İSTANBUL TECHNICAL UNIVERSITY ★ INSTITUTE OF SCIENCE AND TECHNOLOGY

BEHAVIOR-BASED FUZZY CONTROL FOR A MOBILE ROBOT WITH NON-HOLONOMIC CONSTRAINTS

M.Sc.Thesis by

Mustafa ŞİRİN, Mechanical Eng.

(518021024)

Date of submission :	**9 May 2005**
Date of defence examination:	**24 May 2005**
Supervisor (Chairman):	**Assoc. Prof. Dr. Hakan Temeltaş**
Members of the Examining Committee	**Prof.Dr. İbrahim Eksin**
	Assoc. Prof. Dr. Şeniz Ertuğrul

MAY 2005

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

HOLONOMİK OLMAYAN KISITLARA SAHİP BİR MOBİL ROBOTTA DAVRANIŞ TEMELLİ BULANIK KONTROL

Y. LİSANS TEZİ
Mak. Müh. Mustafa ŞİRİN
(518021024)

Tezin Enstitüye Verildiği Tarih : 9 Mayıs 2005
Tezin Savunulduğu Tarih : 24 Mayıs 2005

Tez Danışmanı :	**Doç.Dr. Hakan TEMELTAŞ**
Diğer Jüri Üyeleri	**Prof.Dr. İbrahim EKSİN**
	Doç.Dr. Şeniz ERTUĞRUL

MAYIS 1997

Preface

As one of the first students of the Mechatronics Engineering Masters Program in İTÜ, I would like to thank to all the instructors and coordinators of the program for their courage and enthusiasm in establishing this new and leading discipline in İTÜ.

Also thanks to my thesis advisor Hakan Temeltaş for his guidance in my researches, patience in my long-lasting studies and being open minded to new ideas through this thesis.

Haziran 2005 Mustafa ŞİRİN

Index

List of Tables

List of figures

HOLONOMİK OLMAYAN KISITLARA SAHİP BİR MOBİL ROBOTTA DAVRANIŞ TEMELLİ BULANIK KONTROL

Özet

Bu çalışmada robotik alanında yeni yaklaşımlar olan davranış temelli robotik ve bulanık mantık konuları gerçek zamanda mobil robot uygulamaları bakımından incelenmiş, dört çekerli, dört yönlendirmeli bir mobil robot için "Engelden Sakın", "Hedefe Git", "Duvarı İzle", "Yola Teğet İlerle", "Avare Gez" davranışları oluşturulmuştur. Bu davranışların içinden "Engelden Sakın", "Hedefe Git" ve "Duvarı İzle" davranışları için sonar algılayıcı matematik modelleri oluşturulmuş ve bu davranışların yapısında bulanık mantık yaklaşımı kullanılmıştır. Mobil robot, kinetik ve dinamik olarak holonomik olmayan kısıtları kullanılarak modellenmiştir ve simülasyon sırasında mobil robotun pozisyonu, tekerlek ve robot yönelimleri, tekerlek ve robot hızları, tekerlek momentleri gibi parametreler izlenebilmektedir. Davranışlar da, simülasyon ortamında kazanımları, bulanık mantık işleme yapıları, gerçek zaman uygulanabilirliği ve davranışların koordine edilmeleri bakımından incelenmiştir. Bu çalışma gerçek bir robotta yapılacak deneyler için temel teşkil etmektedir.

BEHAVIOR-BASED FUZZY CONTROL FOR A MOBILE ROBOT WITH NON-HOLONOMIC CONSTRAINTS

SUMMARY

In this study, the new approaches to the robotics subject, behavior-based robotics and fuzzy logic control are investigated for the real-time applications of mobile robots, "Avoid Obstacle", "Move to Goal", "Wall Following", "Head-on", "Wander" behaviors are built up for a four-wheel driven and four-wheel steered mobile robot. Sonar sensor mathematical models are formed for "Avoid Obstacle", "Move to Goal" and "Wall Following" behaviors and fuzzy logic concepts are used in the structure of these behaviors. The mobile robot is modeled kinematically and dynamically considering the non-holonomic constraints. The posture and speed of the robot and the configurations, speeds and torques of the wheels can be obtained from the simulation. The behaviors are investigated regarding their gains, fuzzy inference structures, real-time applicability and their coordination. This study constitutes basis for the experiments on a real mobile robot.

1 Introduction

In spite of the fact that very primitive animals succeed to survive in the nature, which is an uncertain and dynamic environment, robotic researchers still have problems on the real-time mobile robot applications even in isolated environments although they have very fast, complex and powerful computer systems.

This thesis tries to figure out the basic animals' surveillance system and model it to a 4 wheel driven 4 wheel steered mobile robot considering the kinematics and dynamical parameters.

Having seen that the robotic and AI communities are progressing slowly in this area relative to the computer technology, a new approach has emerged in the mid 1980's. This approach redefines the intelligence [1] and tries to mimic the animal behaviors in basic, modular levels on the reactive robotics foundations [2, 3]. The name for this approach is Behavior Based Robotics and it is defined in chapter 2, starting from the biological inspirations [28-30], continuing on the first studies and theoretical foundations of the approach [1-3] and extended with Emotion Based Architecture [5-7]. The recent studies are on the hybrid architectures having interaction between the upper deliberative levels and lower level behaviors. The behaviors are used in the lower levels as reactive modules and some deliberative actions or long-term planning are done on the upper levels that are managing the behaviors.

Another hot subject is using Fuzzy Set Theory [8, 34] in control purposes [9-19]. Fuzzy Logic Control brings a wider understanding and applicability to I/O relations of the control systems, having a similar structure to the human mind. The Fuzzy concepts are clarified in chapter 3. Some of the behaviors of the robot are fuzzified and integrated with sensor models. Combining the fuzzy logic and behavior-based approach has several advantages on real-time applications [19-23]. There are also applications with learning fuzzy behaviors that use neural architectures in the fuzzy inference [18, 22]. These systems should be trained before they successfully operate in the real environment.

The 4-wheel driven 4-wheel steered (4x4x4) mobile robot has a better maneuverability than car-like robots and structurally more stable than 3-wheeled differentially driven robots. The kinematics and dynamic parameters [24-27] are supervised under the non-holonomic constraints of the robot, and the mathematical background is explained in chapter 4.

Behavior Based Robotics, Fuzzy Logic Control, sensor models and 4x4x4 Mobile robot with Non-holonomic constraints are combined in the simulation studies and supported by an animation. The Simulink block structure of the simulation and animation are explained in chapter 5 including some simulation results.

2 Behavior Based Control

2.1 What is Behavior

2.1.1 Biological Inspiration

A behavior is basically, a reaction to a stimulus. Reactive actions play an important role in the real-time applications and in real life. Many experiments have been done on animals proving the reflexive control, i.e. animal behavior.

Scientists severed the connection between a frog’s spine and brain. The goal was to remove all centralized control so that all action was produced reactively and without "thought." Scientists found much of the behavior of a frog was encoded directly into the spine. Stimulating one location will prompt the frog to wipe its head whereas another will cause it to jump [28].

Another example to the animal behavior is the distance and speed control of bee. Bees use a control that is stabilizing image speed of the patterns at both sides. This behavior can be seen in two experiments. If a bee is flying through a tunnel that has a narrower cross-section at somewhere along its length, the bee slows down in the narrow portion [29] (Figure 2-1).

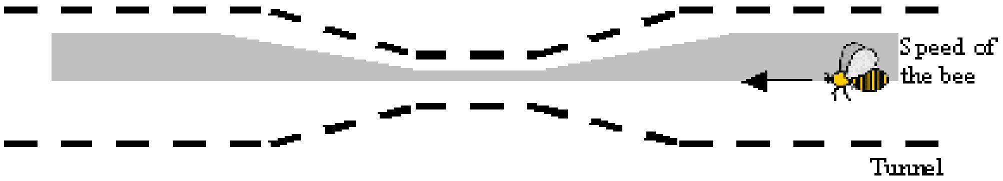


Figure 2-1: Bee in a tunnel

Similarly, the bee is expected to balance the image speeds on the two eyes, staying in the center of a tunnel if both sidewalls are static, and decreasing or increasing the distance to a sidewall moving in the same or opposite direction as the bee, resp. [30] (Figure 2-2).

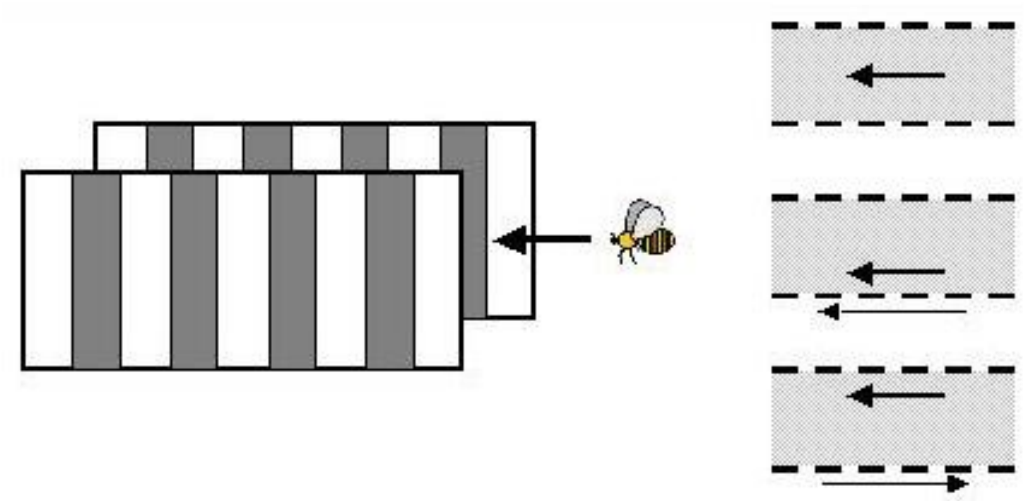


Figure 2-2: Bee between walls

2.1.2 Robot Behavior

In the case of robotics, robot behavior is a direct coupling of the sensory information to the actuators (Figure 2-3). Behaviors do not have deliberative processing levels but may contain representational information as inputs or outputs.

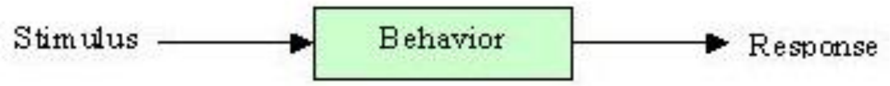


Figure 2-3: Behavior

2.1.2.1 Braitenberg Vehicles

Braitenberg vehicles give the basic understanding for the purely reactive behaviors. These robots were a thought experiment of Valentino Braitenberg [31], and later realized by some other scientists. These vehicles are a set of inflexible vehicles with direct connections of the sensors and motors (Figure 2-4).

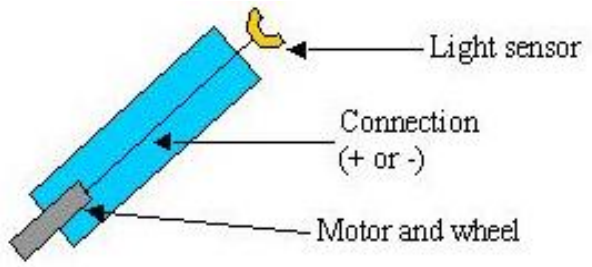


Figure 2-4: Braitenberg vehicle 1

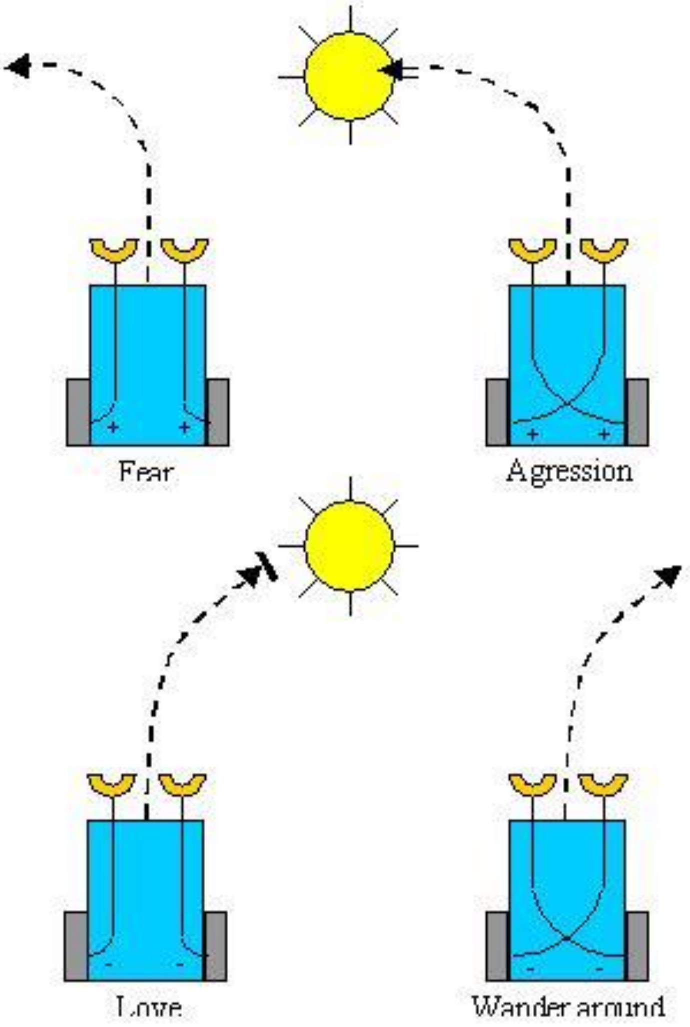


Figure 2-5 : Braitenberg vehicles 2 and 3

The sensors of “Fear” and “Aggression” robots are aversive to light (Vehicle 2). The motor closest to the light goes faster than the other. The “Aggression” robot hits the light source.

The sensors of “Love” and ”Wander around” robots are inhibiting the motor speed in case of strong light, and turns faster in weak light (Vehicle 3). The “Love” robot goes to the light source and stops at a close distance. The “Wander around” robot stays around the light source but also keeps traveling (Figure 2-5).

By choosing some thresholds, nonlinear for the sensors and motors (Vehicle 4), adding some more light sources and sensors, different behaviors can be achieved.

2.1.2.2 Move to Goal Behavior

This behavior forms a vector that is pulling the robot to the goal (Figure 2-7). The position and orientation of this vector is determined according to the relative position of the goal (Figure 2-6). Ether the original position of the goal and the robot, or the relative position of the goal should be known in order to determine the vector position and orientation (Figure 2-7). The Magnitude of this vector is determined according to the distance between the robot and the goal.

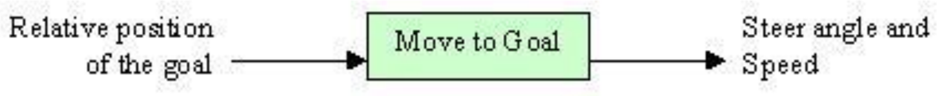


Figure 2-6 : Move to Goal Behavior

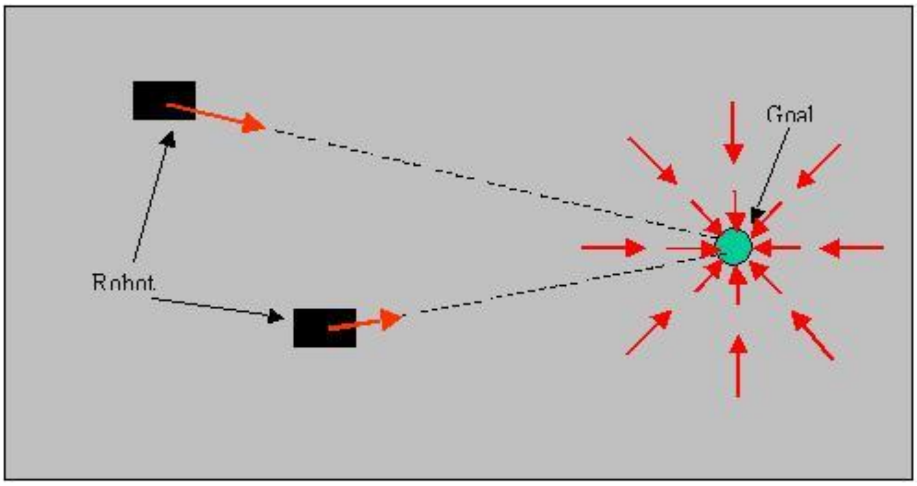


Figure 2-7 : Move to Goal Behavior vectors

2.1.2.3 Avoid Obstacle Behavior

The input of the behavior is the sensory information about the relative position of the obstacle, and the output is the steer angle and speed (Figure 2-8). This behavior forms a vector that is pushing the robot away from the obstacle (Figure 2-9). It is the reverse of the Move to Goal Behavior. The magnitude of the vector increases, as the robot gets closer to the obstacle.

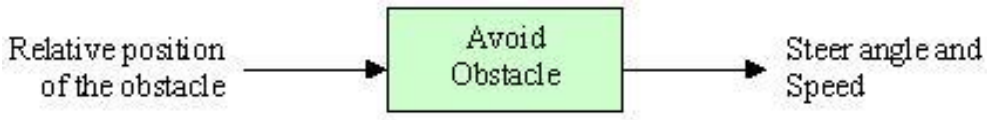


Figure 2-8 : Avoid Obstacle Behavior

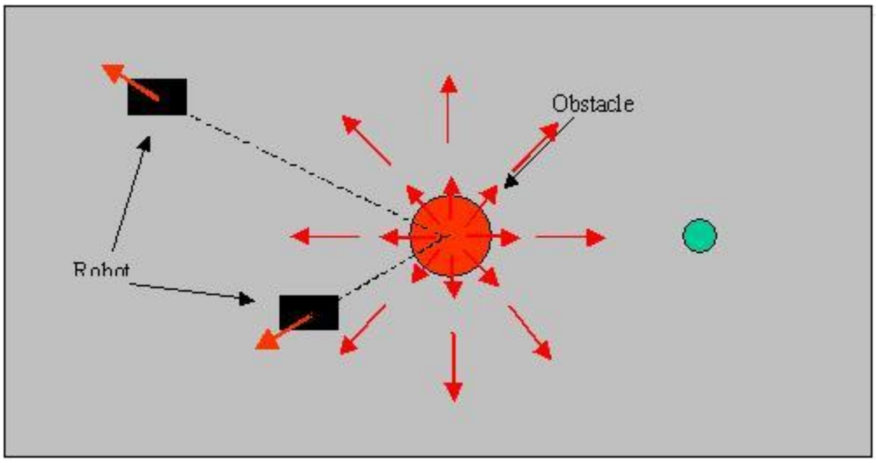


Figure 2-9 : Avoid Obstacle Behavior vectors

2.1.2.4 Track Behavior

The robot tracks a moving object or a route. This behavior is a modified version of Move to Goal Behavior. This time, the Goal is moving (Figure 2-10). The goal attractor vector changes during the voyage (Figure 2-11).

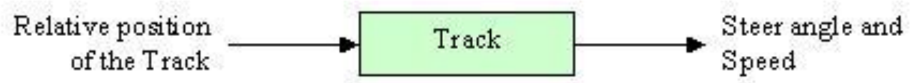


Figure 2-10 : Track Behavior

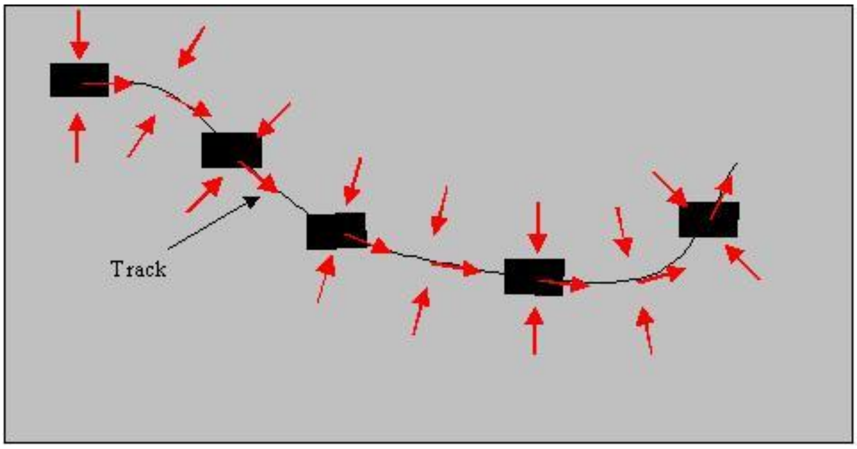


Figure 2-11 : Track vectors

2.1.2.5 Head On Behavior

This behavior provides the robot to look forward while moving (Figure 2-12). The behavior tries to decrease the steer angle down to 0 by changing the orientation θ (Figure 2-13).

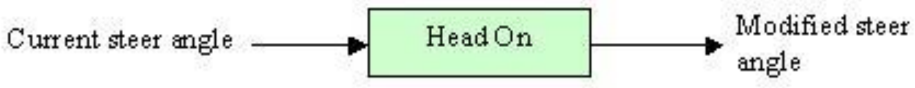


Figure 2-12 : Head-on behavior

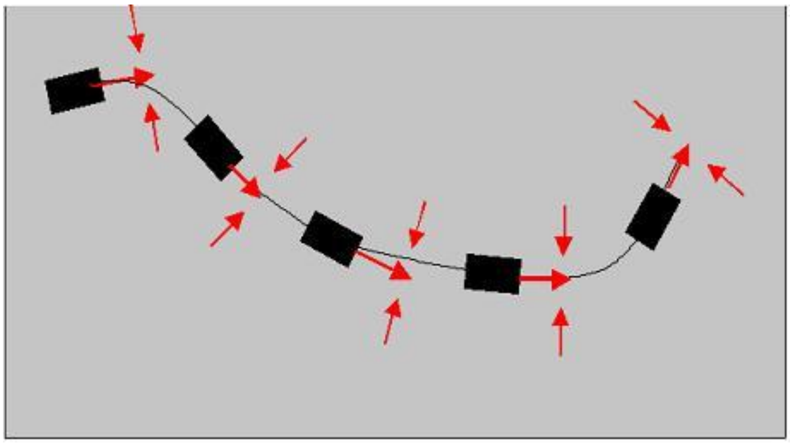


Figure 2-13 : Head-on Behavior vectors

2.1.2.6 Wall Following Behavior

By using this behavior, if the robot goes close to the walls, it starts to follow the wall by trying to keep the distance constant (Figure 2-14), and turn to the empty side if it faces a wall in front (Figure 2-15).

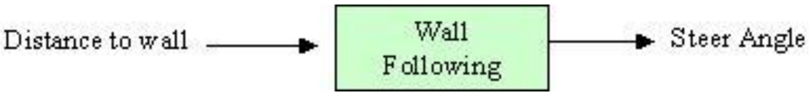


Figure 2-14 : Wall Following Behavior

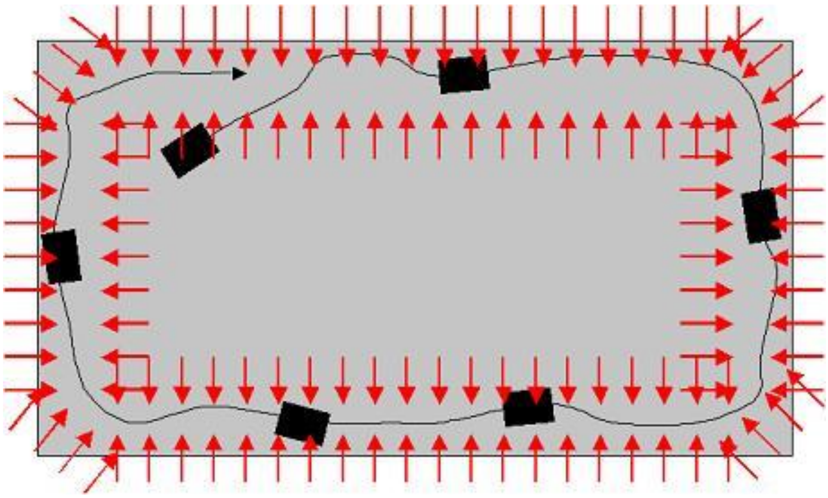


Figure 2-15 : Wall Following Behavior vectors

2.1.3 Behavior gain

The presence of a stimulus is necessary but not enough to evoke a motor-response in a behavior-based robot [3].

The level of the stimulus is a reference for the level of the reaction. There may be a threshold level or a continuous path. The definition of the behavior determines this path. The relation between the sensors and the actuators may be any type (Figure 2-16).

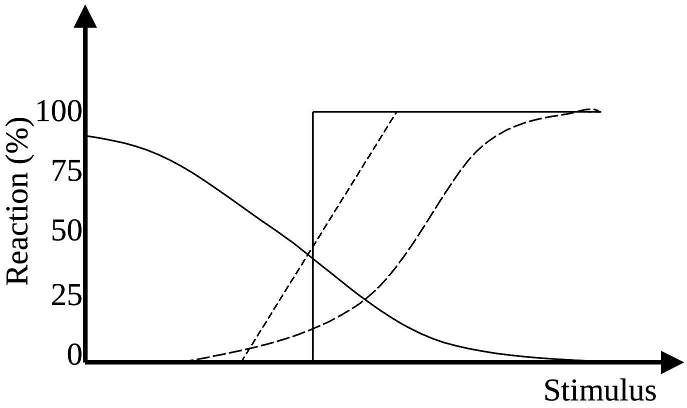


Figure 2-16 : Stimulus vs. Reaction

The reaction level can be controlled with behavior gains. The behavior may be shut down with a zero gain or the force vector magnitude may be increased by increasing gain (Figure 2-17).

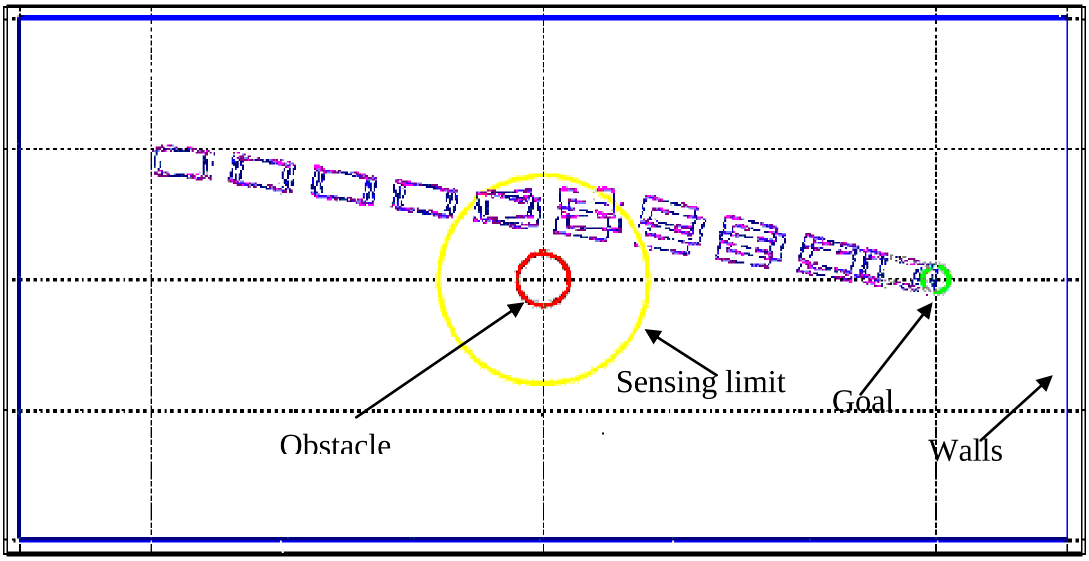


Figure 2-17 : Changing Avoid Obstacle Behavior gain from 0 to 2 and 3.5

2.2 Combining Behaviors

2.2.1 Coordination

The individual behaviors are dedicated to make certain jobs. These behaviors should be coordinated and used in parallel in order to have an intelligent system (Figure 2-18). The combination of these individual behaviors constitutes an undefined behavior to emerge that is called the “Emergent Behavior”.

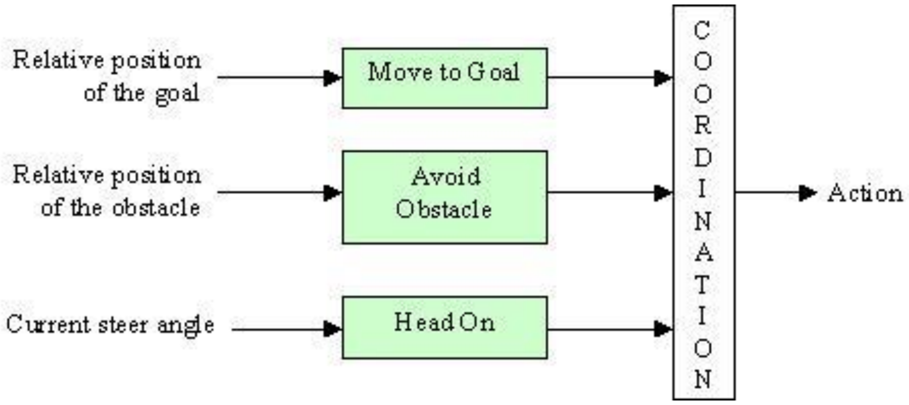


Figure 2-18 : Behavior coordination

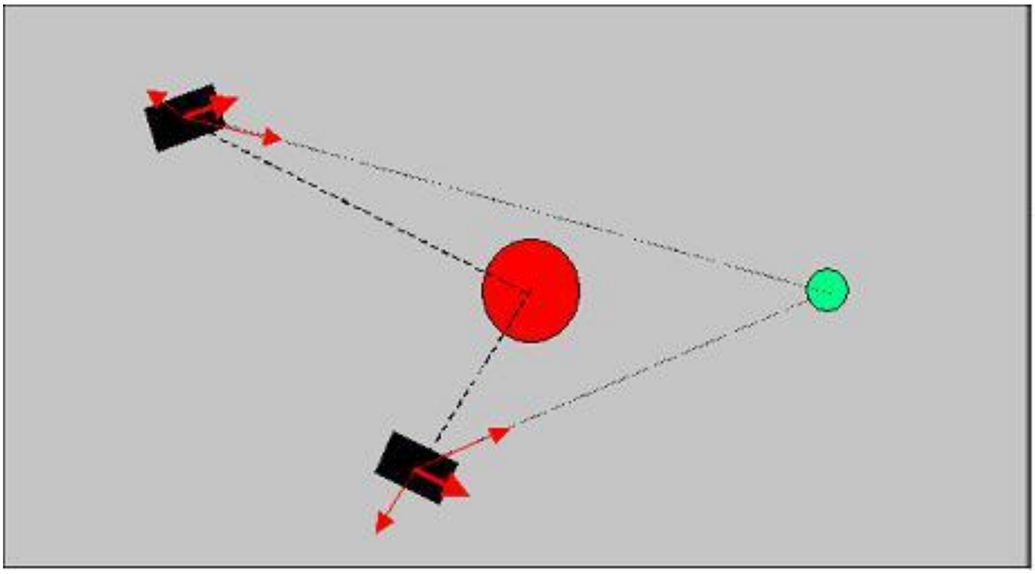


Figure 2-19 : Coordination of Avoid Obstacle and Move to Goal Behaviors

The resulting vectors of the behaviors are combined in a coordination mechanism (Figure 2-19, Figure 2-20). This coordination may be purely additive or may contain some strategies to coordinate the behaviors. Behaviors do not always give similar responses, sometimes there may be conflicts between the responses of different behaviors. There are two main types of coordination functions, competitive methods and cooperative methods.

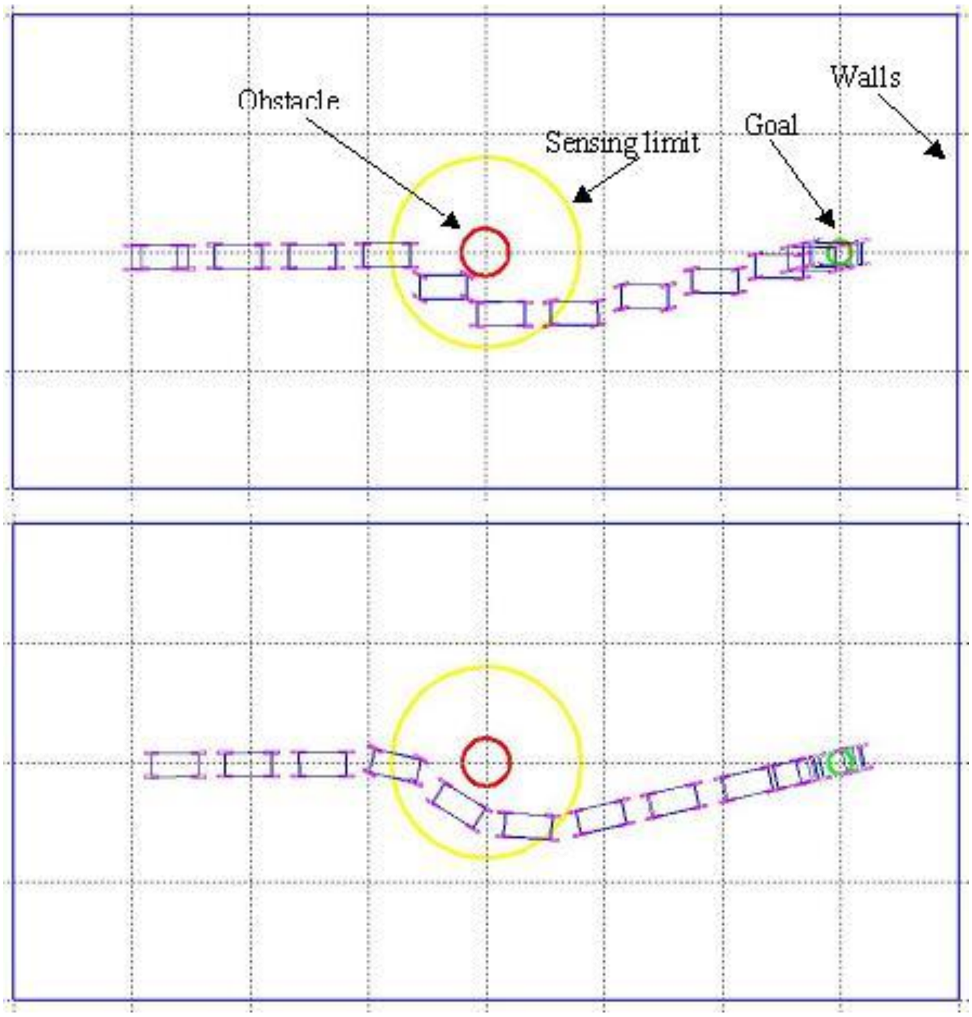


Figure 2-20 : Simulation runs for Avoid Obstacle + Move to goal (upper) and Avoid Obstacle + Move to goal and Head-on (lower) Behaviors

2.2.1.1 Competitive Methods

Different behaviors compete to be the only one whose response is used. The rules of competition are different in different approaches, but at last, only a single behavior's responses or a single response for all behaviors is applied to the motors, and the others are ignored (Figure 2-21).

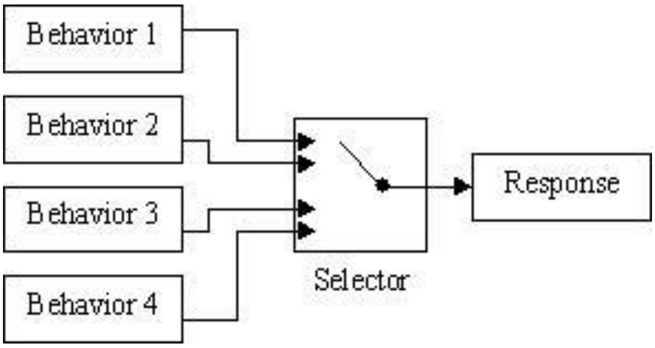


Figure 2-21 : Competitive methods

- **Arbitration:** There is a strict hierarchy between the behaviors. The behavior with a higher dominance competes over the one with a lower dominance.
- **Action-selection:** There is no predefined hierarchy; a behavior is selected according to the present situation or motivation of the robot [32].
- **Voting:** All the behaviors have predefined vote distribution for each response. The response with the most votes is selected to be executed, and other responses are ignored (Figure 2-22). This strategy selects one of the responses instead of selecting one of the behaviors [33].

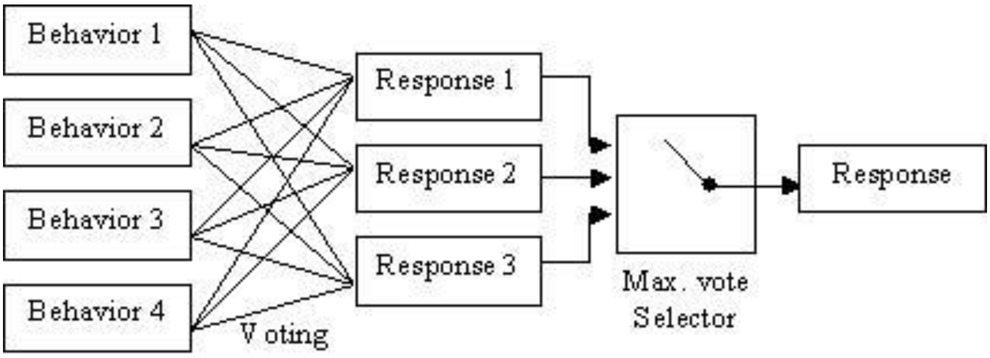


Figure 2-22 : Voting method

2.2.1.2 Cooperative Methods

The advantage of cooperation is the ability to use the responses of different behaviors at the same time (Figure 2-23). Every behavior has some addition to the response at some orders. This order can be arranged by the gains of the behaviors. The resulting response may need to be limited, normalized or modified to avoid extreme conditions.

Potential-fields: The simplest way to cooperate different behaviors is the vector calculation or superposition of the gained responses of each behavior [3].

Fuzzy: The behaviors and responses are processed as fuzzy sets [21-23].

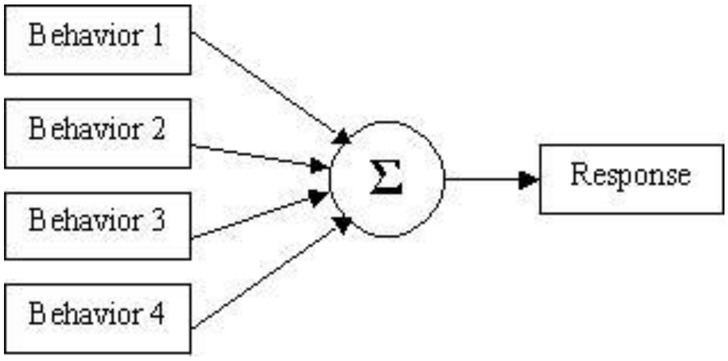


Figure 2-23 : Cooperative methods

2.3 Behavior-Based Architectures

The classical robot architectures lie on vertical “sense-plan-act” strategy. This property of the classical approach has some disadvantages in the real world applications especially because of their complexity, time consuming calculations, and costs. Even this strategy is detrimental to the construction of real working robots and led robotics researchers in the wrong direction [1].

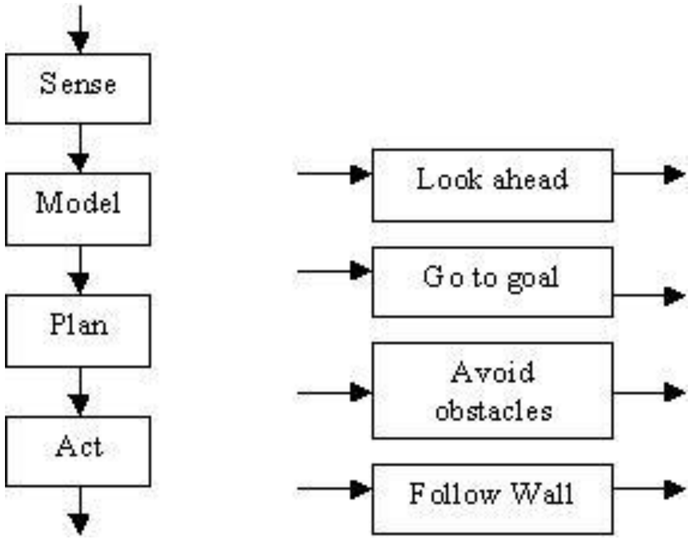


Figure 2-24 : Vertical (left) and horizontal (right) structures

Behavior-based systems have a horizontal architecture (Figure 2-24). This type of reactive organization provides the behavior-based system to have several advantages in real-time applications because of its reactiveness and computation speed (Figure 2-25).

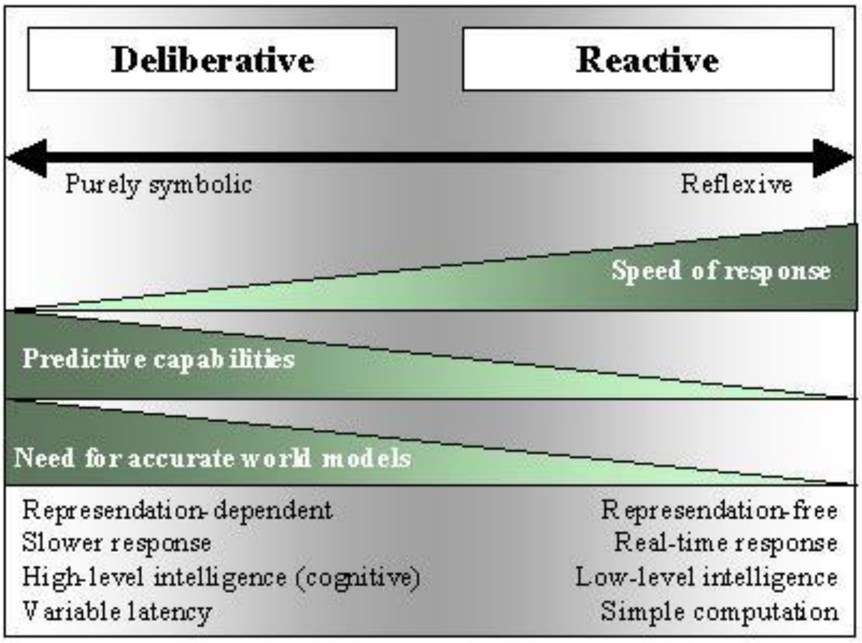


Figure 2-25 : Deliberative vs. Reactive [3]

2.3.1 Subsumption Architecture

Subsumption architecture, developed by R. Brooks in mid-1980s, has a leveled organization of the behaviors. These levels have a hierarchical structure. This hierarchy is built by the coordination of the behaviors in that level (Figure 2-27). Lower levels never rely on the existence of higher levels. Similar to competitive arbitration, the behaviors have two primary mechanisms for coordination:

Inhibition: preventing a signal to be transmitted to the actuators

Suppression: preventing and replacing a signal with a suppressing message.

The lowest level behaviors are called “Augmented Finite State Machine” (AFSM) and they may be reset, inhibited or suppressed by other active AFSMs (Figure 2-26).

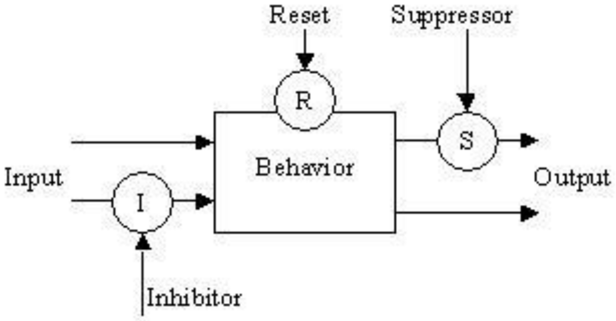


Figure 2-26 : AFSM

All AFSMs have different jobs and they perform their actions by their own perception. There is no global representation or model.

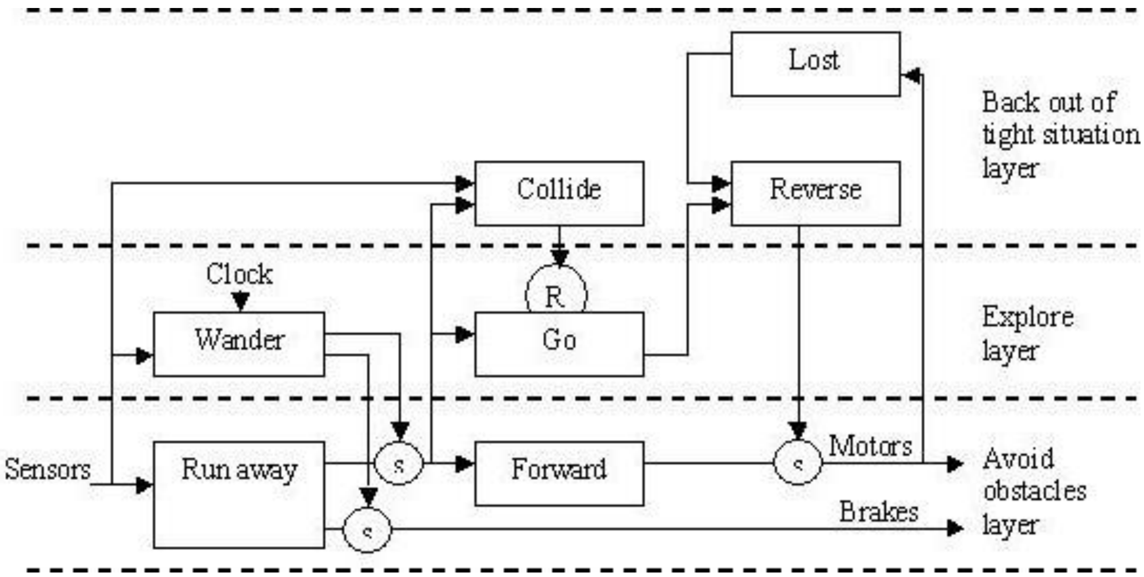


Figure 2-27 : Subsumption Architecture

One important drawback of the subsumption architecture is, the organization between the behaviors and levels gets more complicated by the increasing number of behaviors.

2.3.2 Motor Schemas

Motor schema method [3] uses the potential fields method to coordinate the behaviors. Unlike the subsumption method, there is no predefined hierarchy between the behaviors, all the behaviors may contribute to the overall response of the robot. The software-oriented architecture makes it easy to modify.

There is a perceptual schema in each motor schema (MS). These perceptual schemas (PS) process the information for the motor schema and provide suitable stimuli (Figure 2-28). Each PS can use multiple sensors or outputs of other PS. This property enables the use of multiple sensors for a single sensoriomotor behavior.

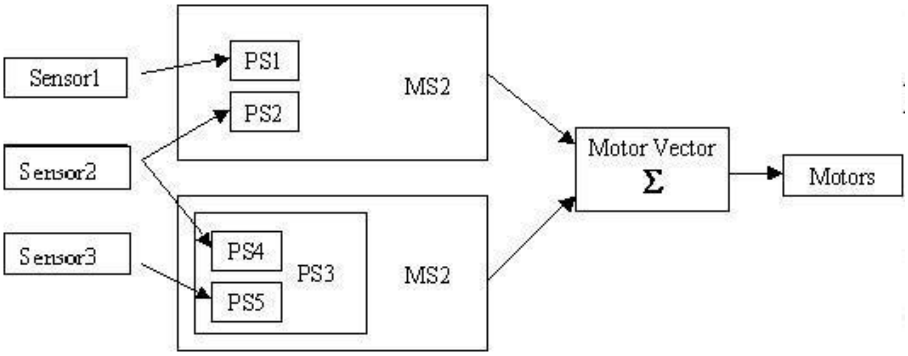


Figure 2-28 : Motor Schema Architecture

2.3.3 Emotion Based Architecture

Considering the nature as a model again, animals manage their behaviors through their motivations, or emotions. The layering problem of subsumption in case of complex behaviors, can be partially solved by emotion-based subsumption. Adding a new behavior becomes easier.

For example, if the robot has "Back-up" behavior in case of a collision, and "Wall following", "Obstacle avoidance", "Wander" behaviors allowing it to safely move, the subsumption hierarchy would be as mentioned. But if a new behavior, "Go to goal" needs to be added to the architecture, a conflict occurs (Figure 2-29). If the "Go to goal" behavior is added at top of "wall following", it will subsume all the behaviors below, including "Obstacle avoidance". If it is placed below "Obstacle avoidance", it will not be able to leave a wall that it is following, and will not be able to go to the goal unless the goal is on the wall.

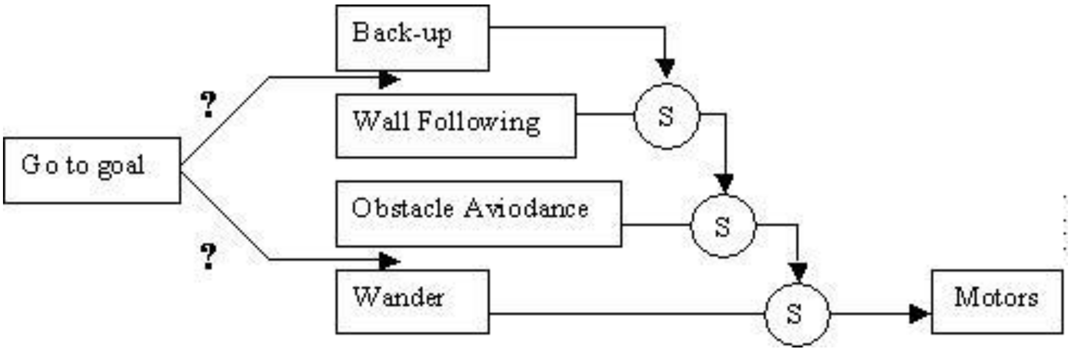


Figure 2-29 : Problem for subsumption

This problem is solved by using emotions at a higher level that is organizing the behaviors (Figure 2-30). If we define the emotions as "Hunger" for the intention to go to the goal, "Secure" for the intention to travel close to wall and "Bored" for the intention to wander in an empty space, the organization with the new behavior "Go to goal" will be as in figure. There is no hierarchy between the behaviors but the emotions.

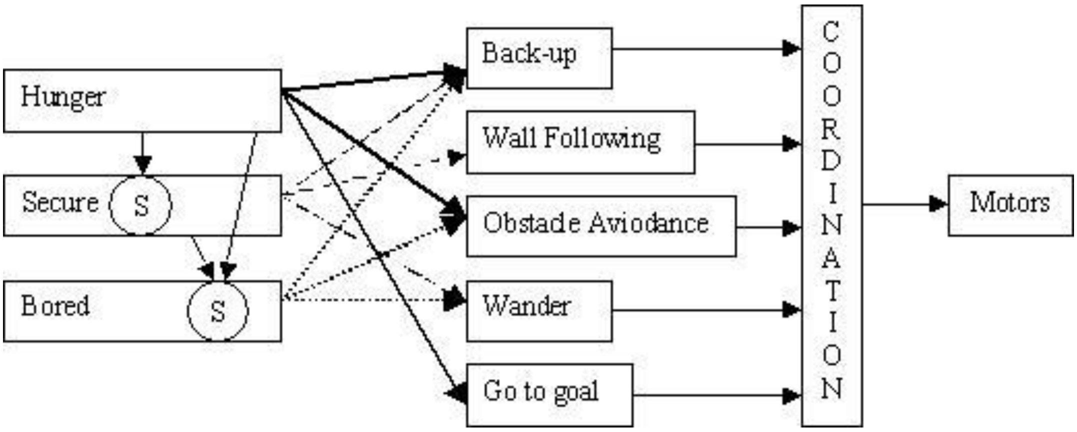


Figure 2-30 : Emotion Based Architecture

2.3.4 Other Methods

There are several other methods used in behavior-based architecture, namely "Circuit Architecture" (Kaelbling and Rosenschein), "Action-selection" (Maes), "Colony Architecture " (Connel), "Animate Agent Architecture (Firby), "DAMN Architecture" (Rosenblatt), "Skill Network Architecture" (Zeltzer), and more.

The common subjects of all these architectures are their avoiding in using representations, using behaviors as building blocks and being reactive. The difference between the architectures is mainly the way they coordinate and manage the behaviors.

3 Fuzzy Behaviors

3.1 Fuzzy Logic

3.1.1 Fuzzy Sets

In order to define a Fuzzy set, the properties of a classical set should be remembered. A classical set has crisp boundaries that strictly define the elements that it contains and does not contain. For example, a classical set of A can be defined as

$A = \{x \text{ I } 30<x<60\}$,

Let this set A be a set of “middle ages”. Looking at this set, one may argue that if 60 is a middle age, then 61 is also a middle age because there is only 1 year between. This problem arises due to the difference in crisp boundaries of classical set and the fuzzy way of human thinking. The sets in the mind work somehow different than classical set theory, they rather work on "fuzzy set theory" [8]. If a normal lifespan is taken as 90 years, a fuzzy set for “middle age”, x, can be defined as

$A = \{(x, \mu_A(x)) \text{ I } x \in X\}$,

Where $\mu_A(x)$ is a membership function (MF) for the fuzzy set A and X is the universe of discourse, i.e. lifespan ($0 \leq X \leq 90$) for this fuzzy set A.

The membership function μ_A can be any function showing the distribution of the universe of discourse on the fuzzy set (Figure 3-1).

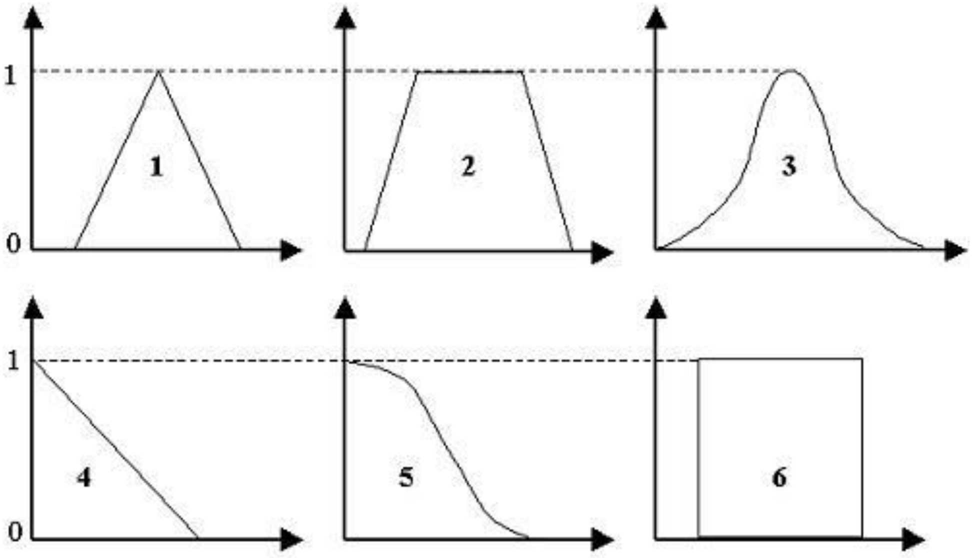


Figure 3-1 : Examples of Membership Functions

The MF value $\mu_A(x)$ varies between 0 and 1 expressing the membership value or membership grade of x to set A. The example number 6 in the figure matches with the definition of classical set theory. This means, the classical sets are a subset of fuzzy sets.

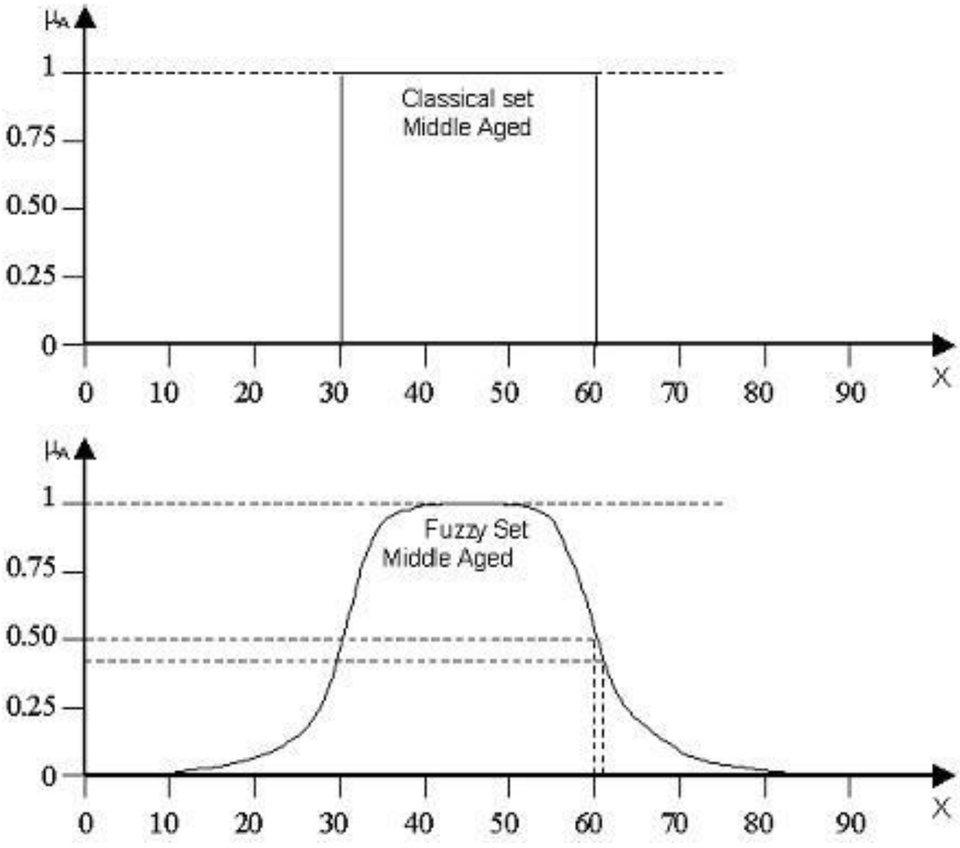


Figure 3-2 : Crisp Set and Fuzzy Set

While the classical set A defines 60 as “Middle Aged” and 61 as not “Middle Aged” the fuzzy set A defines 60 as 0.5 “Middle Aged” and 61 as 0.32 “Middle Aged” (Figure 3-2).

If the whole age groups are defined as “Young” , “Middle Aged” and “Old”, the Figure 3-2 will be Figure 3-3. While the classical set A defines 60 as “Middle Aged”, the fuzzy set A defines 60 as 0,5 “Middle Aged” and 0,3 “Old” (Figure 3-3).

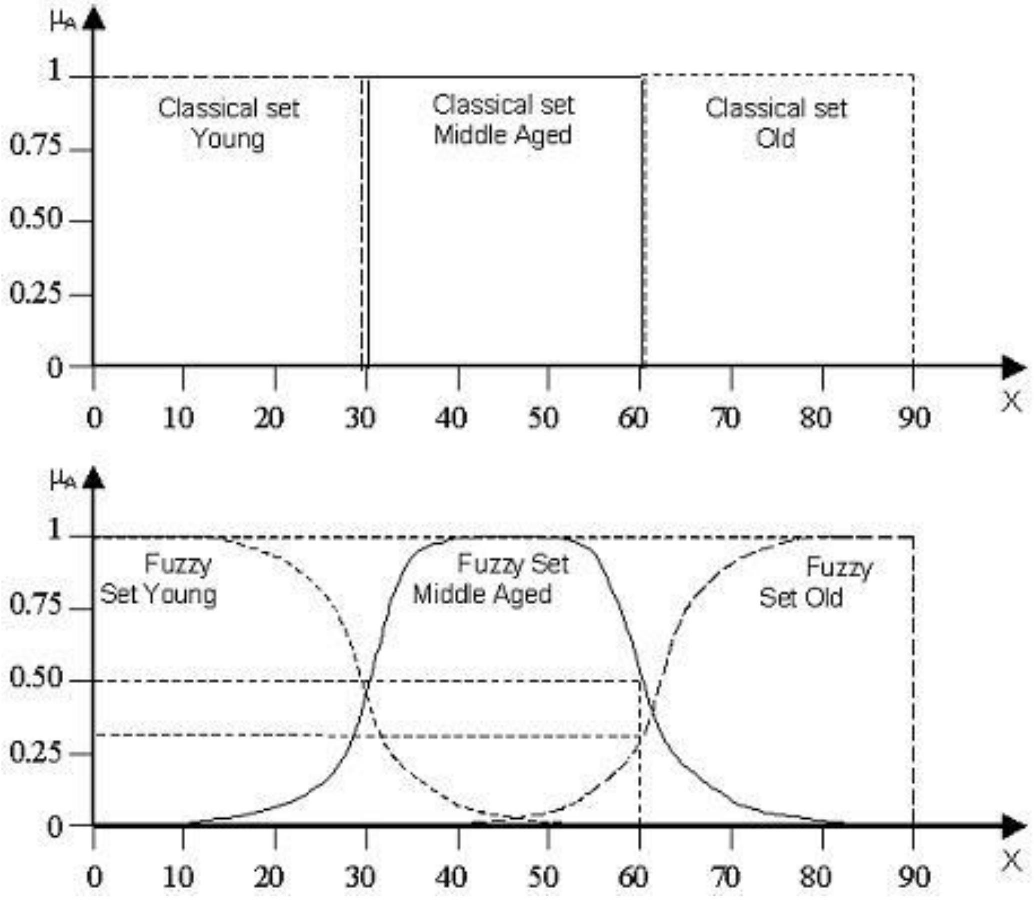


Figure 3-3 : Universe of discourse

3.1.2 If-Then Rules

The If-Then rules are building blocks for a Fuzzy Logic Control structure. In the simplest form, a Fuzzy rule is,

“If (input) is (fuzzy set A1) then (output) is (fuzzy set B2)”

Where the input and output values are linguistic variables with crisp values, A1, A2, A3 are fuzzy sets for input and B1, B2 are fuzzy sets for output. The crisp input value is first converted to a fuzzy value in the fuzzifier. This fuzzy value is carried to the corresponding output fuzzy set according to the fuzzy rule using max-min interfacing. After inferencing, the clipped output fuzzy sets are aggregated and the center of gravity is calculated to obtain the crisp output value (Figure 3-4).

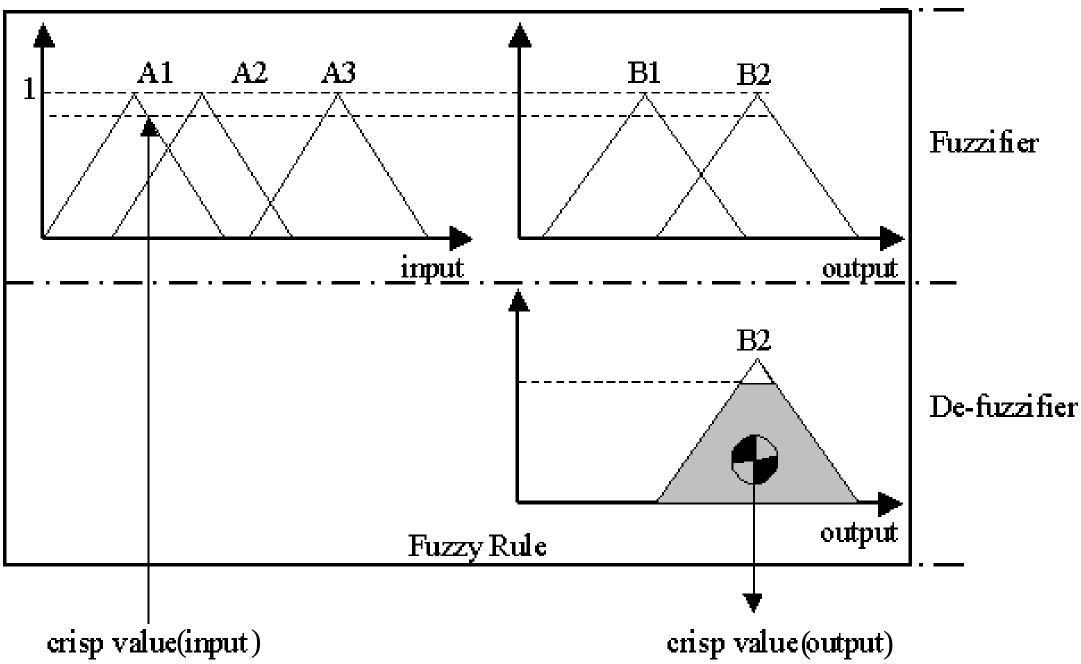


Figure 3-4 : Fuzzification-defuzzification

But In most cases, there are more than one input and one output with logical operations connecting them. These logical operations are basic AND, OR and NOT operations (Figure 3-5).

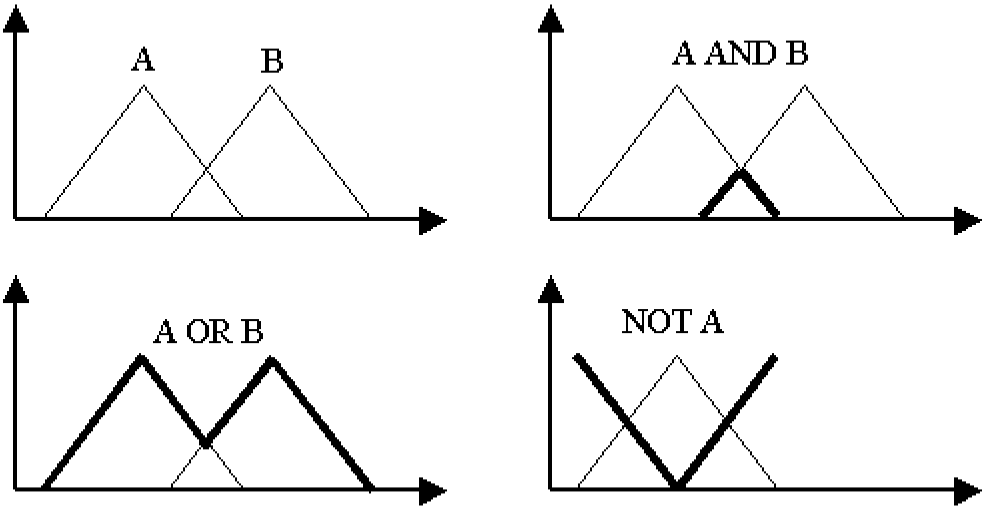


Figure 3-5 : Multivalued logic operations

3.2 Fuzzy Obstacle Avoidance

The obstacle avoidance behavior needs to know the relative place of the obstacle as input and has to output the steering value to the robot. If, in a basic level, the input fuzzy sets from each right and left sensors are “close”, ”far” with universe of discourse 0m < X < 20m and the steer motor output fuzzy sets are “left”, “straight”, “right” with universe of discourse $\pi/2 < X < -\pi/2$ (Figure 3-6).

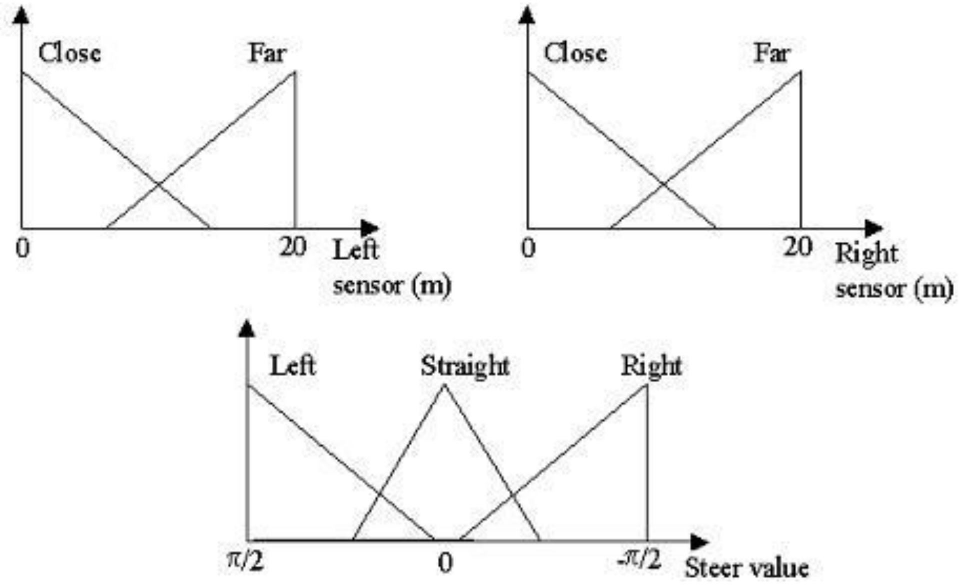


Figure 3-6 : Basic obstacle avoid fuzzy sets

The rule base for the behavior should be in the form

“If (input1) is (fuzzy set A) AND/OR (input2) is (fuzzy set B) then (output1) is (fuzzy set C)”

“If right-sensor is close and left sensor is far then left”

“If left-sensor is close and right-sensor is far then right”

“If right sensor is far and left sensor is far then straight”

The range sensors give the maximum range value when there is no obstacle, i.e., 20m. When the robot encounters with an obstacle at 8m distance on the left side, the first rules work as Figure 3-7.

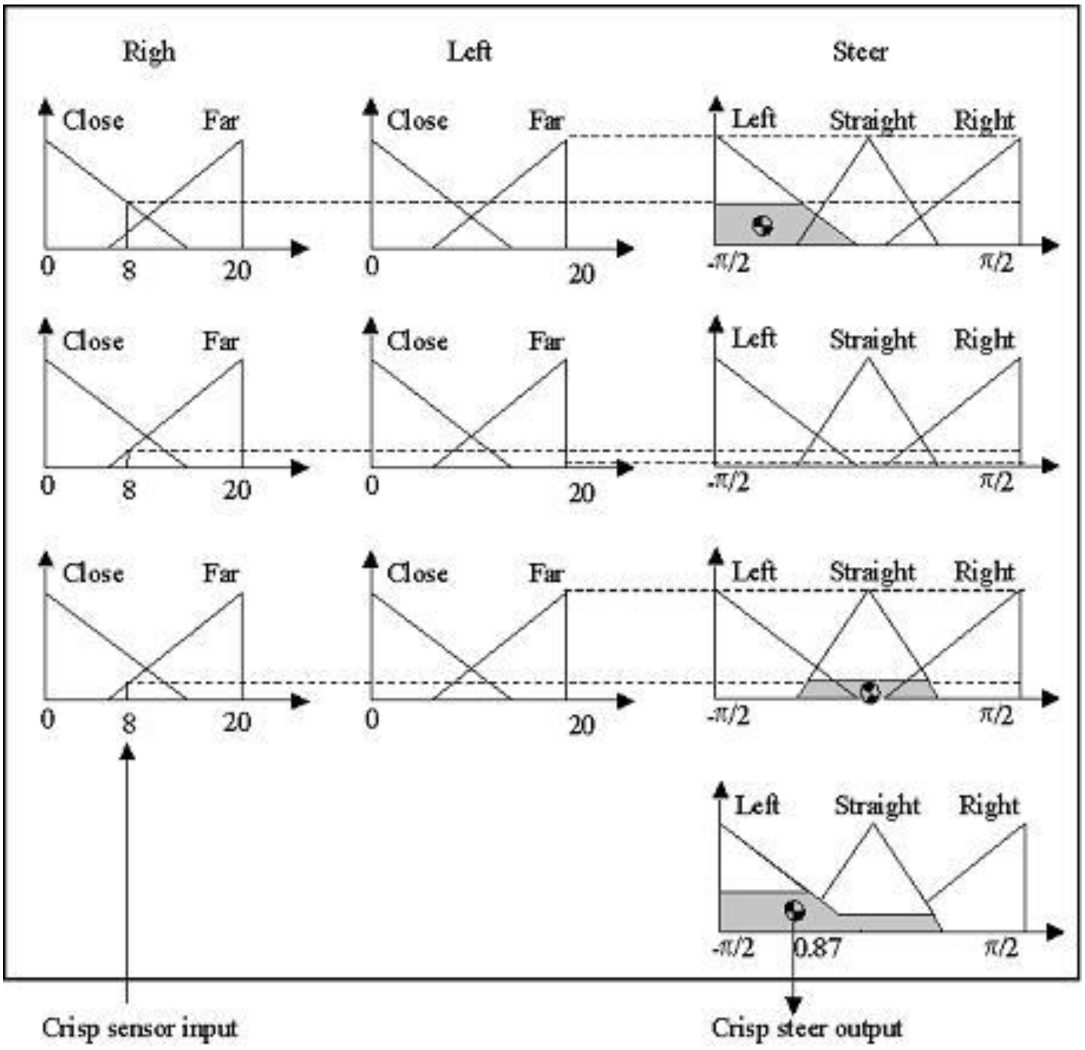


Figure 3-7 : Fuzzy Inference Engine

The output for this situation is calculated as 0.87 rad. in the Fuzzy Inference Engine.

In practice, these sets of fuzzy inputs and output are not enough for the mobile robot for a good obstacle avoidance behavior.

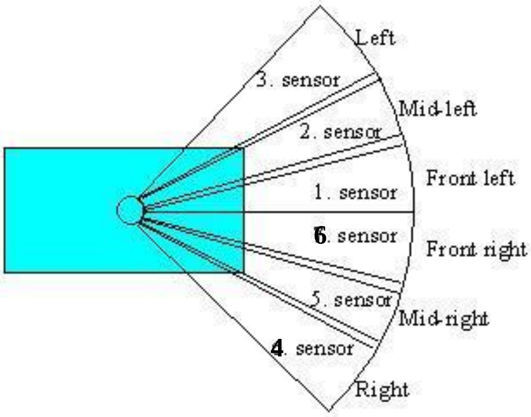


Figure 3-8: Sonar sensor ranges

The mobile robot studied in this thesis has 6 sonar sensors (Figure 3-8) at the front each of which has 16° and 10m range, covering all the front between - 46° and 46° positioned at -38° (most-left), -23° (mid-left), -8° (frt-left), 8° (frt-right), 23° (mid-right), 38° (most-right).

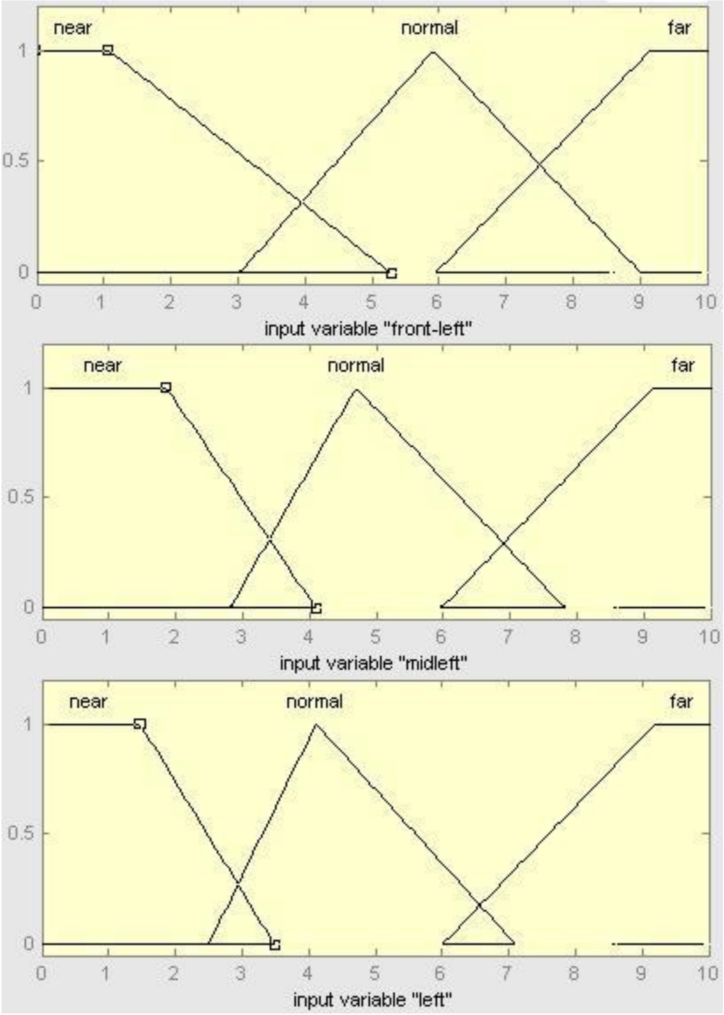


Figure 3-9 : Sensor fuzzy input sets

Each of these sensors has input fuzzy sets “near”, “normal”, “far” (Figure 3-9) with different distributions. The positions of the sensors are important in deciding these distributions; symmetric sensors (1&6, 2&5, 3&4) have the same distribution.

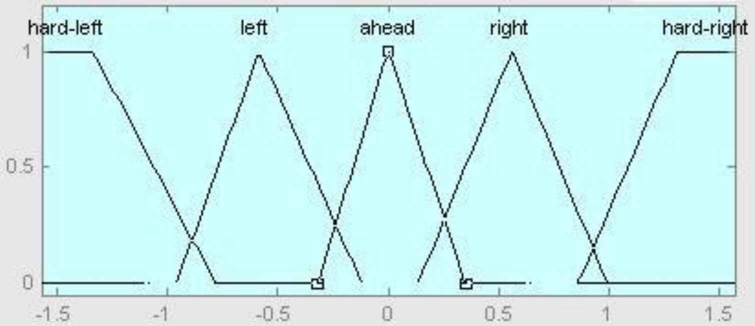
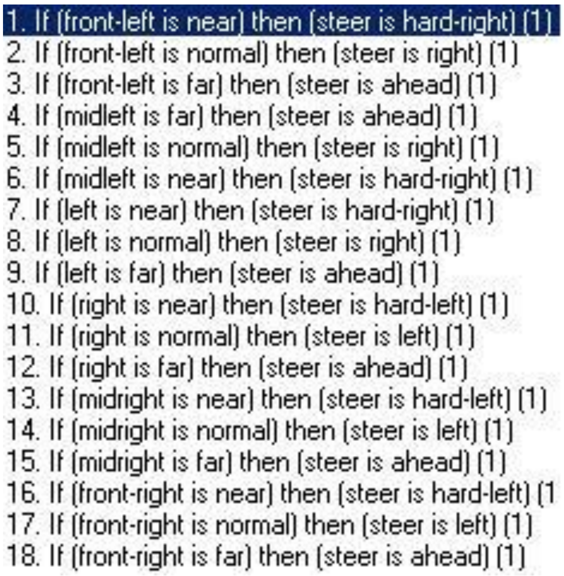


Figure 3-10 : Steer fuzzy output sets

The fuzzy sets for the steering output are “hard-left”, “left”, “ahead”, “right”, “hard-right” and the distribution is in the form in Figure 3-10.

The rule base for obstacle avoidance is in Table 3-1.

Table 3-1: Rule base for fuzzy obstacle avoidance



1. If (front-left is near) then (steer is hard-right) (1)
2. If (front-left is normal) then (steer is right) (1)
3. If (front-left is far) then (steer is ahead) (1)
4. If (midleft is far) then (steer is ahead) (1)
5. If (midleft is normal) then (steer is right) (1)
6. If (midleft is near) then (steer is hard-right) (1)
7. If (left is near) then (steer is hard-right) (1)
8. If (left is normal) then (steer is right) (1)
9. If (left is far) then (steer is ahead) (1)
10. If (right is near) then (steer is hard-left) (1)
11. If (right is normal) then (steer is left) (1)
12. If (right is far) then (steer is ahead) (1)
13. If (midright is near) then (steer is hard-left) (1)
14. If (midright is normal) then (steer is left) (1)
15. If (midright is far) then (steer is ahead) (1)
16. If (front-right is near) then (steer is hard-left) (1
17. If (front-right is normal) then (steer is left) (1)
18. If (front-right is far) then (steer is ahead) (1)

When the obstacle avoidance behavior is tested with move to goal behavior in the simulation, the behavior works perfectly. (Figure 3-13)The sensor outputs and fuzzy engine outputs are shown in Figure 3-11.

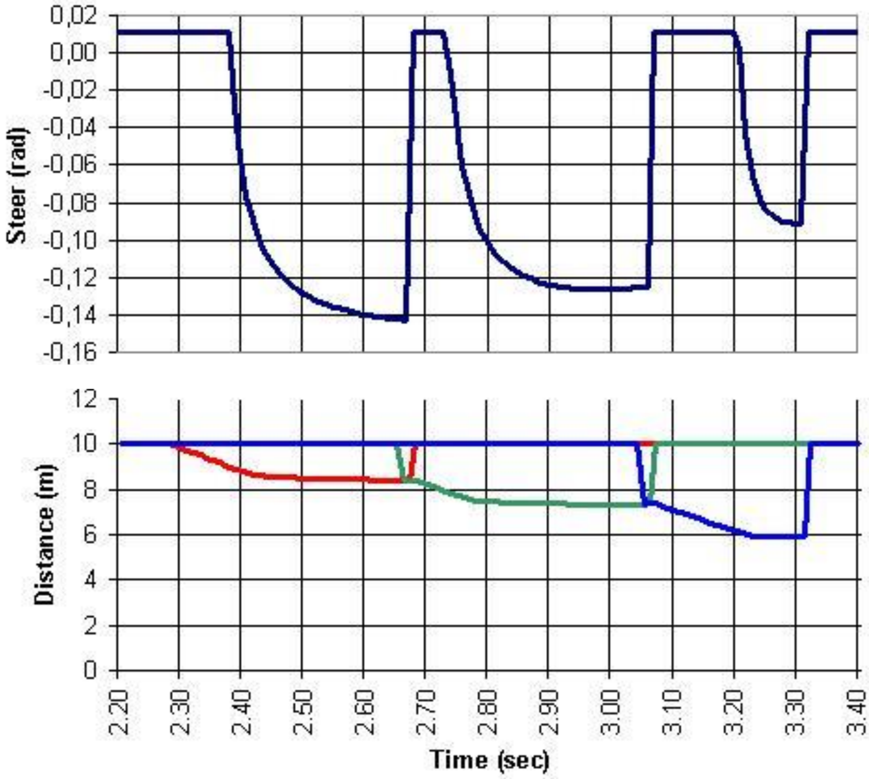


Figure 3-11 : Outputs of the fuzzy engine and the sensors 4,5,6

3.3 Fuzzy Move to Goal

The "Move to Goal" behavior is fuzzified by using the relative distance and the relative angle of the goal as input to the fuzzy engine. There are two outputs of the fuzzy engine, first is the speed and the second is the steer. The input sets are "del-XY" for distance and "del-theta" for the relative angle, and "left", "ahead", "right" for del-theta" (Figure 3-12).

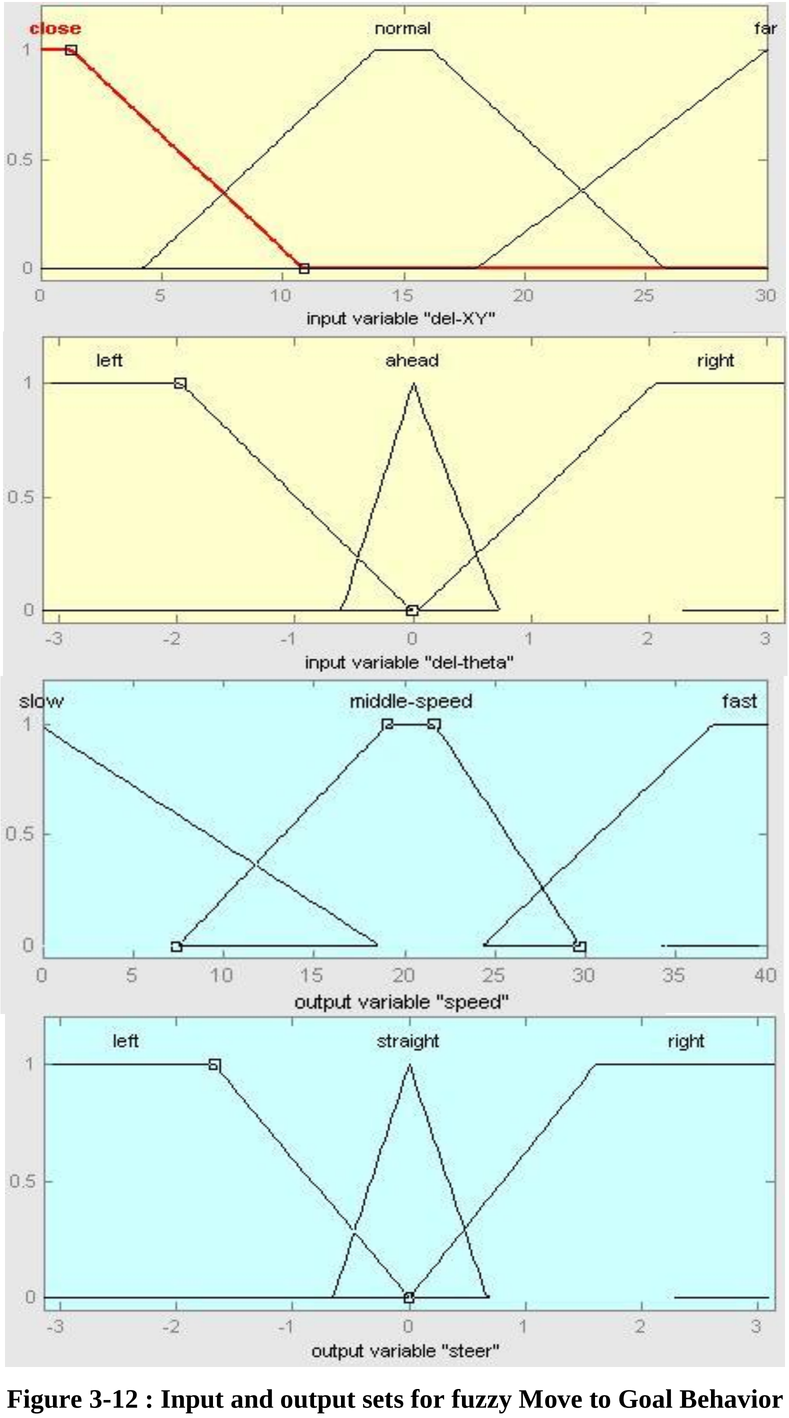
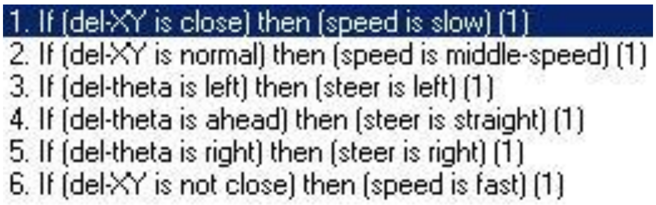


Figure 3-12 : Input and output sets for fuzzy Move to Goal Behavior

There are 6 rules for the Move to Goal fuzzy inference engine. 3 of the rules are for speed output and 3 are for steer output (Table 3-2).

Table 3-2: Rule base for fuzzy Move to Goal



1. If (del-XY is close) then (speed is slow) (1)
2. If (del-XY is normal) then (speed is middle-speed) (1)
3. If (del-theta is left) then (steer is left) (1)
4. If (del-theta is ahead) then (steer is straight) (1)
5. If (del-theta is right) then (steer is right) (1)
6. If (del-XY is not close) then (speed is fast) (1)

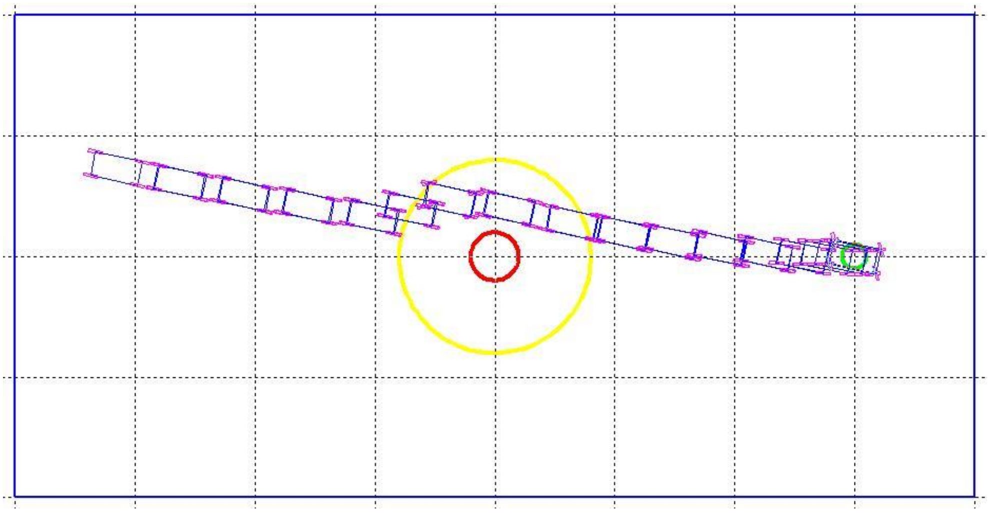


Figure 3-13 : Simulation run with Fuzzy Avoid Obstacle and Fuzzy Move to Goal behaviors

If the simulation is run with the Head-on Behavior, the robot may get into a loop and cannot move away from the obstacle (Figure 3-14). The reasons for this situation are the pure reactive architecture and the positioning of the sonar sensors. The robot does not memorize the place of the obstacle and it goes into the loop "detect obstacle - move away from obstacle - move to the goal - detect obstacle"

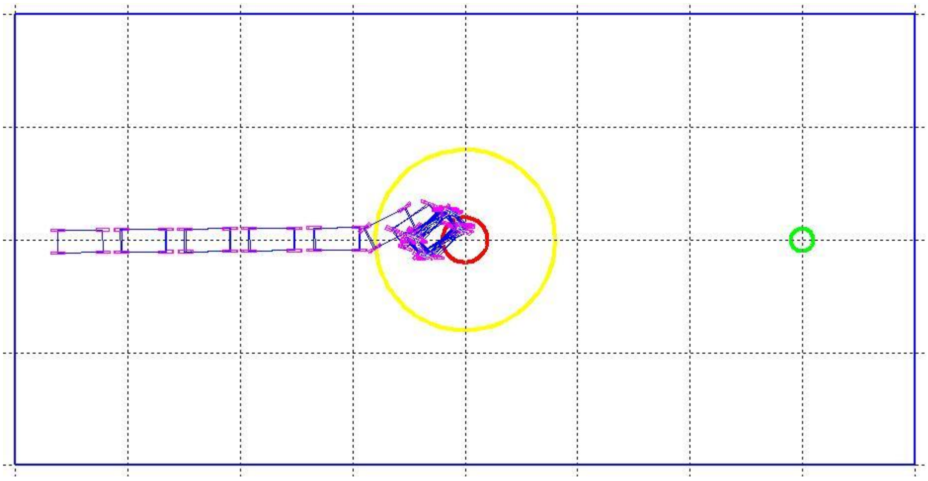


Figure 3-14 : Robot in a loop

One solution for this problem is making the robot less sensitive to the goal by giving it a clear "ahead" concept in the fuzzy engine (Figure 3-15). By doing so, the robot is given a larger steering tolerance at the distances away from the goal. When it approaches to the goal, the angle difference decreases and it has to steer harder to the goal (Figure 3-16).

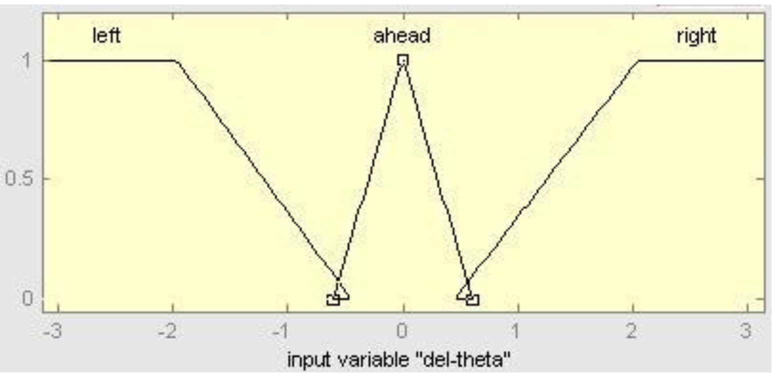


Figure 3-15 : Clear "ahead" concept in the fuzzy engine

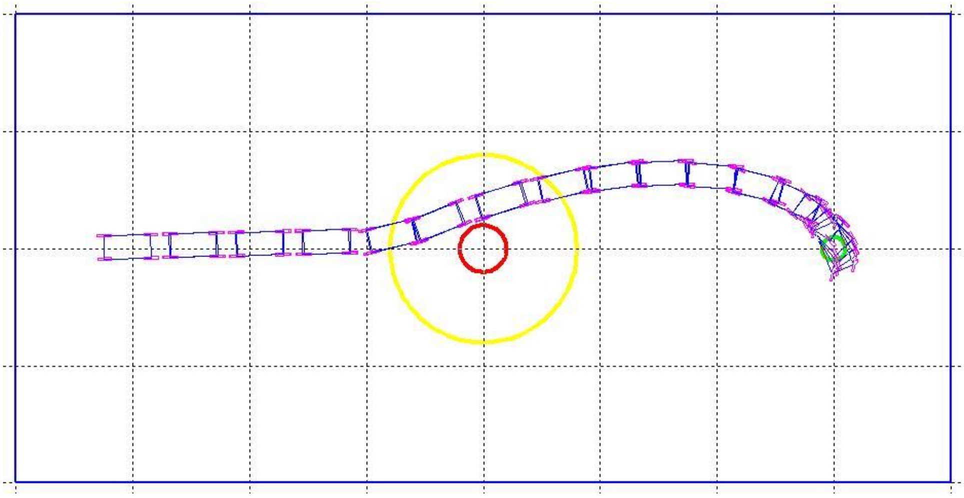
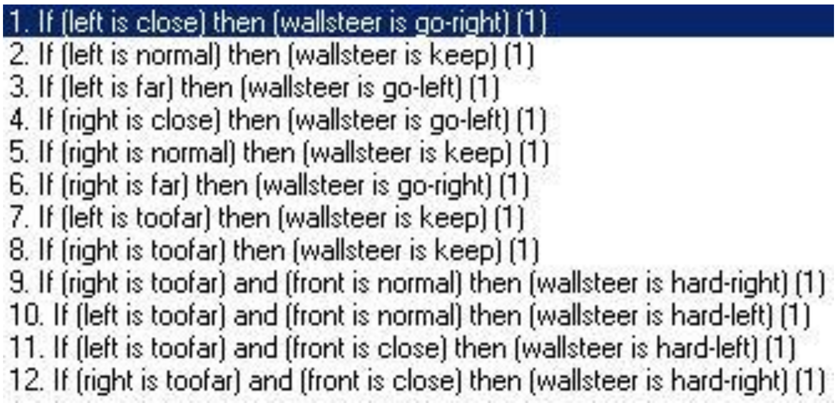


Figure 3-16 : Simulation run with Avoid Obstacle, Head-on and Move-to-Goal Behaviors with clear "ahead"

3.4 Fuzzy Wall Following

The fuzzy wall following behavior uses 3 sensors that are fuzzified as inputs and one steering output. One of these sensors is looking directly to front and two others are looking directly to left and right. The left and right sensors are fuzzified with same membership function sets "close", "normal", "far", "toofar" because of their being symmetric, the front sensor is fuzzified with 3 MFs, "close", "normal", far" (Figure 3-17). The rulebase for Wall Following Behavior is in Table 3-3. The sensor and fuzzy engine outputs of the simulation run (Figure 3-19) is shown in Figure 3-18.

Table 3-3: Rule base for Fuzzy Wall Following Behavior



1. If (left is close) then (wallsteer is go-right) (1)
2. If (left is normal) then (wallsteer is keep) (1)
3. If (left is far) then (wallsteer is go-left) (1)
4. If (right is close) then (wallsteer is go-left) (1)
5. If (right is normal) then (wallsteer is keep) (1)
6. If (right is far) then (wallsteer is go-right) (1)
7. If (left is toofar) then (wallsteer is keep) (1)
8. If (right is toofar) then (wallsteer is keep) (1)
9. If (right is toofar) and (front is normal) then (wallsteer is hard-right) (1)
10. If (left is toofar) and (front is normal) then (wallsteer is hard-left) (1)
11. If (left is toofar) and (front is close) then (wallsteer is hard-left) (1)
12. If (right is toofar) and (front is close) then (wallsteer is hard-right) (1)

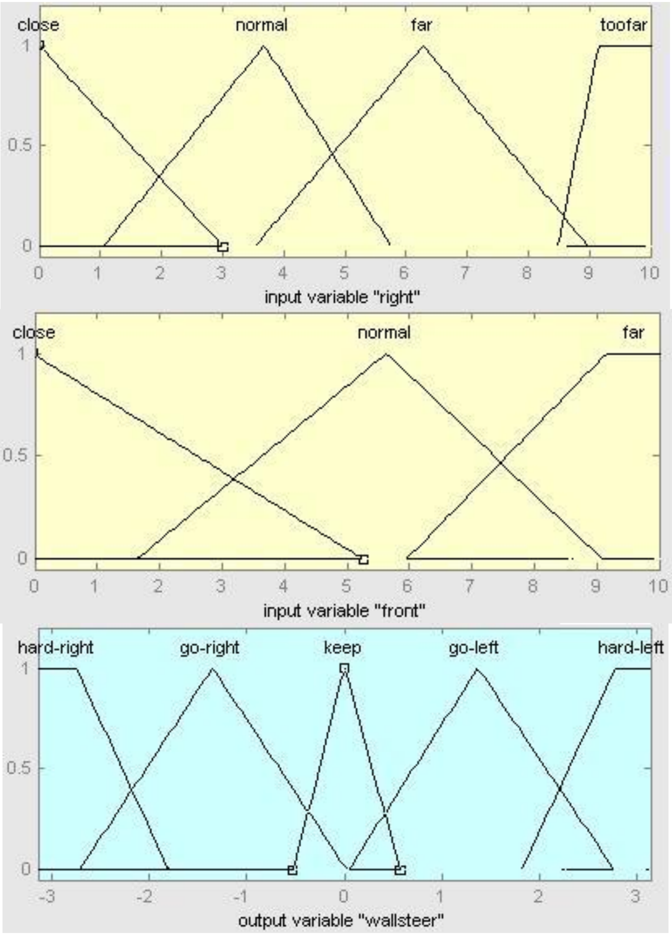


Figure 3-17 : Input and output sets for fuzzy Wall Following Behavior

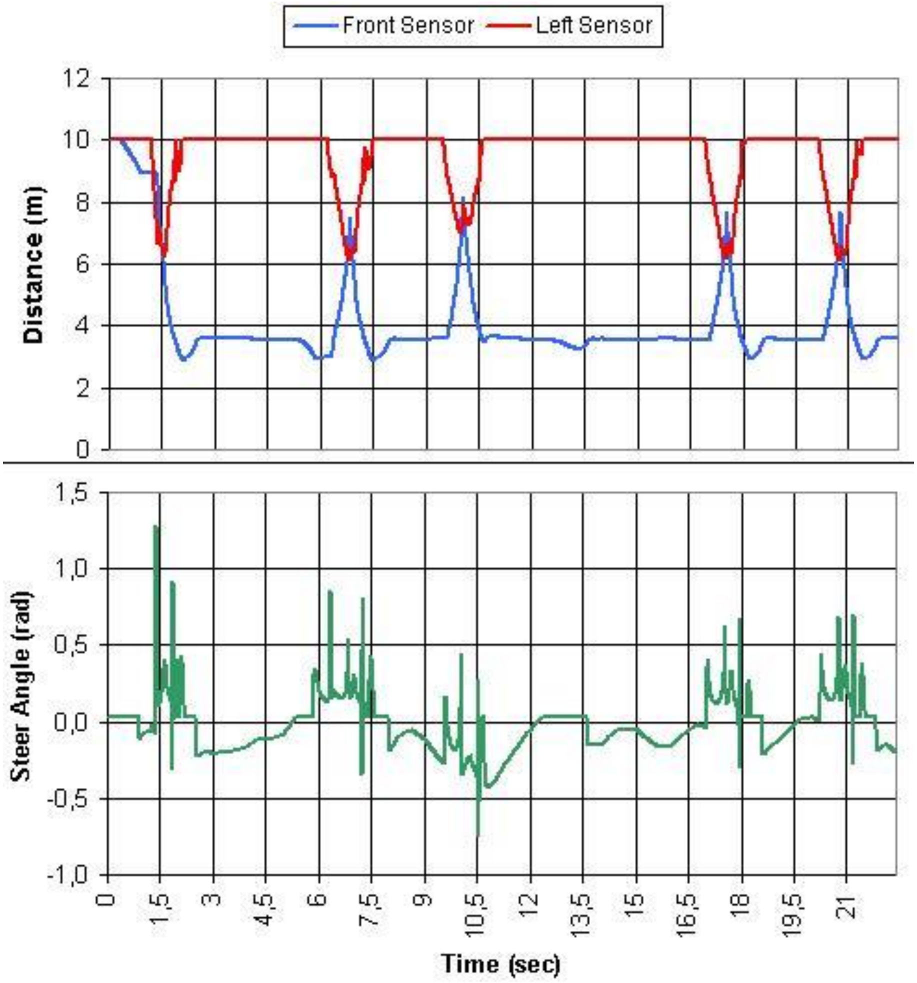


Figure 3-18 : Outputs of Front sensor, Left sensor and Wall Fuzzy engine (top to bottom)

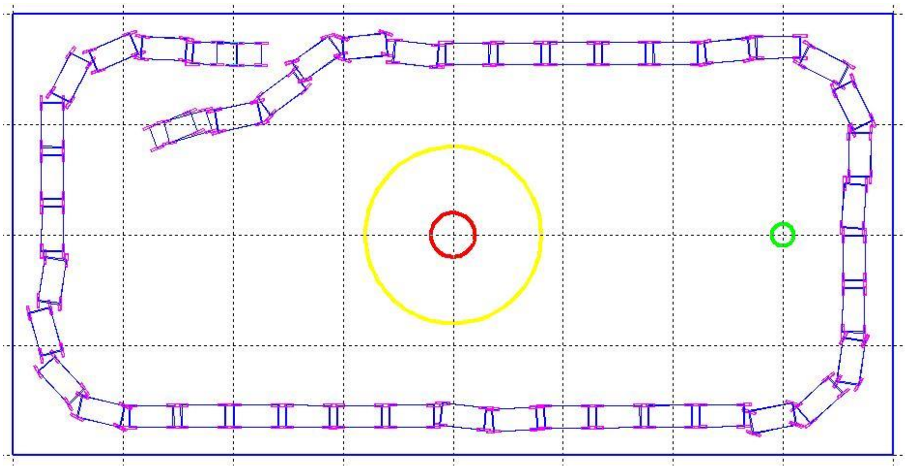


Figure 3-19 : Simulation run with Fuzzy Wall Following, Wander and Head-on Behaviors with constant speed

3.5 Why Fuzzy?

There are several advantages in using fuzzy logic techniques for robotics.

3.5.1 Human-like processing

Fuzzy logic processes the information more similar to human-thinking than other methods. The smooth boundaries and membership concept of fuzzy sets provides the system to “think” wider, and the if-then rules with fuzzy inference, provides the system to make consideration.

The significance of the variables are valued and processed rather than the exact values of them.

3.5.2 Linguistic variables

Unlike other control systems, the information is processed as linguistic variables. It is easy to understand a prebuilt fuzzy system by just looking at the input-output variables and the rules. This property of fuzzy logic makes it easy to modify or change.

The linguistic knowledge and comments of expert or specialist people can also be used in the fuzzy control, making it an experienced system.

3.5.3 Simple Background

The theory behind fuzzy logic is a simple theory with simple mathematical background. There is no need for long challenging formulations. This property also keeps the computational speed at low values for complex systems. As a general rule, the most simple is the best.

3.5.4 Challenging I/O Relations

The input-output relations can be any linear or non-linear function with any complexity. This function can also be implemented into the system by learning-fuzzy techniques [16-18] such as ANFIS (Adaptive Neuro-Fuzzy Inference System).

3.5.5 Blending

Fuzzy logic can be used or blended with other conventional control techniques. The use of fuzzy logic may add simplicity and extend the abilities of the conventional control structures.

3.5.6 Imprecise Data

In practice, there is almost no precise data. All the information that a robot is collecting from the environment has noise and a level of imprecision. The cause of this imprecision in the sensory data is both the sensors and the environment itself. Fuzzy logic is tolerant and shows some buffering to imprecise data with the smooth fuzzy set boundaries.

3.5.7 Sensor Fusion

It is easy to combine different sensory information for one purpose with the rule-based structure of Fuzzy Logic. Also the MFs are helpful for combining different types of sensors because the distribution and universe of discourse of the MFs can be arranged according to the output characteristics of each sensor.

4 Modeling of 4 Wheel Steered 4 Wheel Driven Mobile Robot

4.1 Kinematics

4.1.1 Robot Posture

Robot posture is the position and orientation of the robot frame according to a reference frame O. The robot frame is the x-y frame fixed on the body of the robot as in Figure 4-1.

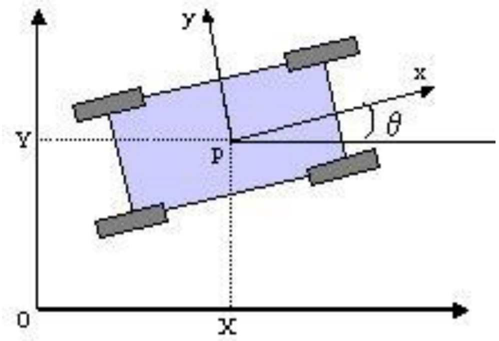


Figure 4-1 : Robot posture

The posture is a vector composed of position “X,Y” and orientation “θ” of the robot as;

$$\xi = \begin{bmatrix} X & Y & \theta \end{bmatrix}^T \qquad \textbf{(4.1)}$$

The rotation matrix for these two frames orthogonal to the plane is;

$$R(\theta) = \begin{bmatrix} \cos(\theta) & \sin(\theta) & 0 \\ \text{-}\sin(\theta) & \cos(\theta) & 0 \\ 0 & 0 & 0 \end{bmatrix} \qquad \textbf{(4.2)}$$

4.1.2 Wheels

4.1.2.1 Pure Rolling, No-Slip

There are four centered orientable wheels, each of which is able to rotate about the vertical axis passing through the contact point of the wheel (steer) and horizontal axis passing through the center of the wheel (drive). The steer angle is “β_i” and the drive angle is “φ_i” for i^{th} wheel (Figure 4-2).

The velocity of the contact point of the wheel to the ground should be zero in order to satisfy the pure rolling no-slip assumption (Figure 4-3). The equations constraining the motion of the wheel are;

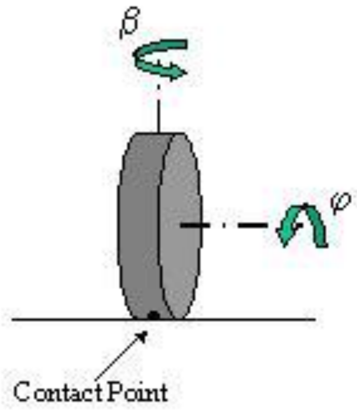


Figure 4-2 : Off-centered wheel

Pure rolling;

$$[-\sin(\gamma+\beta) \quad \cos(\gamma+\beta) \quad l\cos(\beta)]R(\beta)\dot{\xi} + r\dot{\varphi} = 0 \qquad \textbf{(4.3)}$$

No-slip;

$$[\cos(\gamma+\beta) \quad \sin(\gamma+\beta) \quad l\sin(\beta)]R(\beta)\dot{\xi} = 0 \qquad \textbf{(4.4)}$$

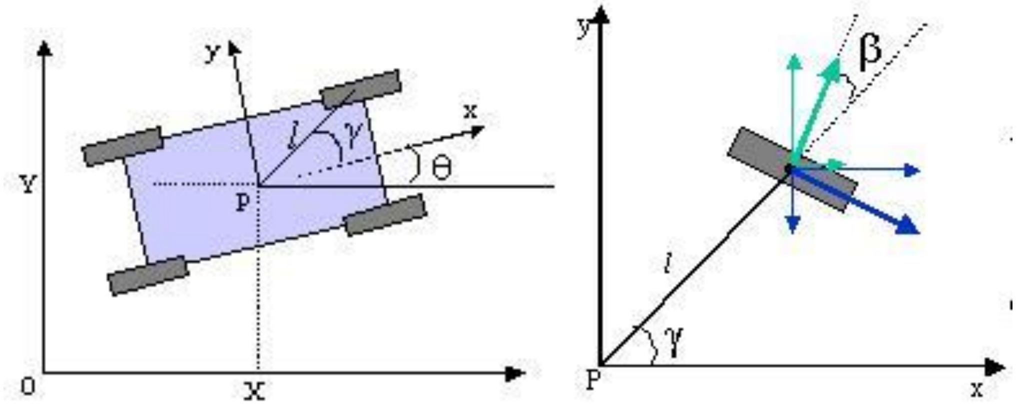


Figure 4-3 : Pure rolling, no-slip constraints

4.1.2.2 ICR

According to Descartes’ principle of instantaneous motion; at each instant, the motion of a planar rigid body coincides either with a pure translation, or with a pure rotation about some point, termed instantaneous center of rotation.

This principle is also applies to any point on the robot which is assumed to be rigid and moving on a horizontal plane. The Instantaneous Center of Rotation (ICR) may be anywhere between the moving point itself and infinity. If we apply this rule to the wheel-centers, we got the following figure (Figure 4-4 : ICR).

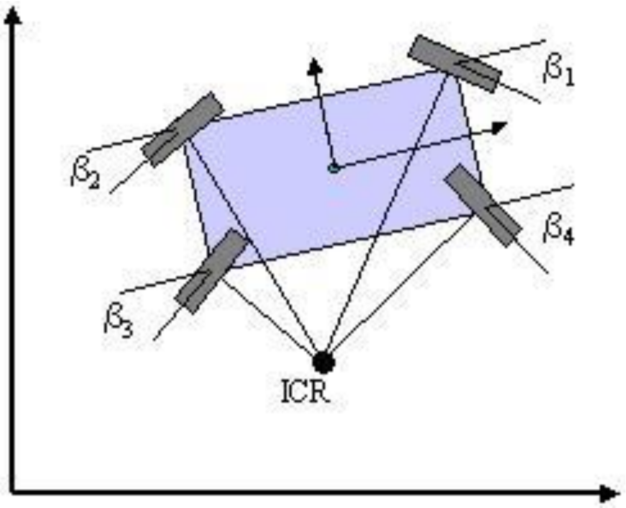


Figure 4-4 : ICR

This phenomenon implies that the orientations of the wheels have some constraints, i.e. the horizontal axes of the wheels should coincide at ICR. These axes do not coincide if and only if the ICR is at infinity i.e. the wheels are parallel and the robot has only transverse motion.

Since two coinciding lines are enough to describe a point, orientations of any two wheels (β_c), for instance β_1 and β_2, are enough to describe the position of the ICR. Once the ICR is located, the orientations of the other two wheels can be calculated as;

$$\beta_3 = \tan^{-1}\left(\frac{\cos(\beta_1)\sin(\beta_2)}{\cos(\beta_1)\cos(\beta_2) + \frac{d_{14}}{d_{12}}\sin(\beta_{2-}\beta 1)} \right) \qquad \textbf{(4.5)}$$

$$\beta_4 = \tan^{-1}\left(\frac{\sin(\beta_1)\cos(\beta_2)}{\cos(\beta_1)\cos(\beta_2) + \frac{d_{14}}{d_{12}}\sin(\beta_{2-}\beta 1)} \right) \qquad \textbf{(4.6)}$$

4.1.2.3 Motion

Considering equations 4.3 and 4.4 for all 4 wheels;

$$J_1(\beta)=\begin{bmatrix}\cos(\beta_1) & \sin(\beta_1) & l_1\sin(\beta_1-\gamma_1)\\ \cos(\beta_2) & \sin(\beta_2) & l_2\sin(\beta_2-\gamma_2)\\ \cos(\beta_3) & \sin(\beta_3) & l_3\sin(\beta_3-\gamma_3)\\ \cos(\beta_4) & \sin(\beta_4) & l_4\sin(\beta_4-\gamma_4)\end{bmatrix} \qquad \textbf{(4.7)}$$

$$J_2=\begin{bmatrix}r_1 & r_2 & r_3 & r_4\end{bmatrix}I_{4\times4} \qquad \textbf{(4.8)}$$

$$C_1(\beta)=\begin{bmatrix}-\sin(\beta_1) & \cos(\beta_1) & l_1\cos(\beta_1-\gamma_1)\\ -\sin(\beta_2) & \cos(\beta_2) & l_2\cos(\beta_2-\gamma_2)\\ -\sin(\beta_3) & \cos(\beta_3) & l_3\cos(\beta_3-\gamma_3)\\ -\sin(\beta_4) & \cos(\beta_4) & l_4\cos(\beta_4-\gamma_4)\end{bmatrix} \qquad \textbf{(4.9)}$$

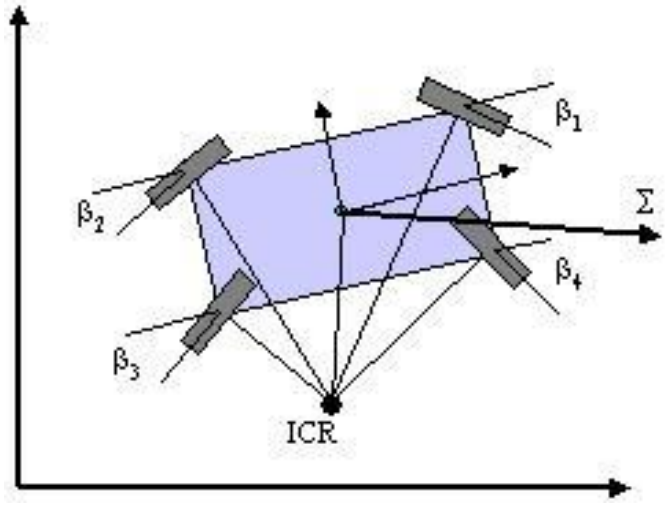


Figure 4-5 : Instantaneous direction of motion

ICR and pure rolling, no-slip constraints describe an instantaneous direction of motion Σ(β_c) for the robot. (Figure 4-5)

Σ (β_c) is perpendicular to the space spanned by C_1, thus;

$C_1(\beta)\Sigma(\beta_c)=0$ and,

$$\Sigma=\begin{bmatrix}l_1\cos(\beta_2)\cos(\beta_1-\gamma_1)-l_2\cos(\beta_1)\cos(\beta_2-\gamma_2)\\ l_1\sin(\beta_2)\cos(\beta_1-\gamma_1)-l_2\sin(\beta_1)\cos(\beta_2-\gamma_2)\\ \sin(\beta_1-\beta_2)\end{bmatrix} \qquad \textbf{(4.10)}$$

On this direction, there is a velocity η(t).

So, the motion of the robot can be described as,

$$\dot{\xi} = R^T(\theta)\Sigma(\beta_c)\eta. \qquad \textbf{(4.11)}$$

This means that, the posture of the robot can be manipulated with a velocity input η(t) at the instantaneous direction of Σ(β$_c$).

4.2 Dynamics

4.2.1 Type of the mobile robot

Wheeled Mobile Robots can be classified into 5 groups. These groups are formed with two parameters; *degree of mobility* δ_m and *degree of steeribility* δ_s, and named as "mobile robot of Type($\delta_{m,}$ δ_s)". Without going into the mathematical descriptions, these parameters can be described as,

Degree of mobility: The number of planar movements (x, y, θ) that a robot can go without changing its wheel configurations. This number is between 1 and 3.

Degree of steeribility: The number of conventional centered orientable wheels that can be oriented independently to steer the mobile robot. This number is between 0 and 2.

According to these explanations, the mobile robot studied in this thesis is a degenerate mobile robot of Type(1,2).

The nondegenerate form of the robot is stated in Figure 4-6.

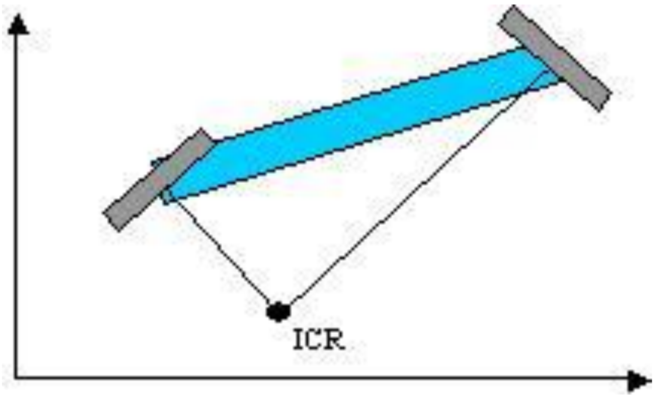


Figure 4-6 : non-degenerate form; bicycle model

4.2.2 Torque calculation

If it is assumed that the mass distribution on the robot frame is symmetric about the x and y axes, the off-diagonal terms of the mass matrix vanishes and the matrix becomes;

$$M = \begin{bmatrix} m_f + 4m_w & 0 & 0 \\ 0 & m_f + 4m_w & 0 \\ 0 & 0 & I_f + m_w \sum_{i=1}^{4} l_i^{\,2} \end{bmatrix} \quad \textbf{(4.12)}$$

Where m$_f$ is the mass of the robot frame and m$_w$ is the mass of the wheel and the moment of inertia of the robot frame is I$_f$ and for each wheel is I$_w$ where J$_\phi$= I$_w$ I$_{4x4}$

Using the Lagrange undetermined coefficients, the general dynamical model can be written as,

$$h_1(\beta)\eta\&+\Phi_1(\beta)\zeta\eta = \Sigma^T E\tau_\phi \quad \textbf{(4.13)}$$

where

$$h_1(\beta) = \Sigma^T\left(M + EJ_\phi E^T\right)\Sigma > 0 \quad \textbf{(4.14)}$$

and

$$\Phi_1(\beta) = \Sigma^T\left(M + EJ_\phi E^T\right)N(\beta_c) \quad \textbf{(4.15)}$$

Where we define

$$E = J_1^T J_2^{-1} \quad \textbf{(4.16)}$$

and

$$N(\beta_c) = [N_1 \quad N_2] \quad \textbf{(4.17)}$$

such that,

$$\begin{aligned} N_1 &= \begin{bmatrix} -l_1\cos(\beta_2)\sin(\beta_1-\gamma_1)+l_2\sin(\beta_1)\cos(\beta_2-\gamma_2) \\ -l_1\sin(\beta_2)\sin(\beta_1-\gamma_1)-l_2\cos(\beta_1)\cos(\beta_2-\gamma_2) \\ \cos(\beta_1-\beta_2) \end{bmatrix} \\ N_1 &= \begin{bmatrix} -l_1\sin(\beta_2)\cos(\beta_1-\gamma_1)+l_2\cos(\beta_1)\sin(\beta_2-\gamma_2) \\ l_1\cos(\beta_2)\cos(\beta_1-\gamma_1)+l_2\sin(\beta_1)\sin(\beta_2-\gamma_2) \\ -\cos(\beta_1-\beta_2) \end{bmatrix} \end{aligned} \quad \textbf{(4.18)}$$

4.2.3 Torque distribution

Having the non-holonomic constraints and the degree of mobility of the robot as 1, theoretically it will be enough to drive only one wheel to move the robot. This phenomenon can be understood better with a “train and railway” analogy (Figure 2-1).

The degree of mobility of a train is also 1, which means that a train can only move on the direction of the railway (i.e. forward or backward). Any one of the wheels or any group of wheels can drive the train supposed that the total torque is enough to move the train. The subject is not the distribution of the total torque, but the total of the torques.

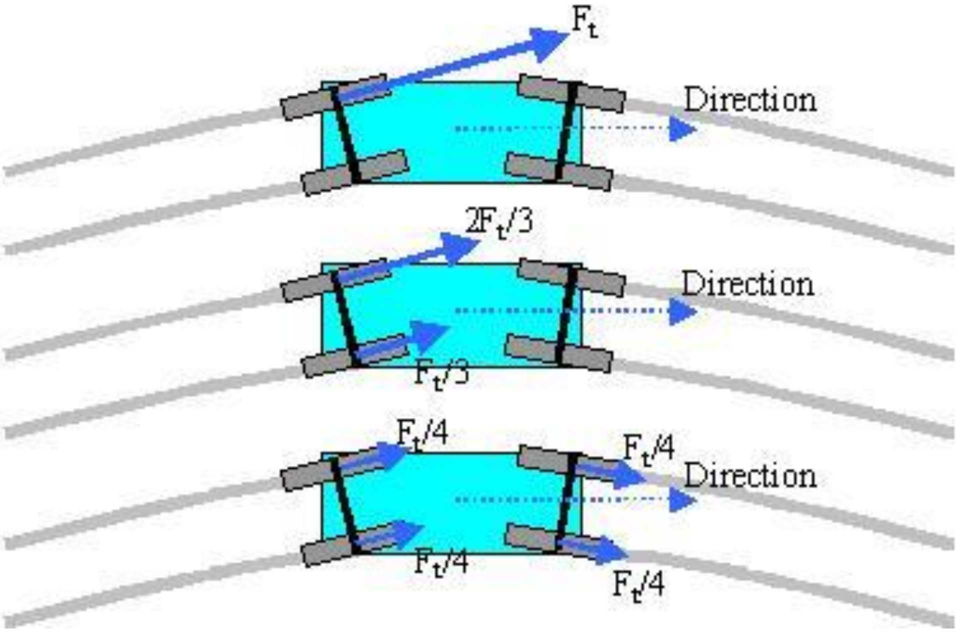


Figure 4-7 : Railway analogy

By dividing the torque to the wheels evenly, the torque value of each wheel is limited in a nominal value (Figure 4-7).

τ_ϕ cannot be extracted from eq. 10 directly. After linearizing eq.10 using computed torque approach, the torques are evenly distributed to each wheel as;

$$\Sigma^T E\tau_\phi = [a_1 \quad a_2 \quad a_3 \quad a_4][\tau_1 \quad \tau_2 \quad \tau_3 \quad \tau_4]^T = h_1(\beta)\eta\dot{\phi} + \Phi_1(\beta)\varsigma\eta = L \qquad \textbf{(4.19)}$$

We set $\tau_\phi = H\tau_0$ and H_i=Lsign(a_i)/σ where σ is the sum of the four elements of the vector Σ^TE.

Now τ_0 can be obtained from

$$\tau_0 = \frac{1}{\Sigma^T EH}\left(h_1(\beta)\eta\dot{\phi} + \Phi_1(\beta)\varsigma\eta\right) \qquad \textbf{(4.20)}$$

4.3 Steering Strategy

4.3.1 Translation

Unlike the car-like vehicles, a 4x4x4 vehicle does not have to be parallel to its track. The vehicle may have a yaw angle between the body orientation and the direction of motion (Figure 4-7).

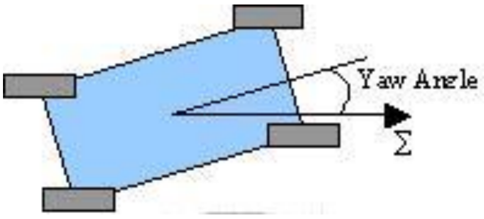


Figure 4-8 : Yaw angle

In order to steer the robot to a point in the plane, simply β_1 and β_2 angles are set to the angle of the vector that is connecting the robot to the point (Figure 4-9).

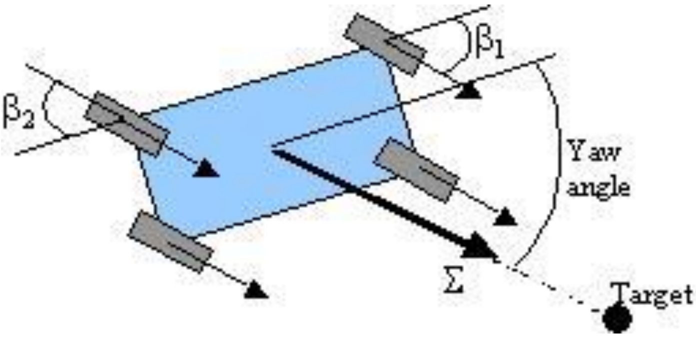


Figure 4-9 : Transverse motion

4.3.2 Rotation

The yaw angle of the robot gives the advantage to control the orientation θ. For a stationary situation, i.e., the ICR is at the center of the robot, the directions of motion for the 4 wheel orientation points on the robot are perpendicular to the line connecting the points to the body center.

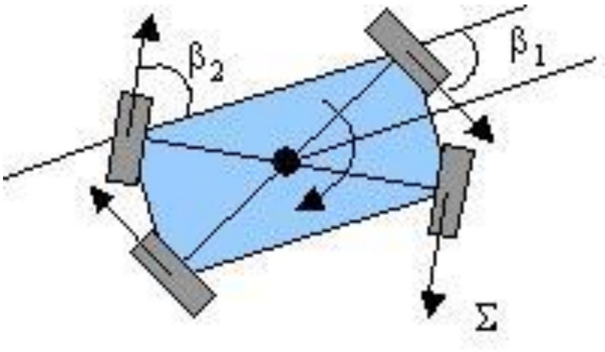


Figure 4-10 : Rotation

The wheel speeds are the same and the wheel angles β_1 and β_2 are the same in magnitude but opposite in sign (Figure 4-10).

4.3.3 Superposition

These two motions can be independently controlled. The resulting robot configuration is the superpositioning of the speed vectors of each wheel. This configuration also satisfies the ICR (Figure 4-11).

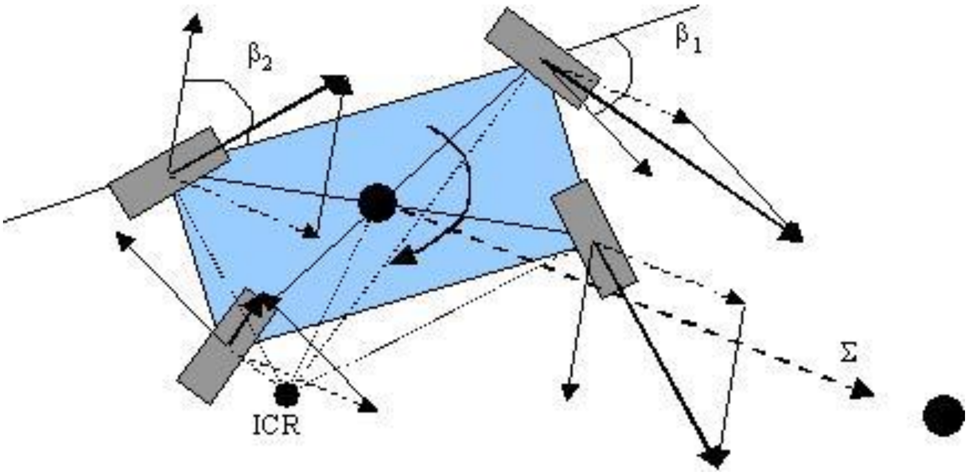


Figure 4-11 : Superpositioning of transverse and rotational motions

The solid arrows show the vectors for rotation, and the dashed arrows show the vectors for translation. For superpositioning, simply by vector calculation, the solid arrows are added to the dashed arrows. The resulting solid bold arrows are the speed vectors of each wheel. This vector superpositioning gives the ICR and speeds of each wheel together with the instantaneous direction of motion Σ.

5 Simulation and Animation

5.1 Simulation Tools

5.1.1 Main Window

Matlab 6.5 software of Mathworks Inc. is used for the simulation and animation studies.

There are several blocks in the main window (Figure 5-1) that are organized as figure

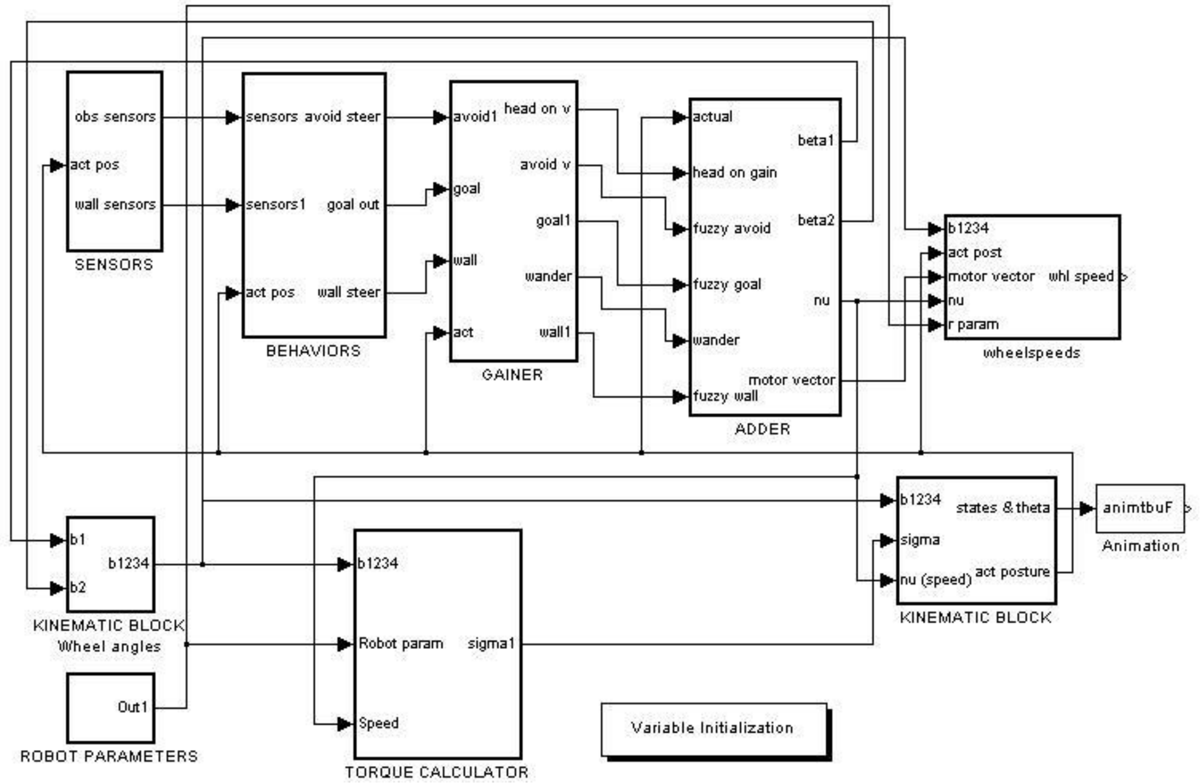


Figure 5-1: Main simulation window

5.1.2 Robot Parameters

The parameters used in the simulation to define the robot are in Table 5-1.

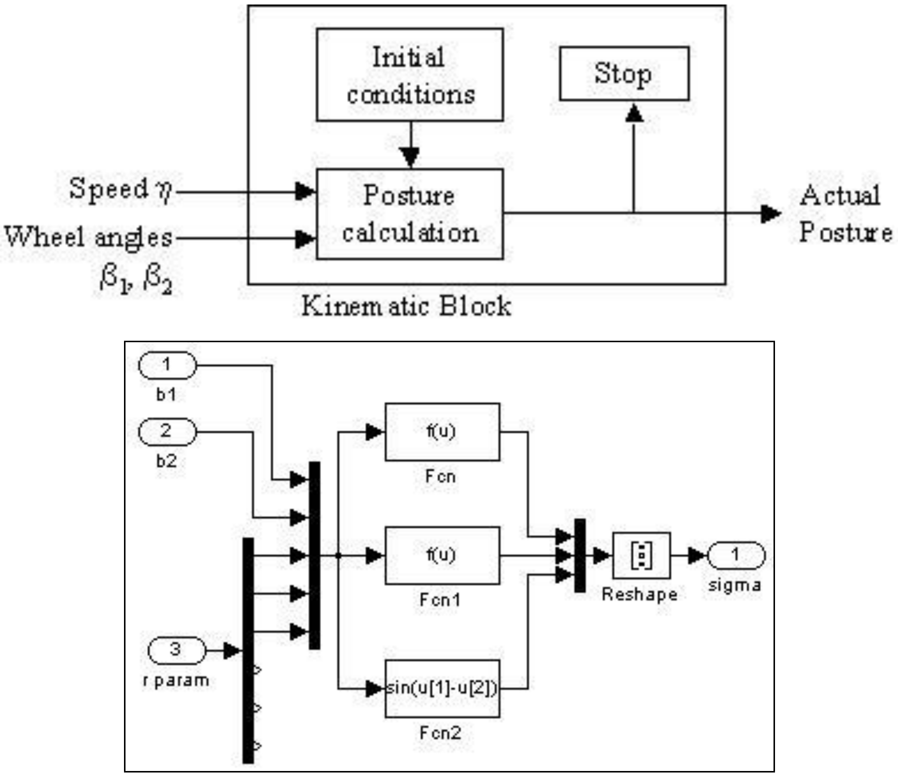
Table 5-1:Robot parameters

Parameter	γ_1(rad)	γ_2(rad)	γ_3(rad)	γ_4(rad)	l(m)	r_w(m)	M_b(kg)	M_w(kg)
Used value	0.69	2.44	-2.44	-0.69	0.21	0.2	25	2

These parameters are entered in the "Robot Parameters" block in the main window. By changing these variables, any 4x4x4 (4 wheel steered 4 wheel driven) mobile robot can be simulated. This parametric definition is an advantage of this simulation structure for further studies about this subject.

5.1.3 Kinematics Block

The Kinematics Block is calculating the actual posture of the mobile robot. The initial posture is entered into this block by the GUI through the animation m-file, and the block calculates the actual posture using speed η and wheel angles β_1, β_2 (Figure 5-2)



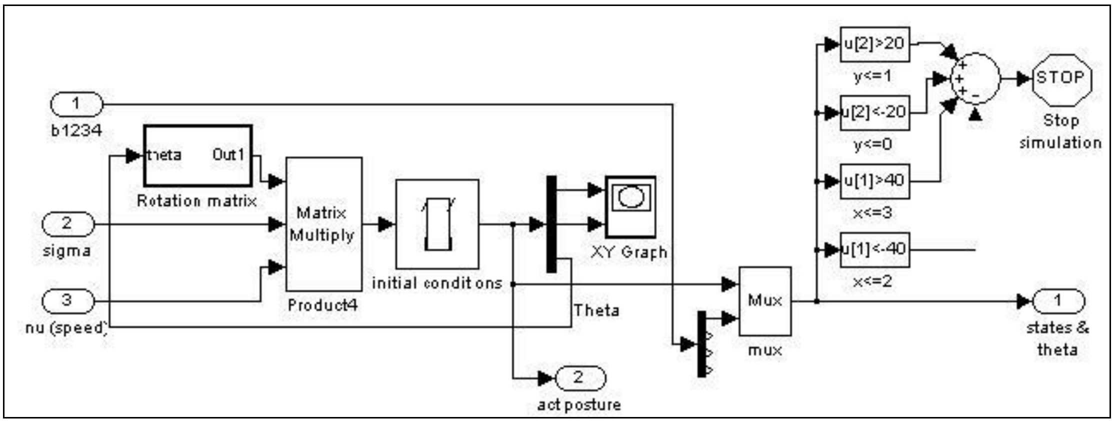


Figure 5-2: Kinematics Block

The kinematics block stops the simulation if the robot position coincides with the walls. This means that the robot hit the wall.

5.1.4 Torque Calculation

The torque for each wheel is calculated by using the posture information from the “Kinematics Block” and the “Robot Parameters” block

5.1.5 Sensors

In the “Sensors” block, there are sensor models for the Obstacle Avoidance and Wall Following behaviors (Figure 5-3). The position of walls and obstacle are also defined in this block. There is one obstacle defined as a point in the plane and 4 walls building up a rectangle area for the robot to travel inside.

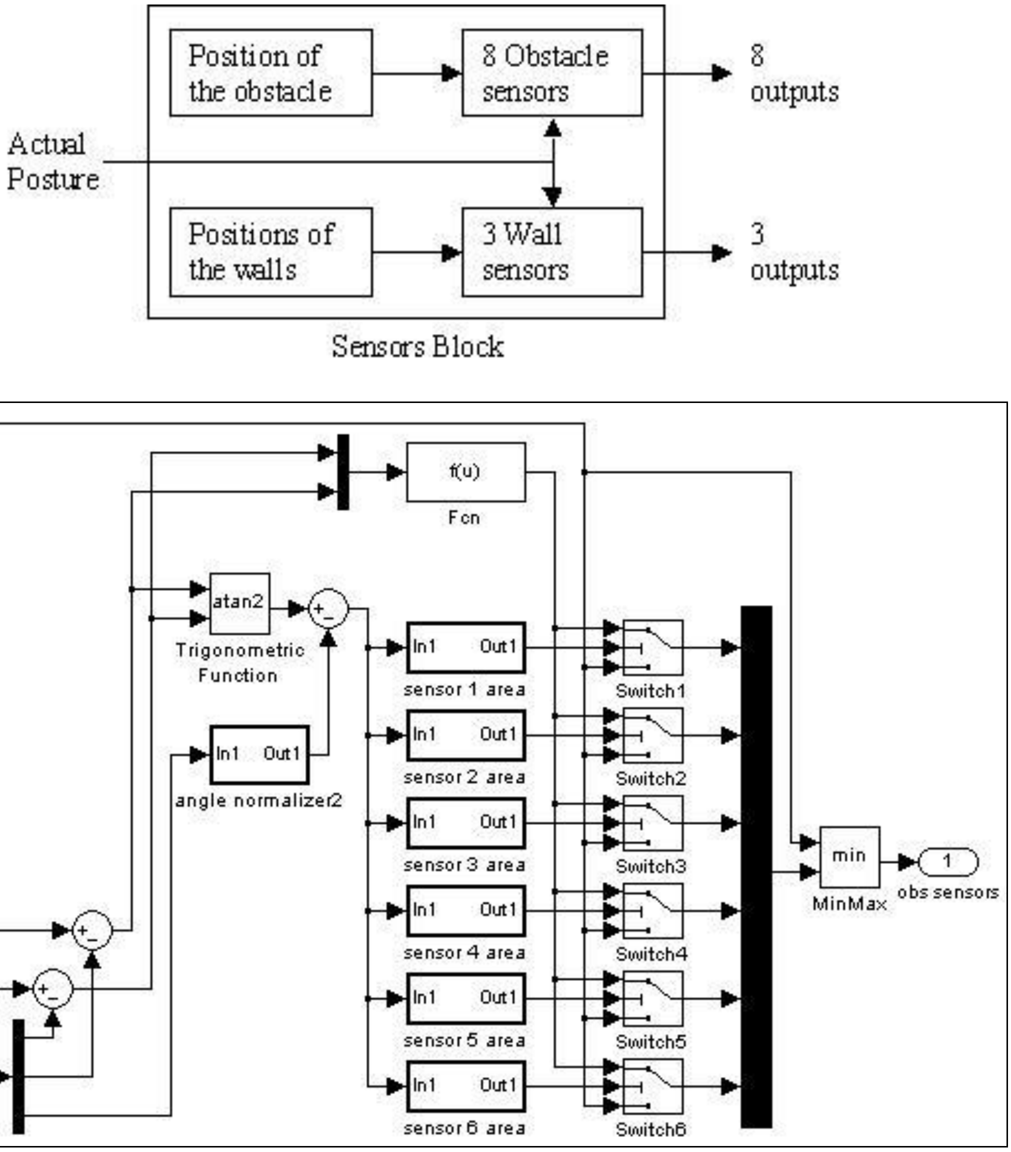


Figure 5-3: Sensors Block, obstacle sensors

5.1.6 Behaviors Block

The Behaviors Block contains the fuzzy behaviors “Go-to-goal Behavior”, “Wander Behavior”, “Obstacle Avoidance Behavior” and “Wall Following Behavior” (Figure 5-4). The sensory input from the “Sensors” block is for the last two behaviors. The first behavior “Go-to-goal” is working with the relative position of the goal given directly as calculated. The “Wander” behavior generates random steer and random positive speed values.

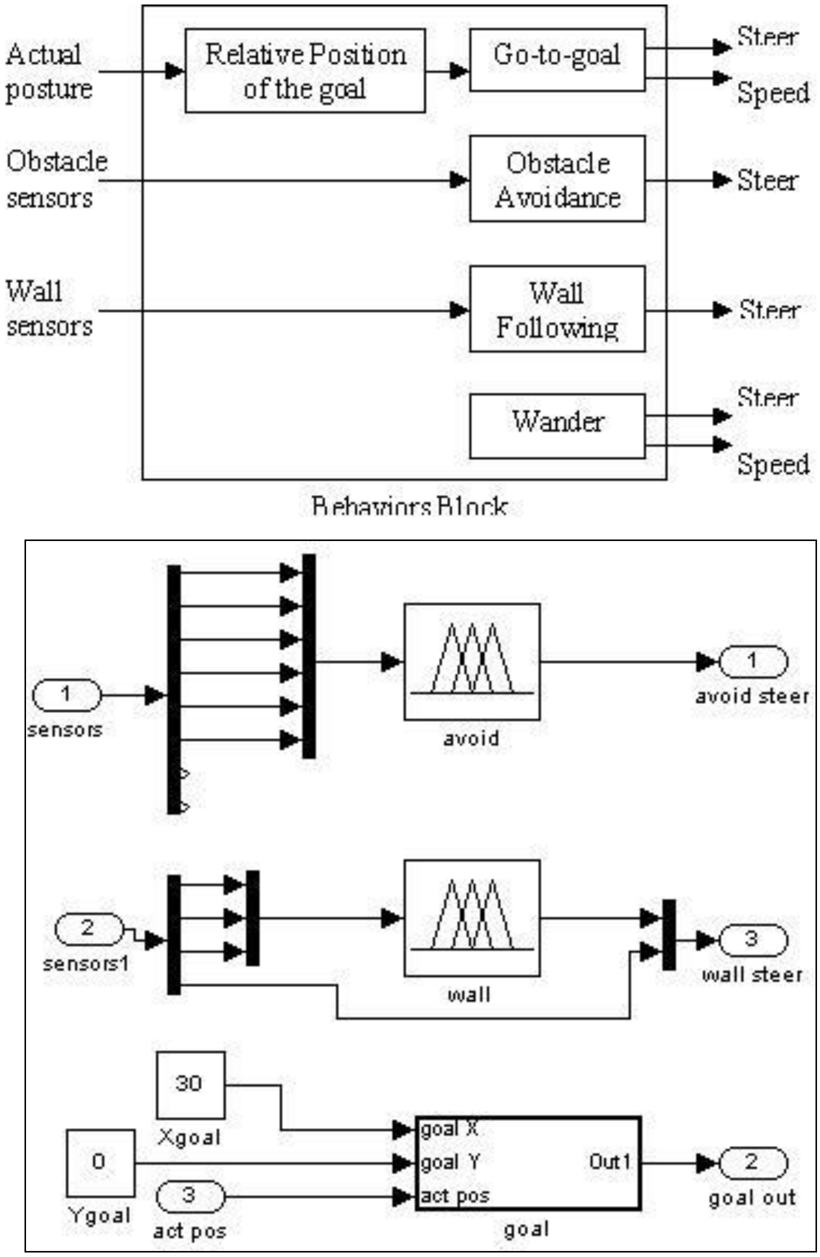


Figure 5-4: Behavior Block

5.1.7 Gainer Block

The Gainer is an upper level supervisor that is managing the gains of each of the behaviors. The behaviors are given some gains according to the current motivation or the emotion of the robot (Figure 5-5). The emotions may be influenced from internal dynamics as well as the environment.

The Gainer has two emotions, “hunger” that is increasing from 0 to 1 by time and “wander” that is decreasing from 1 to 0. If the goal is considered as food, the hunger drops down to 0 and the wander goes up to 1 when the robot reaches the food.

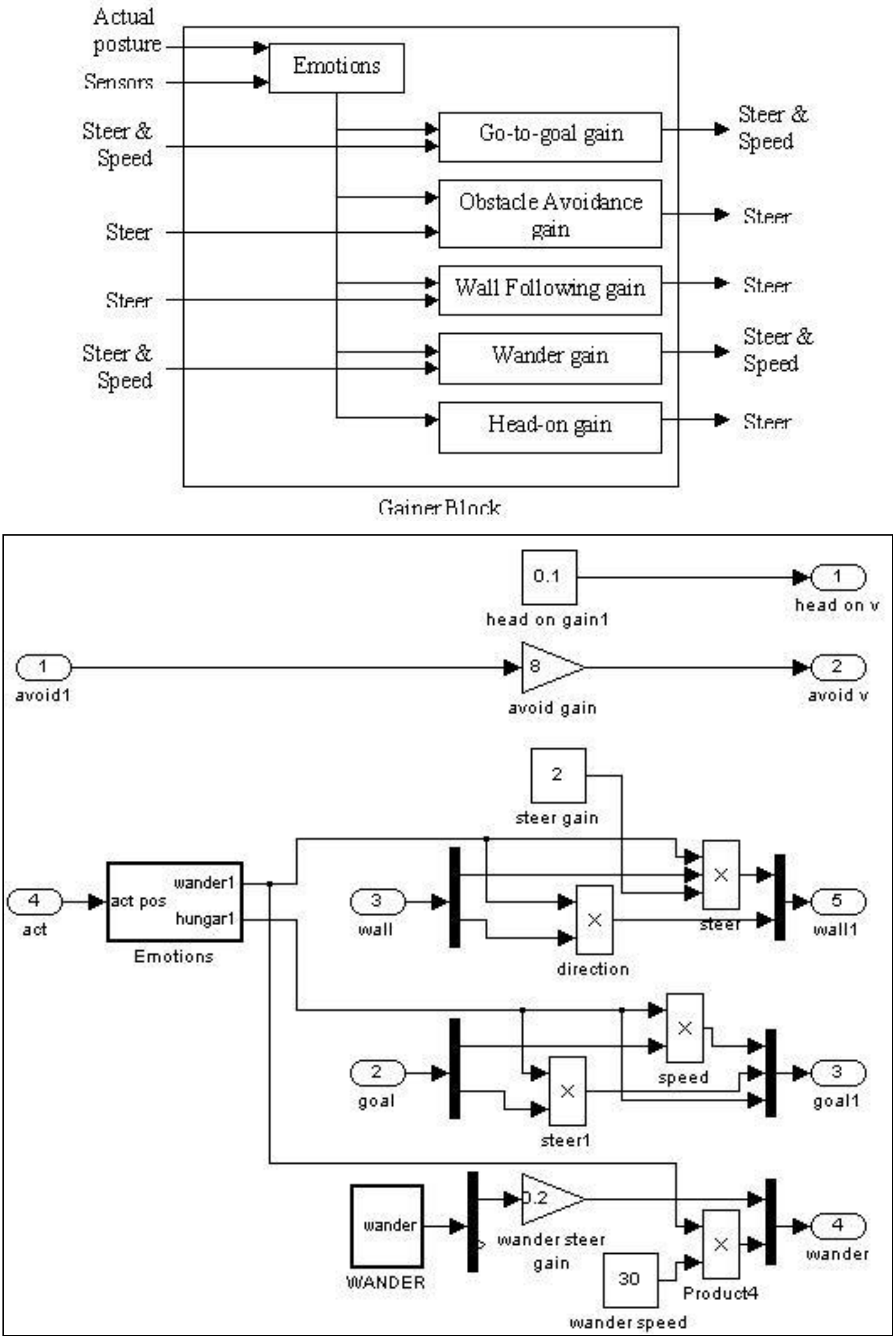


Figure 5-5: Gainer Block

5.1.8 Adder Block

The gained steer and speed values of the behaviors are added in the Adder Block (Figure 5-6). This addition is done separately on the speed, on the 1st wheel and on the 2nd wheel. The separate addition on the two wheels provides a control on the direction Σ and a control on body angle θ. Some directional subjects about the simulation environment are also considered in the addition operations.

The “Head-on Behavior” is built in the Adder Block because it uses the resulting wheel angles as inputs.

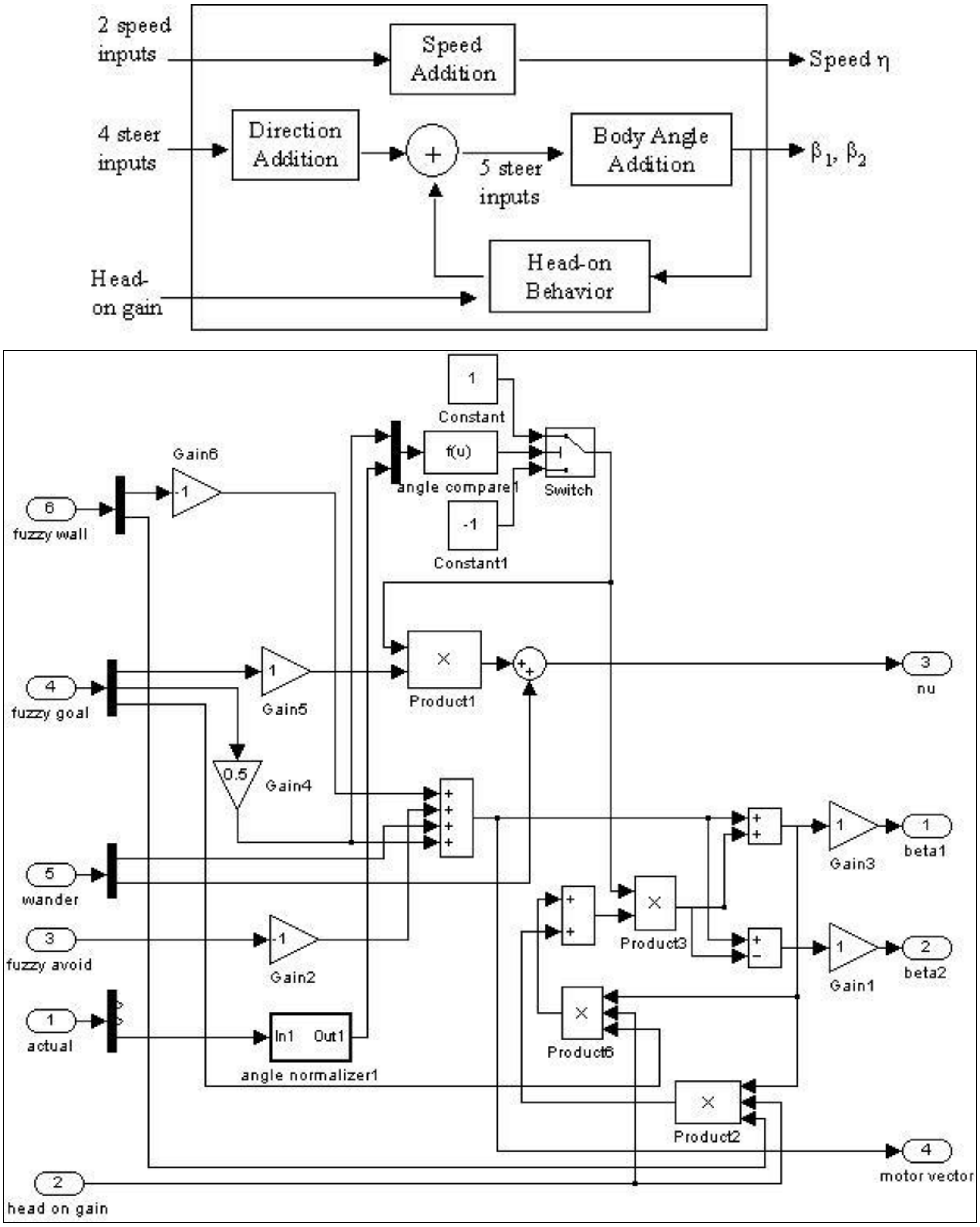


Figure 5-6: Adder Block

5.1.9 Animation Block

The Animation Block is a Simulink s-function containing a Matlab m-file. The inputs to the block is the posture of the robot, and the output of the m-file is the GUI and the initial condition of the posture for time t=0.

The animation m-file and GUI is explained in more detail at 5.3 Animation Tools.

5.2 Matlab 6.5 Fuzzy Logic Toolbox

The Fuzzy Logic concept is explained in detail in the “Fuzzy Behaviors” chapter. The “Avoid-Obstacle Behavior”, “Wall Following Behavior” and “Go to Goal Behavior” are implemented as fuzzy behaviors. The “Head-on Behavior” and the “Wander Behavior” are implemented with crisp values.

5.3 Animation tools

5.3.1 Matlab 6.5 m-file

The m-file contains the information of the GUI and converts the posture information coming from the Kinematics Block to graphical entities.

The m-file also contains the graphical information of the robot, the obstacle, the walls, and the goal. Changing the properties (i.e., position, robot parameters) in the main window of the simulation will affect the posture of the robot, but will not change graphical correspondences of these parameters in the animation, they are separate objects.

For example, if the obstacle position is changed in the simulation but not in the m-file, the GUI will show the obstacle’s position as unchanged, but the robot will detect the obstacle at the new position and the reverse is also correct.

When the “Show Trails” function is enabled, the robot leaves trails on its way for a predefined period of time. This time can be changed in the animation m-file.

The m-file for this simulation is modified from the “Truck-backer-upper” demo in the Matlab Fuzzy Logic Toolbox demos.

5.3.2 Matlab 6.5 GUI

The GUI is a Graphical User Interface in Matlab. GUI is working with the simulation through the animation m-file. There are buttons to start/stop, and during the simulation run, new buttons appear to pause or run the simulation step by step (Figure 5-7).

The robot may leave trails on its way by checking the “Show Trails” box. These trails may be deleted by “Clear Trails” button.

After the simulation stops, the robot may be moved or rotated to have a new initial posture by using the mouse.

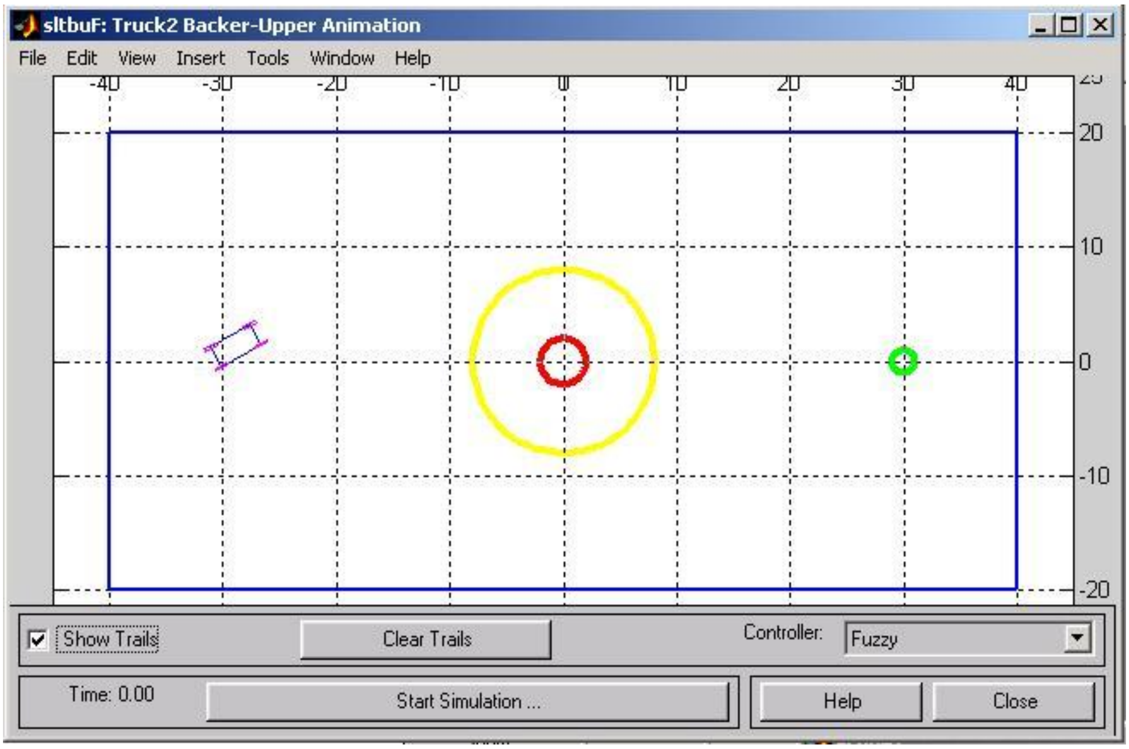


Figure 5-7: GUI

5.4 Simulation Results

5.4.1 Effect of gain on emergent behavior

5.4.1.1 Wall Following gain

The effect of gain on the Avoid Obstacle Behavior is shown on 2.1.3. The head-on behavior was also working at that simulation, but these two behaviors have no conflict between each other.

In case of a conflict, the gain may give the robot a tendency to one of the behaviors. The Wall Following and Move to Goal behavior have a conflict because both are trying to pull the robot to different directions. The gain of the Wall Following behavior is increased from 0 to 2.5 in 5 runs (Figure 5-8). The robot starts close to the wall, and continues to follow the wall as more as the gain is increased figure.

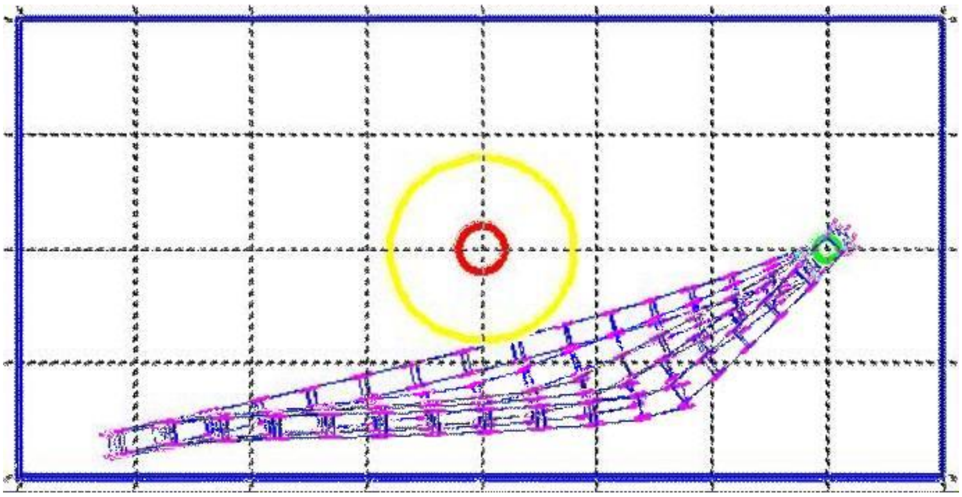


Figure 5-8: Wall gain effect

5.4.1.2 Wander Gain

The Wander Behavior gives the robot a random steer input. As the gain of this random input is increased, the robot to gets apart from Wall and travels in the empty areas (Figure 5-9,Figure 5-10).

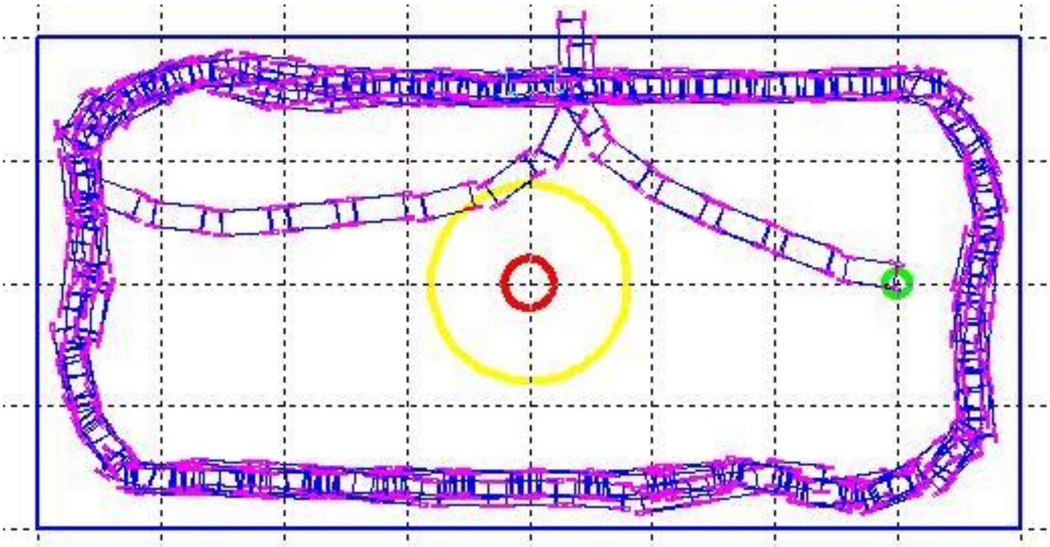


Figure 5-9: Wander gain : 0.2

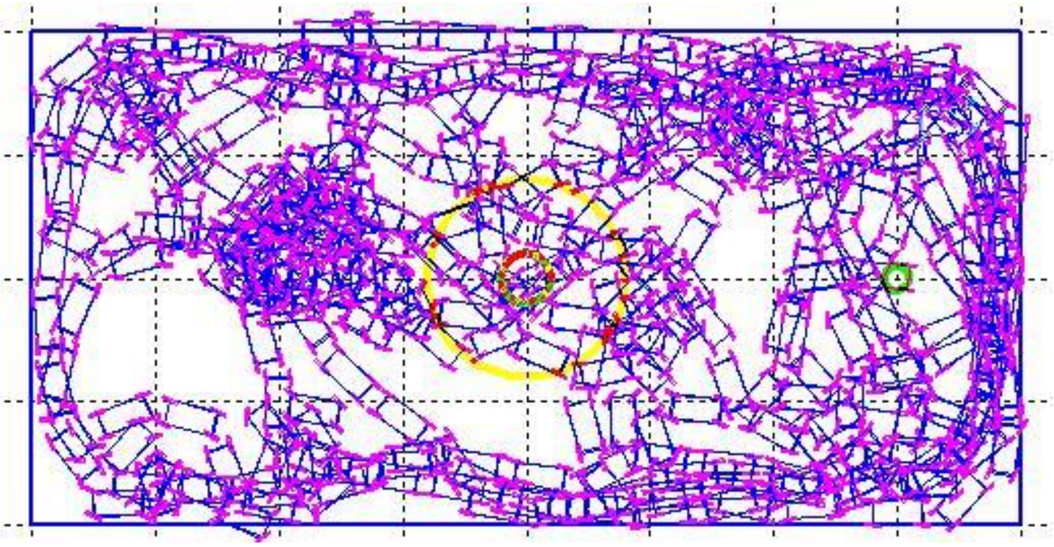


Figure 5-10: Wander gain : 0.4

5.4.2 Obstacle Avoidance

The Avoid Obstacle Behavior works well when the relative place of the obstacle is directly calculated, but there seems to be a problem in the sensory Fuzzy Avoid Obstacle Behavior. The cause of this problem is the distribution of the sensors on the robot. The sensors are looking at a -46º/+46º angle, but the obstacle may be outside this range. The robot does not see the obstacle outside this range and may crash the back or sides to the obstacle.

5.4.3 Emotion Based Architecture Simulation

The constructed emotion architecture has two emotions, Hunger and Wander. These emotions are working with a timer. At the initial condition, Hunger starts with 0 and Wander starts with 1. After 15sec. Hunger emotion steps up to 1 and wander drops down to 0. When the robot reaches the goal, Hunger again drops to 0 and Wander goes to 1 (Figure 5-11). That means, Wander is always 15sec, but Hunger may change.

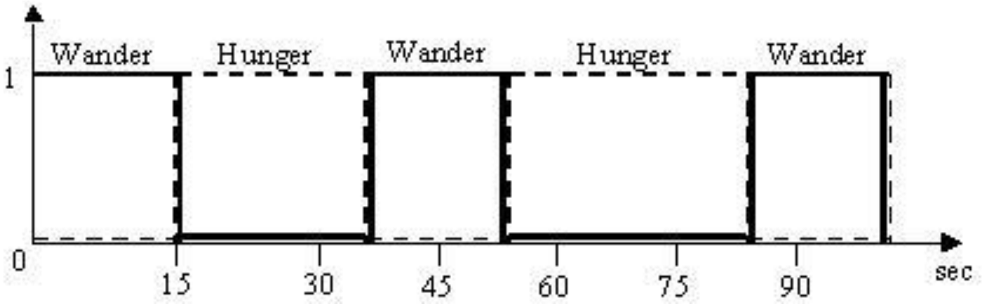


Figure 5-11: Hunger and Wander Emotions

Hunger enables the Move to Goal, Avoid Obstacle and Head-on Behaviors. Wander enables Wall Following, Wander, Avoid Obstacle and Head-on Behaviors (Figure 5-12).

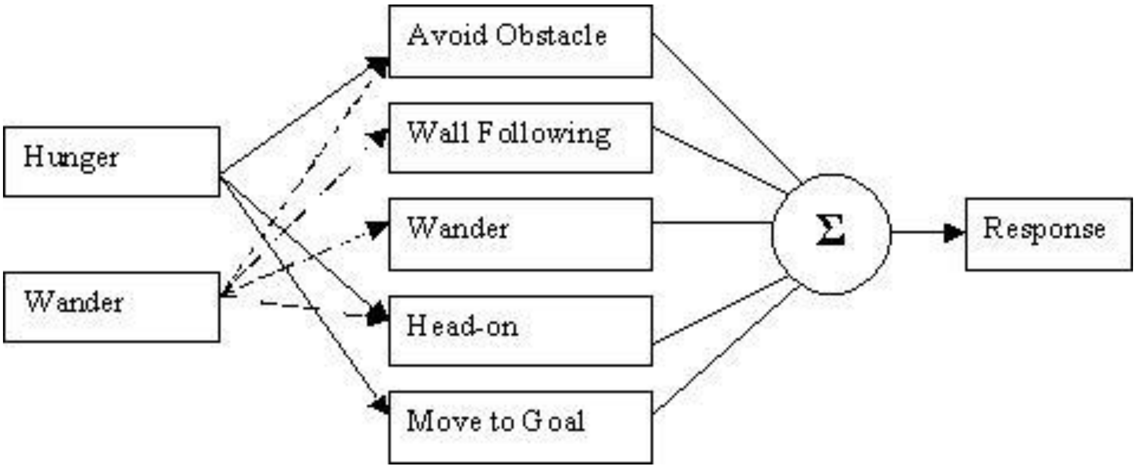


Figure 5-12: Emotion Based Structure

When the simulation has run with this architecture for 150 seconds, Hunger has activated 7 times and the robot has reached the goal every 7 times and almost successfully survived in the simulation area (Figure 5-13). It is seen from the simulation that, the robot has hit the obstacle once.

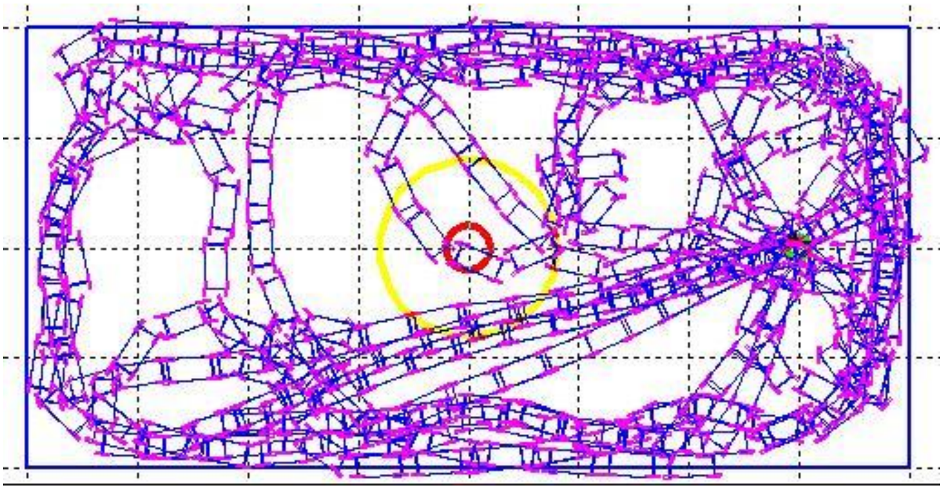


Figure 5-13: Emotion Based simulation run

5.4.4 Kinematic and Dynamic Variables

For a simulation run (Figure 5-14), the kinematics and dynamic variable are shown in Figure 5-15.

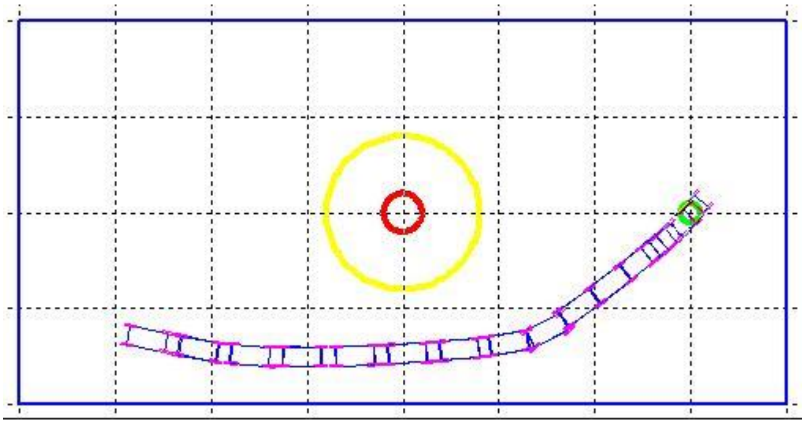
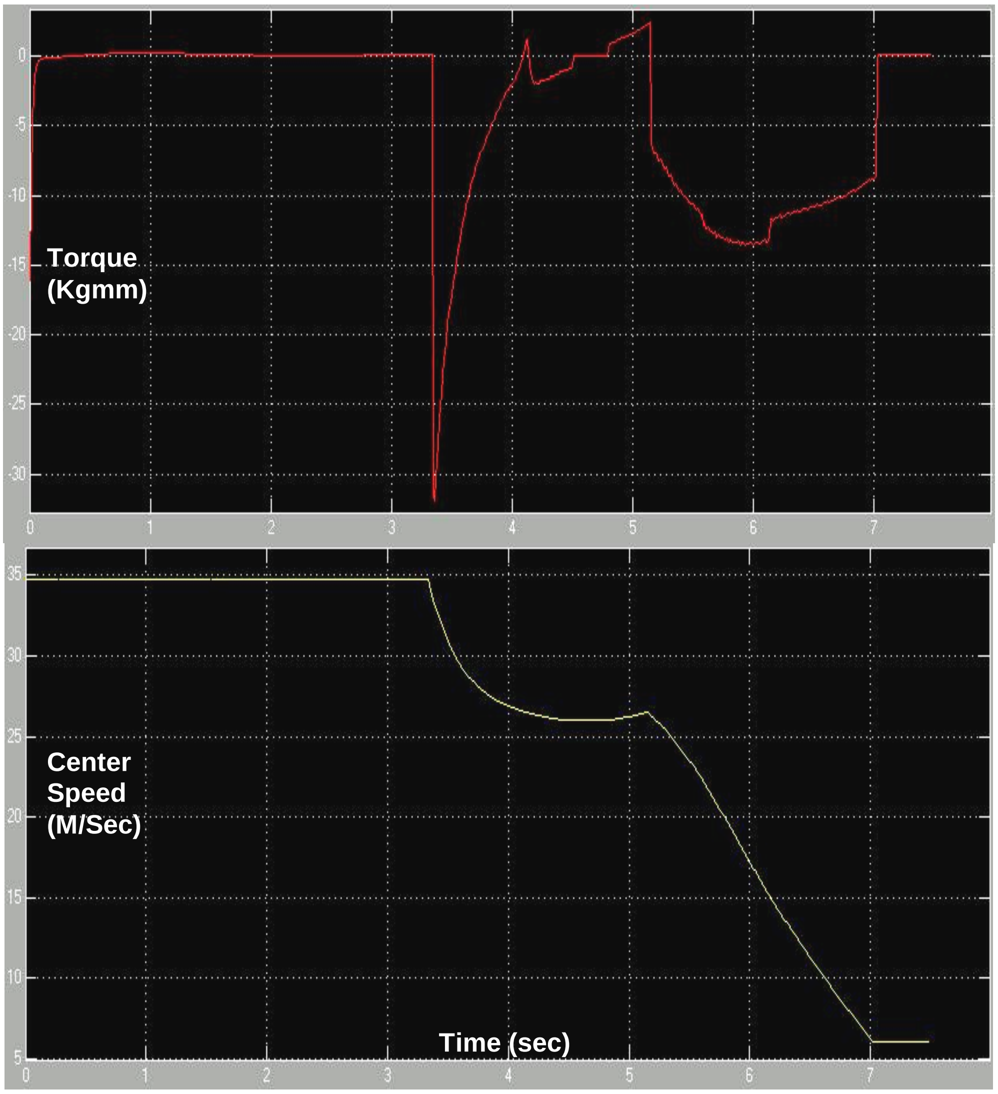


Figure 5-14: Example run



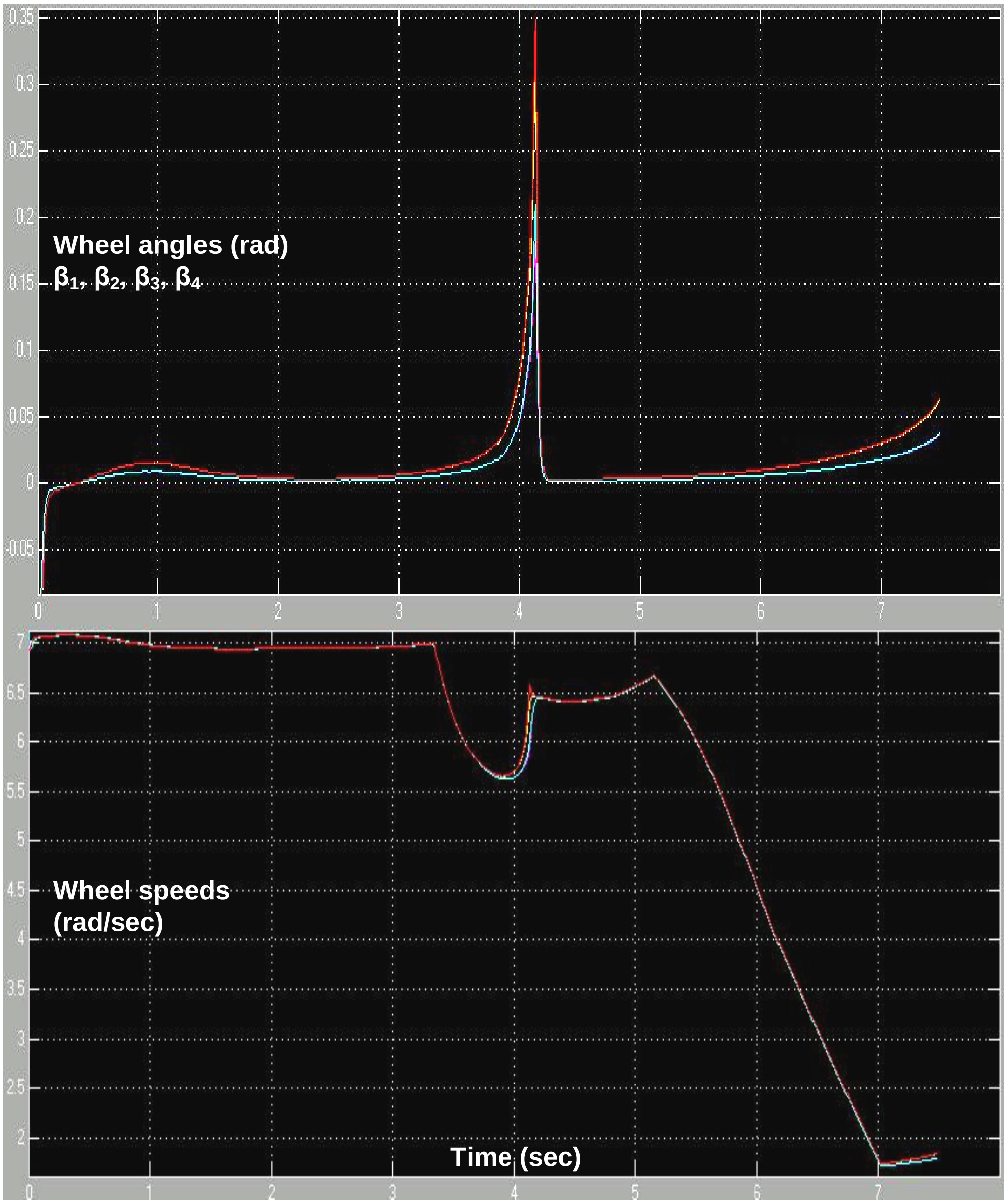


Figure 5-15: Kinematics and dynamic variables for a simulation run

6 Conclusion

The Behavior Base Robotics brought a different and useful side of view to the AI and robotics subjects. Although it has some drawbacks of the Purely Reactive Robotics, the hybrid solutions, which are using behavioral and deliberative concepts together, are promising.

The need for organizing the behaviors is obvious. The organizing level may be a deliberative level that uses representations and makes long-term plans. Emotions are useful tools for organizing the behaviors, but the deliberative level should be also above the emotions and drive the emotions rather than the behaviors.

All the information from the environment is sensory data. There are also internal sensors or dynamics of the robot. All these information should be processed and fed into the behaviors. Fuzzification of the behaviors and processing the sensory information as fuzzy sets bring ease and support the reactivity of the behavior. The configurations of the sensors are important on deciding the membership functions of the fuzzy engines.

Making the experiments in a simulation environment gives a good understanding, but the real robot and real world experiments are detrimental. The simulation environment does not contain the exact robot and world models, and there are approximations and some constraints due to the mathematical background.

The experiments should be carried out to the real world. The real world experiments will be easier and more realistic than preparing a virtual environment and a virtual robot.

New emotions may be added and the emotions may be fuzzified. The rulebase of the Fuzzy Emotions will be a kind of “principles” of the robot that gives it a “personality”.

Some more behaviors may be added to give new abilities to the robot for new purposes.

The results of the kinematics and dynamic calculations may be used to drive a real robot, and a feedback control may be realized in form of behaviors such as “Minimum Power Behavior” that is trying to minimize the power or “Comfort Ride Behavior” that is trying to keep the acceleration in a smooth change by controlling the torque and the speed

The structure of the real mobile robot is different than the one used in the simulation of this thesis. These parameters and formulations should be changed according to the real robot.

The real robot will have a robot arm on its frame. This robot arm will apply some dynamic forces to the mobile robot frame as it moves. These forces and mass distribution changes should be taken into consideration in the kinematics and dynamical calculations.

7 References

[1] **Brooks, R.A**., 1987. Intelligence without representation, MIT Artificial Intelligence Laboratory, USA.

[2] **Holland, O.**, 2002. The first biologically inspired robots, University of Essex, Colchester, UK.

[3] **Arkin, R. C.**, 1999. Behavior Based Robotics, The MIT Press, Cambridge, Massachusets, London, England.

[4] **Likhachev, M., Arkin, R. C.**, Spatio-Temporal Case-Based Reasoning for Behavioral selection, Georgia Institute of Technology, USA.

[5] **Arkin, R. C.**, 2003. Moving up the Food Chain: Motivation and Emotion in Behavior-Based Robots, Georgia Institute of Technology, USA.

[6] **Svetlicic, I.**, 2004. Emotion Based Subsumption Architecture for Mobile Robotics, Masters Thesis, Dept. of Computer Science and System Analysis, Miami University, Ohio, USA.

[7] **Michaud, F.**,2002. EMIB — Computational Architecture Based on Emotion and Motivation for Intentional Selection and Configuration of Behavior-Producing Modules, Hermes Science Publications, Université de Sherbrooke, Canada.

[8] **Zadeh, L.** 1973. Outline of a new approach to the analysis of complex systems and decision processes, IEEE Transactions on Systems, Man, and Cybernetics, Vol. 3, No. 1, January, pp. 28-44.

[9] **Safiotti, A.**, 1997. The uses of fuzzy logic in autonomous robot navigation, Soft Computing 1, Springer-Verlag.

[10] **Li, T.-H. S., Chang, S.-J., Tong, W.**, 2004, Fuzzy Target Tracking Control of Autonomous Mobile Robots by using Infrared Sensors, IEEE Transactions on Fuzzy Systems, Vol. 12, No. 4, August, pp. 491-501.

[11] **Pawlowski, S., Dutkiewicz, P., Kozlowski, K., Wroblewski, W.**, 2001. Fuzzy logic implementation in mobile robot control, Second Workshop on Robot Motion and Control, 18-20 October, pp. 65-70.

[12] **Tsourveloudis, N. C., Valavanis, K. P., Hebert, T.**, 2001. Autonomous Vehicle Navigation Utilizing Potential Fields and Fuzzy Logic, IEEE Transactions on Robotics and Automation, Vol. 17, No. 4, August, pp. 490-497.

[13] **Yang, X.S., Li, H., Meng, M. Q.-H., Liu, P. X.**, 2004. An Embedded Fuzzy Controller for a Behavioral-Based Mobile Robot With Guaranteed Performance, IEEE Transactions on Fuzzy Systemes, Vol. 12 No.4, August, pp. 436-446.

[14] **Kodagoda, K. R. S., Wijesoma, W. S., Teoh, E. K.**, 2002. Fuzzy Speed and Steering Control of an AGV, IEEE Transactions on Control Systems Technology, Vol.10, No. 1, January, pp. 112-120

[15] **Luo, R. C., Chen T. M.**, 2000. Autonomous Mobile Target Tracking System Based on Grey-Fuzzy Control Algorithm, IEEE Transactions on Industrial Electronics, Vol. 47, No. 4, August, pp. 920-931.

[16] **Sim, K.-B., Byun, K.-S., Lee, D.-W.**, 2004. Design of Fuzzy Controller Using Schema Coevolutionary Algorithm, IEEE Transactions on Fuzzy Systems, Vol. 12, No. 4, August, pp. 565-568.

[17] **Kim, S. H., Park, C., Harashime, F.**, 2001. A self-organized Fuzzy Controller for Wheeled Mobile Robot Using an Evolutionary Algorithm, IEEE Transactions on Industrial Electronics, Vol. 48, No. 2, April, pp.467-474.

[18] **Hu, Y., Yang, S. X.**, 2003. A Fuzzy Neural Dynamics Based Tracking Controller for a Nonholonomic Mobile Robot, Proceedings of the IEEE/ASME International conference on Advanced Intelligent Mechatronics (AIM 2003).

[19] **Teiner, M., Rojas, I., Goser, K., Valenzuella, O.**, 2003. A hierarchical fuzzy steering controller for mobile robots, International Symposium on Computational Intelligence for Measurement Systems and Applications, Lugano, Switzerland, 29-31 July.

[20] **Abdessamed, F., Benmahammed, K., Monacelli, E**., 2004. A fuzzy-based reactive controller for a non-holonomic mobile robot, Robotics and Autonomus Systems, **47**, 31-46.

[21] **Vadakkepad, P., Miin O.C., Peng X., Lee T. H.**, 2004. Fuzzy Behavior-Based Control of Mobile Robots, IEEE Transactions on Fuzzy Systems, Vol. 12, No. 4, Auust pp. 559-564.

[22] **Rusu, P., Petriu E. M., Whallen T. E., Cornell, A., Spoelder, H. J. W.**, 2003. Behavior-Based Neuro-Fuzzy Controller for Mobile Robot Navigation, IEEE Transactions on Instrumentation and Measurement, Vol. 52, No. 4, August, pp. 1335-1340.

[23] **Thongchai, S., Suksakulchai, S., Wilkes, D. M., Sarkar, N.**, 2000. Sonar Behavior-Based Fuzzy Control for a Mobile Robot, Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics, Nashwille, Tenessee, 8-11 October.

[24] **Campion, G., Bastin, G., D'Andrea-Novel, B.**, 1996. Structural Properties and Classification of Kinematic and Dynamic Models of Wheeled Mobile Robots, IEEE Transactions on Robotics and Automation, Vol. 12, No. 1, February, pp. 47-62.

[25] **Bak, T., Bentsten, J., Ravn, A.P.**, 2003. Hybrid Control Design for a Wheeled Mobile Robot, Aalborg University, Denmark.

[26] **Thuilot, B., D'Andrea-Novel, B., Micaelli, A.**, 1996. Modelling and Feedback Control of Mobile Robots Equipped with Several Steering Wheels, IEEE Transactions on Robotics and Automation, Vol. 12, No. 3, June, pp. 375-390.

[27] **Makatchev, M., McPee, J. J., Tso, S. K., Lang, S. Y. T.**,1999. Modelling and Control of a Four-Wheel-Steering Mobile Robot, IEEE Transactions on Automatic Control, pp. 100-107.

[28] **Bizzi, E., Mussa-Ivaldi, F., Gistzer, S.**, 1991. Computations Underlying the Execution of Movement: A Biological Perspective, Science, Vol. 253, July, pp. 287-291.

[29] **Srinivasan, M. V., Zhang, S. W., Lehrer, M., Collett, T. S**., *J. Exp. Biol*. **199**, 237.

[30] **Srinivasan, M. V.**, 1992. Distance perception in insects. Current Directions in Psychological Science, 1, 22-26.

[31] **Braitenberg, V.**, 1984. Vehicles: Experiments in Synthetic Psychology, MIT Press, Cambridge, MA.

[32] **Maes, P.**, 1990. Situated Agents Can Have Golas, Robotics and Autonomous Systems, Vol.6, pp. 49-70.

[33] **Rosenblatt, J., and Payton, D.**, 1989. A Fine Grained Alternative to Subsumption Architecture for Mobile Robot Control, Proceedings of the International Joint Conference on Neural Networks, June, pp. 317-323.

[34] **Fuzzy Logic Toolbox**, 2005. The MathWorks Inc. 3 Apple Hill Drive Natick, 01760-2098

Özgeçmiş

1980'de İstanbul'da doğdum. İlk ve orta öğretimimi İstanbul'da Çamlıca İlkokulu ve Özel Fazilet Lisesi'nde tamamladıktan sonra lise 1 ve 2. sınıfları Afyon Fen Lisesinde okudum. Lise 3. sınıfı yine İstanbul'da Özel İstanbul Fen Lisesinde okuduktan sonra 1998'de Ankara'da Ortadoğu Teknik Üniversitesi Makina Mühendisliği bölümüne girdim. 2002'de ODTÜ'den mezun olup İstanbul Teknik Üniversitesi'nde Mekatronik Mühendisliği Yüksek Lisans Programına başladım ve 2005'de bitirdim.