obstacle avoidance. Paromtchik and Laugier [5] presented an approach where a vehicle follows a sinusoidal path in backward motion.

Recently, intelligent control methods are used more extensively, such as fuzzy logic controller, neural network controller, and genetic algorithms (GAs). Yasunobu and Murai [6] proposed a human experience based on fuzzy logic control theory. Jenkin and Yuhas [7] introduced a simplified neural network controller trained on the basis of kinematics data. The methods have also been extended to the applications in autonomous mobile robot control, and yielded much success [8–16]. However, recently published papers considering the issue of car parking only deals with the situation of parallel parking in a single parking space (i.e. the local car parking planner).

Generally, parallel parking involves three distinct problems—recognition of driving circumstance, maneuvering path planning, and vehicle control. Some of the path planning technologies for mobile robot systems can be extended to parking applications. In particular, Petri nets based on discrete event formalisms have been investigated as potential means of achieving collision avoidance, task-preserving human-intervention, and route control. Hwang et al. [17] have proposed that Petri nets are well-suited to the modeling of concurrent systems and there are many analytic techniques [18–21] which allow Petri nets to be verified for the occurrence of potentially undesirable states.

Furthermore, while there are preliminarily experimental park assist systems developed by car manufacturers, the design can only park the car to the nearest parking space.

This paper presents an automatic parallel parking planning system for a car-like vehicle, with the intention of applying the research results of robot motion planning into real-world applications. Unlike traditional design methods, the proposed approach focuses primarily on the path planning in a global parking space and it combines with Petri nets and GAs to determine optimal parking paths. By supposing the information of the parking environment is available, the information implemented in Petri nets is used to recognize suitable parking regions called "Ready-for-parking spaces". The path planner determines a collision-free path, which satisfies the nonholonomic constraints of the vehicle. The path produced is optimized by using GAs that considers three key variables: total distance, number of gear change, and change of the orientation between the front wheels and the main direction of the vehicle. After the previously depicted procedure is completed, a fuzzy logic controller based on the knowledge of experienced drivers is designed and utilized to accomplish the tracking control. Also, the GAs are used here to adjust and optimize the parameters of the membership functions.

In summary, the overall structure solves the problem of parallel parking to the single parking space and extends the result to the situation of multiple parking spaces. The presented strategy can be used in parking aid devices, and it is believed to possess the potential for being integrated into future automobiles.

2 Architecture of Parking System

2.1 Problem Description

Automatic parking is defined as an autonomous car maneuvering from a traffic lane into a parking space to perform parallel, perpendicular or angle parking. The automatic parking aims to reduce parking time and enhance safety of driving in a restrained environment where much attention and experience is required to steer the car. The parking maneuver is achieved by the means of coordinated control of the steering angle and speed, which ensures collision-free motion within available spaces by taking actual situation of the environment into account.

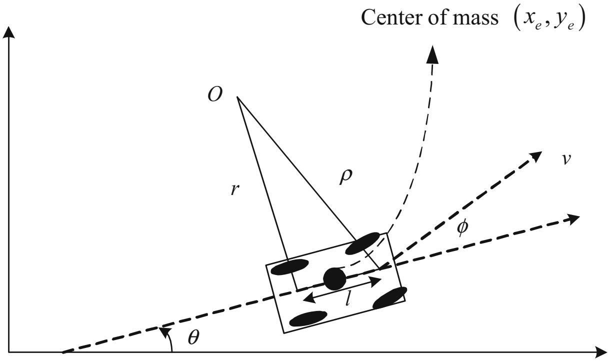
2.2 Modeling of Car-Like Vehicles

Consider the kinematical model of the car-like vehicle shown as in Fig. 1, where the rear wheels are fixed parallel to the car body and allowed to roll or spin but disallowed to slip. This ensures that the rear wheels are always tangent to the automobile orientation. The front wheels can turn either left or right with the constraint that both the front wheels should be parallel. The key variables defining the vehicle's motion illustrated in Fig. 1 are defined as follows

(x_f, y_f)	Position of the center of the front wheels
(x_r, y_r)	Position of the center of the rear wheels
ϕ	Orientation of the steering wheels with respect to the frame of the vehicle
θ	Angle between vehicle frame orientation and X-axis
l	Wheelbase of vehicle
O	Center of curvature
ρ	Distance from O to the midpoint of the front wheels axle
r	Curvature radius
v	Speed of the front wheels
k	Curvature of trajectory

The center of the rear wheel is always tangent to the automobile orientation. Under regular conditions, the reserved velocity is assumed to be about 5 km/h slow; thus, one can suppose that there is no-slipping condition happened and the velocity of the rear wheel in vertical director is close to zero. This is the so-called nonholonomic

Fig. 1 Car-like vehicle



constraint. The kinematics of the car-like mobile with respect to the axis center of the rear wheels are described as

$$\begin{aligned} \dot{x}_r &= v\cos\theta\cos\phi, \\ \dot{y}_r &= v\sin\theta\cos\phi, \\ \dot{\theta} &= v\frac{\sin\phi}{l} \end{aligned} \tag{1}$$

This is used to generate the next backward state position of the vehicle when the present states and control input are given.

2.3 Reference Trajectories for Parallel Parking

It is important to refer a reasonable reference path so that the cars can successfully accomplish the parallel-parking task. In order to implement a feasible and smooth trajectory, Lyon [9] has proposed a derivation of a fifth-order polynomial for the rear reference path. The polynomial is selected because it is the minimum order one capable of giving sufficient degrees of freedom. The rear wheel's path is represented as a function $y_r = f(x_{re})$, where $x_{re} = \frac{x_r}{x_e}$, during a single maneuver. The general form of the polynomial is suggested as

$$f(x_{re}) = a_5x_{re}^5 + a_4x_{re}^4 + a_3x_{re}^3 + a_2x_{re}^2 + a_1x_{re} + a_0 \tag{2}$$

where (x_e, y_e) and (0,0), are, respectively, the vehicle's initial coordinates of the center of mass of the car and terminal position with the following constraints:

$$y_r'(0) = y_r'(x_e) = 0 \tag{3}$$

where $y_r' = {}^{\partial y_r}\!/_{\partial x_r}$ and curvature

$$k(0) = k(x_e) = 0 \tag{4}$$

so that

$$y_r''(0) = y_r''(x_e) = 0 \tag{5}$$

is necessary and sufficient for Eq. 4 to be true.

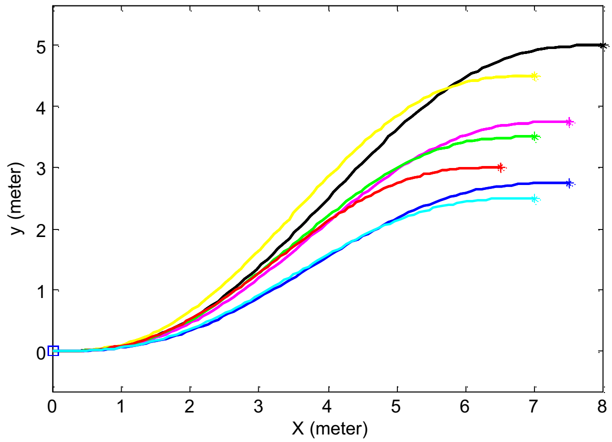


Fig. 2 Reference parking trajectories

Applying Eqs. 3 and 5 to Eq. 2 yields

$$y_r = y_e\left(6x_{re}^5 - 15x_{re}^4 + 10x_{re}^3\right) \tag{6}$$

By utilizing Eq. 6, one can get a variety of reference trajectories as illustrated in Fig. 2.

The use of reference trajectories of the n-th degree polynomial may bound the possible solution of the parking problem and finally increases the number of gear shifting necessary. An alternative could be applied to resolve the problem by using B-Splines as those suggested in [22–24].

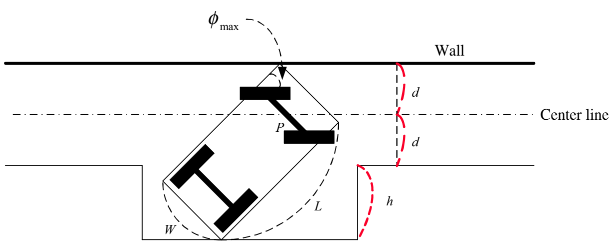
2.4 Ready-for-Parking Space

While an actual parking environment is considered, not all the trajectories produced by Eq. 6 are feasible. To avoid collision, the vehicle must be positioned inside a suitable region before it gets started the parking procedure.

Fig. 3 Extreme scenario of the parallel parking problem

$$h + 2d + \varepsilon_{safe} \geq \sqrt{L^2 + W^2}\,\sin\left(\phi_{\max} + \tan^{-1}\left(\frac{W}{L}\right)\right),$$

$$d + \varepsilon_{safe} \geq \frac{W}{\sqrt{2}}$$



For the admissible parking environment, there is a minimum requirement corresponding to the extreme situation illustrated in Fig. 3. Consider the center line on the road. Suppose that the distance between the line and the wall is d, the depth of the parking space is h, the permissible maximal rotation angle of the front wheel is $\phi_{\max}$, the length and width of the car are, respectively, L and W, and $\varepsilon_{\text{safe}}$ is a constant used to offer an extra safety distance from the obstacles. Then the constraint given for the extreme case is as follows

$$\begin{aligned} h + 2d + \varepsilon_{\text{safe}} &\geq \sqrt{L^2 + W^2} \sin\left(\phi_{\max} + \tan^{-1}\left(\frac{W}{L}\right)\right), \\ d + \varepsilon_{\text{safe}} &\geq \frac{W}{\sqrt{2}} \end{aligned} \tag{7}$$

The preliminary task is to determine a suitable region, which can be determined from two constraints. First, it is the limit of the maximum curvature given by

$$k(x) = \frac{y''}{\left[1 + (y')^2\right]^{\frac{3}{2}}} \leq k_{\max} \tag{8}$$

The second one is the distance from the vehicle to the obstacle's point A with coordinates (x_{ob},y_{ob}), as illustrated in Fig. 4. According to the vehicle's dimensions as defined in Fig. 5, we can solve the values of (x_{rr},y_{rr}) and (x_{fr},y_{fr}) from the reference point (x_{ref},y_{ref}) that produced by the reference trajectory:

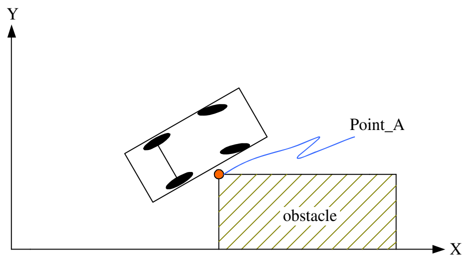
$$\begin{cases} x_{\text{rr}} = x_{\text{ref}} - L_2 \cos\theta + \dfrac{W}{2} \sin\theta, \\ y_{\text{rr}} = y_{\text{ref}} - L_2 \sin\theta - \dfrac{W}{2} \cos\theta, \end{cases} \tag{9}$$

$$\begin{cases} x_{\text{fr}} = x_{\text{ref}} + (L_1 + l)\cos\theta + \dfrac{W}{2} \sin\theta, \\ y_{\text{fr}} = y_{\text{ref}} + (L_1 + l)\sin\theta - \dfrac{W}{2} \cos\theta \end{cases} \tag{10}$$

If $x_{\text{rr}}(t) \geq x_{\text{ob}}$ and $y_{\text{rr}}(t) \leq y_{\text{ob}}$, then the rear-right corner of the vehicle would hit the obstacle. Similarly, if $x_{\text{fr}}(t) \geq x_{\text{ob}}$ and $y_{\text{fr}}(t) \geq y_{\text{ob}}$, the front-right corner of the vehicle would hit the obstacle.

Through this test, the feasible parking region could be characterized as indicated in Fig. 6. The four regions with different colors were characterized by applying the reference parking route into Eqs. 9 and 10 and examine the whole parking route.

Fig. 4 Car parking with obstacle avoidance



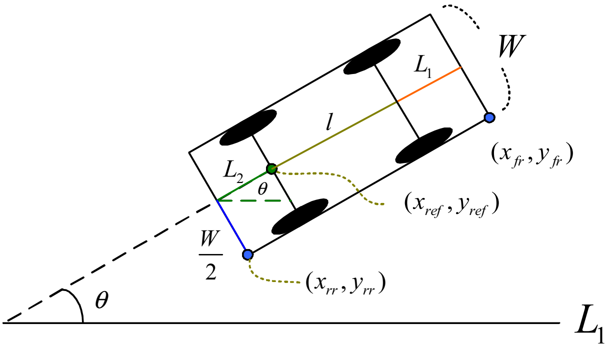


Fig. 5 Definitions of the variables

This shows that, if the car is initially stopped in region 1 then it could be successfully park into the parking space in only one time of gear change. Alternately, the car will hit the obstacle if it was initially stopped in region 2. Region 3 represents the parking paths whose initial curvature exceeded the allowable maximal curvature (assumed to be 45 degrees). Region 4 represents the one with its initial distance to the obstacle being less than $\frac{W}{2}$.

After determining a feasible parking region, the next step would be to navigate the vehicle. The motions of the vehicle involve moving forward and backward; therefore, there would be different paths for the vehicle to navigate into suitable regions since it depends on where the vehicle is situated with respect to the surroundings.

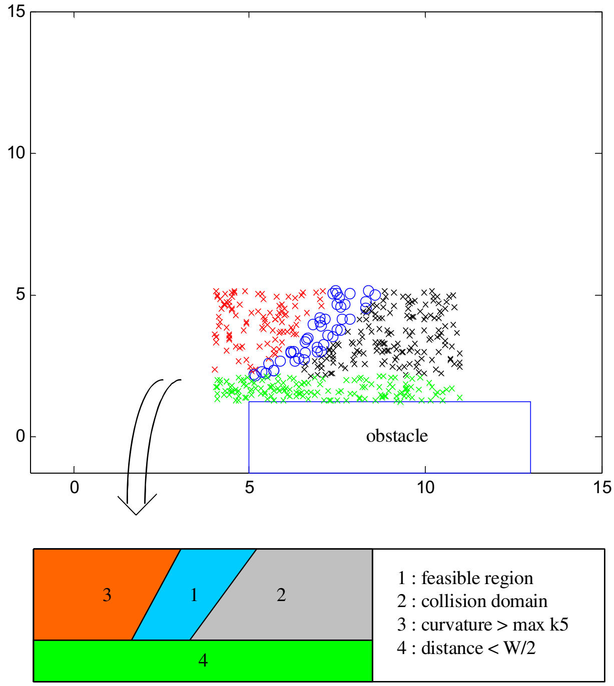


Fig. 6 Different initial positions for feasible and unfeasible vehicle parking region

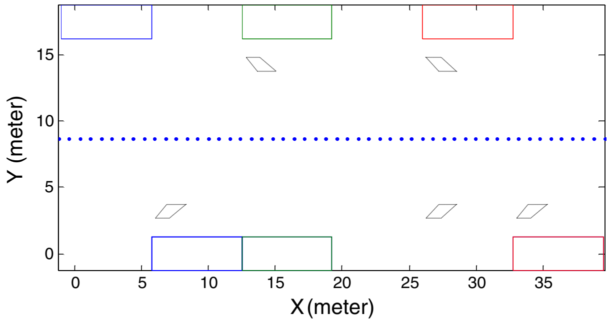


Fig. 7 Ready-for-parking spaces

If the information of the parking lot is available (which might be measured previously or measured via advanced identification/communication techniques, i.e. RFID, Bluetooth, etc.), we can obtain a suitable region for each parking space, which is called "Ready-for-parking space" as shown in Fig. 7.

Before navigating the vehicle, it is essential to determine the most suitable parking space in accordance with current circumstances. It is difficult for human to choose a suitable parking space and decisions are often made instinctively. To resolve the problem an effective method should be developed.

3 Petri Nets and GAs

The parking process is a discontinuous action involving moving forward and backward; therefore, different actions must be combined to find a suitable path for an optimal solution. Petri nets based on discrete event formalisms have been investigated as potential means of achieving collision avoidance, task-preserving human-intervention, and route controlling, which can be served as a tool for the current purpose.

3.1 Introduction of Petri Nets

Petri nets are graphical and mathematical modeling tools applied to many systems. They are promising tools for describing and studying information, processing systems and are characterized as being concurrent, asynchronous, distributed, parallel, nondeterministic, and/or stochastic. As a graphical tool, Petri nets can be used as a visual-communication aid which is similar to flow charts, block diagram, and networks. In addition, tokens are used in these nets to simulate the dynamic and concurrent activities of systems. As a mathematical tool, it is possible to set up state equations, algebraic equations, and other mathematical models governing the behavior of systems.

3.1.1 Classical Petri Nets

A classical Petri net, shown in Fig. 8, consists of four fundamental elements: place, transition, token, and arc. It is a type of directed graph with an initial state called *initial marking*, M_0. The net consists of two kinds of nodes, called *place* and *transition nodes*. *Arc* connects a place to a transition or vice versa. In Fig. 8, places are

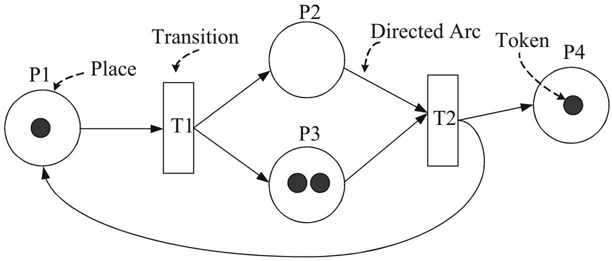


Fig. 8 Four elements of Petri nets (place, transition, token, and arc)

represented as circles and transitions as rectangles. A *marking* assigns a nonnegative integer to each place. If a marking assigns a nonnegative integer k to place P, place P is marked with k tokens. Graphically, k black dots (tokens) are placed in place P. A marking is denoted by M, an $m \times 1$ vector, where m is the total number of places. The pth component of M, denoted by $M(p)$, is the number of tokens in place p.

3.2 Character of Petri Nets

Reachability The reachability of a Petri net is defined as all states that can be reached from a net N with the initial marking M_0, denoted as $R(N,M_0)$. A *reachability graph* is often used to represent the reachability of a Petri net's states. The graph is constructed through a search method where the starting state M_0 is taken, and all possible transitions are evaluated from the starting state. The searching progress expands outwards similarly.

While reachability seems to be a good tool to find erroneous states, the practical problems of the constructed graph still has far too many states to calculate. To solve this problem, a complete Petri net based structure must be constructed.

We use the concept of Petri nets to organize the current parking design system. Since the motion of an automobile that involves gear shifting and direction changing is not a continuous motion, the Petri net is appropriate for the current application. To initialize the problem, let the intermediate parking area be denoted by p and the behavior of the automobile be denoted by t. The intermediate parking area is then divided into nine sub-regions as shown in Fig. 9, where each region corresponds to different navigation paths. The transition for the vehicle is set as in Fig. 10.

By using the concept of Petri nets, the system can be build as in Fig. 11 and a Petri net flow chart with initial and target markings is developed as shown in Fig. 12.

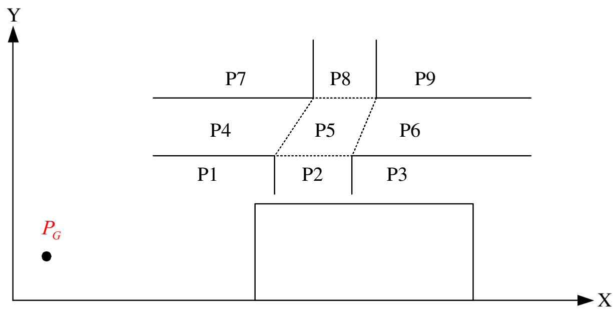


Fig. 9 Nine intermediate parking regions

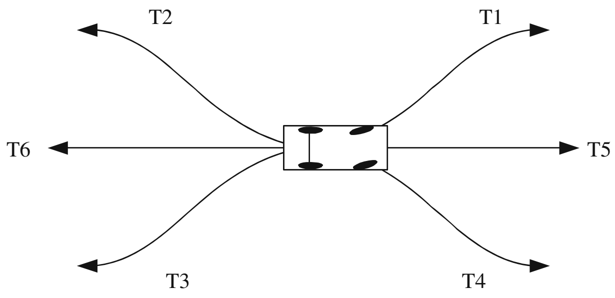


Fig. 10 Transitions for the vehicle

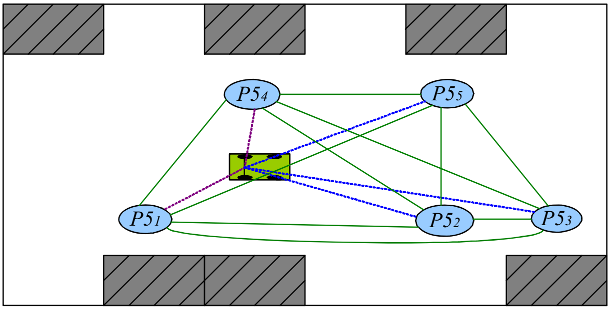


Fig. 11 Organization of Petri nets

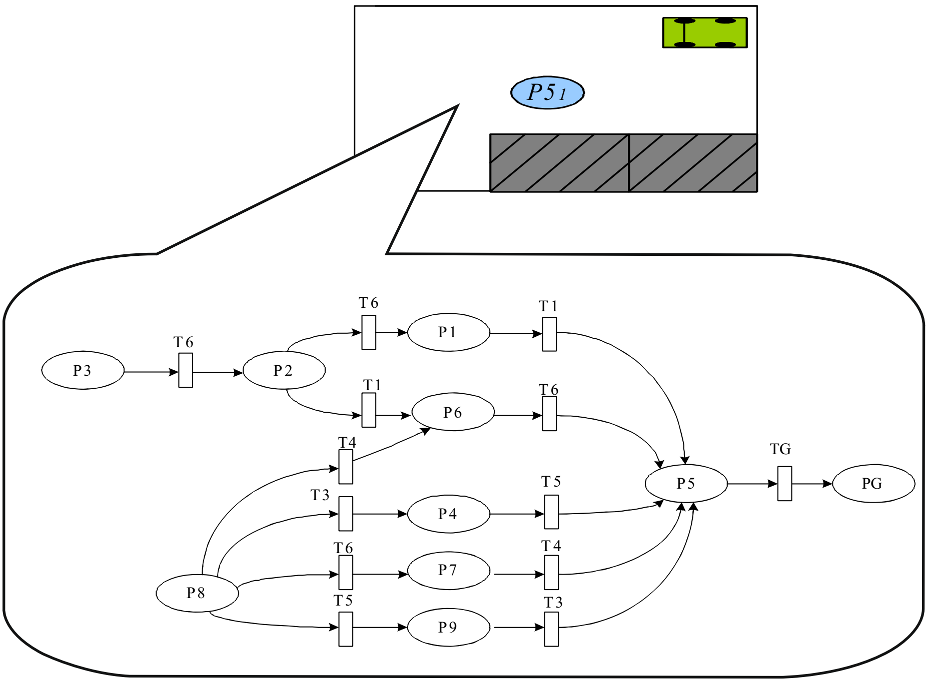


Fig. 12 Petri net flow chart

On the basis of it, we can create a set of reachable paths from the given initial marking to the target marking. Additionally, it is propitious for finding the next place rapidly, if any unexpected error could have occurred. Here we use a case to explain the function of Petri nets.

Demonstrative Case The initial state in $P8$ is

$$M_0 = \left[0\ 0\ 0\ 0\ 0\ 0\ 0\ 1\ 0\right]^T,$$

$$W^+ = \begin{bmatrix} * & T1 & T2 & T3 & T4 & T5 & T6 \\ P1 & 0 & 0 & 0 & 0 & 0 & 1 \\ P2 & 0 & 0 & 0 & 0 & 0 & 1 \\ P3 & 0 & 0 & 0 & 0 & 0 & 0 \\ P4 & 0 & 0 & 1 & 0 & 0 & 0 \\ P5 & 1 & 0 & 1 & 1 & 1 & 1 \\ P6 & 1 & 0 & 0 & 1 & 0 & 0 \\ P7 & 0 & 0 & 0 & 0 & 0 & 1 \\ P8 & 0 & 0 & 0 & 0 & 0 & 0 \\ P9 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}, \quad W^- = \begin{bmatrix} * & T1 & T2 & T3 & T4 & T5 & T6 \\ P1 & 1 & 0 & 0 & 0 & 0 & 0 \\ P2 & 1 & 0 & 0 & 0 & 0 & 1 \\ P3 & 0 & 0 & 0 & 0 & 0 & 1 \\ P4 & 0 & 0 & 0 & 0 & 1 & 0 \\ P5 & 0 & 0 & 0 & 0 & 0 & 0 \\ P6 & 0 & 0 & 0 & 0 & 0 & 1 \\ P7 & 0 & 0 & 0 & 1 & 0 & 0 \\ P8 & 0 & 0 & 1 & 1 & 1 & 1 \\ P9 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}$$

Step1 Look for the firing vector of transitions ρ to obtain the second state. In this case $\rho = \left[0\ 0\ 1\ 1\ 1\ 1\right]^T$ and

$$M' = \rho \otimes W^+ = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix} \text{ means } P8 \rightarrow \left\{ \begin{array}{l} T3 \rightarrow \left\{ \begin{array}{l} P4 \\ P5 \end{array} \right. \\ T4 \rightarrow \left\{ \begin{array}{l} P5 \\ P6 \end{array} \right. \\ T5 \rightarrow \left\{ \begin{array}{l} P5 \\ P9 \end{array} \right. \\ T6 \rightarrow \left\{ \begin{array}{l} P1 \\ P2 \\ P5 \\ P7 \end{array} \right. \end{array} \right.$$

Step2 Check the path's feasibility. In this case the initial position of the car is in $P8$, we observe that the car can move within its boundaries but it is impossible for it to move in parallel to $P5$, so we can get the feasible path as follows

$$P8 \rightarrow \left\{ \begin{array}{l} T3 \rightarrow \left\{ \begin{array}{l} P4\,(o) \\ P5\,(\times) \end{array} \right. \\ T4 \rightarrow \left\{ \begin{array}{l} P5\,(\times) \\ P6\,(o) \end{array} \right. \\ T5 \rightarrow \left\{ \begin{array}{l} P5\,(\times) \\ P9\,(o) \end{array} \right. \\ T6 \rightarrow \left\{ \begin{array}{l} P1\,(\times) \\ P2\,(\times) \\ P5\,(\times) \\ P7\,(o) \end{array} \right. \end{array} \right.$$

Step3 Repeat *Steps 1* and *2* and stop computing when reaching $P5$. In this case

$$P8 \rightarrow \begin{cases} T3 \rightarrow P4 \rightarrow T5 \quad \rightarrow P5 \\ T4 \rightarrow P6 \rightarrow T6 \rightarrow \begin{cases} P1 \\ P2 \\ P5 \\ P7 \end{cases} \\ T5 \rightarrow P9 \rightarrow T3 \rightarrow \begin{cases} P4 \\ P5 \end{cases} \\ T6 \rightarrow P7 \rightarrow T4 \rightarrow \begin{cases} P5 \\ P6 \end{cases} \end{cases}$$, so the reachability path is

$$P8 \rightarrow \begin{cases} T3 \rightarrow P4 \rightarrow T5 \rightarrow P5 \\ T4 \rightarrow P6 \rightarrow T6 \rightarrow P5 \\ T5 \rightarrow P9 \rightarrow T3 \rightarrow P5 \\ T6 \rightarrow P7 \rightarrow T4 \rightarrow P5 \end{cases}$$

Step4 Combined with the GA depicted later to determine the permissible parking paths. Select the optimal one subsequently.

3.3 Optimal Path Planning with GA

In this section, we used the binary GA to design an optimal path. GAs are artificial genetic systems based on the process of natural selection and natural genetic. They have been proven to be efficient in functioning optimization. Unlike traditional search algorithms, GAs don't require a large number of mathematical equations to search for the optimal solutions.

3.3.1 Performance Specifications

While designing an optimal path for vehicle parking, there are various performance indices that are generally considered. In [2] and [11], distance and curvature were considered for the problem of parking path optimization. Here, an additional factor was placed into consideration, that is, the number of gear change. Utilizing the number of gear change can ensure that the total time the parking process taken is minimized. It is believed that a parking procedure that requires more than two gear shifts would be considered inefficient.

Distance In the process of car parking, the relative position of the parking spot is significant to be recognized. To illustrate, a car parallel to a parking space will not be able to move directly into the parking spot although they seems really close. Therefore, it is essential to consider the distance along with the forward and backward movement.

Curvature The curvature of vehicle movement affects the rotary angle of the wheels. Although every vehicle has a different limitation on the rotary angle, its maximum value, in common, does not exceed 45 degrees. An overlarge curvature that appears when the vehicle is in motion could cause passengers a physical discomfort or even dizziness.

Number of Gear Change Generally, a parking procedure requires more than simply reversing a vehicle into the desired parking space. The routine must also be involving the characteristics of driving forward and backward which is equivalent to the number of gear change. The lesser times the forward and reverse gear shifts have engaged, the easier it is to have the parking procedure smoothly. In addition, more gear changes can result in extended distances; therefore, this is also a performance index which should be minimized.

On the basis of the previous definitions, we define the term "optimal parking" which means that the car can follow the optimal path which admits the least parking distance, curvature, and number of gear changes. This, in other words, means the least parking effort for the car driving mechanism. The fitness function reflecting the optimal parking can be designed as follows

$$\text{Fitness} = \frac{1}{W_1 \frac{D}{D_{\max}} + W_2 \frac{\mathrm{Ng}}{\mathrm{Ng}_{\max}} + W_3 \frac{\sum_{i=1}^{n} |\phi_i|}{\left[\sum_{i=1}^{n} |\phi_i|\right]_{\max}}} \tag{11}$$

where

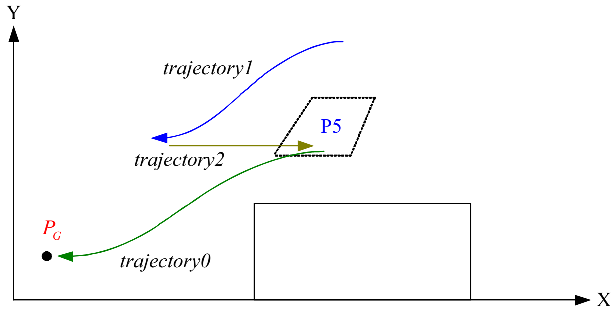
W_i	weighting scalar
$D\left(=\sum_{i=1}^{N} \text{trajectory}_i\right)$	total distance moved
Ng	number of gear change
ϕ_I	orientation of the steering wheels with respect to the frame of the vehicle of the *i*-th moving segment
$\bullet_{\max}$	prespecified maximum value of the designated variable

3.3.2 GA Operators

Procedures The design of the optimal parking path is organized in the following steps. The initial position (x_0, y_0) is first randomly selected.

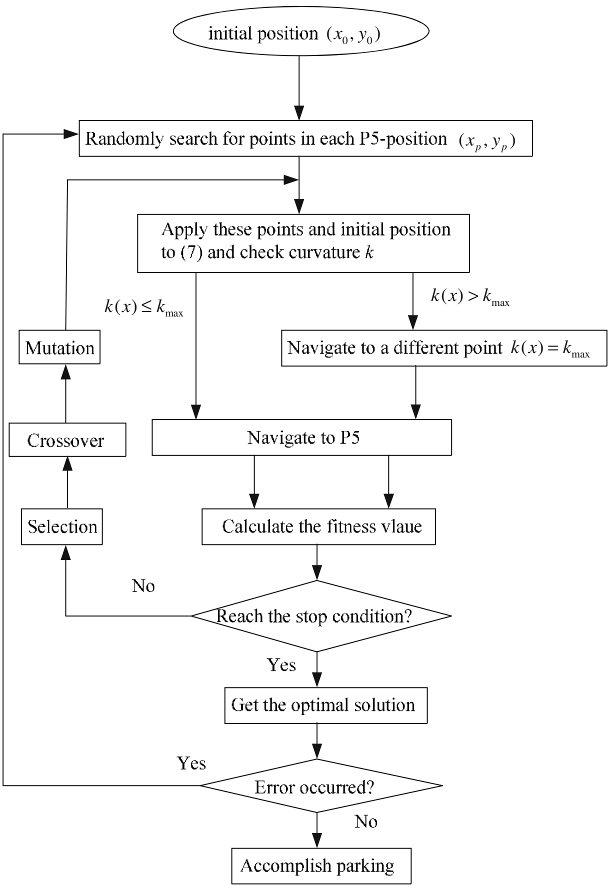
1. Generate the points (x_p, y_p) randomly in each $P5$-position, $p = 1, 2, n$.
2. Apply these points and initial position to (6) and make sure that the curvature k does not exceed $k_{\max}$. If $k(x) \leq k_{\max}$, the path allows the vehicle to navigate to $P5$; *trajectory1* is produced. If $k(x) > k_{\max}$, it is impossible for the vehicle to navigate directly to $P5$; therefore, the vehicle should be navigated to a different point where it is able to perform the maximum curvature. Obviously, the resulting path we obtained will consist of two parts; a curve *trajectory1* and a line *trajectory2* as shown in Fig. 13. P_5 in Fig. 13 is the region of $P5$ in Fig. 9 which was selected only for demonstration. Actually, the car could be initially stopped in anyone of nine regions in Fig. 9.
3. Apply (x_p, y_p) and (x_G, y_G) to obtain *trajectory0*.
4. Find the number of gear change. For example, $P8 \rightarrow P4 \rightarrow P5 \rightarrow P_G$: The vehicle moved backward, forward, then backward, needs three gear change.
5. Find the change of ϕ.
6. Calculate the fitness and select the better chromosomes with probabilities based on the fitness value.

Fig. 13 Illustrations for parking trajectories



7. If reaching the stop condition or obtaining the optimal solution, we may stop the process, or else repeat Steps 2–7 until reaching the stop condition.
8. Get the optimal solution.
9. Check if the error occurred when the vehicle moves. (For example: the parking space is occupied.)

Fig. 14 Overall flow chart



The overall flow chat is illustrated in Fig. 14. After finding the optimal path, the next step is to make the vehicle follow the trajectory accurately; a trajectory tracking controller is implemented next to fulfill the remaining task.

4 Fuzzy Controller Design

One of the major advantages is that fuzzy logic control can integrate linguistic information, expert knowledge, and experience to design the controller without requiring a precise system model. We focus in this section on designing the tracking controller for the vehicle to complete the task of parallel parking.

4.1 Problem Description and Analysis

Suppose that the goal is characterized by (x_G, y_G, θ_f), the current vehicle position is (x, y), the orientation of the vehicle is θ_c, and the angle between the vehicle center line and the goal is θ_G, see Fig. 15:

Referring to Fig. 15, we can get θ_G, d_e, u_1 and u_2 as follows

$$\theta_G = \tan^{-1}\left(\frac{y_G - y}{x_G - x}\right), \tag{12}$$

$$d_e = \sqrt{(x_G - x)^2 + (y_G - y)^2}, \tag{13}$$

$$u_1 = \theta_G - \theta_c, \tag{14}$$

$$u_2 = \theta_f - \theta_G, \tag{15}$$

We propose a two-stage fuzzy logic controller to command the steering angle of the front wheels for parking task as shown in Fig. 16. Input variables of the *Stage1* controller are u_1 and u_2 and the output variable is θ_e. The fuzzy control rules are listed in Table 1. The membership functions for characterizing the input and output variables are illustrated in Fig. 17. In practical applications, the sensor module is used to identify the position of the car in the inertia frame.

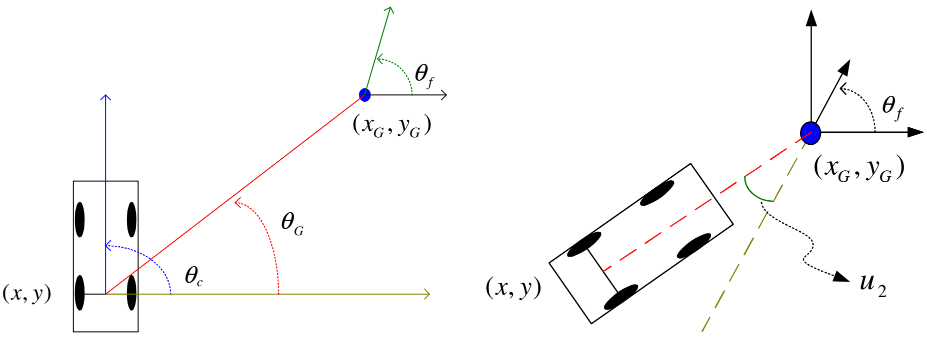


Fig. 15 Target tracking control system with terminal orientation angle θ_f

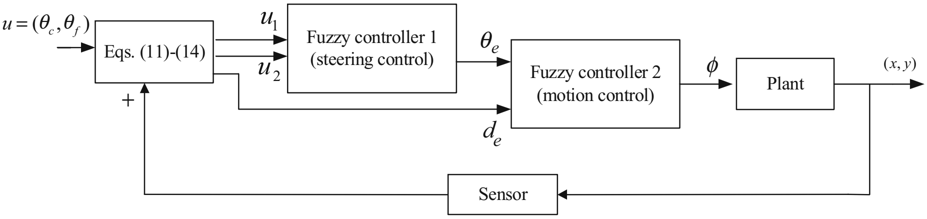


Fig. 16 Two-stage fuzzy logic controller

Table 1 Fuzzy rules for steering control

θ_e		u_1				
		NB	NS	ZE	PS	PB
u_2	NB	NM	NB	NB	NB	NB
	NM	NS	NM	NM	NM	NB
	NS	PM	PS	NS	NS	NM
	ZE	PM	PS	ZE	NS	NM
	PS	PM	PS	PS	NS	NM
	PM	PB	PM	PM	PM	PS
	PB	PB	PB	PB	PB	PM

Fig. 17 Coding in chromosome for membership function parameters

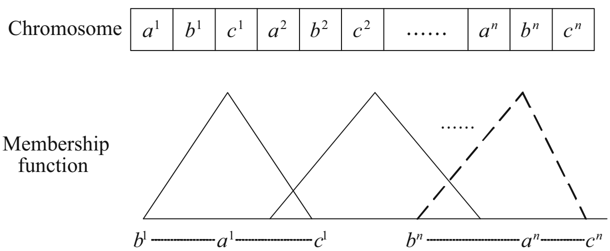


Table 2 Fuzzy rules for motion control

ϕ		d_e				
		VS	S	M	B	VB
θ_e	NB	PS	PM	PM	PB	PB
	NS	PS	PS	PM	PM	PB
	ZE	ZE	ZE	ZE	ZE	ZE
	PS	NS	NS	NM	NM	NB
	PB	NS	NM	NM	NB	NB

For the *stage2* controller, d_e and θ_e are used as the input linguistic variable. d_e is in the range of distance [0,100] (m) and θ_e is in the range of angle [−180,180] (degree). The output variable is the steering angle ϕ. For the actual condition of the vehicle, the range of ϕ is [−45,45] (degree). Fuzzy control rules are listed in Table 2. The smaller the value of d_e, the closer the vehicle is to the goal; the smaller the value of θ_e, the more accurate and direction-wise the vehicle is heading. The smaller the value of u_2, the closer the vehicle is to the straight line extended from the terminal orientation angle.

When $u_1 = 0$ and $u_2 = 0$, it means that the vehicle is already on the extended line and is heading towards the goal correctly. The objective of the controller is to make d_e, u_1, and u_2 converge to zero when $t \to \infty$. As a result, the vehicle would just need to go straight for reaching a specified goal. It should be remarked, however, that it is not always necessarily the car parking to be perfectly precise in practice. Therefore, when the tracking error has reached within a permissible range, the parking operation is thought to be accomplished. Thus, it actually admits some modifications for real-world applications.

4.2 Parameter Optimization

To assure that the fuzzy controller achieves the goal more accurately, we use GAs to adjust the parameters of the membership function. Each input and output membership function possesses the following form:

$$\mu^j(x) = \begin{cases} 1 + \dfrac{x - a_j}{a_j - b_j}, & b_j \le x \le a_j \\ 1 + \dfrac{a_j - x}{c_j - a_j}, & a_j \le x \le c_j \\ 0, & \text{otherwise} \end{cases} \tag{16}$$

The parameters for the corresponding membership function should be symmetric to the stationary point of the input and output variables. The objective function is defined as

$$E = \sum_{i=1}^{k} \left[\frac{d_{e_i}^2}{(x_G - x_0)^2 + (y_G - y_0)^2} + \frac{\theta_{e_i}^2}{|\theta_G - \theta_0|^2} \right] \tag{17}$$

and the corresponding fitness function is

$$\text{Fitness} = \frac{1}{E} \tag{18}$$

As shown in Fig. 17, parameters in the membership function are coded in the chromosome and the process of genetic evolution involves reproduction, crossover and mutation. The course of evolution is performed individually for each population. While these procedures are being carried out, a chromosome that possessed the highest fitness value is selected from the population and then proceeds to the next generation. Such adjustment for input and output terms is performed repeatedly.

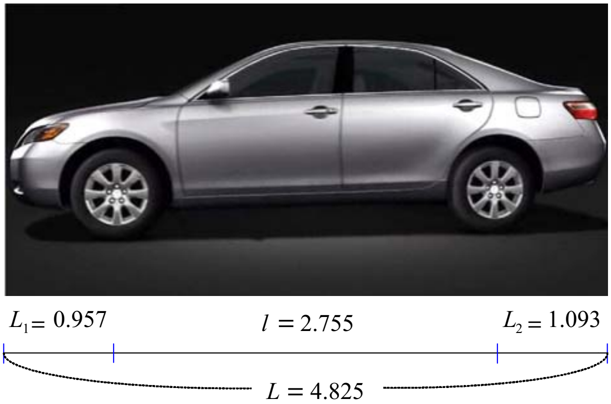


Fig. 18 Dimensions of the demonstrative vehicle

4.3 Simulation Results

The simulation results presented here were based on Toyota Camry Sedan 2008 (Fig. 18) with the following dimensions: length 4.825 m, width 1.82 m, $l = 2.755$ m and $\phi_{\max} = 45$ degree. Each parking space is suppose to be of the standard size 6.5×2.5 m. For the minimum requirement of parking environment, $d \geq 1.6$ m, $\varepsilon_{\text{safe}} = 0.3$ m.

In the first case, the car is with the starting position $(20{,}30{,}-50) \equiv (m{,}m{,}\text{degree})$ and the goal (50,70,60). The experiments are designed to verify the performance of the fuzzy steering control scheme with its membership functions with and without parameter optimization. For the unoptimized case, the membership functions in the fuzzy term sets equally partition the corresponding physical domains. The trajectories of the simulation results for the unoptimized and optimized cases are illustrated on the left and right sides of Fig. 19 respectively. The membership functions with and without optimization are displayed in Fig. 20. In Fig. 21, the figures placed on the left column show the result without optimization.

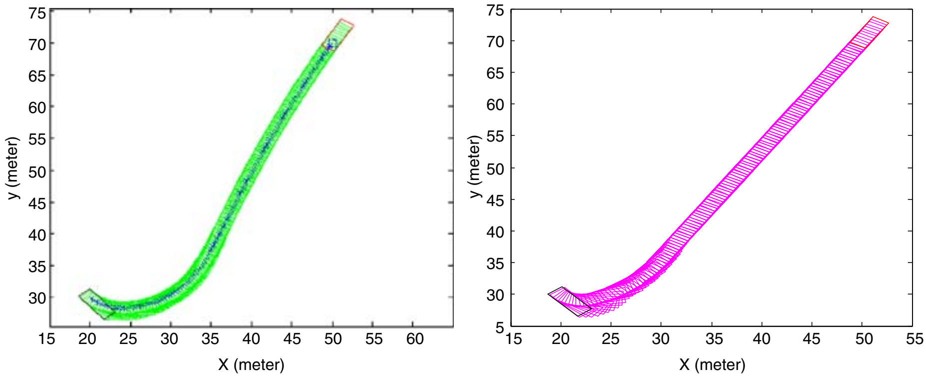


Fig. 19 Experiments without (*left*) and with (*right*) parameter optimization for the fuzzy controller

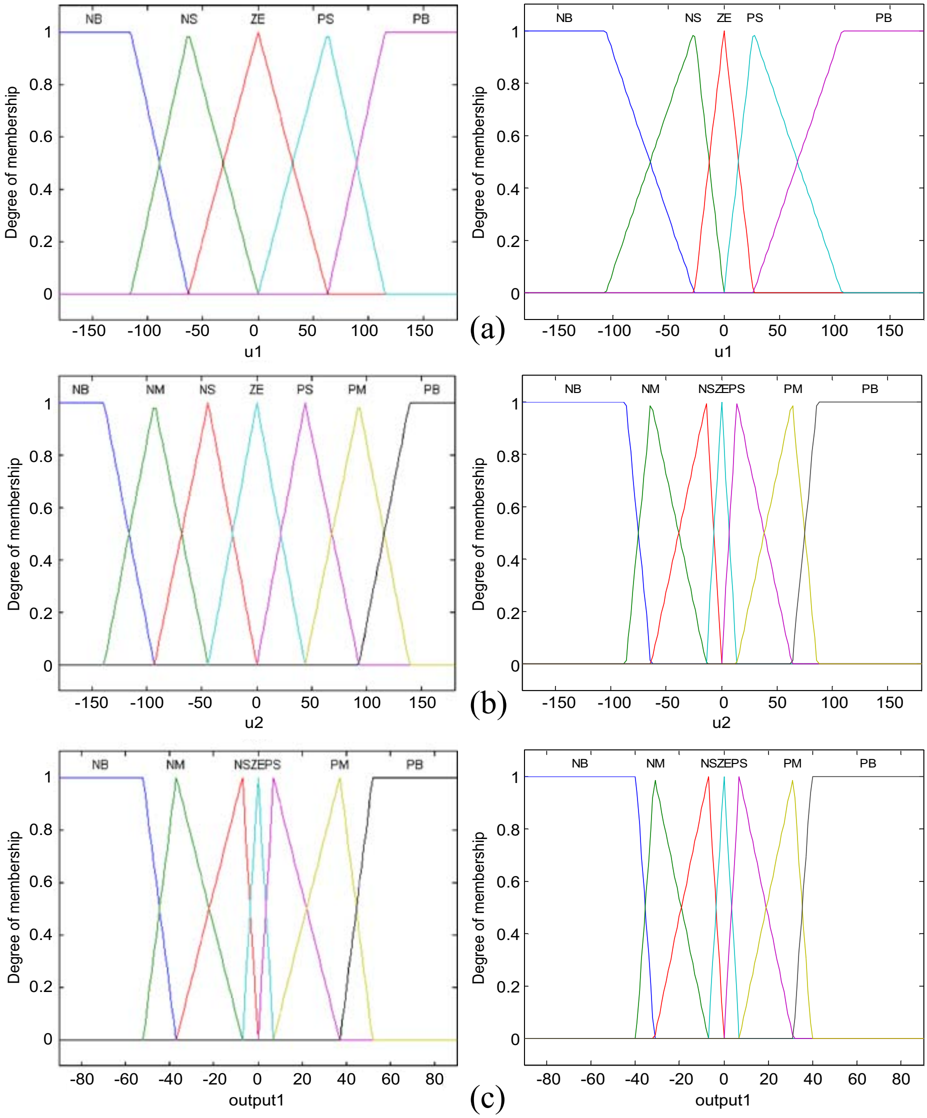


Fig. 20 Membership functions without (*left column*) and after (*right column*) optimization; **a** u_1, **b** u_2, and **c** ϕ

After optimizing the controller, the focus is switched to verify tracking control to the reference path. In Fig. 22, the working efficiency for the trajectory tracking of the reference path is shown.

In the case of Figs. 23 and 24, the car is in the ready-to-parking space *P*5, navigating to the parking space from different sides.

Next, we apply the path optimization method and the fuzzy controller for finding the optimal path in the parking area with multiple parking spaces. We set the generation number 50 and the population size 20. The resulting parking path is shown

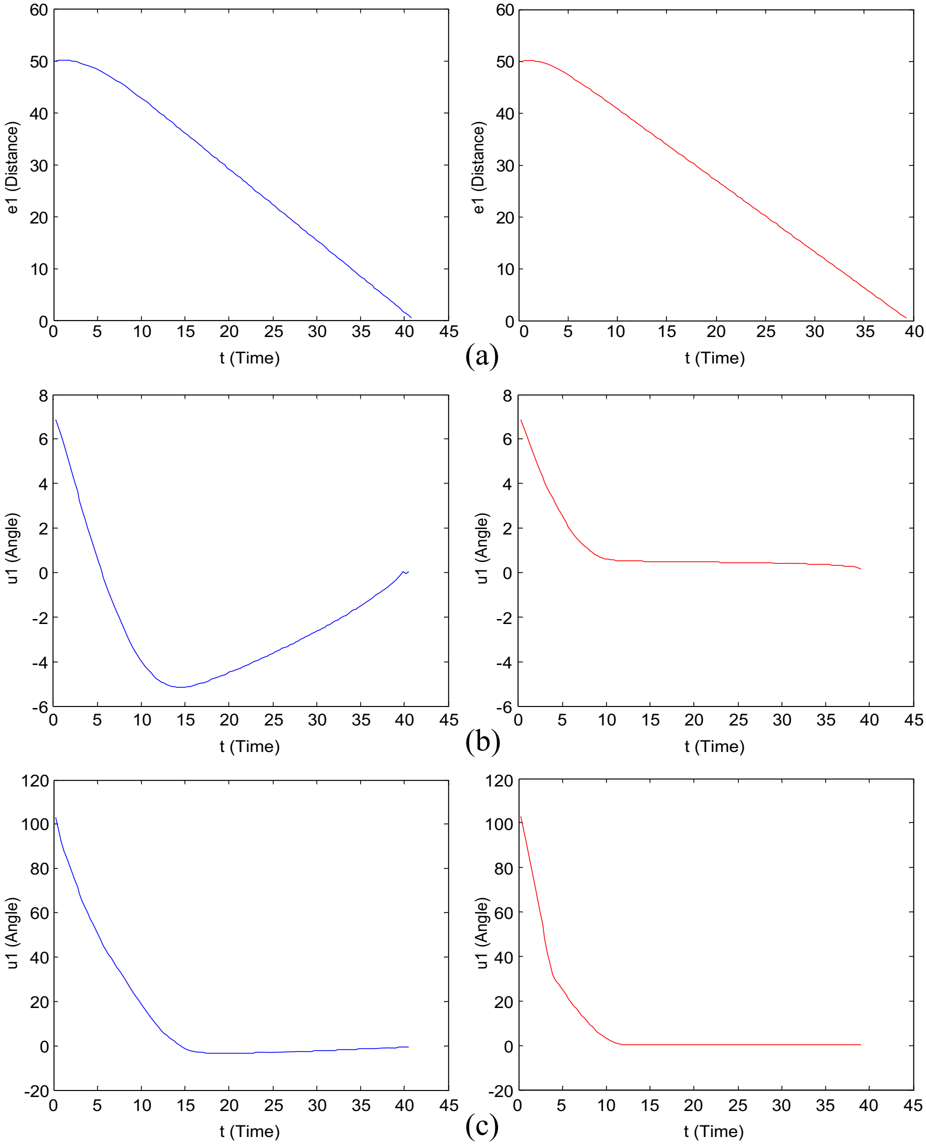


Fig. 21 Comparisons of the responses of **a** d_e, **b** u_1, and **c** u_2 for the fuzzy controller without (*left column*) and with (*right column*) parameter optimization

in Fig. 25. Change of fitness, distance, number of gear change, and steering angle are displayed in Fig. 26. In this case, we adjusted the weight function so that the GA preferred to emphasize the number of gear change so that the car is able to be driven into the parking space with less effort.

In Fig. 27, the car is merely required to shift into the reverse gear to reach *P*5; therefore, one gear change, shortest parking distance and smallest steering angle are ultimately obtained, as shown in Fig. 28.

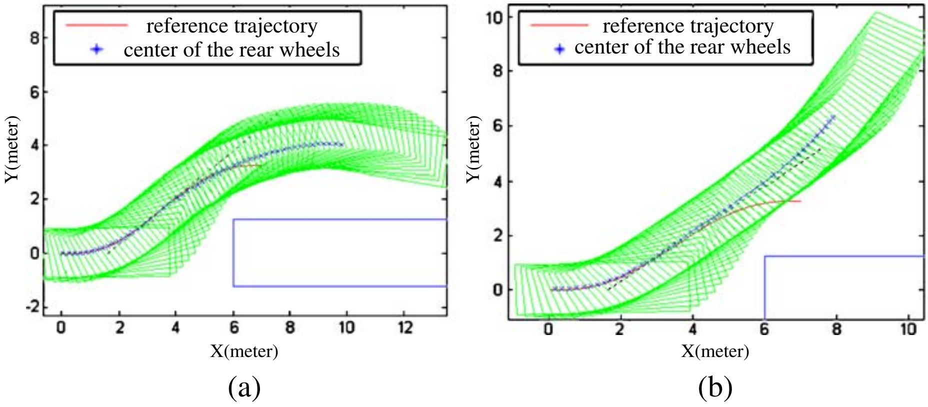


Fig. 22 Parallel parking trajectories with the initial condition (10,4,–10) (*left*), (8,6.5,60) (*right*) and terminal condition (0,0,0)

For further demonstration, three animations have been created and ready for demonstration of our proposed design. Please visit the websites listed below.

Case 1 http://140.120.31.124:2930/movie01.wmv
Route planning and parking execution without considering obstacles.

Case 2 http://140.120.31.124:2930/movie02.wmv
Route planning and parking execution considering the condition that the initial parking space was occupied.
From this particular case, the effect of the Petri net is clearly illustrated; when encountering a sudden obstacle, the token is transferred from the

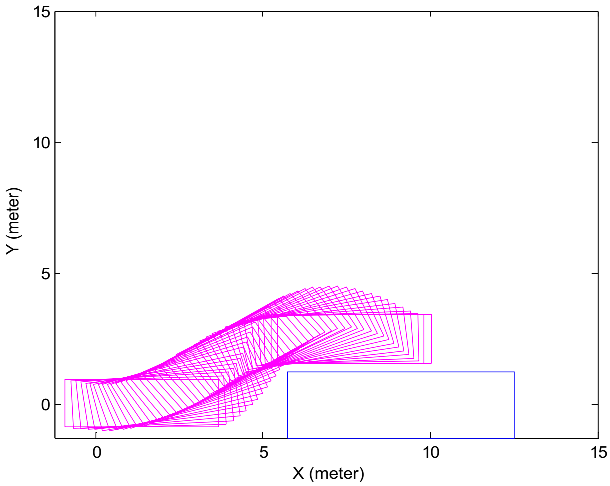


Fig. 23 Initial condition (6.5,2.5,0) and terminal condition (0,0,0)

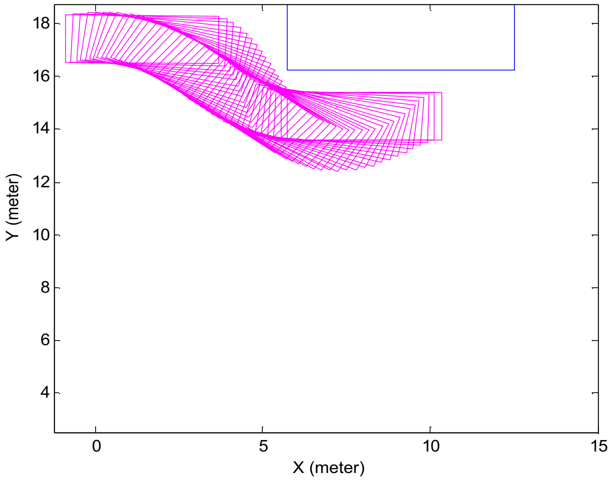


Fig. 24 Initial condition (6.8,14.5.0) and terminal condition (0,17.5,0)

original place to an alternative position, so that the car is able to respond immediately and resort to another empty parking space.

Case 3 http://140.120.31.124:2930/movie03.wmv
Route planning and parking execution considering the condition that the original planned parking space becomes occupied.

4.4 Practical Concerns

For practical applications, the proposed parking system can be implemented in actual parking lots. RFID components can be included in each parking space and vehicle so that one is able to detect the parking environment prior to fulfill the parking

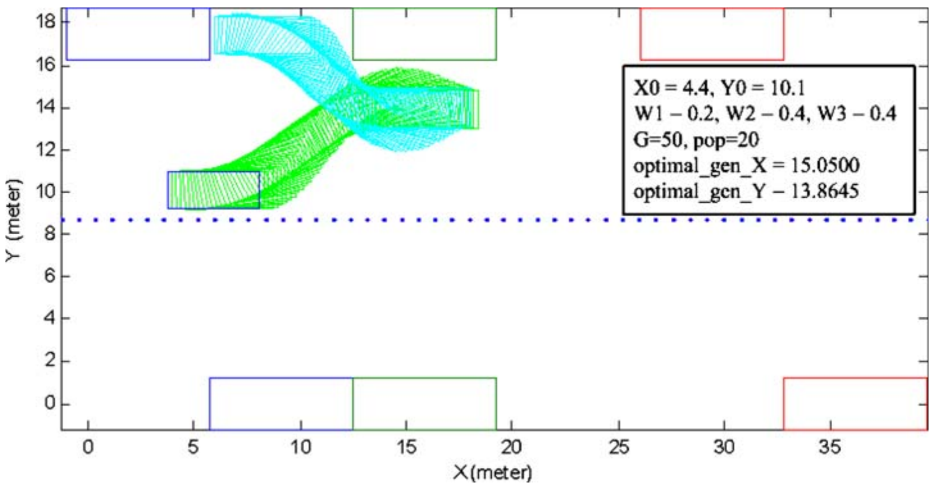


Fig. 25 Simulation result for random selection 2

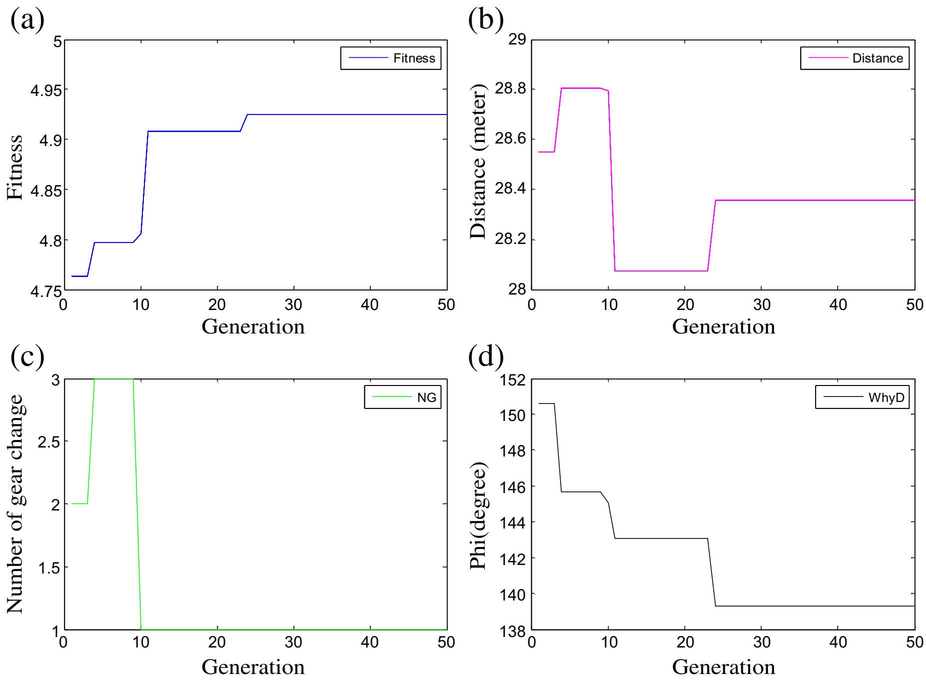


Fig. 26 Overall performance **a** fitness, **b** distance, **c** NG, **d** ϕ

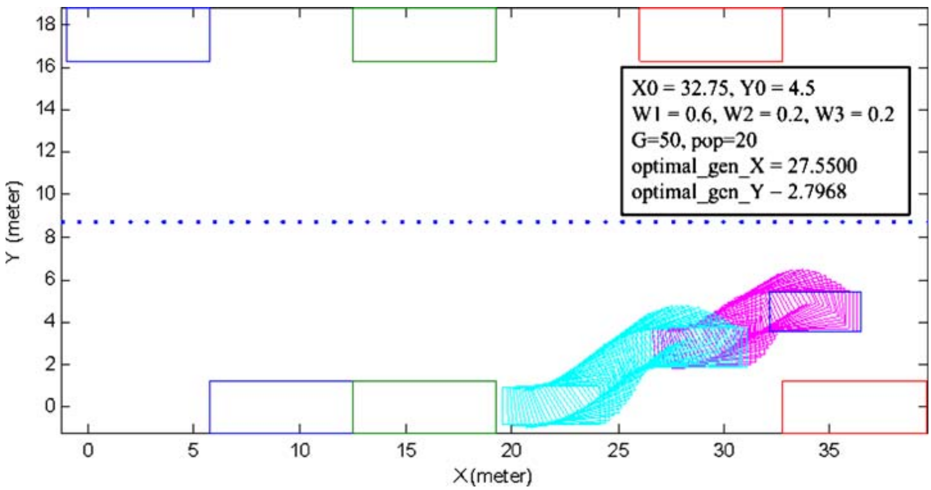


Fig. 27 Simulation result for random selection 3

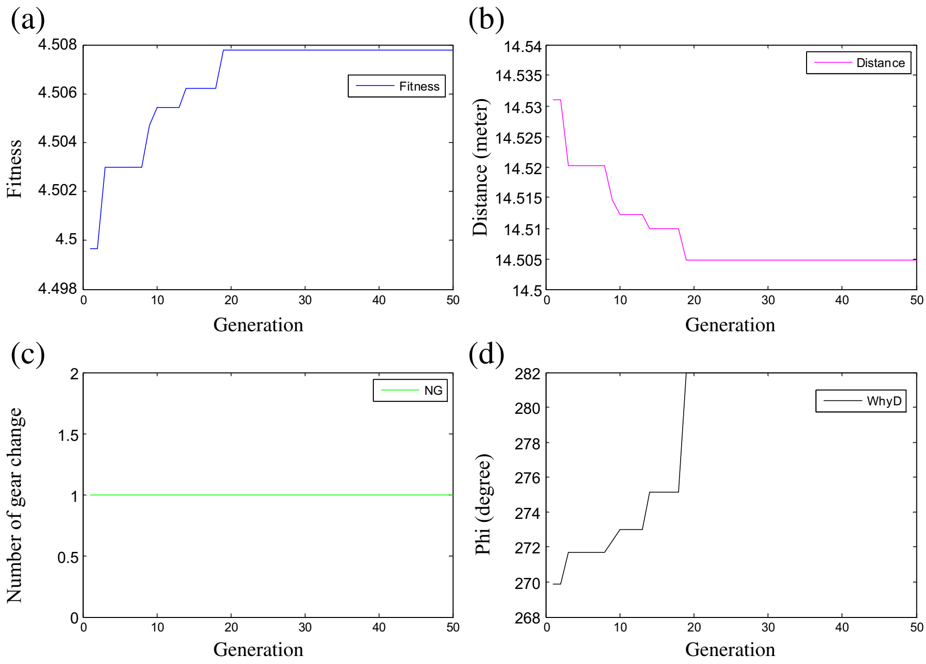


Fig. 28 Overall performance **a** fitness, **b** distance, **c** NG, **d** ϕ

procedure. After gathering the required information on available parking spaces through RFID components and wireless communication components, the parking availability can be determined and the algorithm can be performed for automatic parking.

5 Conclusion

In this paper, a complete parking mechanism for autonomous car-like vehicles has been introduced and applied to solve the parallel parking problem. This architecture includes a Petri-net, off-line global trajectory planner, a decisive kernel based on GAs, and a trajectory tracking controller. The major advantage of the proposed method is, providing an effective parking path and strategy and further extending the case of single parking space to the case of multiple parking spaces. The simulation results have illustrated the effectiveness of the proposed approach.

Acknowledgements The authors would like to thank Dr. V. Kanakakis for brining the extreme scenario Fig. 3 and references [22–24] to their attention. This research was sponsored in part by the Ministry of Education, Taiwan, ROC under the ATU plan.

References

1. Dubins, L.E.: On curves of minimal length with a constraint on average curvature, and with prescribed initial and terminal positions and tangents. Am. J. Math. **79**, 497–516 (1957). doi:10.2307/2372560
2. Reeds, J.A., Sheep, L.A.: Optimal paths for a car that goes both forwards and backwards. Pac. J. Math. **145**, 367–393 (1990)
3. Lamfferriere, G., Sussmann, H.: Motion planning for controllable systems without drift. In: Proc. IEEE Int. Conf. Robotics Automation, vol. 2, pp. 1148–1153 (1991)
4. Murray, R., Sastry, S.: Steering nonholonomic system using sinusoid. In: Proc. IEEE Int. Conf. Decis. Control, pp. 2097–2101 (1990)
5. Paromtchik, I.E., Laugier, C.: Autonomous parallel parking of a nonholonomic vehicle. In: Proc. IEEE Int. Conf. Vehicle Symp, pp. 13–18 (1996)
6. Yasunobu, S., Murai, Y.: Parking control based on predictive fuzzy control. In: Proc. IEEE Int. Conf. Fuzzy Systl, pp. 1338–1341 (1994)
7. Jenkins, R.E., Yuhas, B.P.: A simplified neural network solution through problem decomposition: the case of the truck backer-upper. IEEE Trans. Neural Netw. **4**(4), 718–720 (1993)
8. Nishimori, K., Hirakawa, S., Tokutaka, H.: Fuzzification of control timing in driving control of model car. Proc. IEEE Conf. Fuzzy Syst. **1**, 297–302 (1993)
9. Lyon, D.: Parallel parking with curvature and nonholonomic constraint. In: Proc. IEEE Intell. Vehicles Symposium, pp. 341–346 (1992)
10. Watanable, K., Tang, J., Nakamura, M., Koga, S., Fukuda, T.: A fuzzy-Gaussian neural network and its application to mobile robot control. IEEE Trans. Control Syst. Technol. **4**(2), 193–199 (1996)
11. Tang, K.S., Man, K.F., Liu, Z.F., Kwong, S.: Minimal fuzzy memberships and rules using hierarchical genetic algorithms. IEEE Trans. Ind. Electron. **45**(1), 162–169 (1998)
12. Zhou, Y.S., Lai, L.Y.: Optimal design for fuzzy controller by genetic algorithm. IEEE Trans. Ind. Appl. **36**(1), 93–97 (2000)
13. Kim, S.H., Park, C.: A self-organized fuzzy controller for wheeled mobile robot using an evolutionary algorithm. IEEE Trans. Ind. Electron. **48**(2), 467–474 (2001)
14. Chang, S.J.: Autonomous fuzzy parking control of a car-like mobile robot. IEEE Trans. Syst. Man Cybern., Part A **33**(4), 451–465 (2003)
15. Chiu, C.S., Lian, K.Y.: Fuzzy gain scheduling for parallel parking a car-like robot. IEEE Trans. Control Syst. Technol. **13**(6), 1084–1092 (2005)
16. Suruz, M.M., Wail, G.: Intelligent parallel parking of a car-like mobile robot using RFID technology. In: Proc. of the ROSE 2007 International Workshop on Robotic and Sensors, pp. 1–6 (2007)
17. Hwang, F., King, J., Hale, R., Rokonuzzaman, M.: Discrete event control of mobile robotic systems with potential applications in healthcare. In: Proc. IEEE Ann. EMBS Int. Conf, pp. 431–435 (2000)
18. Looney, C.G.: Fuzzy Petri nets for rule-based decision making. IEEE Trans. Syst. Man Cybern. **18**(1), 178–183 (1988)
19. Murata, T.: Petri nets: properties, analysis and applications. IEEE Proc. **77**(4), 541–580 (1989)
20. Ahson, S.I.: Petri nets models of fuzzy neural networks. IEEE Trans. Syst. Man Cybern. **25**(6), 926–932 (1995)
21. Canales, J.C., Xiaoou, L., Wen, Y.: Fuzzy knowledge learning via adaptive fuzzy Petri net with triangular function model. In: Proc. of the World Congress on Intelligent Control and Automation, 2, vol. 1, pp. 4249–4253 (2006)
22. Nikolos, I.K., Valavanis, K.P., Tsourveloudis, N.C., Kostaras, A.N.: Evolutionary algorithm based off-line 3-D path planner for UAV navigation. Automatika J. **42**(3–4), 143–150 (2001)
23. Kanakakis, V., Tsourveloudis, N.: Evolutionary path planning and navigation of autonomous underwater vehicles. In: Proceedings of the 15th IEEE Mediteranean Conference on Control and Automation, pp. 1–6 (2007)
24. Kanakakis, V., Valavanis, K.P., Tsourveloudis, N.C.: Fuzzy logic based navigation of underwater vehicles. J. Intell. Robot. Syst. **40**, 45–88 (2004). doi:10.1023/B:JINT.0000034340.87020.05