Fuzzy Logic Based Autonomous Parallel Parking of a Car-Like Mobile Robot

Mina Khosh Nejad

A thesis

In

The Department

Of

Mechanical and Industrial Engineering

Presented in Partial Fulfillment of the requirements

For the Degree of Master of Applied Science (Mechanical Engineering) at

Concordia University

Montreal, Quebec, Canada

April 2006

Your file Votre référence
ISBN: 0-494-14309-6
Our file Notre référence
ISBN: 0-494-14309-6

ABSTRACT

Fuzzy Logic Based Autonomous Parallel Parking Of A Car-Like Mobile Robot

Mina Khosh Nejad

Intensive research is carried out in the automotive industry for the development of intelligent automobiles that can make the driving procedure easier and more secure. For automated control of vehicles, autonomous parking problems of the Car-Like-Mobile-Robot have attracted a great deal of attention recently from research organization and automobile industries. The research in car parking problem is derived from the general motion planning problem and is usually defined as: finding a path that connect the initial configuration to the final one with collision free capabilities and by considering the non-holonomic constraints.

This study aims at developing an autonomous parallel parking controller for a Car-Like-Mobile-Robot. Numerous studies have proposed various control strategies on this topic in which the most famous one is the path tracking techniques. But they suffer from major drawbacks as they have to acquire an accurate estimate of the vehicle's state during the maneuver which is difficult to obtain. They also fail in the case the parking space dimensions can not be identified. In this research we have presented a Neuro-Fuzzy control strategy to back drive the car to the parking lot based on the sonar data. We have focused on the case where the parking space dimensions are not known which has not been investigated by the previous approaches. Our controller is able to decide about the

motion direction at each time interval by starting form the initial position and by processing the information obtained by the sensors. On the other hand, in the traditional approaches the path is planned in offline by knowing the parking space dimensions and then will be tracked based on path tracking algorithms. The simulation results demonstrate the effectiveness of the developed algorithm and show that the proposed scheme can be considered as a step towards autonomous intelligent vehicles which can successfully adapt their parking behavior in unknown parking spaces.

Acknowledgment

I would like to thank Dr. K. Demirli for his professional supervision, guidance, and wisdom through out this study.

I also would like to say my special regards to my family in the last two years for their love, support and patience.

Table of Contents

List of Figures

List of Tables

CHAPTER 1: Introduction and Definition of the Problem

One of the ultimate objectives in robotics is to create autonomous robots capable of executing tasks without human intervention. An autonomous robot has to automatically decide which motions to execute in order to accomplish a task. So creating autonomous robots definitively requires the development of motion planning algorithms. The basic motion planning problem is to plan a collision-free path specified by initial and target configuration for a rigid object among static and moving obstacles. An extension of this problem is to consider nonholonomic constraints, that is, nonintegrable equations involving the derivatives of the configuration parameters, which restricts the admissible direction of motions of the robot [1].

Car-like mobile robots have received a great amount of interest in the framework of motion planning for nonholonomic systems. The path planner of a car-like robot has to meet nonholonomic constraints which means; 1) the movement direction must always be tangent to its trajectory. 2) Furthermore, the turning radius is mechanically limited to a minimum value, which is equivalent to say that the vehicle curvature is upper bounded [1].

Intensive research is carried out in automotive industry for the development of intelligent automobiles that can make the driving procedure easier and more secure. For automated control of vehicles, there are 2 kinds of problems: a) driving on road and b) automatic parking. For the first one, a lot of approaches have been developed for the cases like lane

following and lane changing. The second one which is a typical motion control problem is more challenging and has attracted a great deal of attention in recent year from research organizations and automobile industries and has been widely studied. Basically the parking problem can be classified into two different categories: garage-parking problem and parallel parking problem.

1-1. Literature survey

Although some research has been done on the development of parking assistance systems [9], fully automatic parking algorithm has not been investigated yet.

General parking algorithm consists of 3 phases:

1) Building a local map from the parking space, based on the information of the empty slot that can be obtained from sensor data or vision system. So in the first phase, the parking space dimensions should be detected by navigating the vehicle forward.
2) Reaching to a ready-to-reverse position while setting the orientation parallel to the parking space.
3) Planning the motion that brings the car from its initial posture to its goal posture. This motion planning problem for CLMR has been widely investigated and is still an open problem.

Current approaches to solve this problem can be classified into: Path planning approaches and Skill-based approaches:

Path planning approaches

In this approach, a feasible geometry path is planned in advance taking into account the environmental model as well as the nonholonomic constraints, and then control commands are generated to follow the reference path.

For offline path planning in the absence of obstacles, the parking problem can be defined as finding the shortest paths connecting two given initial and final configurations. It has been shown by Dubins [6] that the shortest path for a car-like vehicle is arcs of circle (with minimum turning radius) connected with straight line segments. But the problem about this trajectory is that the curvature profile is discontinuous between arcs and straight lines, so that the robot has to stop at each discontinuity point at line-arc transition to reorient the front wheels. Since instantaneous changes in steering mechanisms are physically impossible, it results in errors of the state of the vehicle at these transition points. To overcome this inconvenience, several authors have proposed continuous-curvature path planners using differential geometric methods. These planners generate clothoids, cubic spirals, B-splines, quintic polynomials, etc., which are then followed by using a path-tracking technique [1].

Trajectory tracking techniques basically consist of designing a time sequence of configuration (orientation together with position) for the vehicle and corresponding reference inputs to motion control system so that the vehicle move along the specified path.

Among path tracking approaches, Jiang and Seneviratre [7] have studied sensor guided autonomous parallel parking, where the process consists of scanning, positioning and

maneuvering. The path in the maneuvering phase is 2 circular arcs of minimum radius tangentially linked to each other and to avoid collision, a forbidden area inside the parking slot is defined.

The authors in [10] address the parking problem of a car-like mobile robot by tracking feasible reference trajectories via a fuzzy sliding-mode control.

Good summaries of these approaches can be found in [7] and [17].

Skill-based approach

Skill-based approaches are usually based on getting feedback from the environment to decide at each step size the next small motion of the robot towards the goal posture.

It is well known that fuzzy logic is dealt with linguistic description of complex systems and it can be utilized to formulate human experiences to control strategies. In skill-based approaches, usually fuzzy logic or neural network is used to acquire and transfer heuristic knowledge of expert driver's skill to an automatic parking controller. There is no reference path to follow and the control commands are usually generated by considering the orientation and position of the vehicle relative to the parking space.

In general, fuzzy logic approaches are trying to emulate human-like qualitative reasoning. The controller provides linguistic (fuzzy) commands to turn left or right and speed up or slow down. Good summaries of these studies can be found out in [11]-[16].

For the skill-based parking in [8], an artificial neural network is trained to directly map the vide sensor's CCD-image of the environment to the corresponding steering angle in the direct neural control architecture. They have also implemented a fuzzy hybrid control

architecture where the controller is configured as a combination of an artificial neural network and fuzzy network.

Authors in [2] have proposed autonomous fuzzy behaviour control by implementing human-like driving skills. They have synthesized four kinds of Fuzzy logic control systems: fuzzy wall following control, fuzzy corner control, fuzzy garage-parking control and fuzzy parallel parking control, to accomplish fuzzy autonomous control.

In the most recent study addressed in [1], an automatic fuzzy behaviour-based controller for diagonal parking is proposed where the fuzzy controller consists of four main modules in charge of four different tasks: Deciding the driving direction to meet the constraints of the parking lot, selecting the speed magnitude, and planning the short paths when driving forward and backward. While the global structure of the fuzzy control system has been obtained by emulating expert knowledge as drivers, the design of the different constituent modules has mixed heuristic and geometric-based knowledge. So some modules have been designed by translating heuristic expert knowledge into fuzzy rules. On the other side, the modules which should perform forward and backward maneuvers are designed base on the fuzzy rules that have been extracted by an identification and tuning process from numerical data corresponding to geometrically optimal paths [1].

Advantages of these fuzzy planners are that their design is simple, rapid, and inexpensive. On the other hand, and contrary to path planning approaches, these typical fuzzy planners do not consider nonholonomic constraints, and they are not optimal in the sense that do not take into account neither minimal-length path nor low control energy requirements [1].

1-2. Description of the problem

Parallel parking is the placement of the vehicle in parallel to its moving direction in a confined space which is wider than its dimension. A four-wheel vehicle is simulated as a rectangular body backing up to the parking space of a rectangular shape.

General parallel parking problem of a non-holonomic vehicle can be defined as finding a path that connects the initial configuration to the final one with collision free capability while taking into account the non-holonomic constraints.

The control systems described in this study do not consider the avoidance of unforeseen obstacles in the parking space but focuses on parking the car on a rectangular parking space. We have assumed that the parking space is not too tight so that one backward maneuver is enough to perform parallel parking and the vehicle does not need to go forward and backward to reach the goal position.

We have also assumed that the vehicle is equipped with ultrasonic sensors and it will navigate forward to its ready-to-reverse position before maneuvering to the parking place. Based on the fact that the parking space dimensions can be identified or not by navigating the vehicle forward, we classified the parking problem into 3 different categories:

Class A- Parking parallel to the curb while the width of the parking space can be detected

In this case, the goal position will vary by different sizes of the parking space width. So the control scheme should be capable of producing an appropriate path to perform a maneuver parallel to the curb. In the proposed control algorithms which will be described in detail, we always assumed a minimum length of the parking space for parking. The strategy for parking lengths bigger than L_{min} is continuing on a straight line after reaching the primary goal position corresponds to this L_{min}. It is clear that the width of the parking space play a significant role in the goal position and on the path. This class can be defined by 2 different cases.

Case 1:

In this case the parking space is defined by the parked car in front and the back as well as the curb. We consider this case as the easiest case for parallel parking, as we can get a local map of the parking space by processing the information obtained by ultrasonic sensors which has been mounted on the car. This case is shown in Figure 1-1.

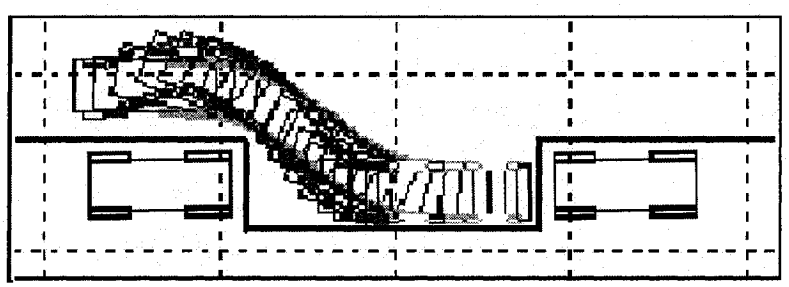

Figure 1-1. Parallel parking to the curb: Case1- ClassA

Case 2:

In this case the parking space is defined by the parked car in front and the curb. The width of the parking space can be identified but as there is no back car we have no idea of the length of the parking bay. This case is shown in Figure 1-2.

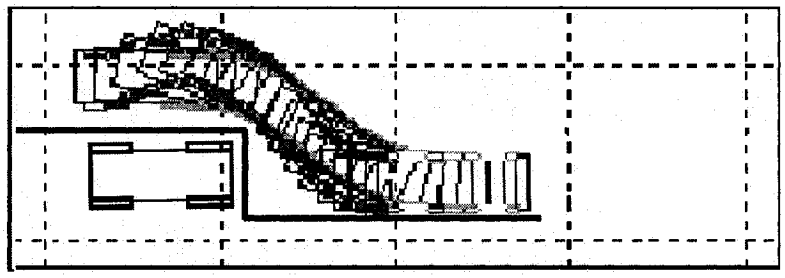

Figure 1-2. Parallel parking to the curb: Case2 - Class A

Class B- Parking parallel to the outside edge of the parked car in front

In this case, we do not need to know the width of the parking space as the strategy is to park parallel to the outside edge of the parking space and not the curb. So there is no need of any intelligent control strategy as the goal position is always fixed irrespective of the width of the parking space. There is just one single path that can be planned just based on the dimension of the car and should then be followed for any size of the parking space dimensions. This case is shown in Figure 1-3.

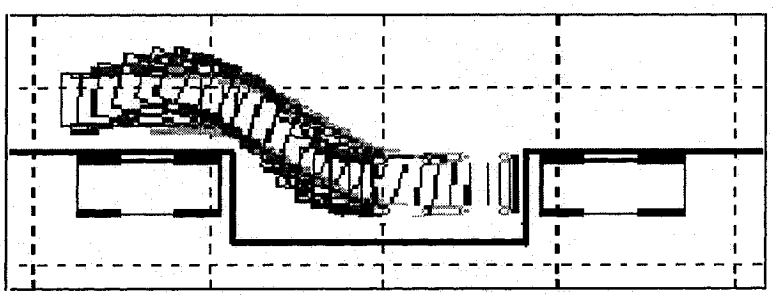

Figure 1-3. Parallel parking to the outside edge: Class B

Class C- Parking parallel to the curb while the width of the parking space can not be detected

In this case the parking space is defined just by the parked car in front. The strategy is to park the car parallel to the inside edge of the parking space. There is no curb and no back car. As there is no curb, the vehicle will not be able to detect the parking space dimensions by navigating forward to its start position. We consider this case as the most difficult case for parallel parking, because we have no idea of the parking space dimensions and so the goal position can not be identified to plan the path. It is clear that if the vehicle will be able to autonomously park in this case, it will be able to park in the previous 2 cases as well. This case has been fully investigated in this study and a control strategy has been proposed to autonomously navigate the vehicle into the parking bay by processing the sonar sensors information at each time interval during the path. So the vehicle will reach the proper goal position directly without knowing the parking area width by processing sensor information reflected off of the stationary car parked in front. This case is shown in Figure 1-4.

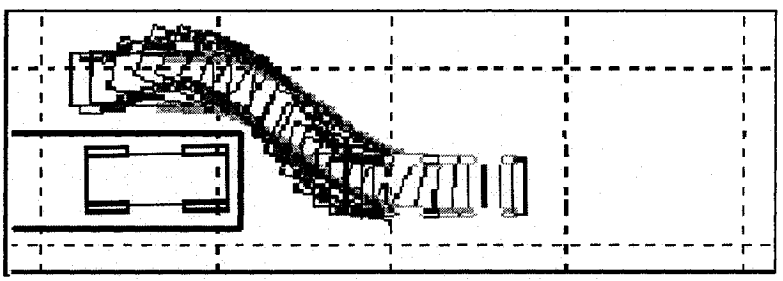

Figure 1-4. Parallel parking to the curb: Class C

1-3. Disadvantages of traditional control methods for parallel parking

Classical control methods for parking problem usually combine an offline path planner with a controller capable to track the nominal trajectory. So the overall approach consists of 2 steps:

1) Path planning: Generate a path offline connecting the desired initial and final configurations by using a nonholonomic path planner.
2) Trajectory tracking: Apply a real-time tracking controller to follow the trajectory

There have been many path planning and trajectory tracking algorithms proposed in the literature in this respect which has been discussed in section 1-2.

There are 2 main disadvantages associated with the traditional approaches:

1) In order to define a path offline, we should be able to detect the parking space dimensions to assign a goal position to our path planner. In class C, where there is no curb we can not detect the width of the parking space and so the goal position is not defined to produce the path for tracking. In general, classical approaches fail in case C.

2) In any path tracking algorithm we would need to have an estimate of the position and orientation of the car during the maneuver, which is usually difficult to obtain. Many path tracking algorithms will rely on linearization of the kinematic equations of the car as well as the wheel encoders mounted on

the motor to estimate the state of the car. But it has been experimentally observed that the vehicle localization errors accumulate over time due to inaccuracies in the kinematic model and precision limitations of the encoders [14].

1-4. An overview of the proposed methodology

This research is aimed at developing an automatic parallel parking controller which is an important task in autonomous ground vehicles. In this dissertation the parallel parking problem has been investigated in the absence of obstacles and the focus is on parking spaces not too tight so that one backward maneuver is enough for performing parallel parking.

The overall architecture of the control system proposed here is an ANFIS structure, which is actually a fuzzy inference system implemented in the framework of adaptive neural network. By using a hybrid learning procedure, the proposed ANFIS can construct an input-output mapping in the form of fuzzy if-then rules based on the stipulated input-output data set.

The path planner described in this study is a Neuro-Fuzzy system that processes information from sensors and execute motion control that provide continuous-curvature paths, thus taking into account the nonholonomic constraints. Our controller does not need to know the desired trajectories to follow, but it finds them, by starting form the

initial configuration of the robot and processing the information obtained by sensors. So they do not track trajectories, but make the robot follow feasible trajectories that will be generated online, which allows reducing the computational cost.

Two types of Neuro-Fuzzy control systems for parallel parking have been proposed. The first architecture which is a behaviour-based approach is based on the acquisition and transfer of an experienced human driver's skills by the use of sensor information to an automatic parking control system and the second one is a sensor-based control architecture which performs parallel parking by just relying on sonar data. The proposed schemes consist of 2 steps:

1) Generation of training data

At first a set of input-output data has been generated based on the two different criteria explained above. In the behaviour-based approach, the input data has been constructed based on expert driver's experiences from sensor information. In the sensor-based approach, the input data consists of direct distance measurements obtained from 3 sonar sensors mounted on the front left corner of the car. The output data in both approaches is the turning angle, which decides motion direction at each time interval.

For the first approach, the training data has been generated by considering just 2 extreme parking dimensions, while in the second approach 3 parking space dimensions have been used for training. So the geometrical paths corresponding to these parking dimensions have been planned in offline and descritized to obtain the input-output data sets.

2) Construction of the ANFIS architecture

Once the training data have been obtained, the rules of sugeno-type fuzzy system have been extracted by applying subtractive clustering on them. Then the fuzzy inference system has been implemented in the framework of adaptive networks for fine tuning the premise and consequent parameters.

The control command (turning angle) is generated in real time according to the input data corresponding to the current state. Compared with other existing subjective fuzzy logic techniques and path tracking methods, our approach will be able to approximate the reference trajectory by starting from the initial configuration and taking into account the nonholonomic constraints.

The goal of this work is two-fold. First we have sought a planner that is exact and yet requires little computational time and can be implemented easily. Second we wish that the proposed strategy can be an efficient step toward fully automatic parallel parking system in the most difficult case (case C) which has not been investigated neither by tracking algorithms, nor skill-based approaches.

CHAPTER 2: Kinematic Equations and Offline Path Planning Algorithm

This chapter deals with the problem of planning constrained motions where the constraints are nonholonomic in nature. A nonholonomic constraint is expressed as a non-intergrable equation involving the derivatives of the configuration parameters. The main consequence of a nonholonomic constraint is that an arbitrary path in the admissible configuration space does not necessarily correspond to a feasible trajectory for the robot. So the adopted trajectory should be collision-free and respect nonholonomic constraints.

In this chapter we start by presenting a model of the Car-Like Mobile Robot (CLMR) and its kinematic equations. Then the nonholonomic constraints which restrict its admissible direction of motion are described. Then we define a 5^{th} order polynomial path which is a smooth and feasible trajectory that would be considered as the nominal trajectory in the simulations.

2-1. Kinematic model of a CLMR

Consider a kinematic model of the CLMR shown in Figure 2-1, where the rear wheels are fixed parallel to car body and allowed to roll or spin but not slip. So they are always tangent to the orientation of the car. The front wheels can turn to left or right, but the left and right front wheels must be parallel. All the corresponding parameters of the CLMR depicted in Figure 2-1 are defined in Table 2-1, as follows [12].

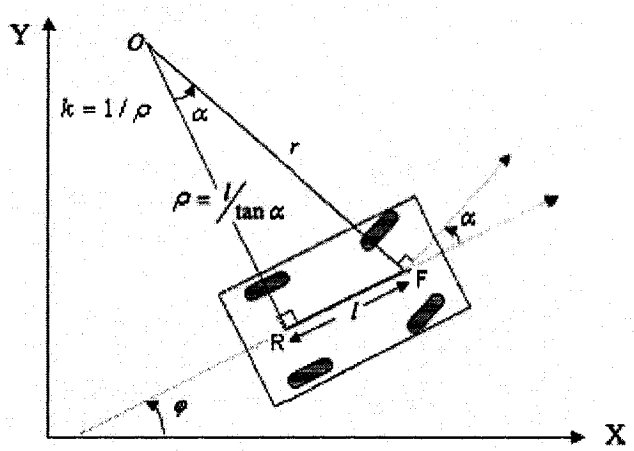

Figure 2-1. Kinematic Model of a car-like mobile Robot

Table 2-1. Parameters of the kinematic model of a CLMR

φ	Orientation of the car with respect to the X axis
α	Turning angle
l	Wheel base of CLMR
O	Center of rotation
r	Distance form point O to the midpoint of the front wheels axle
ρ	Curvature radius
k	Curvature of the fifth-order polynomial

The general expression for curvature of any curve is: $k = d\theta / ds$, where s is the path-length variable and θ is the angle of the tangent to the path. For a path defined in Cartesian coordinates, curvature of $y(x)$ given by [5]:

$$k = (d^2y / dx^2) / [1 + (dy / dx)^2]^{3/2} \qquad (2\text{-}1)$$

The simple kinematic model usually used for CLMR [1]:

$$\begin{aligned} \dot{x} &= v.\cos(\varphi) \\ \dot{y} &= v.\sin(\varphi) \\ \dot{\varphi} &= k.v \end{aligned} \qquad (2\text{-}2)$$

Where x and y are the coordinates of the vehicle rear axle midpoint and v, is the velocity of the vehicle which we assume it to be constant.

The (2-1) and (2-2) have been used to model the vehicle in the simulations.

Once current configuration (x, y, φ, k) is known, the robot has to control the value of its new turning angle in order to drive the car backward into the parking space. Next we discuss nonholonomic constraints which restrict the admissible directions of motion.

2-2. Nonholonomic Constraints

Let ***A*** be a car-like robot moving on a planner workspace. ***A*** has two rear wheels and two directional front wheels. A configuration of ***A*** is defined by a triple (x, y, φ) where (x, y) are the coordinates of the rear axle midpoint R and φ the orientation of ***A,*** i.e. the angle between the x axis and the main axis of ***A*** (Figure 2-2) [3].

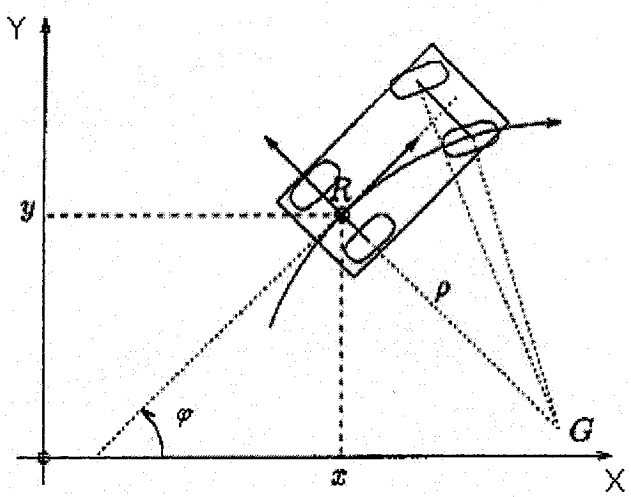

Figure 2-2. A model of a CLMR

A body moving on the plane has only one centre of rotation. Under perfect rolling assumption, a wheel must move in a direction which is normal to its axle. Therefore, when ***A*** is moving, the axles of its wheels intersect at G, the centre of rotation of ***A.*** The orientation of the rear wheels being fixed, G must be located on the rear wheels axle (possibly at an infinite distance) and R must move in a direction which is normal to this axle. In other words, the following constraint holds [3]:

$$-\dot{x}\sin(\varphi) + \dot{y}\cos(\varphi) = 0 \qquad (2\text{-}3)$$

Besides, owing to the fact that the front wheels orientation is mechanically limited, the distance ρ between *R* and *G*, i.e. the curvature radius at point *R*, is lower bounded by a certain value $\rho_{\min}$ and the following constraint holds [3]:

$$x^2 + y^2 - \rho_{\min}^2 .\dot{\varphi}^2 \geq 0 \qquad (2\text{-}4)$$

It can be shown that (2-3) and (2-4) are non-holonomic; the derivative terms cannot be eliminated. Accordingly they restrict the shape of the feasible paths [3].

2-3. Definition of a feasible path

A path defined in *x-y* plane is **feasible** if it respects (2-3) and (2-4); which means if and only if [3]:

a) Its tangent direction is piecewise continuous with opposite semi-tangents at the cusp points (the passage from forward to backward motion occurs at these cusp points. It would be possible to reorient the front wheels at these cusp points where the speed becomes null. So the tangent direction changes its signs at these points.)

b) The curvature at each point of path is less than $1/\rho_{\min}$, where $\rho_{\min}$ is the minimum turning radius of ***A.***

In addition a path is **smooth**, if and only if its tangent direction is continuous [3].

As mentioned earlier, many research work that plan collision free paths for CLMR compute paths made up of straight line segments connected with tangential circular arcs. The primary reason for this is that, it has been shown that the shortest path for a CLMR between two configurations is such a path [6]. The secondary reason is that these paths are easy to deal with from computational point of view [3]. Thus according to the definition of feasible paths, paths made up of straight segments connected with tangential circular arcs of minimum radius are feasible. They can also be smooth if their tangent direction is continuous [3].

As mentioned earlier however, the curvature of this type of path is discontinuous; the discontinuities occurring at the transitions between straight line segments and arcs. Accordingly, when ***A*** follows such a path, it has to stop at each curvature discontinuity in order to reorient its front wheels (an instantaneous reorientation of the wheels is physically impossible). The continuity of the curvature is therefore a desirable property for paths. This leads us to define a **continuous-curvature** path as a path whose curvature profile is continuous [3].

2-4. Reference Trajectory

We adopted a fifth-order polynomial which is a smooth and feasible trajectory such that the CLMR can follow [4]. In fact, several other curves have been used as reference trajectory which the most famous one is straight line segments connected with tangential circular arcs of minimum radius. The problem about this kind of trajectory has been explained in the previous section.

Among polynomial curves it has been shown that a fifth-order polynomial is the least polynomial behaving the parallel parking [4].

The curvature of this trajectory, k, as a function of x, shown in Figure 2-3, confirms that the desired continuous-curvature path is provided and also the vehicle will be able to straighten out its wheels at the maneuver end points. Because of the zero curvature at these points, in the case where vehicle navigates forward to its start position or if the vehicle has to continue on a straight line after reaching to the goal position, there will be no discontinuity in the curvature profile. As can be seen in Figure 2-3, the maximum curvature increases directly with the ratio (y_s / x_s). This ratio must be chosen sufficiently small, so that the resulting continuous steering-function does not violate the peak turning angle and hence the second non-holonomic constraint [4,5].

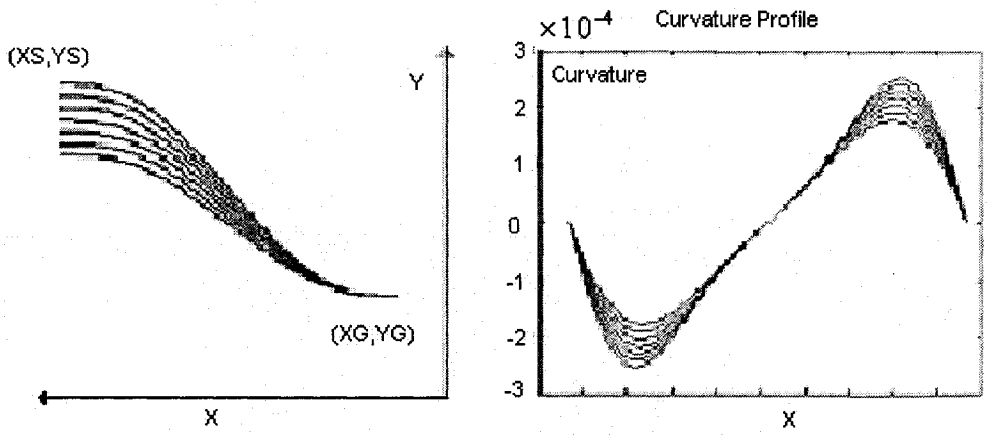

Figure 2-3. Family of fifth-order polynomials and their curvature profile

2-5. Derivation of the fifth-order polynomial path for the mid-point of the rear wheel axle

Constraints on this curve include initial and final positions, orientations and curvatures as well as the non-holonomic constraints [4].

The path of the mid-point of the rear axle is represented as a function: $y = f(x)$. The general form of a fifth-order polynomial is given by:

$$y(x) = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + a_4 x^4 + a_5 x^5 \qquad (2\text{-}5)$$

That is subjected to zero slope and curvature at start and goal configuration,

$$
\begin{aligned}
y(x_s) &= y_s & y(x_g) &= y_g \\
\dot{y}(x_s) &= 0 & \dot{y}(x_g) &= 0 \\
\ddot{y}(x_s) &= 0 & \ddot{y}(x_g) &= 0
\end{aligned}
\tag{2-6}
$$

So the six coefficients in (2-5) can be found to satisfy the position, slope and curvature constraint of this maneuver.

CHAPTER 3: Fuzzy Rule-Based Modeling

Conventional System Modeling tools are not well suited for dealing with ill-defined and uncertain systems. Contrary, a fuzzy inference system which employs fuzzy if-then rules can model the qualitative aspects of human reasoning processes without employing precise quantitative analyses. This is called subjective system modeling.

The objective fuzzy modeling or Fuzzy Identification which has first introduced by Takagi and Sugeno, has found numerous practical applications in control. However one of the basic aspects of this approach is the need for effective methods for tuning the membership functions (MF's) in order to minimize the output error measure of the system [18].

That is why **Adaptive-Network-based Fuzzy Inference System,** or simply **ANFIS,** can serve as a basis for constructing fuzzy rule-base with appropriate membership functions to generate a mapping between input-output data pairs [18].

In this chapter, first the fuzzy if-then rules and fuzzy inference system will be discussed then, the architecture and learning procedure underlying ANFIS is presented. ANFIS is a fuzzy inference system embedded in the framework of adaptive networks. By using a hybrid learning procedure, the proposed ANFIS can construct a mapping between input-output data pairs in the form of fuzzy if-then rules.

3-1. Fuzzy If-Then Rules and Fuzzy Inference Systems

Fuzzy if-then rules are statements in the form of: **IF A THEN B,** where **A** and **B** are, fuzzy sets characterized by appropriate membership functions.

One of the forms of fuzzy if-then rule, proposed by Takagi and Sugeno, has fuzzy sets involved only in the premise part while, the consequent part is described by a polynomial function of the input variables [18].

Fuzzy inference systems which are also known as **fuzzy-rule-based system,** composed of five functional blocks (Figure 3-1) [18]:

1) **Rule Base** which contains a number of fuzzy if-then rules.

2) **Database** which defines the membership functions of the fuzzy sets used in the Fuzzy rules.

3) **Decision-making Unit** which performs the inference operations on the rules.

4) **Fuzzification Interface** which transforms the crisp inputs into degrees of match with linguistic values.

5) **Defuzzification Interface** which transform the fuzzy results of inference into a crisp output.

Usually the rule base and the database are jointly referred to as the **knowledge base.**

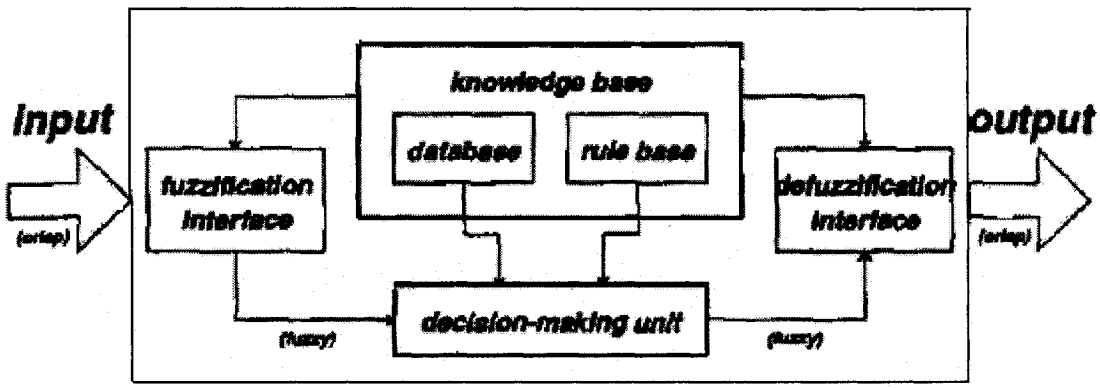

Figure 3-1. Fuzzy Inference Engine

The steps of **fuzzy reasoning** (inference operations upon fuzzy if-then rules) performed by fuzzy inference systems are [18]:

1) Comparing the input variables with the membership functions on the premise part to obtain the membership values (or compatibility measures) of each linguistic label.

2) Combining (through a specific T-norm operator, usually multiplication or min.) the membership values on the premise part to get **firing strength (weight)** of each rule.

3) Generating the qualified consequent (either fuzzy or crisp) of each rule depending on the firing strength.

4) Aggregating the qualified consequents to produce a crisp output.

Several types of fuzzy reasoning have been proposed in the literature. The inference procedure of a two-rule two-input Sugeno fuzzy inference system is shown in Figure 3-3 and the details are explained in section 3-4.

3-2. Adaptive networks

An adaptive network, as the name indicates, is a network structure consisting of a number of nodes connected through directional links. Each node represents a process unit, and the links between nodes specify the casual relationship between the connected nodes. All or parts of the nodes are adaptive, which means the outputs of these nodes depend on modifiable parameters pertaining to these nodes. Each node performs a static node function on its incoming signals to generate a single node output, a node function is a parameterized function with modifiable parameters; by changing these parameters we change the node function as well as the overall behavior of the adaptive network The **learning rule** specifies how these parameters should be updated to minimize a prescribed error measure, which is a mathematical expression that measures the discrepancy between the network's actual output and a desired output. In other words adaptive networks are used for system identification and our task is to find appropriate network architecture and a set of parameters which can best model an unknown target system that is described by a set of input-output data pairs [19].

Conceptually, a feedforward adaptive network is actually a static mapping between its input and output spaces; this mapping may be either a simple linear relationship or a highly nonlinear one, depending on the network structure (node arrangement and connections and so on) and the functionality of each node. The goal is to construct a network for achieving a desired nonlinear mapping that is regulated by a data set consisting of desired input-output pairs of a target system to be modeled. This data set is usually called the **training data set** [19].

3-3. Backpropogation for feed-forward networks

The basic learning rule for adaptive networks is in essence the simple steepest descent method. The central part of this learning rule, concerns how to recursively obtain a gradient vector in which each element is defined as the derivative of an error measure with respect to a parameter. The procedure of finding a gradient vector in a network structure is generally referred to as backpropogation because the gradient vector is calculated in the direction opposite to the flow of the output of each node. Once the gradient is obtained, a number of derivative-based optimization techniques are available for updating the parameters. In particular, if we use the gradient vector in a simple steepest descent method, the resulting learning paradigm is often referred to as the **backpropogation learning rule** [19].

Suppose that a given feed-forward adaptive network in the layered representation has L layers and layer l (l =0, 1,..., L; l =0 represents the input layer) has $N(l)$ nodes. Then the output and function of node i [i =1, ...,$N(l)$] in layer l can be represented as $x_{l,i}$ and $f_{l,i}$ respectively, as shown in Figure 3-2. Since the output of a node depends on the incoming signals and the parameter set of the node, we have the following general expression for the node function $f_{l,i}$ [19]:

$$x_{l,i} = f_{l,i}(x_{l-1,1},...,x_{l-1,N(l-1)},\alpha,\beta,\gamma,...) \qquad (3\text{-}1)$$

Where α , β , γ etc. are the parameters of this node.

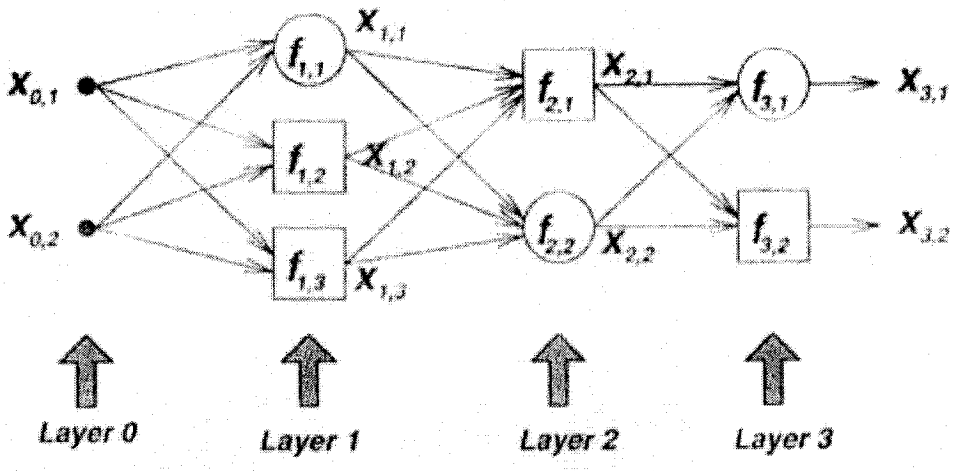

Figure 3-2. Layer representation

Assuming that the given training data set has P entries, we can define an error measure for the pth $(1 \le p \le P)$ entry of the training data set as the sum of squared errors [19]:

$$E_P = \sum_{k=1}^{N(L)} (d_k - x_{L,k})^2 \qquad (3\text{-}2)$$

where d_k is the kth component of the pth desired output vector and $x_{L,k}$ is the kth component of the actual output vector produced by presenting the pth input vector to the network (for notational simplicity, we omit the subscripts p for both d_k and $x_{L,k}$). Obviously, when E_p is equal to zero, the network is able to reproduce exactly the desired output vector in the pth training data pair. Thus the task is to minimize an overall error measure, which is defined as $E = \sum_{p=1}^{P} E_P$ [19].

To use steepest descent to minimize the error measure, first the gradient vector should be obtained.

First, we define the error signal $\varepsilon_{l,i}$ as the derivative of the error measure E_p with respects to the output of node i in layer l :

$$\varepsilon_{l,i} = \frac{\partial^+ E_P}{\partial x_{l,i}} \tag{3-3}$$

For an internal node output $x_{l,i}$, the partial derivative is equal to zero, since E_p does not depend on $x_{l,i}$ directly. However, it is obvious that E_p does depend on $x_{l,i}$ indirectly, since a change in $x_{l,i}$ will propagate through indirect paths to the output layer and thus produce a corresponding change in the value of E_p [19].

The error signal for the *ith* output node (at layer L) can be calculated directly:

$$\varepsilon_{L,i} = \frac{\partial^+ E_P}{\partial x_{L,i}} = \frac{\partial E_P}{\partial x_{L,i}} \tag{3-4}$$

This is equal to $\varepsilon_{L,i} = -2(d_i - x_{L,i})$ if E_p is defined as in equation (3-2). For the internal node at *ith* position of layer l, the error signal can be derived by the chain rule [19]:

$$\varepsilon_{l,i} = \underbrace{\frac{\partial^+ E_P}{\partial x_{l,i}}}_{\substack{\text{error signal} \\ \text{at layer l}}} = \sum_{m=1}^{N(l+1)} \underbrace{\frac{\partial^+ E_P}{\partial x_{l+1,m}}}_{\substack{\text{error signal} \\ \text{at layer l+1}}} \frac{\partial f_{l+1,m}}{\partial x_{l,i}} = \sum_{m=1}^{N(l+1)} \varepsilon_{l+1,m} \frac{\partial f_{l+1,m}}{\partial x_{l,i}} \tag{3-5}$$

Where, $0 \le l \le L-1$. That is, the error signal of an internal node at layer l can be expressed as a linear combination of the error signal of the nodes at layer $l+1$.

Therefore for any l and i $[0 \le l \le L-1$ and $1 \le i \le N(l)]$ we can find $\varepsilon_{l,i} = \frac{\partial^+ E_P}{\partial x_{l,i}}$, by first applying equation (3-4) once to get error signals at the output layer, and then applying equation (3-5) iteratively until we reach the desired layer l. The underlying procedure is called **backpropogation** since the error signals are obtained sequentially from the output layer back to the input layer [19].

The gradient vector is defined as the derivative of the error measure with respect to each parameter, so we have to apply the chain rule again to find the gradient vector. If α is a parameter of the *ith* node at layer l, we have [19]:

$$\frac{\partial^+ E_P}{\partial \alpha} = \frac{\partial^+ E_P}{\partial x_{l,i}} \frac{\partial f_{l,i}}{\partial \alpha} = \varepsilon_{l,i} \frac{\partial f_{l,i}}{\partial \alpha} \tag{3-6}$$

The derivative of the overall error measure E with respect to α is

$$\frac{\partial^+ E}{\partial \alpha} = \sum_{p=1}^{P} \frac{\partial^+ E_P}{\partial \alpha} \tag{3-7}$$

For steepest descent, the update formula for the generic parameter α is:

$$\Delta \alpha = -\eta \frac{\partial^+ E}{\partial \alpha} \tag{3-8}$$

In which η is the learning rate, which can be further expressed as

$$\eta = \frac{k}{\sqrt{\sum_{\alpha} (\frac{\partial E}{\partial \alpha})^2}} \tag{3-9}$$

where k is the step size, the length of each transition along the gradient direction in the parameter space. Usually we can change the step size to vary the speed of convergence [19].

There are two types of learning paradigms that are available to suit the needs for various applications. In **off-line learning,** the update formula for parameter α is based on equation (3-8), and the update action takes place only after the whole training data set has been presented, that is only after each epoch. On the other hand, in on-line learning, the parameters are updated immediately after each input-output data pair has been presented, and the update formula is based on equation (3-6). In practice, it is possible to combine these two learning modes and update the parameter after k training data entries have been presented, where k is between 1 and P, and it is sometimes referred to as the epoch size [19].

3-4. ANFIS: Adaptive Neuro Fuzzy Inference System

ANFIS is a class of adaptive network that is functionaly equivalent to fuzzy inference systems. In what follows, a description of how the parameter set should be decomposed to facilitate the hybrid learning rule for ANFIS architecture , will be given [19].

ANFIS Architecture

For simplicity, we assume that the fuzzy inference system under consideration has two inputs x and y and one output z. For a first-order Sugeno fuzzy model, a common rule set with two fuzzy if-then rules is shown below:

Rule 1: If x is A_1 and y is B_1, then $f_1=p_{11}x+p_{21}y+p_{31}$

Rule 2: If x is A_2 and y is B_2, then $f_2=p_{12}x+p_{22}y+p_{32}$

Figure 3-3, illustrates the reasoning mechanism for this Sugendo model; the corresponding equivalent ANFIS architecture is as shown in Figure 3-4, where the nodes of the same layer have similar functions, as described next. Here we denote the output of the *ith* node in layer l as $O_{l,i}$ [19].

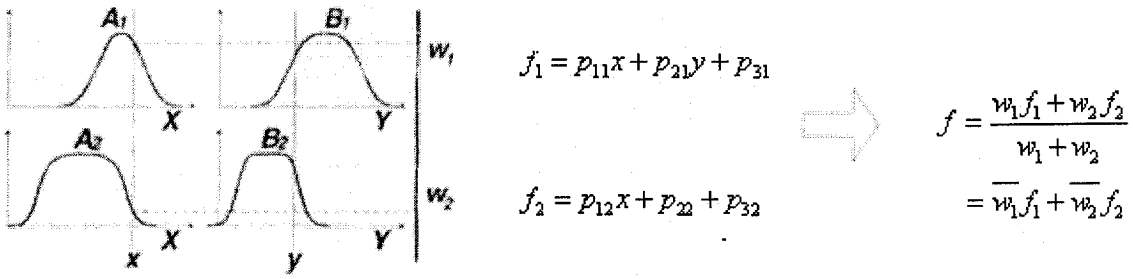

Figure 3-3. A two input, first-order sugeno fuzzy model with 2 rules

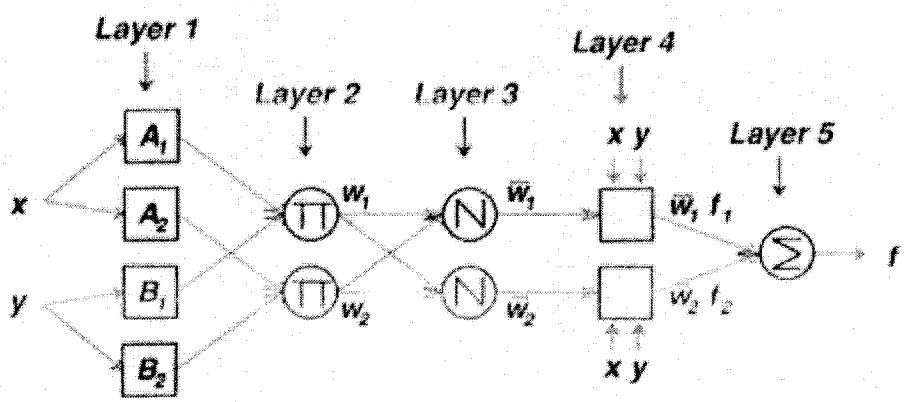

Figure 3-4. Equivalent ANFIS architecture

Layer 1: Every node *i* in this layer is an adaptive node with a node function

$$O_{1,i} = \mu A_i(x), \qquad for\ i = 1, 2$$
$$O_{1,i} = \mu B_{i-2}(x), \qquad for\ i = 3, 4 \tag{3-10}$$

where x (or y) is the input to node i and A_i (or B_{i-2}) is linguistic label associated with this node. In other words, $O_{1,i}$ is the membership grade of a fuzzy set A (=A_1, A_2, B_1 or B_2)and

it specifies the degree to which the given input x (or y) satisfies the quantifier A. Here the membership function for A can be any appropriate parameterized membership function, such as the sigmoidal function:

$$\mu_A(x) = \exp(\frac{-0.5(x-c)^2}{\delta^2}) \qquad (3\text{-}11)$$

where c is the cluster center and δ is the standard deviation. As the values of these parameters change, the sigmoidal function varies accordingly, thus exhibiting various forms of membership functions for fuzzy set A. Parameters in this layer are referred to as **premise parameters**.

Layer 2: Every node in this layer is a fixed node labeled Π, whose output is the product of all the incoming signals:

$$O_{2,i} = w_i = \mu A_i(x)\mu B_i(y), \quad i = 1,2, \qquad (3\text{-}12)$$

Each node output represents the firing strength of a rule. In general, any other T-norm operators that perform fuzzy AND can be used as the node function in this layer.

Layer 3: Every node in this layer is a fixed node labeled N. The *ith* node calculates the ratio of the *ith* rule's firing strength to the sum of all rules' firing strengths:

$$O_{3,i} = \overline{w}_i = \frac{w_i}{w_1 + w_2}, \quad i = 1,2, \tag{3-13}$$

For convenience, outputs of this layer are called normalized firing strengths.

Layer 4: Every node *i* in this layer is an adaptive node with a node function

$$O_{4,i} = \overline{w}_i f_i = \overline{w}_i (p_{1i} x + p_{2i} y + p_{3i}) \tag{3-14}$$

Where $\overline{w}_i$ is a normalized firing strength from layer 3 and $\{p_{1i}, p_{2i}, p_{3i}\}$ is the parameter set of this node. Parameters in this layer are referred to as **consequent parameters**.

Layer 5: The single node in this layer is a fixed node labeled Σ, which computes the overall output as the summation of all incoming signals:

$$\text{overall output} = O_{5,i} = \sum_i w_i f_i = \frac{\sum_i w_i f_i}{\sum_i w_i} \tag{3-15}$$

Thus we have constructed an adaptive network that is functionally equivalent to a Sugeno fuzzy system [19].

3-4. Hybrid Learning Algorithm

Form the ANFIS architecture shown in Figure 3-4, we observe that when the values of the premise parameters are fixed, the overall output can be expressed as a linear combination of the consequent parameters. In symbols, the output f in Figure 3-4 can be rewritten as [19]:

$$\begin{aligned} f &= \frac{w_1}{w_1 + w_2} f_1 + \frac{w_2}{w_1 + w_2} f_2 \\ &= \overline{w}_1(p_{11}x + p_{21}y + p_{31}) + \overline{w}_2(p_{12}x + p_{22}y + p_{32}) \\ &= (\overline{w}_1 x)p_{11} + (\overline{w}_1 y)p_{21} + (\overline{w}_1)p_{31} + (\overline{w}_2 x)p_{12} + (\overline{w}_2 y)p_{22} + (\overline{w}_2)p_{32} \end{aligned} \tag{3-16}$$

which is linear in the consequent parameters: $p_{11}, p_{21}, p_{31}, p_{12}, p_{22}$ and p_{23}. From this observation, we have:

S = set of total parameters,

$S1$= set of premise (nonlinear) parameters,

$S2$ = set of consequent (linear) parameters,

Hybrid learning algorithm consists of a forward and backward path. In the forward pass of the hybrid learning algorithm premise parameters being fixed, node outputs go forward until layer 4 and the consequent parameters are identified by the least-square method. In the backward pass, the error signals propagate backward and the premise parameters are updated by gradient descent. Table 3-1 summarizes the activities in each pass [19].

Table 3-1. Two passes in the hybrid learning procedure for ANFIS

	Forward pass	Backward pass
Premise parameters	Fixed	Gradient descent
Consequent parameters	Least-squares estimator	Fixed
Signals	Node outputs	Error signals

CHAPTER 4: Behaviour-based Neuro-Fuzzy Control Scheme

In this approach the concept of behavior-based control is used to design a Neuro-Fuzzy control scheme to perform parallel parking in a rectangular constrained space with just one backward maneuver. So we will focus on class A, which has been described in section 1-3.

To accomplish the autonomous fuzzy behavior control, the Car-Like Mobile Robot has trained to park in just 2 parking dimensions based on the training data obtained from sensor information generated offline by adopting a fifth-order polynomial as the reference trajectory. The proposed controller is an ANFIS architecture that generates turning angle as output. As long as the states (positions and orientations) of the robot are measurable at each discrete-time step during the control process, this controller can make the robot follow feasible trajectories by knowing the initial configuration of the robot and park successfully at the prescribed goal position. The simulation results, which are based on real dimensions of a typical car, demonstrate the feasibility and effectiveness of the proposed controller in practical car maneuvers.

In this chapter, first the generation of training data which is based on heuristic knowledge of human driver's experience is discussed in section 4-1, then the design of the ANFIS architecture and the control strategy to generate the reference path online is described in section 4-2.

4-1. Generation of training data based on human-like driving skills

According to the drivers' experiences, which describe their parking strategy in terms of linguistic rules, two kinds of information is necessary for performing parallel parking. The first one is the orientation of the car and the second one is an idea of how much we are inside the parking bay. Based on this description of the parking behavior we can derive a 2input-1output fuzzy control scheme to command the steering wheel.

Our methodology is to extract the fuzzy rules from numerical data corresponding to geometrical paths which are obtained based on heuristic knowledge of expert driver's experiences. In order to generate the training data, we assumed that the relative distances between the car and the parking space are measurable by processing the information obtained from ultrasonic sensors mounted on the corners of the car. Figure 4-1, illustrates the distances that have been used for the generation of the input training data.

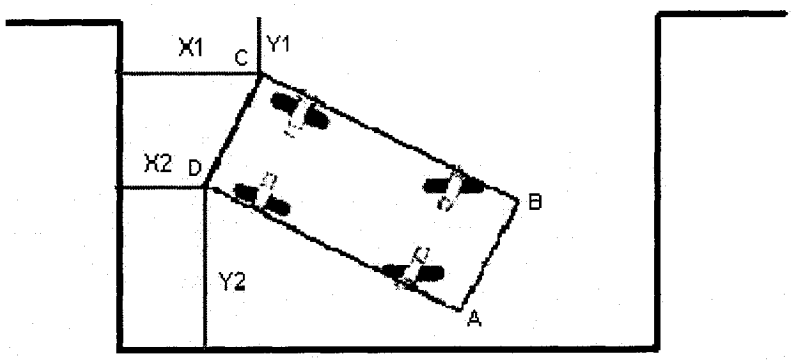

Figure 4-1. Relative distances between the CLMR and the parking space

Where x_1 and x_2 are the distances of the front-right and front-left corners of the car and the parked car in front respectively, y_2 is the distance of the front-left corner of the car and the curb and y_1 is the relative distance of the front-right corner from the open parking area.

Based on the definition of illustrated distances, orientation of the car can be obtained by $x_1 - x_2$, and the amount we go inside the parking space can be expressed as $y_2 - y_1$.

We just rely on information of the front of the car, as it seems back information does not have significant effect in our parking behavior; which can be interpreted as how much you have to turn the steering wheel to accomplish a good maneuver.

So by implementing the input data as (x_1-x_2) and (y_2-y_1) and output data as turning angle, we can derive a 2-input 1-output fuzzy control scheme to describe the motion direction at each sampling time during the maneuver.

The data that has been used for training corresponds to the geometrical paths of parking dimensions depicted in Figure 4-2. The parking space described by a rectangle of dimension W (for width) and L (for length). Also another path in between has been used for checking. The interpolative reasoning of the derived fuzzy rule-based system is demonstrated in simulation results.

It is clear that, positions of the corners can be written in terms of position of the mid-point of the rear axle as well as orientation. The set of equations, which describe these relations are:

$$
\begin{aligned}
X_A &= X_F + (a - l/2).\cos\varphi + b.\sin\varphi \\
Y_A &= Y_F + (a - l/2).\sin\varphi - b.\cos\varphi \\
X_B &= X_F + (a - l/2).\cos\varphi - b.\sin\varphi \\
Y_B &= Y_F + (a - l/2).\sin\varphi + b.\cos\varphi \\
X_C &= X_F - (a + l/2).\cos\varphi - b.\sin\varphi \\
Y_C &= Y_F - (a + l/2).\sin\varphi + b.\cos\varphi \\
X_D &= X_F - (a + l/2).\cos\varphi + b.\sin\varphi \\
Y_D &= Y_F - (a + l/2).\sin\varphi - b.\cos\varphi
\end{aligned}
\qquad (4\text{-}1)
$$

Where *A, B, C* and *D* are the 4 corners of the car (as depicited in Figure 4-1), a and b are half of the length and width of the car respectively and l is the wheel base.

As long as corner's positions are identified, the input data pairs can be constructed easily for simulation. Each path has been sampled to 20 segments so the training data is a 40×3 matrix, which contains all the discrete information.

Dimension of a typical car; length=474 cm, width=178 cm, wheel base=267 cm (Mazda 6), has been used in simulation to show the ability of the controller in real world applications.

In our approach the parking behavior is independent of the distance of the back of the car and the right wall. A typical behavior is shown in Figure 4-3.

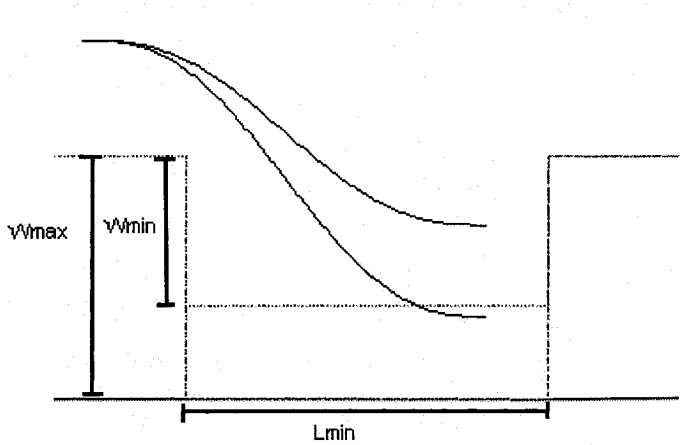

Figure 4-2. Paths that have been used for training

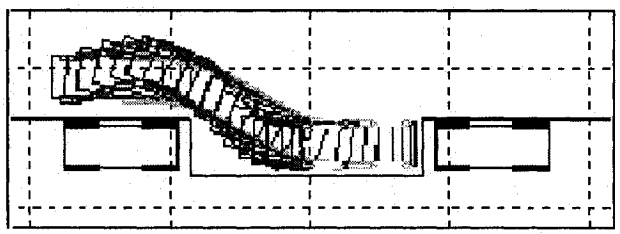

Figure 4-3. A typical behavior

It is clear that the size of the parking space has significant effect on our parking behavior. In practice, the narrower the parking space, the more forward and backward maneuver is necessary to perform parallel parking. But as we have just considered parking behaviors in ordinary dimensions not tight ones, W_{min} and L_{min} are set to 230 cm and 800 cm respectively so that the car can be parked with a single backward maneuver. The minimum size of the parking space that has been considered is 1.687 times the length and 1.292 times the width of the vehicle. We assumed W_{max} and L_{max} to be 370 cm and 1100 cm respectively. Also checking data corresponds to W=300 cm and L=800 cm, which has been obtained in a similar manner has been used for model validation. The input1, input2 and output of training and checking data are shown in Figure 4-4, Figure 4-5, and Figure 4-6, respectively.

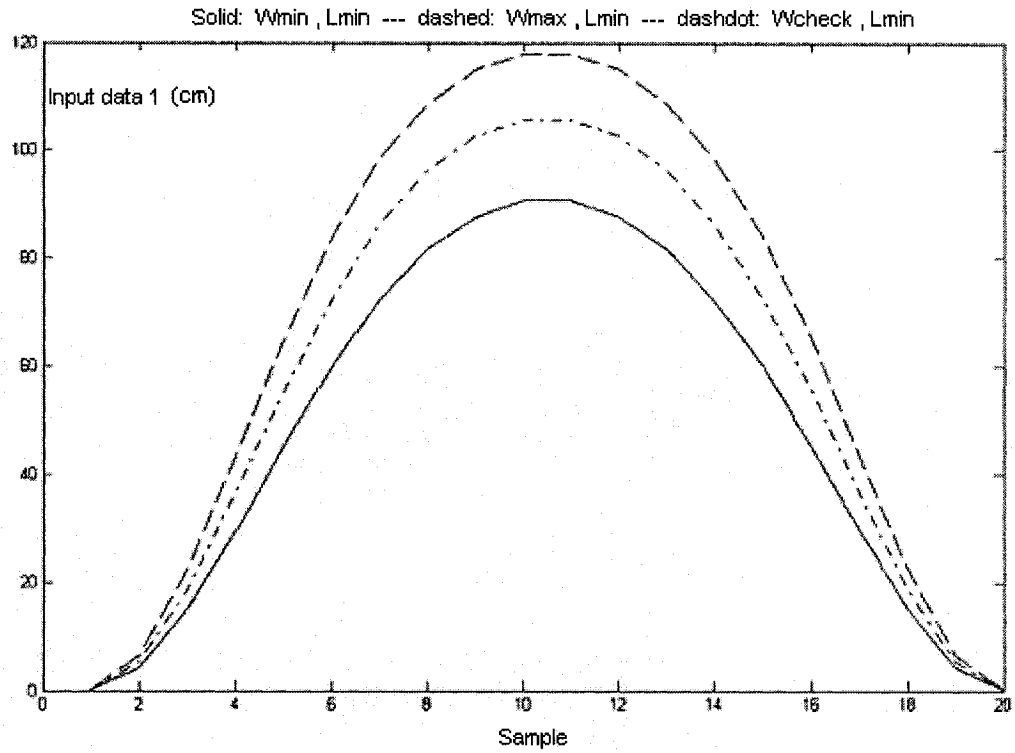

Figure 4-4. First Input data versus sampling point

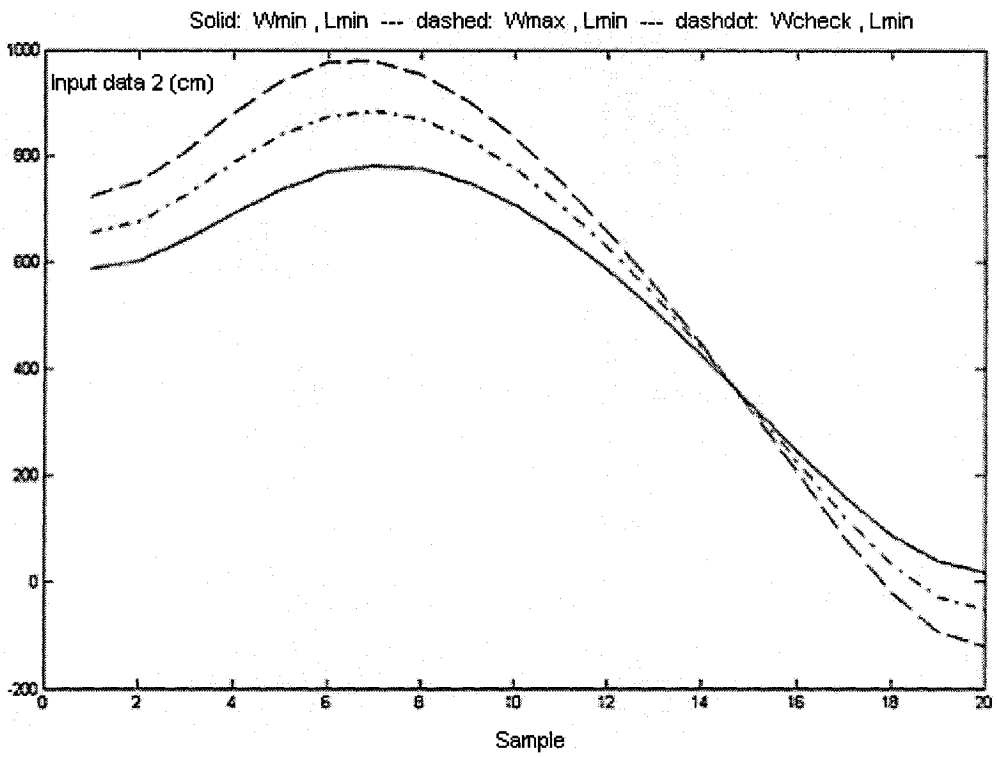

Figure 4-5. Second input data versus sampling point

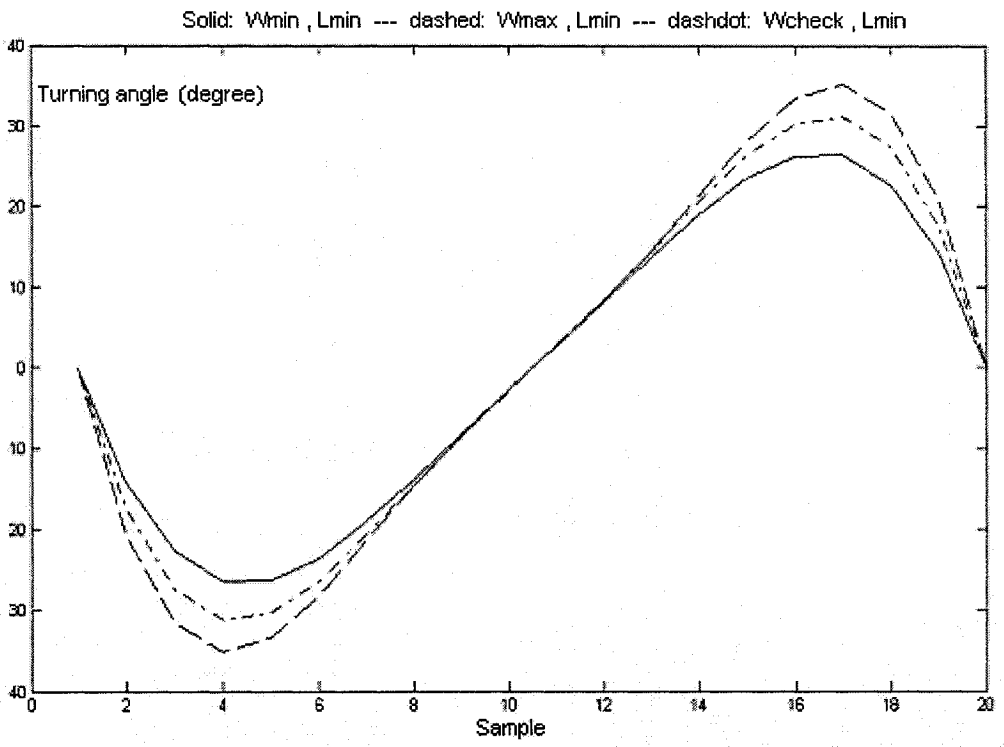

Figure 4-6. Output data versus sampling point

4-2. ANFIS Design and the control scheme

The main theme of implementing this Neuro-Fuzzy controller is to make the robot generate a fifth-order polynomial trajectory online, from a known initial configuration. As mentioned earlier, the controller is in charge of generating the turning angle by taking the distances measured by the sensors as inputs.

The overall control methodology is depicited in Figure 4-7. Once the robot knows its current configuration at each sampling time, the inputs can be calculated directly and be applied to the controller. After the proper turning angle has been generated by the use of

ANFIS controller, the configuration of the next time interval can be found out by considering the kinematics equations.

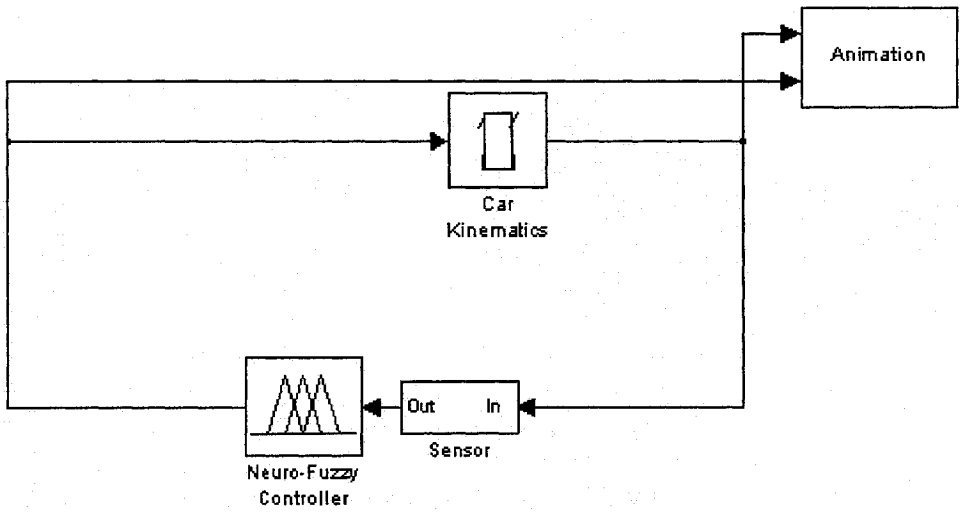

Figure 4-7. Overall control scheme

Our approach has been to generate two of the nominal trajectories to obtain a set of training data. Once the training data have been obtained, the rules of the sugeno-type fuzzy system have been extracted by applying subtractive clustering on these numerical data. Then the fuzzy inference system has been implemented in the framework of adaptive neural networks for fine tuning the premise and consequent parameters.

Our approach is different from the previous traditional path planners, as the robot does not need to know the desired polynomial path to track, but it will approximate such a path by starting from the known initial configuration and processing the sensor information.

The clustering parameters that have been used for extracting the rules are: radii=0.5, squash factor=1.5, accept ratio=0.5 and reject ratio=0.15, which results in 7 fuzzy rules.

The total number of parameters in the neural network structure is 49 which consist of 21 consequent parameters and 28 premise parameters. These parameters which are the cluster centers and standard deviations of the extracted sigmoidal membership as well as the linear parameters associated with the consequents of fuzzy rules are given in Table A-1, in the Appendix. The number of nodes associated with this Neuro-Fuzzy structure is 47. The membership functions of the input variables, the neural network structure and the corresponding fuzzy inference system are illustrated in Figure 4-8, Figure 4-9 and Figure 4-10, respectively.

The general Hybrid Learning Algorithm has been used to update premise and consequent parameters. Backpropogation which used for modifying the premise parameters has implemented the Steepest Descend to minimize the gradient vector of the Error Measure. Least Square Estimator has been used to update the consequent parameters.

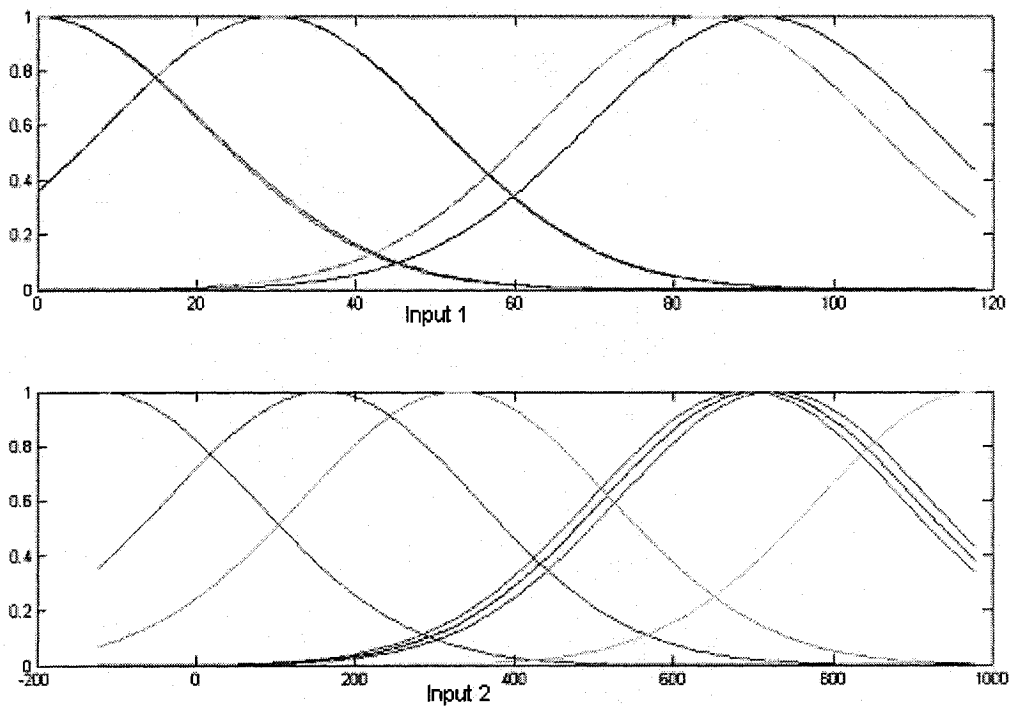

Figure 4-8. Membership functions of the input variables

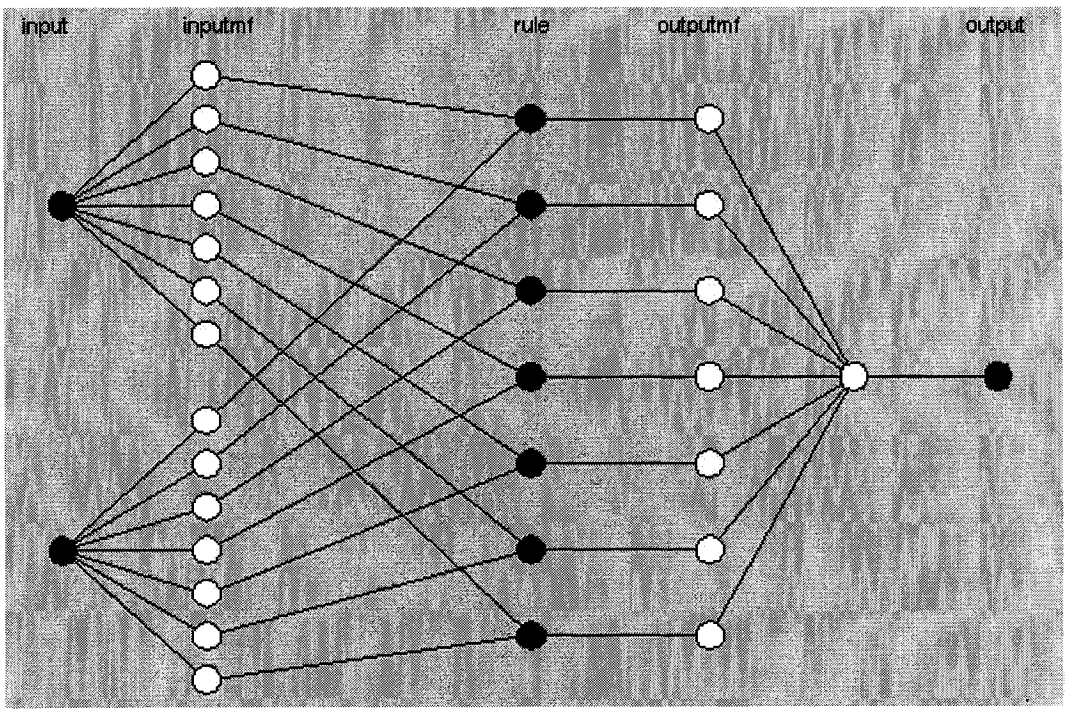

Figure 4-9. Neural network structure

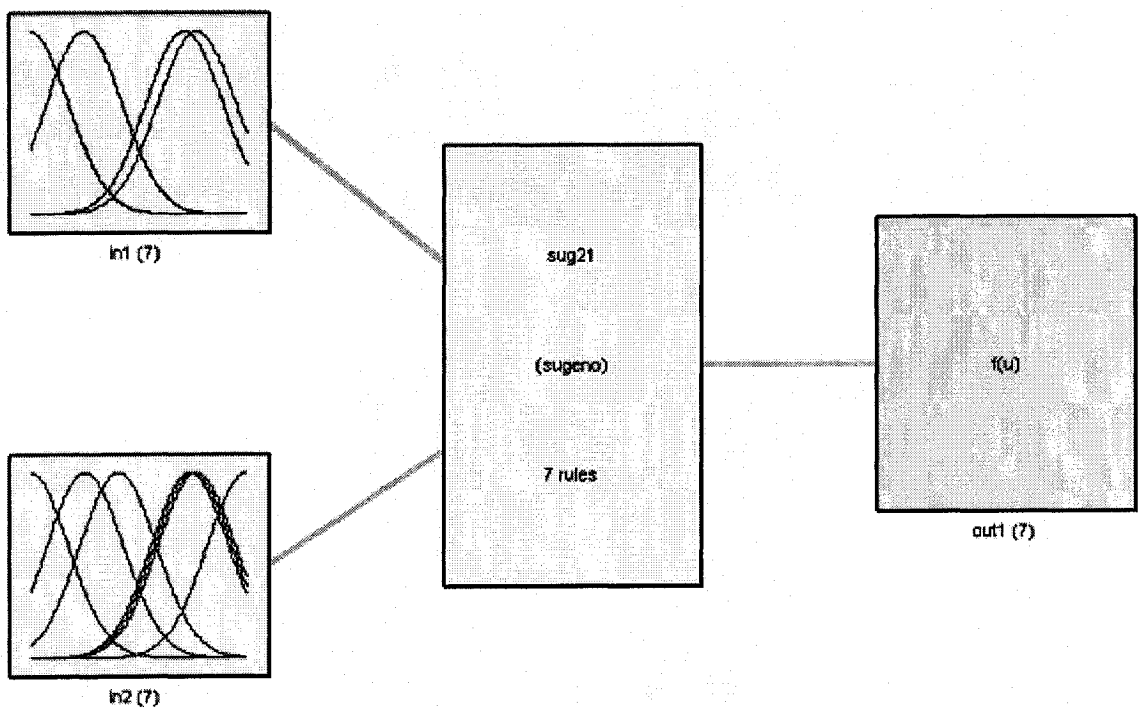

Figure 4-10. Sugeno Fuzzy Inference System

The mean square error representing checking error signal up to 40 epochs is depicted in Figure 4-11. This mean square error basically represents the mean square of the error which is the difference between actual checking data output and the output of the controller after the checking input data has been applied to it. The actual checking data output versus the output of the ANFIS architecture corresponds to the input checking data is shown in Figure 4-13, which illustrates the generalization ability of the Neuro-Fuzzy controller to produce proper steering command for the parking dimensions it has not been trained. The inference procedure of the fuzzy rule-base system is depicited in Figure 4-12. Surface plot is also depicted in Figure 4-14, which shows the smoothness of turning angle profile.

The interpolative reasoning of the fuzzy rule-base system to generate the nominal trajectories for the parking dimensions (corresponding to W=230, 260, 290, 330 and 360 cm) that have not been used in training are shown in Appendix as Figure A-1 to Figure A-5. The maximum position error corresponds to these parking space dimensions is given in Table 4-1, which shows the maximum vertical distance difference between the generated trajectory planned online and the ideal desired trajectory planned offline. As can be seen the maximum position error is below 10 cm which shows a good performance for the proposed methodology.

Table 4-1. Maximum vertical position error (cm)

W=230 cm	W=260 cm	W=290 cm	W=330 cm	W=360 cm
6.7	7.8	8.5	9.4	8.3

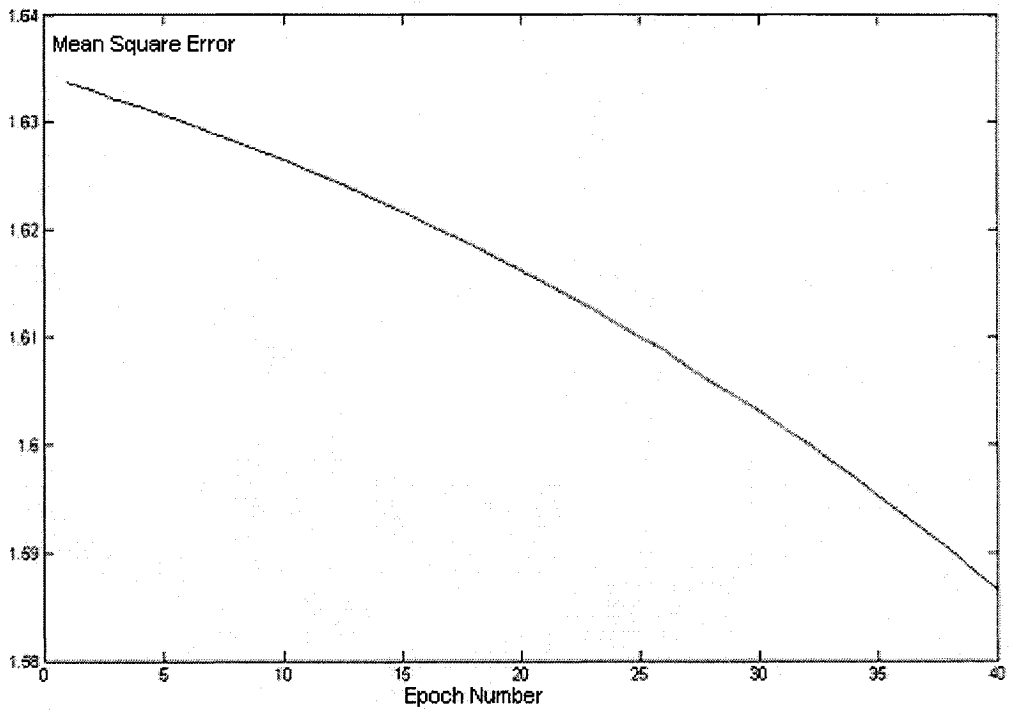

Figure 4-11. Checking error versus epoch number

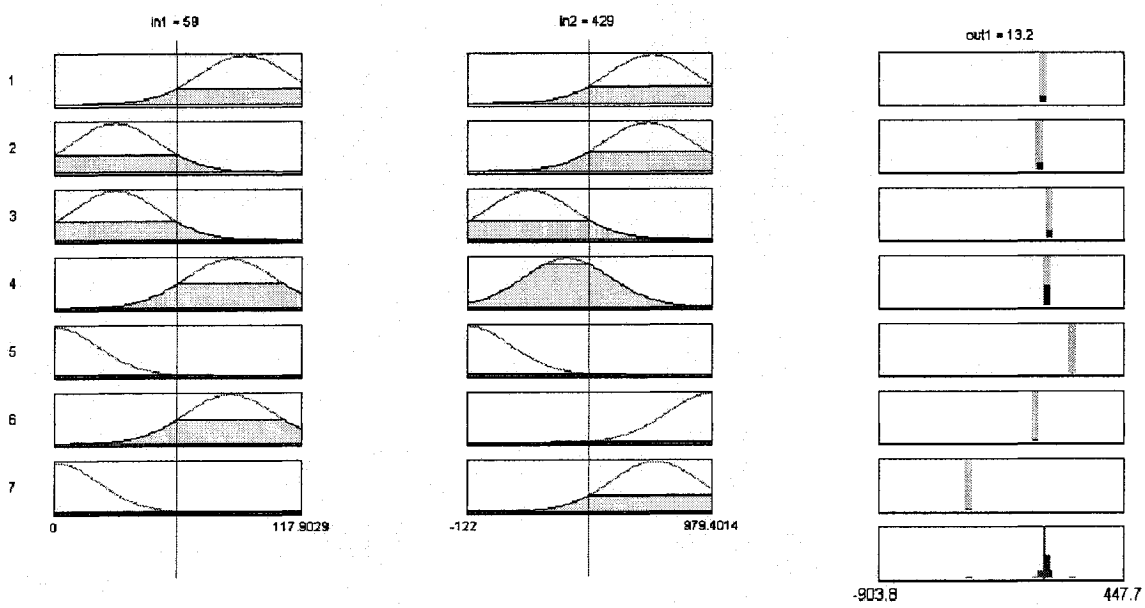

Figure 4-12. Inference procedure of the fuzzy rule-base system

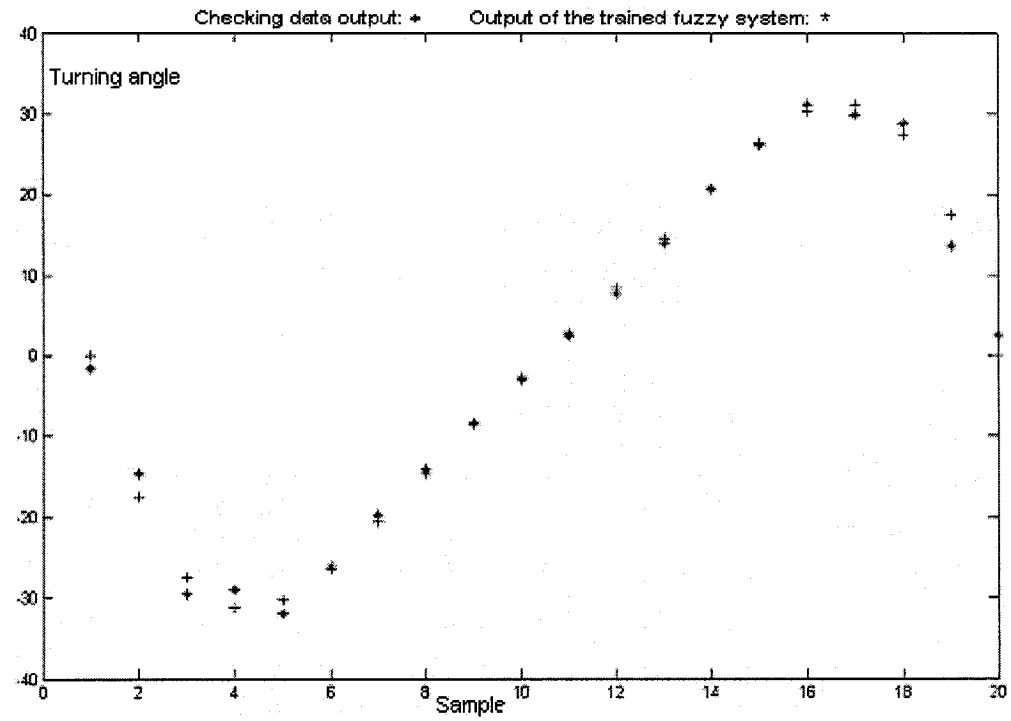

Figure 4-13. Actual checking data output versus output of the controller

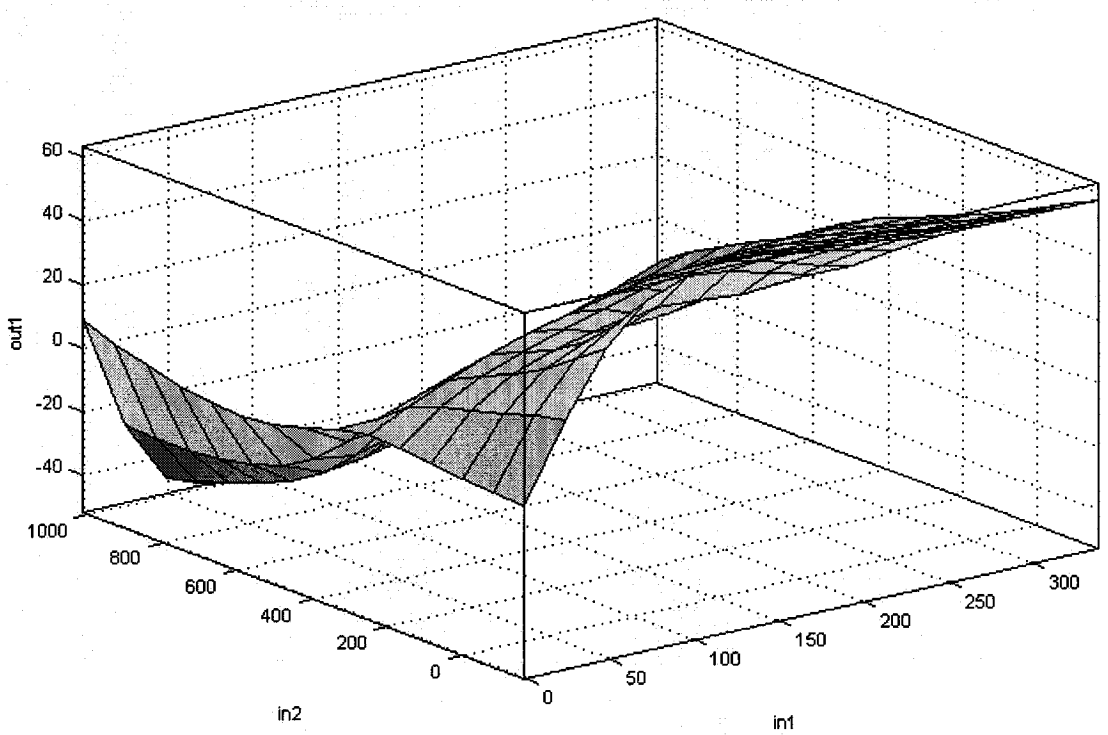

Figure 4-14. Inputs versus turning angle

4-3. Advantages and disadvantages

Advantages:

The main advantage of this approach compared to the other existing path tracking approaches which has been discussed in the literature survey is that the robot does not need to know the desired polynomial path to track, but it will generate such a path online, by starting from a known initial configuration and processing the data which has been acquired based on the expert drivers experiences through sensor measurements. So by applying the input data to the controller, the proper steering angle will be generated online which shows the motion direction at each time interval during the maneuver. On the other hand, in the path tracking algorithms which has been discussed in [7] and [17], first the ideal path should be planned offline and then be tracked. So basically these approaches are not computationally efficient compared to the proposed methodology which does not need any offline path planning.

Disadvantages:

The main disadvantage is the fact that the distances used as input signals to the controller can not be found out directly by the ultrasound sensors. Although they can be expressed in terms of the position and orientation of the car, which enable us to construct them at each time interval during the maneuver, the problem mentioned in the traditional methods about obtaining the accurate position and orientation at each time interval still exists. So the lack of accurate state information is a major drawback of the proposed scheme. This drawback is overcome by the approach presented in the next chapter.

CHAPTER 5: Sensor-based Neuro-Fuzzy Control Scheme

This approach is a Neuro-Fuzzy control system for autonomous sensor-based parallel parking in an unknown parking space. In this chapter we focus on case C, which will automatically cover cases A and B as discussed in chapter 1. One of the advantages of this control scheme in comparison with the previous behaviour-based approach is behaviour fusion based on direct sonar sensor measurement during the maneuver. Another main advantage is its ability of autonomous parking when parking space dimensions can not be identified.

For the sensor-based navigation scheme presented here, the 4 dimensional input space consists of three range measurements obtained from the three sonar sensors that have been mounted at the front left corner of the car, plus the calculated discrepancy between the current measured distance and the previous measured one. The input data have been implemented to generate the control output signal which is turning angle.

In order to carry out sensor-based maneuvers, we have generated a set of 4 input-1 output training data set by recording the sonar sensor measurement during the maneuver to the parking space by simulating their behaviour. We have considered 3 known parking widths for training. Three polynomial paths corresponding to these three widths have been generated with the offline path planner that has been described in section 2-5. The 3 produced paths have been sampled and the training data have been generated by simulating the sonar sensor behaviour. The sonar sensor behaviour has been described in

detail in section 5-1, 5-2 and 5-3. The generation of training data has been presented in section 5-4. After the training data has been obtained, a neuro-fuzzy structure has been built by applying subtractive clustering on the input data. Construction of the ANFIS structure is illustrated in detail in section 5-5.

The simulation results for unknown parking spaces show that our controller is able to decide the motion direction online, based on the sonar readings at each time interval on the path, which serve as inputs.

The results demonstrate the effectiveness of the developed algorithm to park the vehicle into the parking lot without knowing the parking space dimensions.

5-1. Sonar Sensors

Ultrasonic sensors or sonar sensors which also called TOF ranging systems, measure the traveling time needed for a pulse of emitted energy to detect a reflecting surface and then return back to the receiver. The distance is obtained from the measured traveling time (round trip time) and the velocity of energy wave. Nowadays many researchers are using sonar sensors on their mobile robots.

In the automobile industry, there is a vast application of sonar sensors for parking aid applications. They are especially useful when there is poor front and rear visibility as they can detect still and moving objects in blind area of the vehicle. They are usually installed at the front and rear bumper for obstacle avoidance purposes. Their large automotive application is generally due to the low cost of these sensors, and the ease with which the sonar data can be processed directly to provide range information. However,

ultrasonic range measurements suffer from some drawbacks which are described in detail below.

Most sonar ranging systems currently employ a single acoustic transducer that acts as both a transmitter and receiver. After the transmitted pulse hits an object, an echo is detected by the same transducer acting as a receiver. An example of a typical echo is shown in Figure 5-1. A conventional Time-Of-Flight (TOF) system produces a range value when the echo amplitude waveform first exceeds a threshold level. This is shown to occur at time t_0 in Figure 5-1. A range measurement R_0 is obtained from the roundtrip time-of-flight by the following formula [21]:

$$R_0 = ct_0 / 2 \qquad (5\text{-}1)$$

Where c is the speed of sound in air.

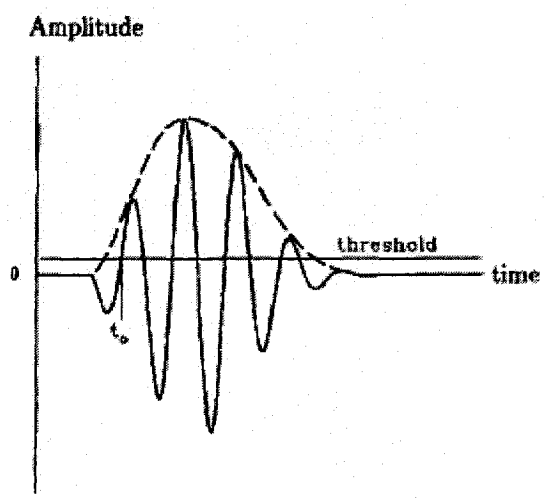

Figure 5-1. Typical echo observed in an ultrasound ranging system

The echo amplitude decreases as the transducer orientation deviates from normal incidence. This deviation is measured by the *inclination angle* as shown in Figure 5-2.

Inclination Angle (or incidence angle) is the angle from the transducer line-of-sight to the normal of the surface. When the magnitude of the inclination angle is greater than a critical angle, which is a function of operating frequency and surface roughness the echo amplitude falls below the threshold level and so no TOF measurement is produced. It can be shown that the threshold level determines the effective angular beam width of the transducer [21].

If a sonar reading returns while the incidence angle is more that the critical angle of the surface, this means that the reflected pulse may not be detected or it will be detected after being bounced off of some objects in the environment. This phenomenon is called *false reflection.*

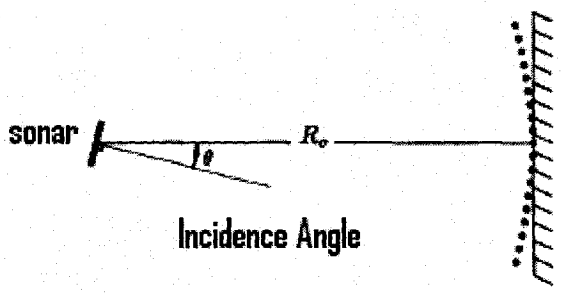

Figure 5-2. Definition of Incidence Angle

5-2. Beam Pattern and Radial Imprecision

When the radius of the transmitting aperture is much larger that the acoustic wavelength, the radiation forms a beam as shown in Figure 5-3, in which the acoustic energy has a beam pattern.

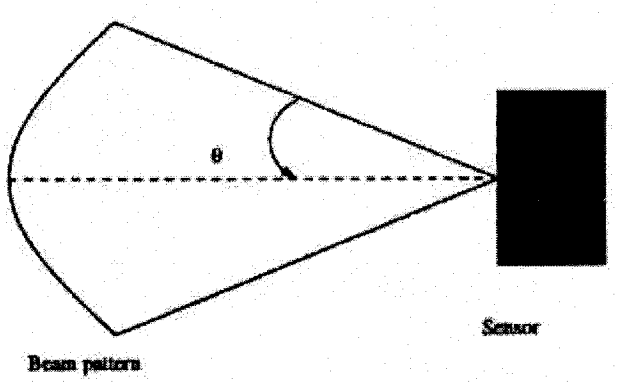

Figure 5-3. Beam pattern

The presence of the beam width adds *radial Imprecision* on the distances measured by the sonar sensors. It results from the fact that the beam is reflected from the portion of the target closest to the sensors [22].

In Figure 5-4, where the incidence angle is θ , the expected reading is |OB|, but due to the fact that the beam is reflected from the portion of the target closet to the sensor, the reading obtained by the sensor is |OA|. The difference between the expected reading and the measured one is called *radial imprecision* [22].

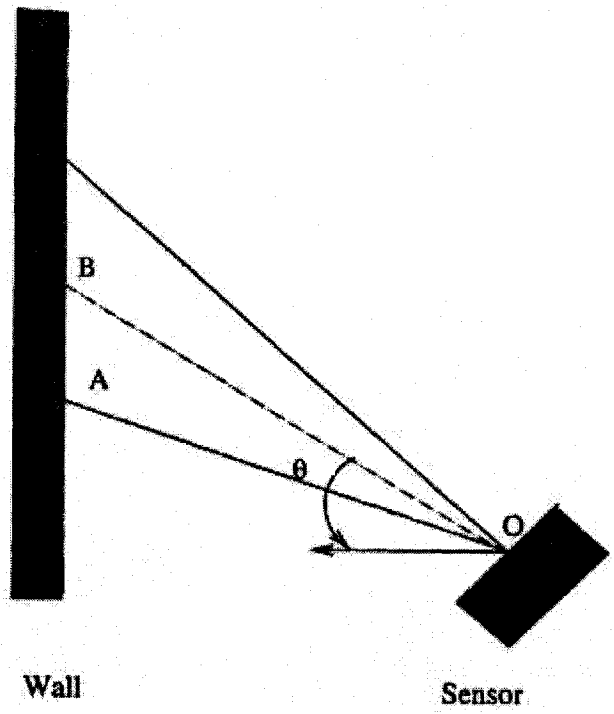

Figure 5-4. The effect of the beam width on sonar reading

5-3. Sonar specification used in simulations

The sonar sensor used in our simulations has a beam width of ± 30 : as mentioned, for incidence angles up to 30 degree the sonar will obtain a reading which is due to the reflection from the portion of the target closest to the sensor. When the sensor is not be able to measure the distance, which is the case where the incidence angle is higher than 30, the sensor will show a large value.

Although the traditionally assumed beam width is around ± 22, the new piezo transducers have much greater beam width. For example, piezo transducers produced by Senscomp

Corp have a variety of beam angle from 10 to 125. So the 60 degree view angle which is used in simulations is realistic.

The key elements of a piezo transducer are crystal or ceramic piezo material bounded to a metal case or cone. As this piezo material is excited by an electric potential, the physical size of the piezo material expands or contracts. The moving can or cone displaces air and generates a burst of ultrasonic sound.

5-4. Generation of Training Data

The key idea of the proposed scheme is to construct a mapping between sonar sensor measurements and the turning angle based on a training data set in order to plan and carry out sensor-based maneuvers. So the first step is the generation of the training data set.

The input-output data pairs have been obtained by adopting a 5^{th} order polynomial trajectory and discretizing it uniformly into 100 sampling points. The offline generation of the adopted nominal trajectory which is based on knowing the initial and goal positions has been described in detail in section 2-5. The training data set has been generated of the three geometrical paths corresponding to W_{min} , W_{max} and another path in between. These 3 paths that has been used for training is depicited by red, green and blue respectively in Figure 5-5.

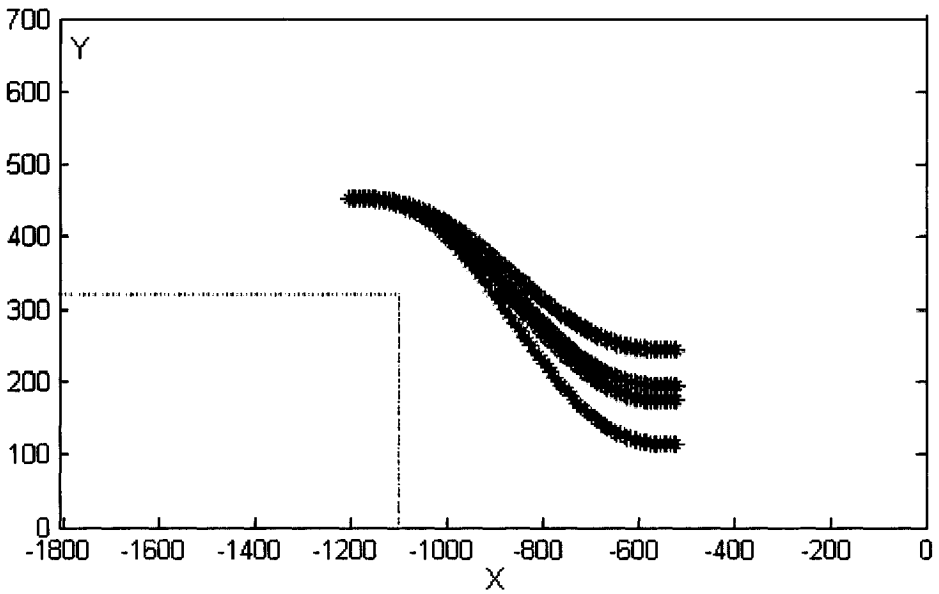

Figure 5-5. Paths used for training: (Red,Blue,Green)--Path used for checking: Black

We put 3 ultrasonic sensors (*S1, S2* and *S3*) at the front left corner of the car at the directions specified in the Figure 5-6, to measure distances of the car and the parked car in front while the car is following the three geometrical paths. The sensor measurements simulated based on sonar sensor behaviours that have been described in section 5-1, 5-2 and 5-3. Bumper shape of the parked car has also been modeled in order to get continuous reading during the maneuver without blind spot.

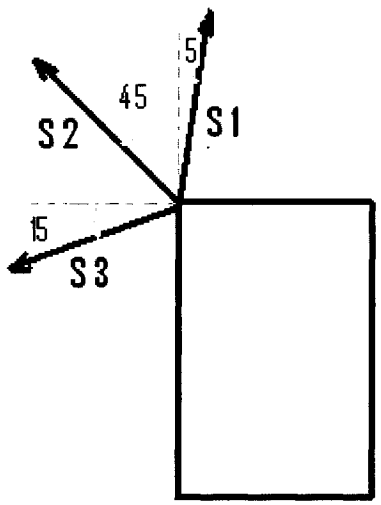

Figure 5-6. Sensor's directions

Bumper shape

The 3 sensors which have been mounted on the front left corner of the car are supposed to measure the distances of the car and the parked car in front. In order to have continuous readings from these 3 sensors, we have modeled the bumper curves of the corners of the parked car in front with a 10^{th} order polynomial as follows:

$$y(x)=0.000000000008x^{10}-0.000000002188x^{9}+0.000000274093x^{8}-0.000019687336x^{7} +0.000894331892x^{6}-0.026715168495x^{5}+0.528041009626x^{4}-6.757473267485x^{3}+ 52.789660973707x^{2}-220.254219988371x+48.995876111479 \quad (5\text{-}2)$$

The polynomial equation which describes the curves of the corners of the parked car in front has been derived from a top view of a car as shown in Figure 5-7.

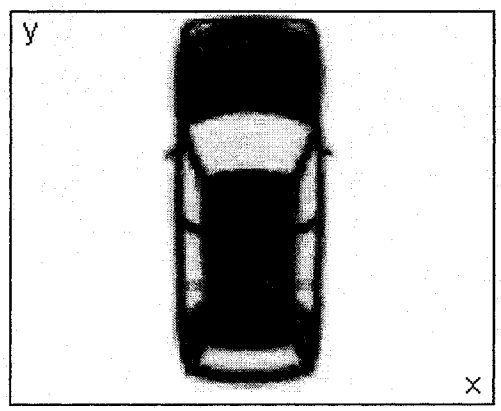

Figure 5-7. Top view of a real car

By fitting the polynomial to show the corner curves, we have simulated the car as shown in Figure 5-8.

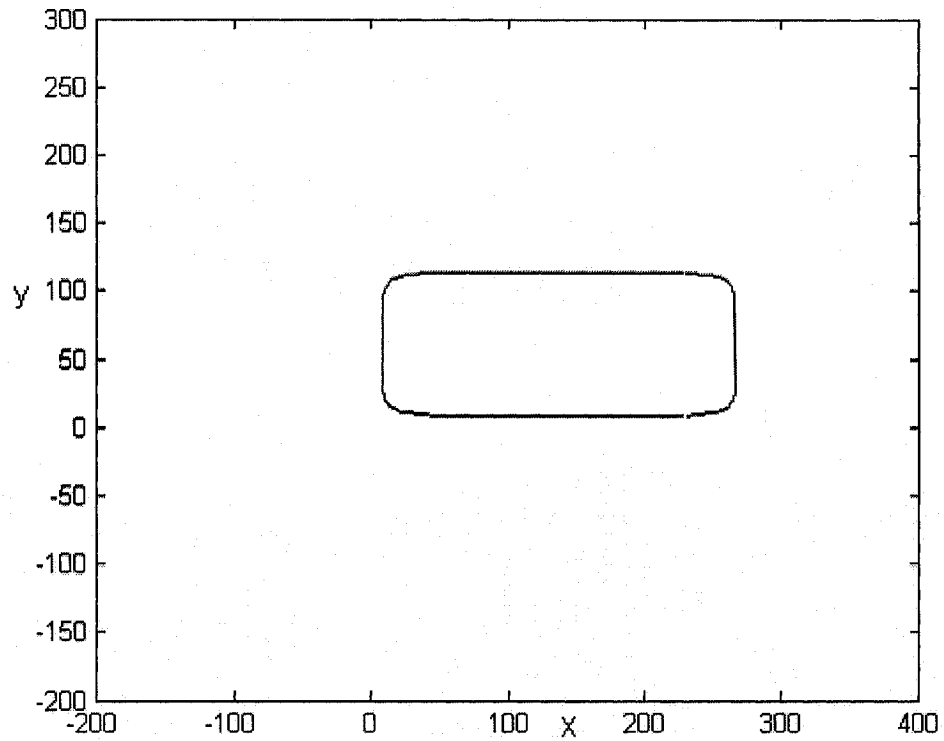

Figure 5-8. Model of a car with bumper shape

Sensor measurements

Distance measurements of the sensors are simulated based on the principles of sonar behviour that has been described in detail in section 5-1, 5-2 and 5-3.

The range measurements acquired by *S3, S2,* and *S1* during the maneuver corresponding to W=260 cm are shown in Figure 5-9, Figure 5-11 and Figure 5-13 respectively. The recorded distance by these sensors from the parked car in front are depicited in Figure 5-10, Figure 5-12 and Figure 5-14 respectively.

As can be seen in Figure 5-15, the three sensors together can fully cover the range information necessary for the car's navigation and there is no blind spot during the maneuver. The total distance measurements acquired by the combination of 3 sensor readings is illustrated in Figure 5-16.

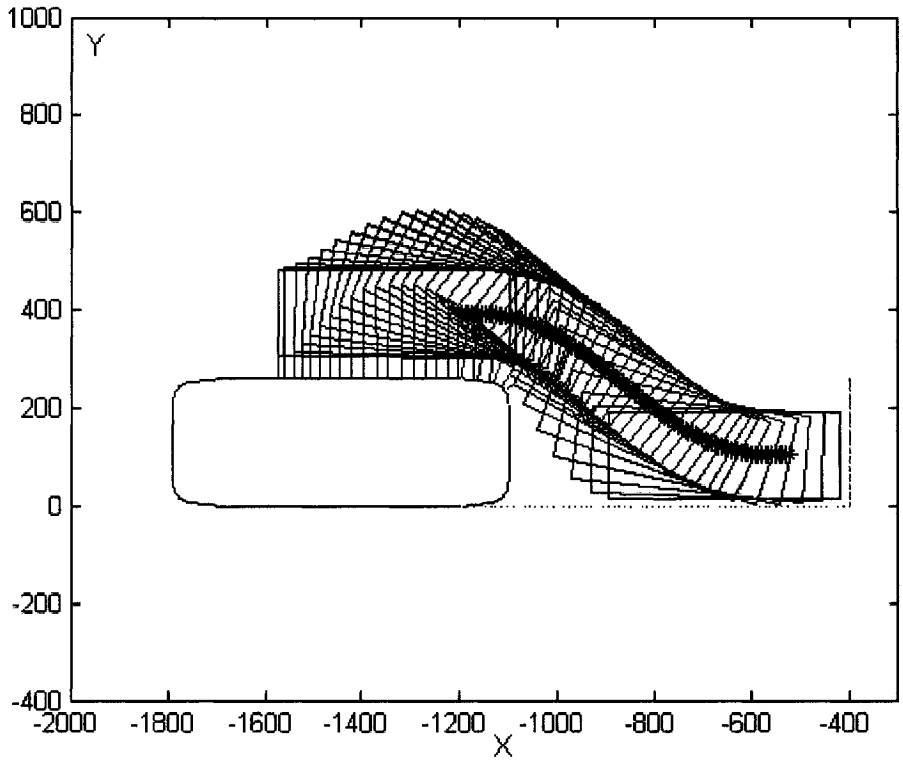

Figure 5-9. Sensor measurements acquired by S3

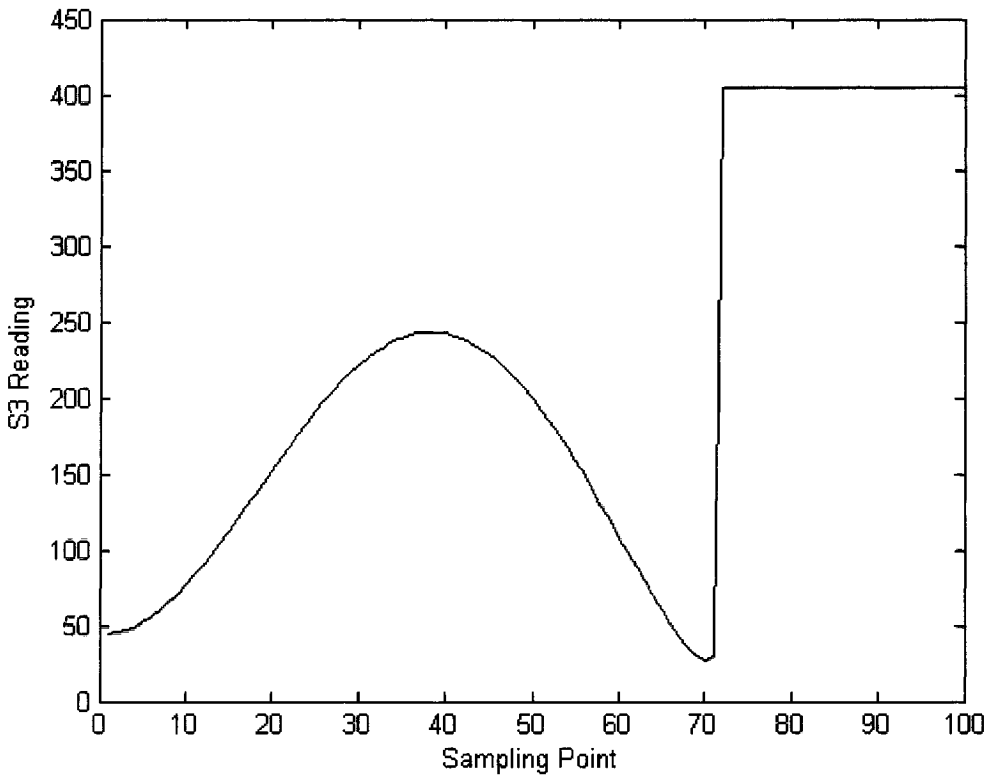

Figure 5-10. Distance measurements of S3 versus sampling Point

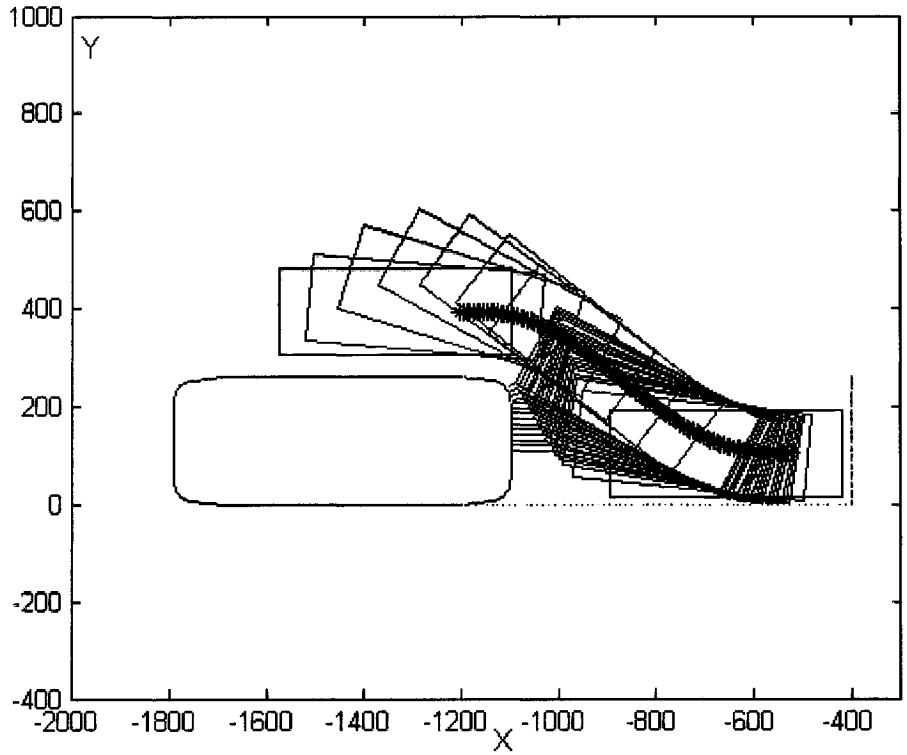

Figure 5-11. Sensor measurements acquired by S2

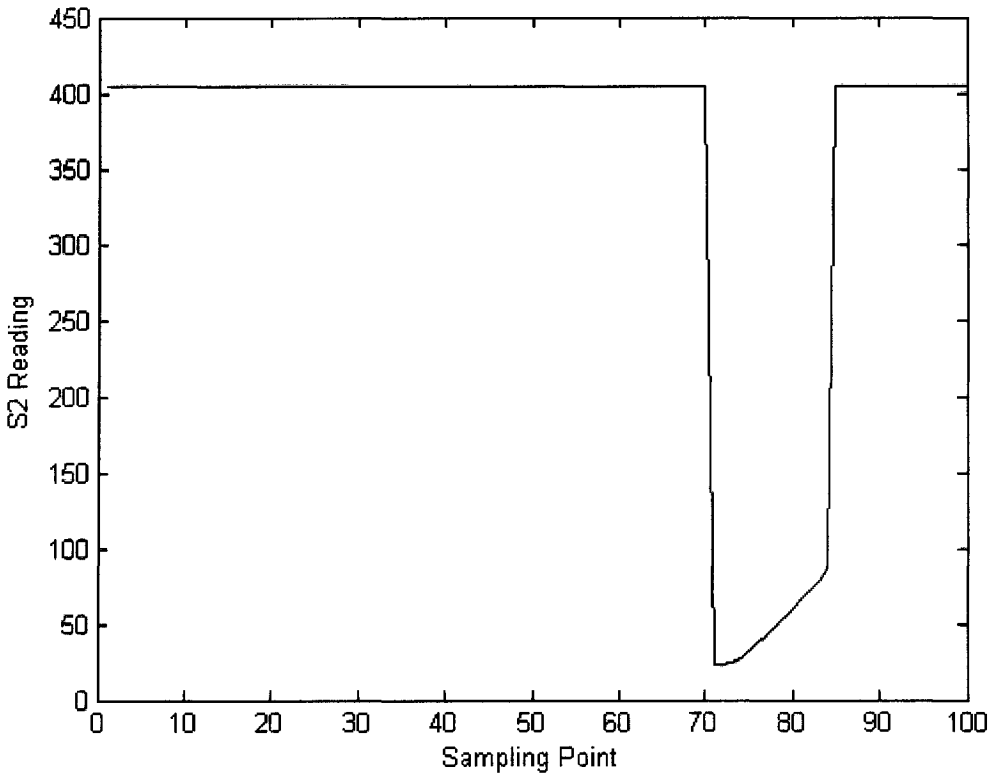

Figure 5-12. Distance measurements of S2 versus sampling point

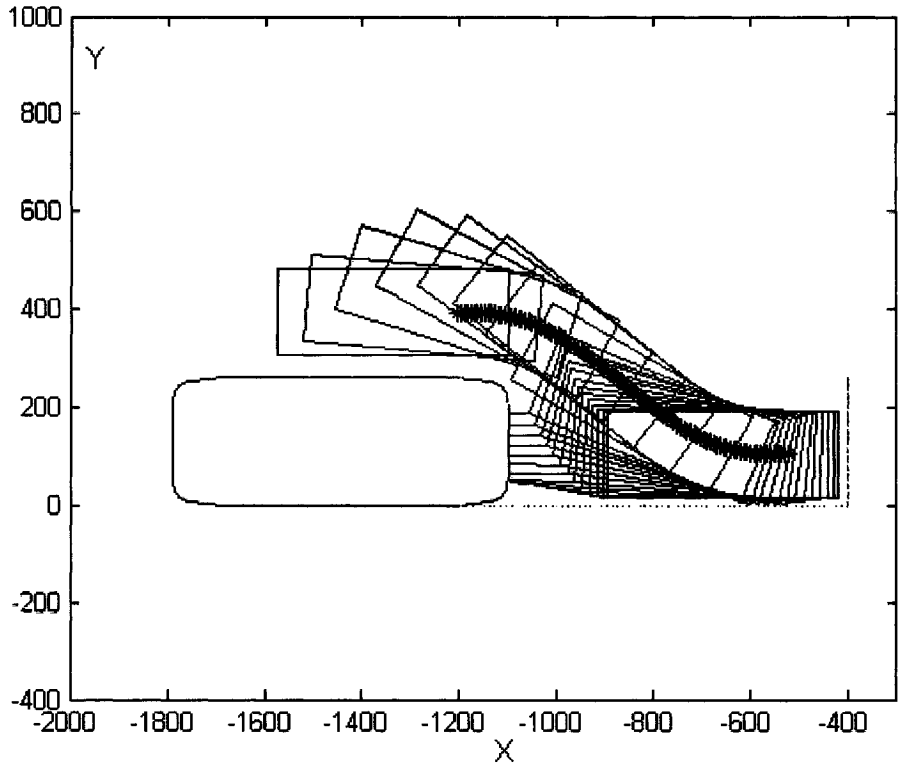

Figure 5-13. Sensor measurements acquired by S1

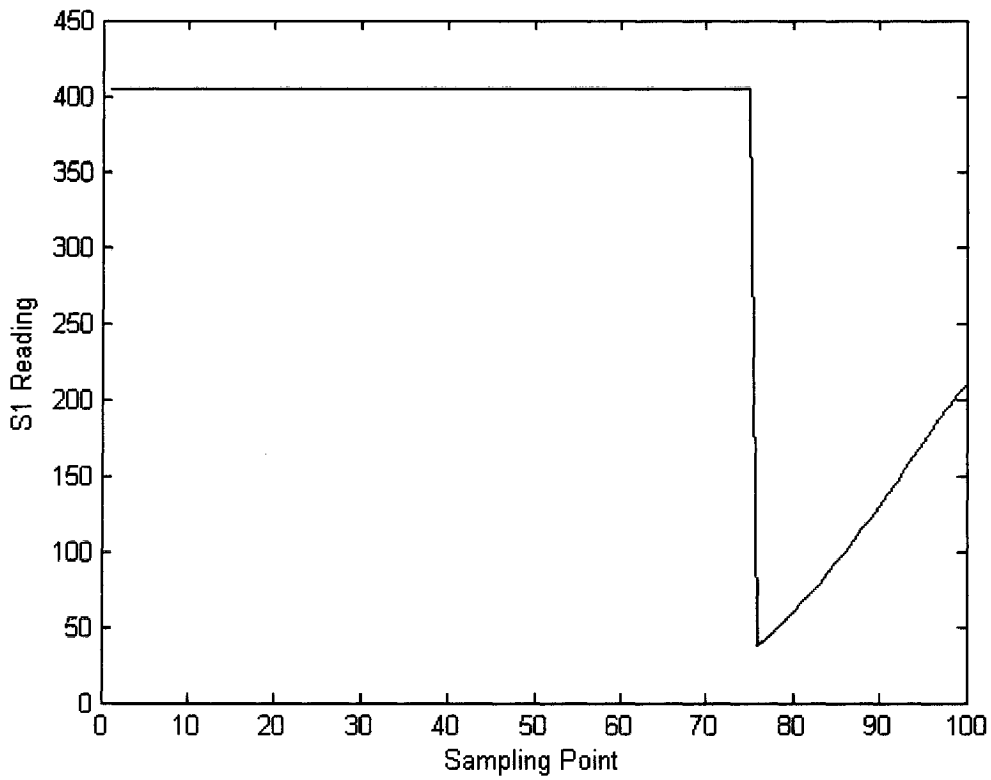

Figure 5-14. Distance measurements of S1 versus sampling point

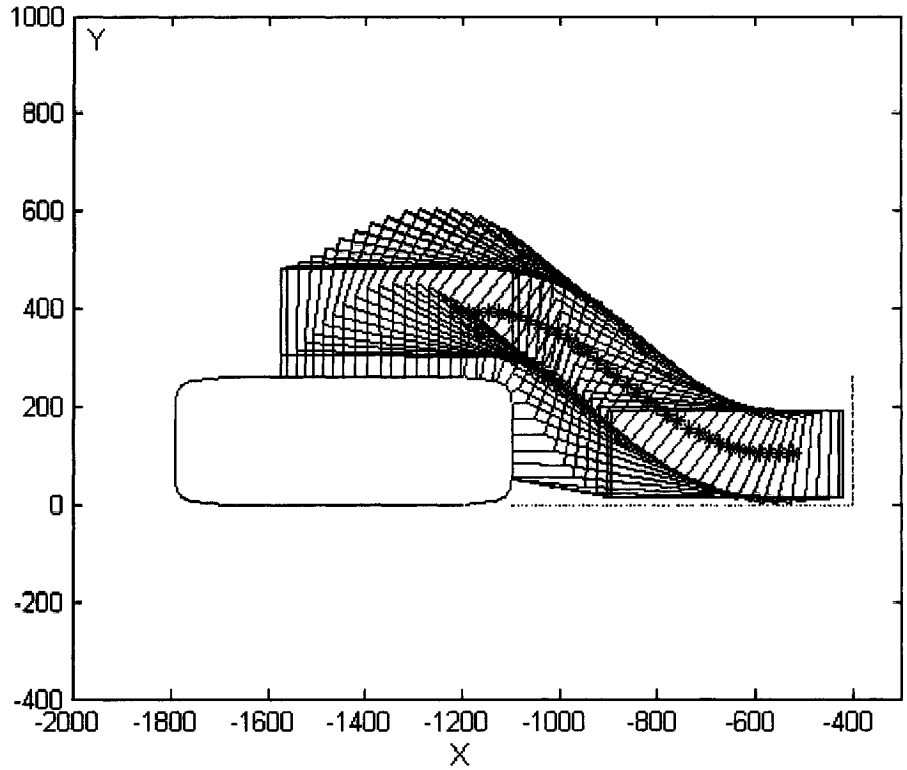

Figure 5-15. All 3 sensor measurements

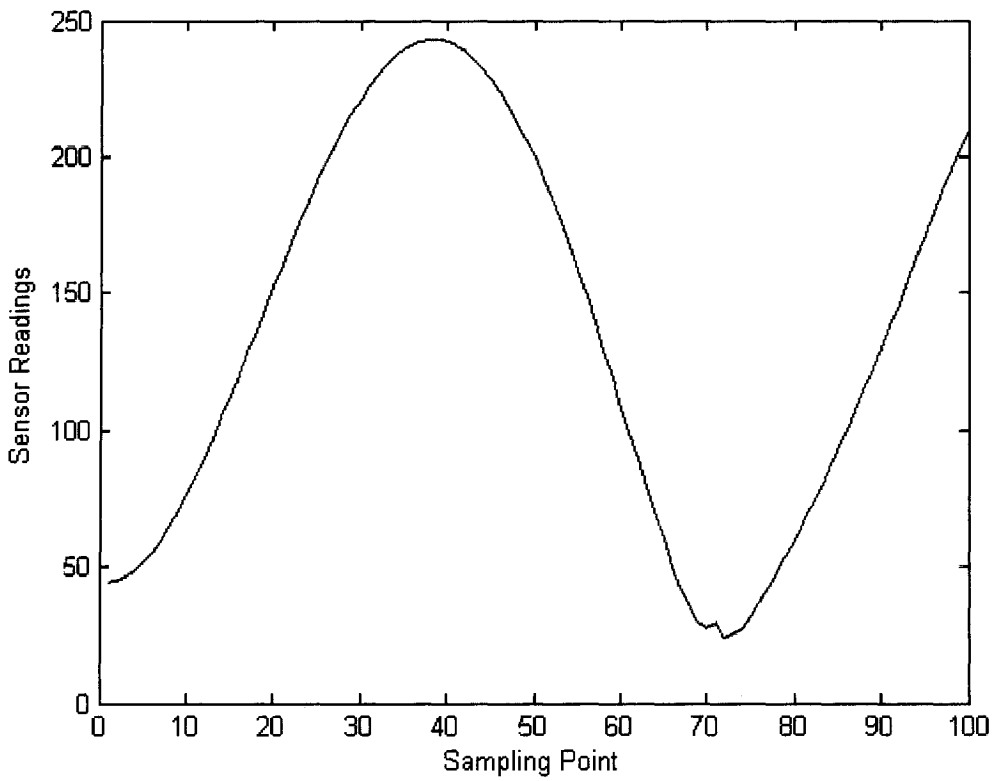

Figure 5-16. Distance measurements of all 3 sensors versus sampling point

Rationale for the proposed approach

The idea behind this approach is that the distances between the front left corner of the car and the parked car in front has all the necessary information required to perform parallel parking. In case C where there is no curb to help us detect the width of the parking space, the car will still be able to autonomously maneuver to the parking bay without knowing the width of the parking space.

The input signals to the controller are the range information acquired by the ultrasonic sensors as well as the difference of the previous distance and the new measured one. The output of the controller is the turning angle. The reason for implementing the 4^{th} input is that our controller should be able to provide us the proper motion direction, when we have the same distance measurements by the sensors during the maneuver. For example by considering *S3* readings in Figure 5-10, we can see that at the sampling points 20 and 56 we have the same distance measurement of 150 cm. But the turning angle corresponding to these 2 points of the path is different. In order to enable the controller distinguish between these 2 sampling points to provide the proper turning angle, the trend of the sensor measurements should also be trained to the controller. As can be seen in Figure 5-10, at the sampling point 20 the discrepancy between the current distance measurement and the previous one is positive while at the sampling point 56, it is negative. Hence by training the increasing or decreasing nature of the sensor measurements, this problem will be solved. It should be emphasized that the sonar data d_i ($i = 1,2,3$) have been independently processed by the neuro-fuzzy system.

Three paths corresponding to (W_1, $L_{\min}$), (W_2, $L_{\min}$) and (W_3, $L_{\min}$) have been used for training, where W_1 is the minimum width of the parking place that the car is able to park in, W_3 is the maximum width and W_2 is a width between W_1 and W_3. These 3 paths are shown with red, blue and green respectively in Figure 5-5. A forth path shown in black in Figure 5-5, has been used for model validation. A checking data set has been obtained in a similar manner which corresponds to (W_{chk}, $L_{\min}$). The idea is that for the parking widths that have not been used in training, the controller should be able to interpolate between the three paths used for training and generate the appropriate path online, by starting form the initial configuration and processing the sonar data without knowing the width of the parking space. As the nominal trajectory is a 5^{th} order polynomial which has zero curvature at the maneuver end points, there will be no discontinuity on curvature profile if the car navigates on a straight line after reaching the primary goal position.

Each path has been sampled to 100 segments so the training data is a 300×4 matrix. Dimension of a typical car; length=474 cm, width=178 cm, wheel base=267 cm (Mazda 6) has been used in simulations to show the ability of the controller in real world applications.

As we have considered parking behavior in ordinary dimensions such that one backward maneuver would be enough to accomplish parallel parking, $W_{\min}$ and $L_{\min}$ are set to 190 cm and 700 cm respectively. The minimum size of the parking space that has been considered is 1.476 times the length and 1.067 times the width of the vehicle. We assume $W_{\max}$ and $L_{\max}$ to be 320 cm and 1100 cm respectively and W_2 considered 260 cm. Also checking data corresponds to W_{chk}=240 cm and L=700 cm.

5-5. Design of the Neuro-Fuzzy Controller

After the 4-input 1-output training data set has been obtained, fuzzy rules have been extracted from sonar data which can be represented as fuzzy variables through membership functions. Then the fuzzy inference system which utilizes the fuzzy set theoretic operations has been imbedded into a 4 layer standard back-propogation neural network to fine tune the premise and consequent parameters by the hybrid learning algorithm.

The clustering parameters that have been used for extracting the rules are: radii=0.3, squash factor=0.9, accept ratio=0.5 and reject ratio=0.15, which results in 16 fuzzy rules. The membership functions of the input variables and the inference procedure of the fuzzy rule base system are illustrated in Figure 5-17 and Figure 5-18 respectively. The fuzzy inference structure is depicited in Figure 5-19.

The total number of parameters in the neural network structure is 208, which consist of 128 premise parameters and 80 consequent parameters. The premise parameters which are the cluster centers and standard deviations of the extracted sigmoidal membership functions are shown in Table A-2 in the Appendix. The consequent parameters which are the linear parameters associated with the consequents of the extracted fuzzy rules are given in Table A-3 in the Appendix. The number of nodes associated with this Neuro-Fuzzy structure is 167. The neural network structure is shown in Figure 5-20.

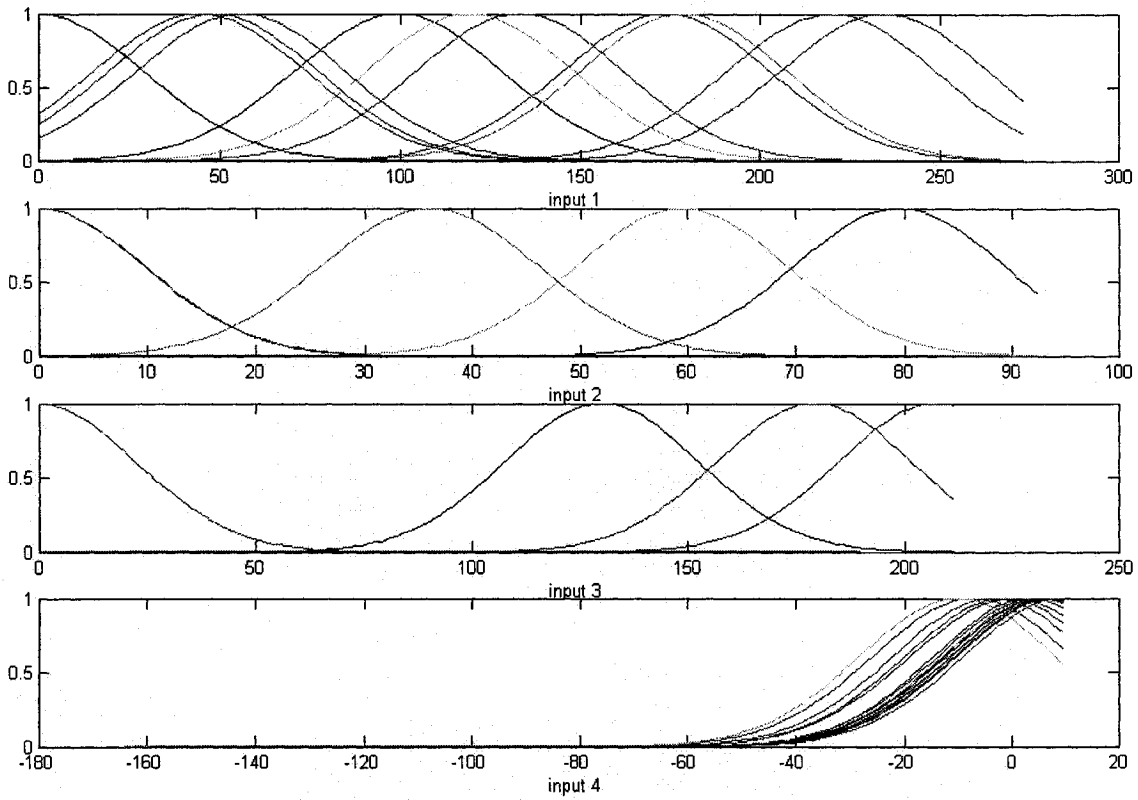

Figure 5-17. Membership functions of the input variables

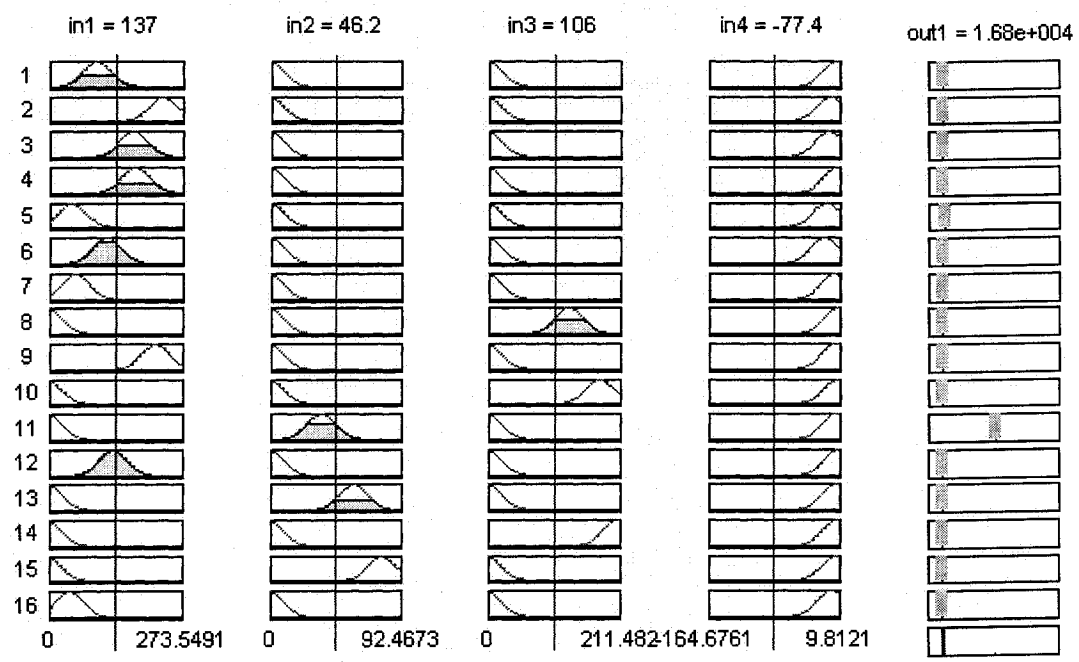

Figure 5-18. Inference procedure of the fuzzy rule-base system

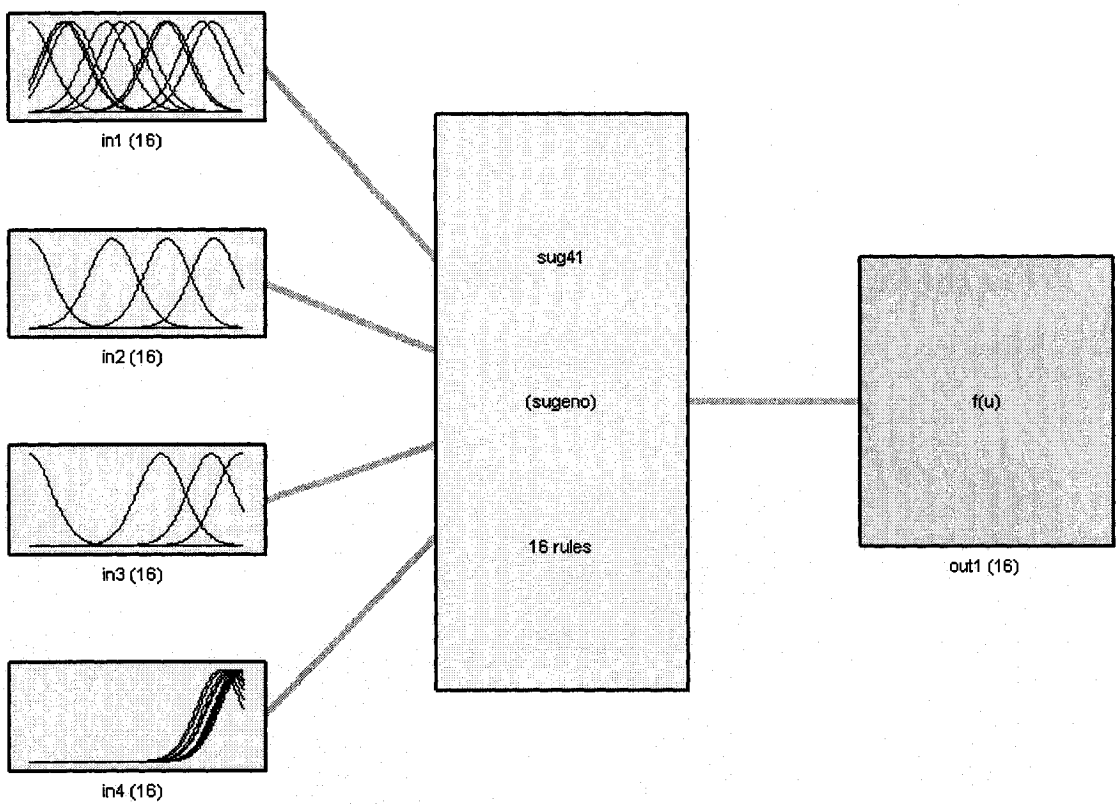

Figure 5-19. Derived 16 rule fuzzy inference system

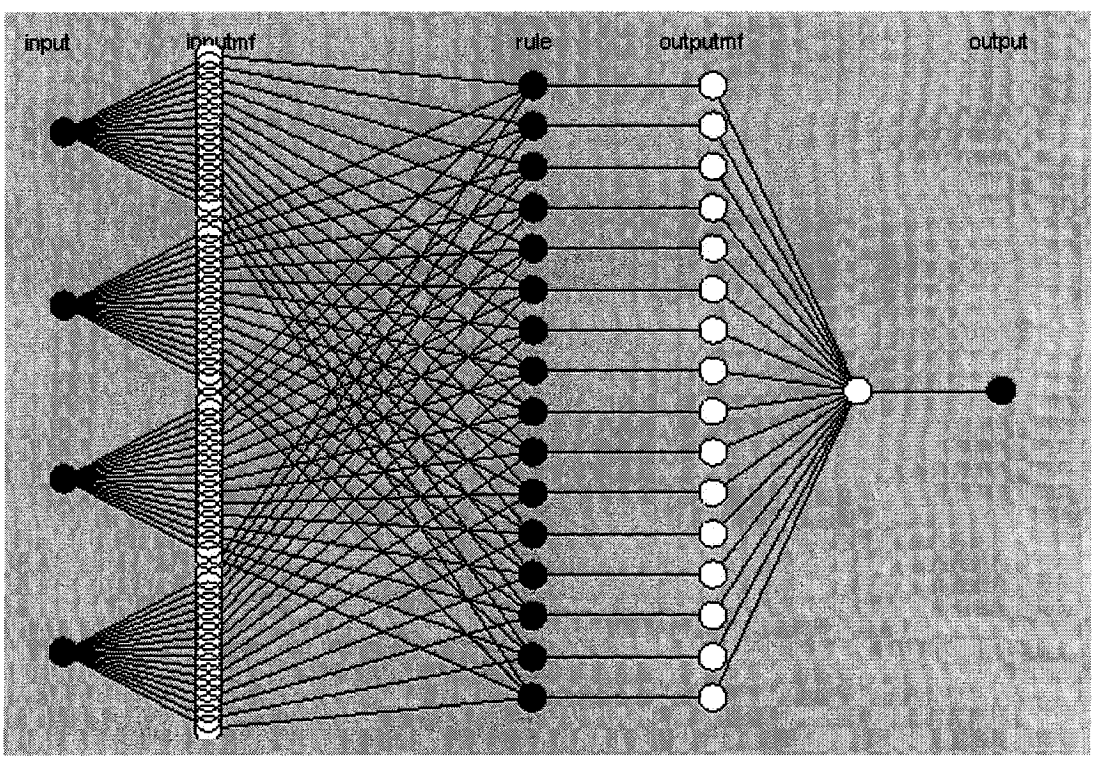

Figure 5-20. Corresponding neural network structure

The constructed controller is in charge of producing the proper turning angle to generate a 5^{th} order polynomial path by processing the input signals at each discrete point of the path. A typical behaviour derived from simulations is shown in Figure 5-21.

The root mean square error representing the checking data error signal up to 60 epochs is depicted in Figure 5-22. The actual checking data output versus the output of the ANFIS architecture corresponds to the input checking data is shown in Figure 5-23, which illustrates the ability of the Neuro-Fuzzy controller to produce proper steering command for the parking dimensions it has not been trained.

Simulation Results

In order to demonstrate the feasibility of autonomous sensor-based parallel parking, five parking space dimensions corresponding to W=200, W=225, W=250, W=275 and W=300 cm are used for simulation.

The results for W=275 cm which is shown in Figure 5-21, illustrate the actual path produced online by the controller and the estimated path planned by offline path planner assuming that we know the width of the parking space. The maximum position error in Figure 5-21, which is the maximum vertical distance error between the actual path and estimated path is 14.7 cm. The simulation results for the other parking space widths are shown in Figure A-6 to Figure A-9 in the Appendix and the maximum position error of all the parking space widths used for simulation is given in Table 5-1. The detailed position error of all the points on the paths corresponding to these widths, representing the vertical distance difference between the actual path and estimated path is illustrated in Table A-4 in the Appendix. As can be seen from Table 5-1, the maximum position error

is below 15 cm. This shows a very good performance and demonstrates that the proposed control scheme enable the vehicle to intelligently and successfully adapt its parallel parking behavior in unknown parking spaces.

To summarize, our proposed online path planner provides the proper motion direction based on sensor measurements without knowing the parking space dimensions. Hence the actual path is constructed piece by piece based on what sonar sees at each time interval during the maneuver. On the otherhand, in traditional path tracking approaches the estimated path is planned based on the offline path planning algorithm by assuming that we know the parking space dimensions. The offline path planner generates the ideal polynomial path offline, by assigning the goal position to the known parking space width.

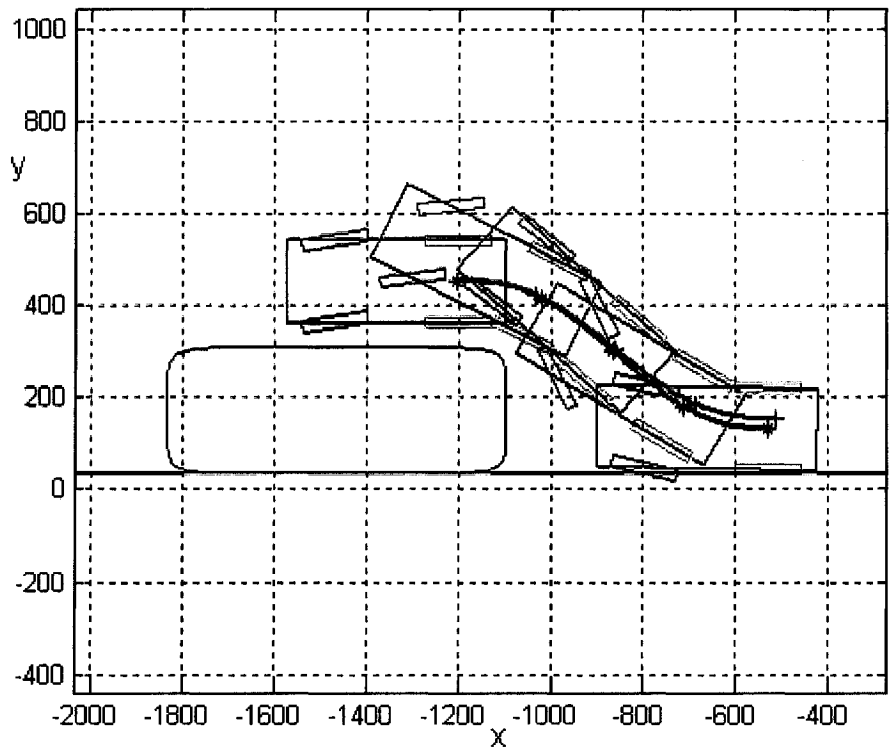

Figure 5-21. Simulation results for W=275----Red: Actual path, Black: Estimated path

Table 5-1. Maximum vertical position error (cm)

W=200 cm	W=225 cm	W=250 cm	W=275 cm	W=300 cm
13.8	14.5	14.7	12.6	13.0

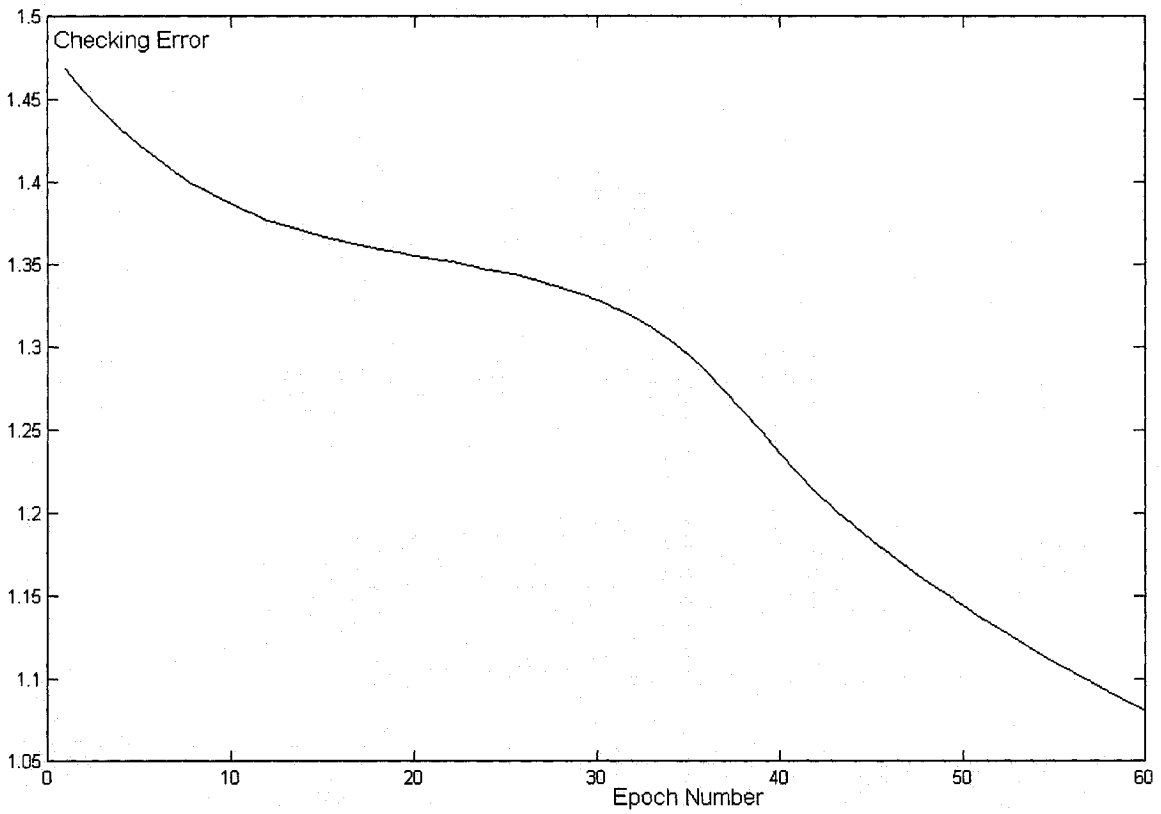

Figure 5-22. Mean square error corresponding to checking data versus epoch number

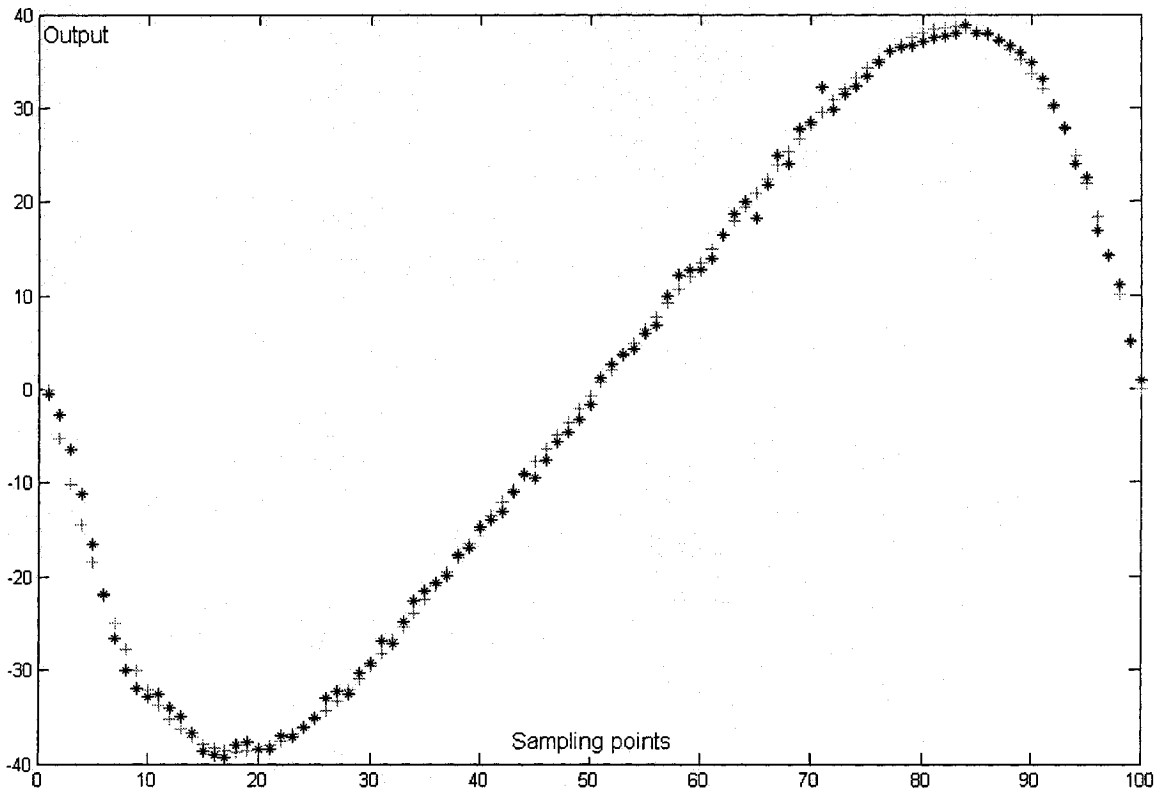

Figure 5-23. Actual Output: +, Output of the ANFIS: *

5-6. Advantages

1) As mentioned earlier this approach proposes a generic solution for the case we can not detect the parking space widths, which has not been investigated neither by behaviour-based algorithms nor traditional approaches. The controller simply relies on sensor measurement to command the steering angle. It covers all the cases for parking parallel to the curb, so it is a general and efficient solution for parallel parking.

2) The input signals to the controller can be obtained directly by the ultrasonic sensor, so we do not need to have an estimate of the state of the vehicle to construct them (which was the case of behaviour-based approach).

3) In this approach, the desired path is produced online based on what sonar sees at each time interval contrary to the path tracking approaches where the ideal path produced offline is tracked by the path tracking techniques. So the algorithm is computationally efficient, as there is no need of offline path planning.

CHAPTER 6: Summary and Conclusions

In this study, a complete solution for autonomous backward parallel parking of a CLMR under the guidance of sensor data is proposed. Two navigation approaches have been proposed: Behavior-based method and sensor-based method.

In the first approach the concepts of Fuzzy Logic control and behavior-based control are merged to emulate human like driving skills. In this approach we have focused on case A, where the parking space dimensions can be identified. According to heuristic knowledge of expert driver's experience, two kinds of information are necessary for performing parallel parking. The first one is the orientation of the car and the second one is the idea of how much we are inside the parking bay. In order to accomplish the autonomous fuzzy behaviour control, a 2 input- 1 output neuro-fuzzy system has been constructed based on training data set obtained from two geometrical paths corresponding to $W_{\min}$ and $W_{\max}$. To generate the training data, we assumed the relative distances between the car and parking space are measurable by processing the information obtained from ultrasonic sensors mounted on the 2 front corners of the car. So the input data pairs have been constructed by these distance measurements based on the description of driver's experiences. The output data set is the turning angles.

The simulation results show that as long as the states of the car are measurable at each discrete time step during the control process, the proposed controller can produce the proper turning angle to make the robot follow feasible trajectories by starting form the initial configuration.

In the second approach, we have focused on case C where the parking space dimensions can not be identified. In this approach, the controller combines the features of Fuzzy Logic and sensor-based navigation to design a neuro-fuzzy system to autonomously plan the motion towards the goal position. In this approach we have focused on the most difficult case of parallel parking when the parking space dimensions can not be identified. For this navigation scheme, the 4 dimensional input space consists of three range measurements obtained from the three sonar sensors mounted at 3 directions at the front left corner of the car, plus the discrepancy between the current measured distance and the previous measured one. The control output signal is the turning angle. Three paths have been used for training. Our training data set has been obtained by discritizing these paths and simulating the sonar sensors behaviour.

The simulation results show that the car can decide about the motion direction at each sampling point of the path without knowing the parking space width, based on the direct sonar readings which serve as inputs.

Based on the obtained simulation results we observe that:

1) The proposed sensor-based parking strategy for motion planning is quite effective for automatic parking task in the most difficult case when the parking space dimensions can not be identified. In other words, the proposed scheme will enable the vehicles to intelligently and successfully adapt their parallel parking behavior in unknown parking spaces.

2) Compared with other existing subjective fuzzy logic techniques and path tracking methods, our approach will be able to approximate the reference trajectory online by processing the sensor information at each time interval and by taking into account the non-holonomic constraints.

3) The results which demonstrate the suitability of the reverse-motion maneuvering algorithm for variety of parking space dimensions show that the proposed scheme can be considered as a step towards achieving a fully automatic parking system and has the potential to be implemented in real life applications.

4) The algorithm is computationally efficient because the path will be generated online based on the sensor measurement and there is no need of any offline path planner which reduces the computational cost.

Future Work

1) Experiments (hardware implementation) should be conducted, to verify the simulation results.

2) The proposed control strategies can be developed to address parallel parking problem with the existence of obstacles. The problem will become more complicated as there are two interacting goals that should be considered simultaneously by the control system. In other words, the controller should be able to avoid obstacles and to reaches the goal position in the parking slot. So the control system should be decomposed in to modules in charge of each goal.

REFERENCES:

[1] I. Baturone, F.J. Moreno-Velo, S. Sanchez-Solano and A. Ollero, "Automatic design of fuzzy controllers for Car-Like autonomous robots", IEEE Transaction on Fuzzy System, vol. 12, No. 4, pp. 447-465, Aug 2004.

[2] T- H S.Li, S-J. Chang, and Y-X. Chen, "Implementation of human-like driving skills by autonomous Fuzzy Behavior Control on an FPGA-Based Car-Like mobile robots", IEEE Transaction on Industrial Electronics, vol. 50, No. 5, pp. 867-880, Oct 2003.

[3] A. Scheuer and TH. Fraichard, "Planning continuous-curvature paths for Car-Like robots", IEEE Transactions on Intelligent Robots and Systems, vol. 3, pp. 1304-1311, Nov 1996.

[4] D. Lyon, "Parallel parking with curvature and nonholonomic constraints", in Proceedings of Intelligent Vehicles 92nd Symposium, Detroit, MI, pp. 341-346, July 1992.

[5] W. Nelson, "Continuous-curvature paths for autonomous vehicles", IEEE Proceeding on Robotics and Automation, vol. 3, pp. 1260-1264, May1989.

[6] L. E. Dubins, "On curves of minimal length with a constraint on average curvature and with prescribed initial and terminal positions and tangents", American Journal of Math, vol. 79, pp. 497-516, 1957.

[7] K. Jiang, "A sensor guided autonomus parking system for nonholonomic mobile robots" IEEE Transaction on Robotics and Automation, Detroit, MI, vol. 1, pp. 311-316, May 1999.

[8] W.A. Daxwanger and G K. Schmidt, "Skill-based visual parking control using neural and fuzzy netwoks", IEEE Transactions on Systems, Man And Cybernetics, vol. 2, pp. 1659-1664, Oct 1995.

[9] M. Wada and K S.Yoon, H. Hashimoto, "Development of advanced parking assistance system", IEEE Transactions on Industrial Electronics, vol. 50, No. 1, pp. 4-17, Feb 2003.

[10] G.G. Rigates, S.G Tzafestas and G.J Evangelidis, "Reactive parking control of nonholonomic vehicles via a fuzzy learning automation", IEE Proc on Control Theory and Applications, vol. 148, Issue. 2, pp. 169-179, March 2001.

[11] Y. Zhao and E G. Jr. Collins, "Fuzzy parallel parking control of autonomous ground vehicles in tight spaces", IEEE International Symposium on Intelligent Control, pp. 811-816, Oct 2003.

[12] T-H S. Li and S-J. Chang, "Autonomous fuzzy parking control of a Car-Like mobile robot", IEEE Transaction on Systems, Man, And Cybernetics-Part A: Systems and Humans, vol. 33, No. 4, pp. 451-465, July 2003.

[13] M. Khoshnejad and K. Demirli, "Autonomous parallel parking of a Car-Like Mobile Robot by a neuro-fuzzy behaviour-based controller", in proceedings of NAFIPS, pp. 814-819, June 2005.

[14] Y. Zhao and E G. Jr. Collins, "Robust automatic parallel parking in tight spaces via fuzzy logic", Journal of Robotics and Autonomous Systems, vol. 51, pp. 111-127, 2005.

[15] G. Chen and D. Zhang, "Back-driving a truck with suboptimal distance trajectories: A fuzzy logic control approach", IEEE Transactions on Fuzzy Systems, vol. 5, No. 3, pp. 369-380, Aug 1999.

[16] C-W Cheng, S-J Chang and T-H S. Li, "Parallel parking control of autonomous mobile robot", IEEE proceedings on Industrial Electronics, Control and Instrumentations, vol. 3, pp. 1305-1310, Nov 1997.

[17] A W. Divelbiss and J T. Wen, "Trajectory tracking control of a car-trailer system", IEEE Transactions on Control Systems Technology, vol. 5, No. 3, pp. 269-278, May 1997.

[18] J-S Roger. Jang, "Adaptive-Network-Based fuzzy inference system", IEEE Transactions on Systems, Man, And Cybernetics, vol. 23, no. 3, 1993.

[19] J-S Roger Jang, C-T Sun and E. Misutani, Neuro-Fuzzy and Soft Computing: A Computational Approach to Learning and Machine Intelligence, Pearson Education; 1st edition (Sep 26, 1996)

[20] J-C Latombe, Robot Motion Planning, Kluwer Academic Publishers, 1991.

[21] B. Barshan and R. Kuc, "Differentiating sonar reflections from corners and planes by employing an intelligent sensor", IEEE Transactions on Pattern Analysis And Machine Intelligence, vol. 12, No. 6, June 1990.

[22] M. Molhim, "Possibilistic sonar modeling and localization for mobile robots", Master Thesis, 1997.

APPENDIX

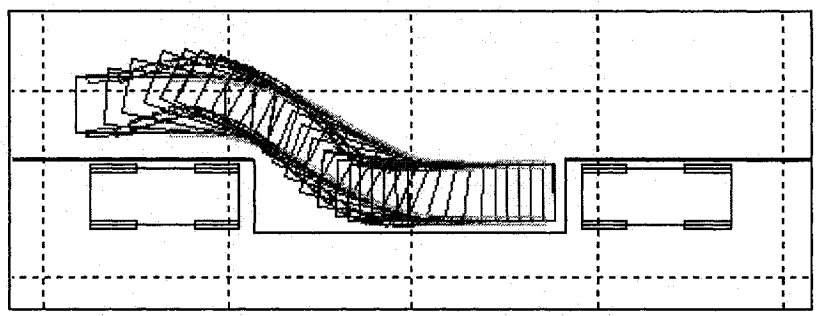

Figure A-1. Parking dimension: W=230

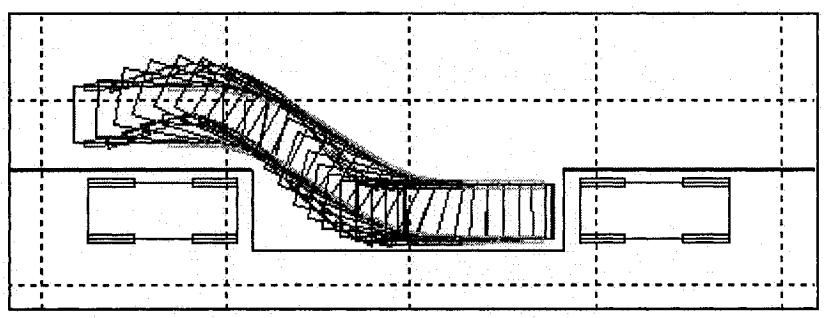

Figure A-2. Parking dimension: W=260 cm

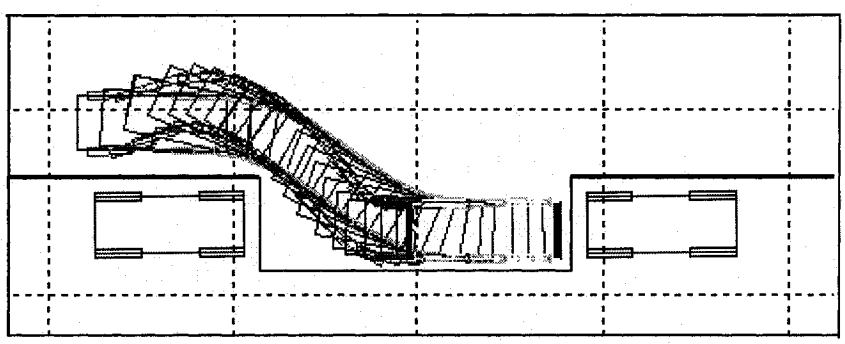

Figure A-3. Parking dimension: W=290 cm

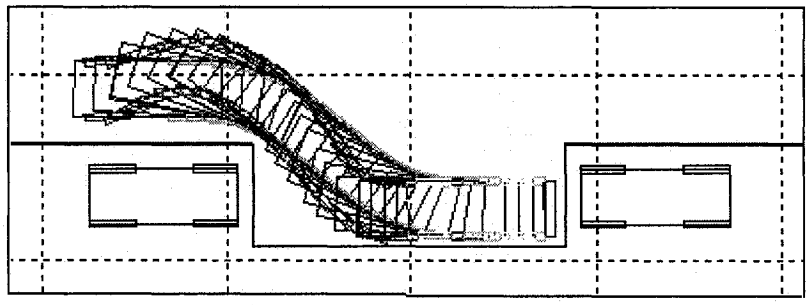

Figure A-4. Parking dimension: W=330 cm

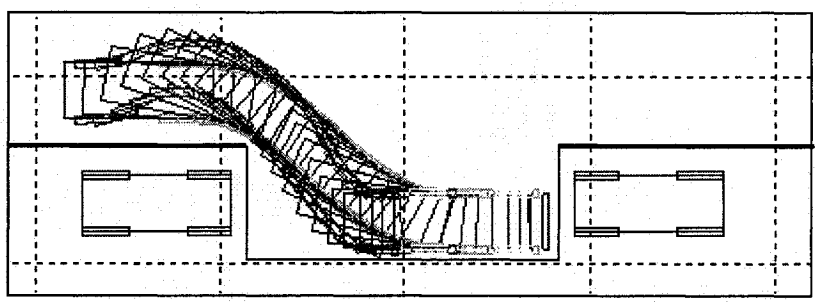

Figure A-5. Parking dimension: W=360 cm

Table A-1. Premise and Consequent parameters

Rule Number	Input1 [c , δ]	Input2 [c , δ]	Consequent Parameters: $P_1(input1)+P_2(input2)+P_3$
Rule1	[60.19 247.4]	[208.1 694.9]	[-0.0163 -0.1719 114.4]
Rule 2	[59.92 81.34]	[208.2 161.2]	[0.2681 0.06365 -9.617]
Rule 3	[59.67 81.26]	[208.2 657]	[-1.193 0.07231 124.7]
Rule 4	[60.15 250.5]	[208.1 334.4]	[-0.007198 -0.06481 49.86]
Rule 5	[60.14 289.9]	[208.1 1015]	[0.119 -0.1327 86.7]
Rule 6	[59.96 -0.2025]	[208.1 -162]	[1.239 0.02488 8.817]
Rule 7	[60.26 -0.1231]	[208.2 725.9]	[-2.969 0.112 -150]

Table A-2. Premise Parameters (Sigmoidal MF Parameters)

Input1 [c , δ]	Input2 [c , δ]	Input3 [c , δ]	Input4 [c , δ]
[29.03 99.44]	[9.808 2.839e-006]	[22.43 -2.174e-008]	[18.5 5.682]
[29 234.6]	[9.808 7.225e-019]	[22.43 -2.705e-020]	[18.51 -1.482]
[29.06 174]	[9.808 1.274e-011]	[22.43 -2.412e-013]	[18.5 -3.703]
[28.97 178.8]	[9.808 5.257e-012]	[22.43 1.699e-013]	[18.49 7.146]
[28.87 48.27]	[9.809 0.002419]	[22.43 -0.0001017]	[18.69 -7.252]
[29.06 119.3]	[9.808 5.488e-007]	[22.43 3.782e-010]	[18.31 -10.1]
[28.82 55.77]	[9.839 0.005395]	[22.43 0.0001202]	[17.89 3.174]
[29.01 -7.037e-010]	[9.808 2.661e-011]	[22.43 130]	[18.51 7.745]
[29.02 219.9]	[9.808 -5.648e-017]	[22.43 -4.811e-018]	[18.51 4.568]
[29.01 2.078e-017]	[9.808 -2.176e-017]	[22.45 179.2]	[18.51 8.067]
[29 0.002561]	[10.17 36.07]	[22.43 1.288e-008]	[18.52 5.487]
[29 133.2]	[9.808 -6.909e-008]	[22.43 -8.665e-010]	[18.51 9.313]
[29.01 -1.484e-010]	[9.815 59.45]	[22.43 1.86e-014]	[18.51 6.06]
[29.01 -3.041e-022]	[9.808 3.812e-023]	[22.44 207.3]	[18.51 6.93]
[29.01 -8.124e-017]	[9.807 79.54]	[22.43 2.142e-020]	[18.51 6.065]
[29.35 44.45]	[9.773 -0.008332]	[22.43 9.111e-005]	[19.04 0.06269]

Table A-3. Consequent Parameters

Consequent parameters	$P_1(input1)+P_2(input2)+P_3(input3)+P_4(input4)+P_5$ [P_1 , P_2 , P_3 , P_4 , P_5]
Rule 1	[7.861 -42.26 -4.031 23.45 -683.3]
Rule 2	[-0.345 3.136e-011 2.266e-013 1.951 197.9]
Rule 3	[-2.199 9.487e-005 6.827e-006 38.69 2607]
Rule 4	[6.889 -0.0001148 -9.288e-006 24.49 -3466]
Rule 5	[-73.15 -100.6 214.8 -14.63 5931]
Rule 6	[-3.493 -1.152 0.2652 51.95 2503]
Rule 7	[80.18 -383.4 -129.7 -125.4 -9869]
Rule 8	[0.1798 -0.001227 0.05679 -7.635 88.02]
Rule 9	[-3.887 -3.497e-009 -3.981e-010 -5.387 776.2]
Rule 10	[5.237e-008 -3.778e-010 -0.6628 23.6 -80.94]
Rule 11	[6333 0.2138 -2.556 -0.02317 27.96]
Rule 12	[-1.8 -0.8289 -0.0564 32.76 -881.8]
Rule 13	[0.03729 0.3357 -1.622e-005 -6.029 55.21]
Rule 14	[1.025e-012 -5.982e-015 -1.958 -6.509 478]
Rule 15	[1.797e-008 0.3815 -7.818e-012 -9.348 71.02]
Rule 16	[67.91 280 -107.4 225.5 205.2]

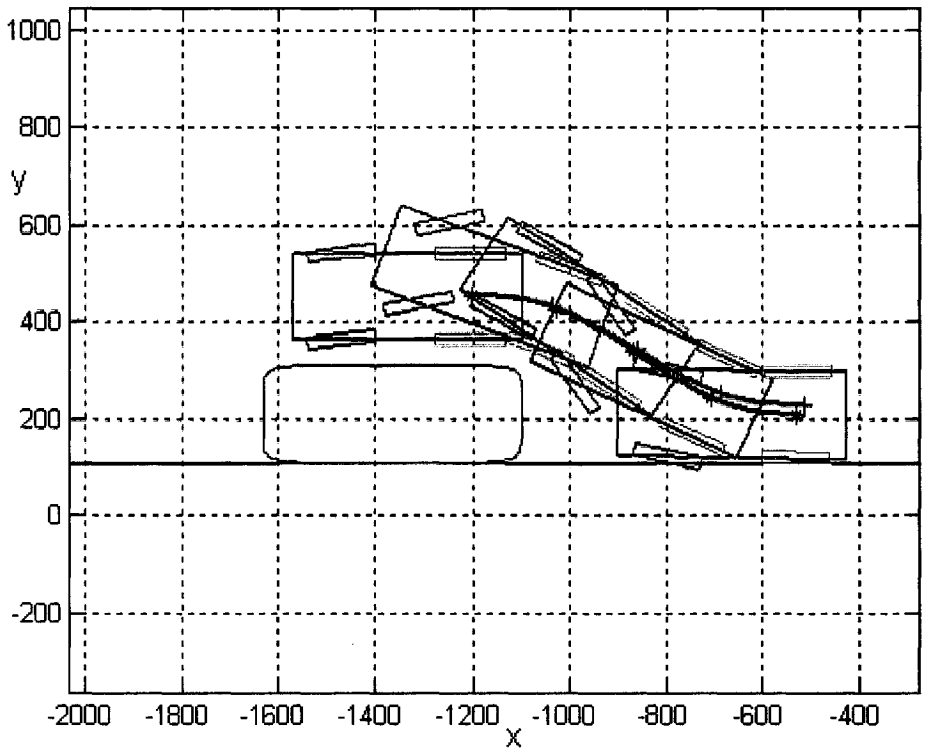

Figure A-6. Simulation results: W=200 cm----Red: Actual path, Black: Estimated path

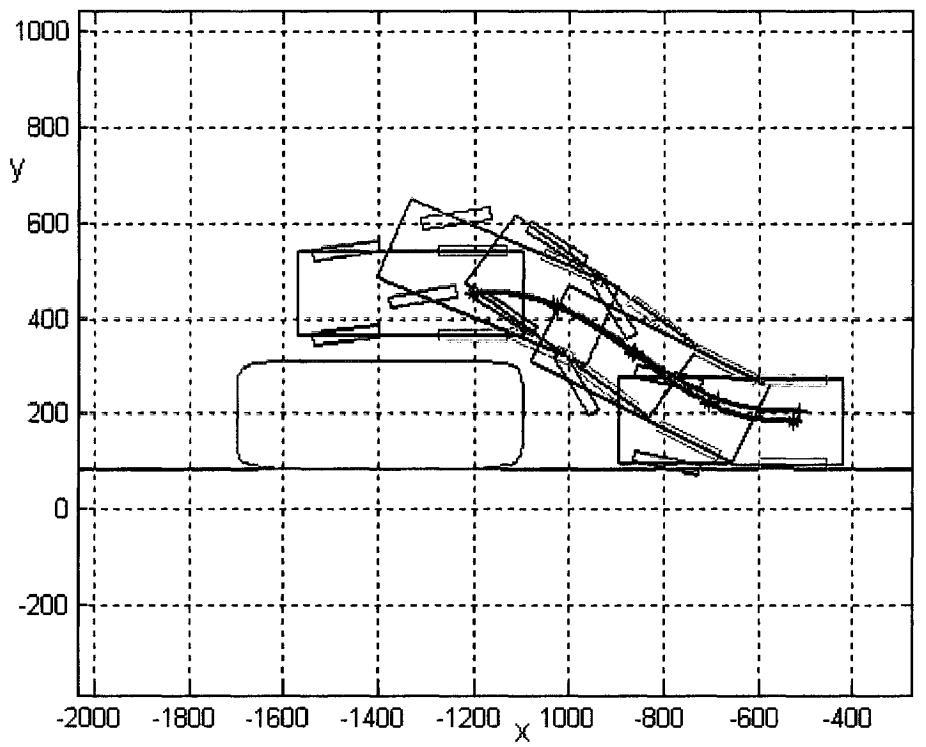

Figure A-7. Simulation results: W=225 cm----Red: Actual path, Black: Estimated path

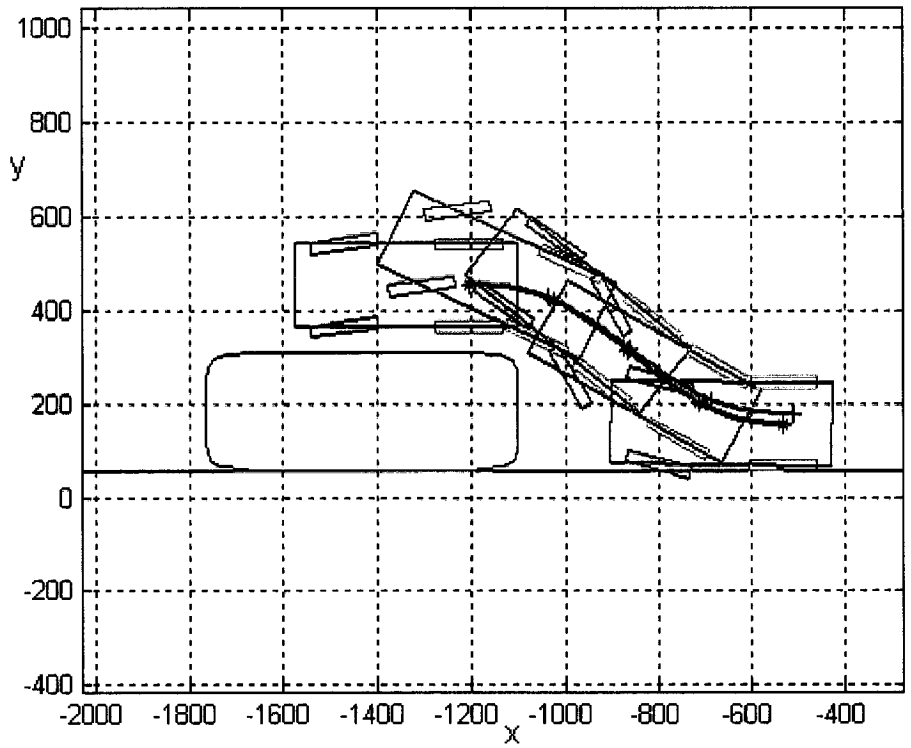

Figure A-8. Simulation results: W=250 cm----Red: Actual path, Black: Estimated path

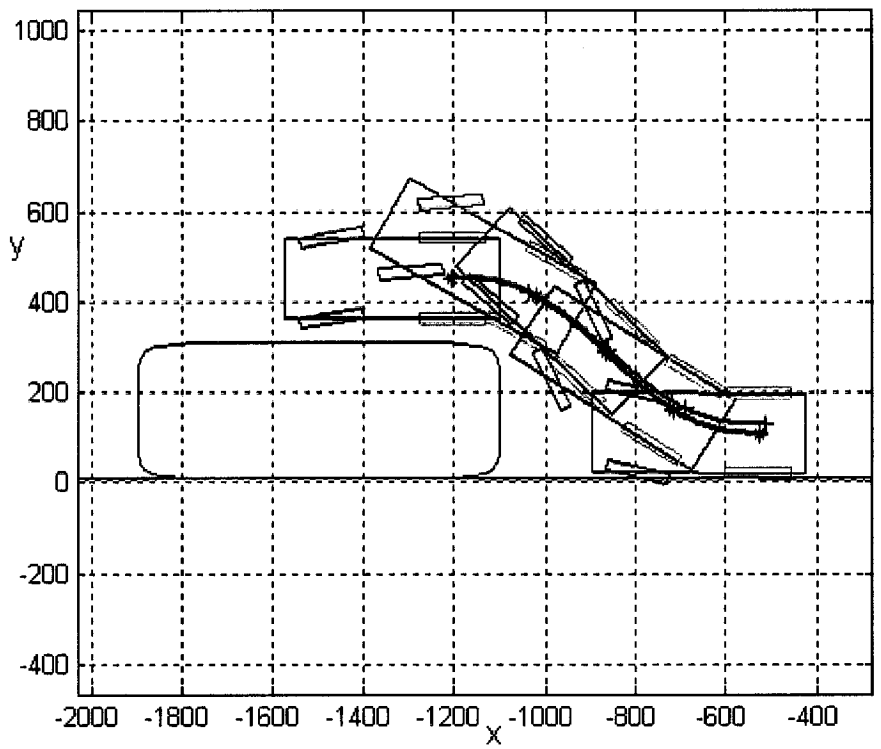

Figure A-9. Simulation results: W=300 cm----Red: Actual path, Black: Estimated path

Table A-4. Vertical distance error between the ideal path (offline) and generated path (online)

W=200	W=225	W=250	W=275	W=300
0.0000	0.0000	0.0000	0.0000	0.0000
0.0097	0.0120	0.0139	0.0160	0.0181
0.0339	0.0429	0.0507	0.0592	0.0676
0.0663	0.0870	0.1046	0.1240	0.1435
0.1028	0.1409	0.1729	0.2088	0.2447
0.1416	0.2040	0.2560	0.3151	0.3742
0.1829	0.2785	0.3573	0.4480	0.5384
0.2295	0.3690	0.4828	0.6150	0.7459
0.2859	0.4820	0.6403	0.8248	1.0057
0.3582	0.6252	0.8375	1.0850	1.3244
0.4530	0.8053	1.0800	1.3992	1.7038
0.5763	1.0266	1.3697	1.7671	2.1415
0.7319	1.2902	1.7045	2.1842	2.6327
0.9208	1.5930	2.0792	2.6446	3.1723
1.1409	1.9301	2.4877	3.1427	3.7567
1.3880	2.2951	2.9240	3.6739	4.3835
1.6565	2.6819	3.3832	4.2351	5.0502
1.9409	3.0855	3.8615	4.8233	5.7519
2.2360	3.5015	4.3557	5.4344	6.4800
2.5372	3.9264	4.8624	6.0620	7.2225
2.8410	4.3571	5.3772	6.6975	7.9658
3.1443	4.7903	5.8944	7.3307	8.6964
3.4444	5.2221	6.4072	7.9511	9.4020
3.7387	5.6482	6.9080	8.5485	10.0716
4.0247	6.0637	7.3894	9.1139	10.6959
4.2996	6.4635	7.8444	9.6392	11.2670
4.5606	6.8429	8.2667	10.1175	11.7785
4.8047	7.1971	8.6507	10.5433	12.2253
5.0293	7.5221	8.9918	10.9119	12.6032
5.2318	7.8144	9.2862	11.2199	12.9091
5.4100	8.0711	9.5311	11.4645	12.1410
5.5621	8.2898	9.7242	11.6441	12.2973
5.6865	8.4690	9.8643	11.7576	12.3775
5.7824	8.6076	9.9506	11.8047	12.3816
5.8491	8.7051	9.9830	11.7858	12.3103
5.8865	8.7615	9.9622	11.7019	12.1649
5.8949	8.7774	9.8891	11.5545	12.9472
5.8747	8.7538	9.7653	11.3457	12.6596
5.8271	8.6922	9.5930	11.0781	12.3049
5.7533	8.5942	9.3747	10.7547	11.8866
5.6550	8.4622	9.1131	10.3789	11.4083
5.5340	8.2986	8.8116	9.9545	10.8744
5.3925	8.1062	8.4737	9.4859	10.2893

5.2329	7.8881	8.1033	8.9774	9.6581
5.0577	7.6476	7.7046	8.4339	8.9861
4.8696	7.3881	7.2819	7.8606	8.2790
4.6715	7.1134	6.8398	7.2627	7.5427
4.4664	6.8272	6.3830	6.6457	6.7833
4.2573	6.5334	5.9164	6.0153	6.0072
4.0473	6.2359	5.4448	5.3773	5.2211
3.8395	5.9387	4.9733	4.7376	4.4315
3.6371	5.6458	4.5070	4.1020	3.6453
3.4431	5.3611	4.0507	3.4766	2.8693
3.2606	5.0885	3.6095	2.8671	2.1104
3.0924	4.8316	3.1881	2.2793	1.3753
2.9414	4.5941	2.7912	1.7189	0.6707
2.8102	4.3794	2.4233	1.1912	0.0030
2.7013	4.1907	2.0887	0.7016	0.6214
2.6170	4.0310	1.7913	0.2549	1.1967
2.5596	3.9030	1.5350	0.1444	1.7170
2.5308	3.8090	1.3230	0.4919	2.1773
2.5325	3.7512	1.1583	0.7839	2.5726
2.5660	3.7313	1.0434	1.0172	2.8987
2.6327	3.7508	0.9805	1.1890	3.1519
2.7337	3.8108	0.9713	1.2969	3.3291
2.8697	3.9120	1.0170	1.3395	3.4280
3.0414	4.0550	1.1182	1.3156	3.4468
3.2492	4.2398	1.2753	1.2249	3.3847
3.4934	4.4665	1.4880	1.0675	3.2416
3.7738	4.7345	1.7557	0.8443	3.0182
4.0900	5.0432	2.0774	0.5568	2.7160
4.4410	5.3913	2.4516	0.2067	2.3375
4.8253	5.7774	2.8764	0.2032	1.8858
5.2408	6.1990	3.3495	0.6700	1.3648
5.6841	6.6532	3.8678	1.1899	0.7792
6.1513	7.1358	4.4276	1.7588	0.1342
6.6369	7.6413	5.0243	2.3715	0.5639
7.1348	8.1632	5.6521	3.0217	1.3083
7.6382	8.6937	6.3035	3.7022	2.0909
8.1403	9.2245	6.9703	4.4040	2.9026
8.6351	9.7481	7.6432	5.1176	3.7334
9.1178	10.2579	8.3130	5.8331	4.5725
9.5853	10.7494	8.9719	6.5415	5.4097
10.0354	11.2193	9.6138	7.2357	6.2367
10.4662	11.6650	10.2341	7.9105	7.0472
10.8761	12.0836	10.8287	8.5611	7.8361
11.2628	12.4715	11.3925	9.1814	8.5974
11.6236	12.8244	11.9194	9.7634	9.3226
11.9558	13.1381	12.4016	10.2965	9.9989

12.2574	13.4103	12.8317	10.7693	10.6101
12.5276	13.6412	13.2048	11.1735	11.1402
12.7671	13.8343	13.5207	11.5072	11.5798
12.9783	13.9955	13.7844	11.7770	11.9317
13.1641	14.1313	14.0040	11.9945	12.2097
13.3277	14.2474	14.1884	12.1721	12.4307
13.4711	14.3479	14.3447	12.3195	12.6095
13.5952	14.4344	14.4771	12.4429	12.7556
13.6988	14.5061	14.5867	12.5441	12.8736
13.7792	14.5594	14.6715	12.6213	12.9629
13.8314	14.5888	14.7264	12.6689	13.0188