YILDIZ TECHNICAL UNIVERSITY
FACULTY OF ELECTRIC AND ELECTRONICS
DEPARTMENT OF COMPUTER ENGINEERING

SENIOR PROJECT

MULTI ROBOT SIMULATION PURPOSEFUL SYNCHRONOUS POSITION DETERMINATION AND MAP BUILDING

Project Supervisor: Assist. Prof. Dr. Sırma YAVUZ

Project Group

03061036 Belgin TAŞDELEN

İstanbul, 2008

CONTENTS

ABBREVIATION LIST

RF : Radio Frequency

IR : Infra Red

SFD : Start Frame Delimiter

CRC : Cyclic Redundancy Check

FIGURE LIST

TABLE LIST

PREFACE

I would like to thank Assistant Professor Sırma YAVUZ, my supervisor, for her kind support, new ideas and teaching me new concepts especially in simulation area. I enjoyed being a member of her active research group. Also, I thank to Aytunç BEKEN and Ümit KAVALA for their valuable help.

ÖZET

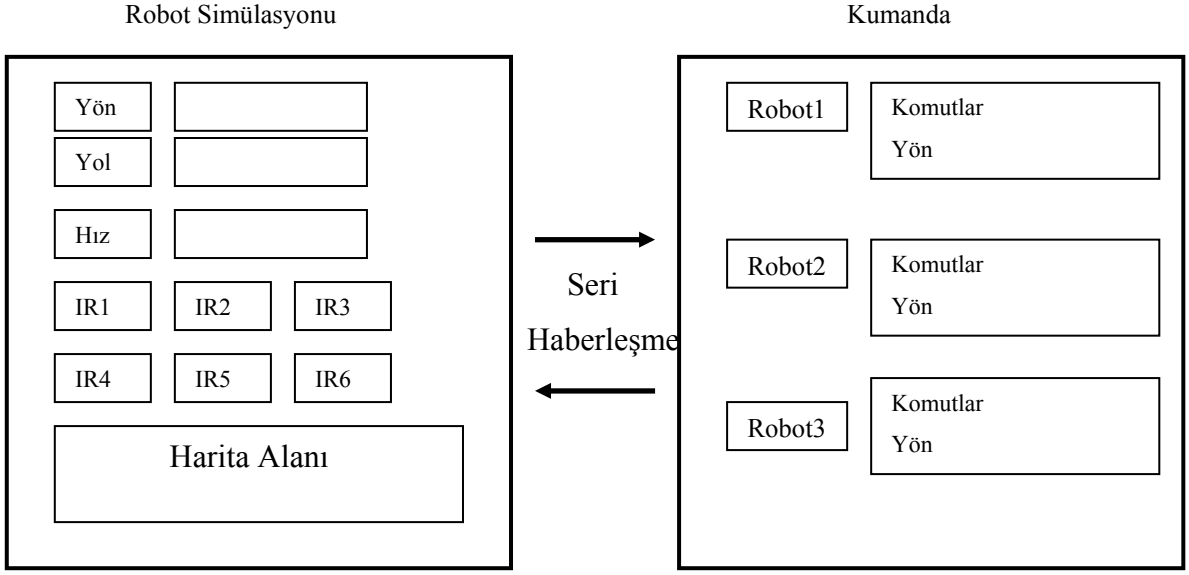
Tasarlanılacak projede temel amaç gerçek hayatta eş zamanlı konum belirleme ve haritalama yapan otonom üç robotun çalışma prensiplerinin modellenmesi olacaktır. Burada gerçek hayattaki robotlar, bilinmeyen bir yerin bilinmeyen üç farklı noktasına bırakıldığında, bu mekânı dolaşmaya başlayarak kendi konumlarını hesaplarken aynı zamanda da çeşitli algoritmalar vasıtası ile bu mekanın haritasını çıkartmaya çalışacaklardır.

Robotlar bir kumandadan gelen önceden belirlenmiş bir haberleşme protokolüne göre aldığı yön ve hız bilgileri doğrultusunda hareket etmekte, istenildiğinde de üzerlerindeki kızıl ötesi, ultrasonik sensör, encoder, optik kodlayıcı ve rotary sensör bilgilerini yine aynı protokole göre isteği yapan kumandaya iletecektir. Uygulamada virtual olarak oluşturulacak 2 com port bağlantısından aynı bilgisayar üzerinde iletişim oluşturma imkanı elde edilecektir.

Simülasyonu yapılacak robotlar bir ön ve iki arka olmak üzere, üç tekerlek üzerine yerleştirilen bir platformdan oluşmaktadır. Çalışmadaki tüm robotlar aynı yapıya sahip olacak şekilde tasarlanmıştır. Merkeziyetçi bir yapı kullanılacağından tüm robotlar sensör ölçülerini merkez bilgisayara iletirken, bilgisayar gerekli planları yapıp, robotlara kontrol işareti gönderecektir.

Tasarlanılacak yapı temel 2 alt bölümden oluşmaktadır:

1. Robot Simülasyonu
2. Kumanda



Robotun kenarlarına yerleştirilmiş 6 adet kızılötesi algılayıcı, ön ve arkada 1'er yanlarda 2'şer tanedir. Sensörler gerçek robotta 10–80 cm arası algılama kabiliyetine sahiptirler. Simülasyonda ise 0–80 cm arası ölçüm yapılabilmesi kararlaştırılmıştır. Simülasyon tarafındaki yön bilgisi rotary sensörü, kumanda tarafında yer alan yön bilgisi ise servo ile ilişkilendirilecektir. Robotlara ilişkin kinematik denklemler yardımı ile hareket fonksiyonları düzenlenilecektir.

Aynı donanıma sahip olan robotlarda planlama tamamen merkez tarafında gerçekleştirilecektir. Her robota bir sonraki hareketi gönderilecektir. Robot-Bilgisayar arası gerek komut alış verişi gerekse sensör ve konum/yön verisi iletimi Bluetooth alıcı verici modüller üzerinden sağlanacaktır.

Simülasyon uygulamasında kullanıcı robot simülasyonu tarafında robotları koordinat değeri girerek yerleştirecek, harita oluşturabilecek, istediği haritayı kaydedebilecek ve gerekli seçimler yapıldıktan sonra kumanda portu dinlemeye alınacaktır. Aynı zamanda gerçek uygulamaya yakınlık oluşturabilmek amacı ile sensör gürültüsü ekleyebilme opsiyonu da projeye dâhil edilecektir. Bu sayede gerçeğe daha yakın değerler elde etme şansı artırılmış olacaktır. Kumanda kısmında kullanıcı kontrolünü kolaylaştırmak amaçlı olarak, robotların verilen komutlar doğrultusunda ilerlemeleri sırasında, komutların değiştirilmemesi halinde doğrultularına bakılarak çarpışma risklerinin olup olmadığı hakkında bilgi verilmesi amaçlanmaktadır. Ayrıca robotların duvara çarpma durumları da incelenecek ve uyarı verilerek kontrol kolaylığı arttırılacaktır.

ABSTRACT

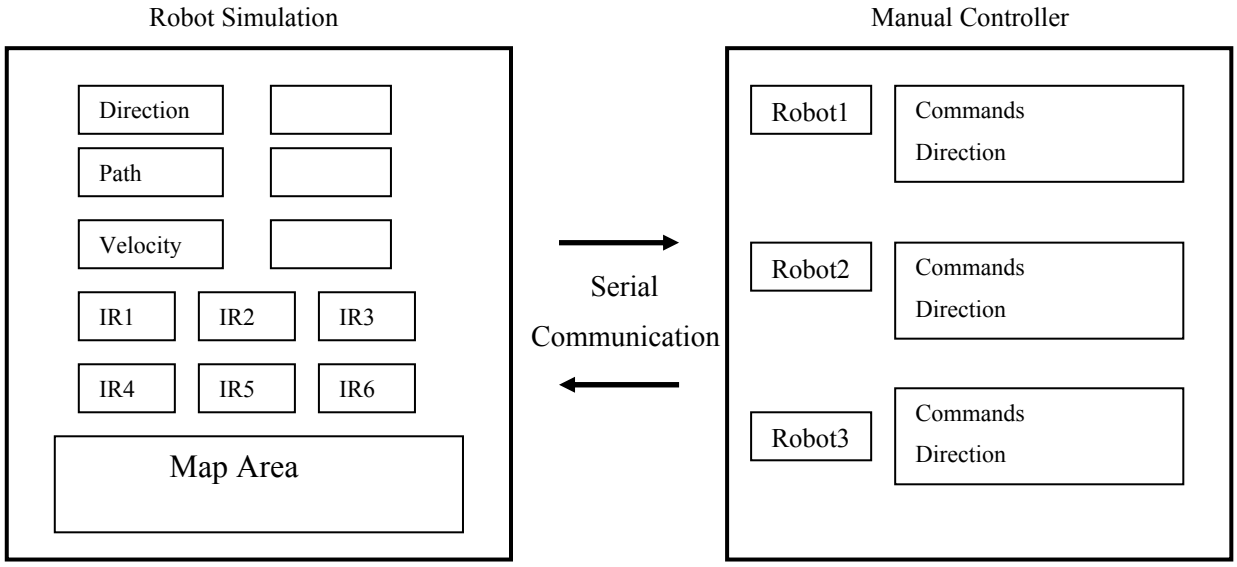
The main aim of the proposed project is, modeling the system principles of the autonomous three robots that determine position synchronously and map building as real time systems. In the scenario of the simulation, real time systems (robots) will try to map building the place that they have not known (trained about) before.

Robots will move through the information of velocity and direction that will be received from the controller by predefined communication protocol, and values of the devices on them as IR sensors, ultrasonic sensor, encoder and rotary sensors can be transmitted by the same protocol to the controller that sends the request. In the application, the opportunity of using virtual communication ports will be utilized.

The simulated robots have the same structural properties, and have a platform that is fixed on 3 wheels, two of that are at the back of it. Because of being used a centralist structure in the project, all the robots will send their sensor values to the central computer, the computer will implement the required calculations and send control signal to robots back.

The main structure of the project will have 2 subparts;

1. Robot Simulation
2. Manuel Conroller



The 6 IR sensors that are fixed on the robot are 1 on the front and back part, and 2 on the left and right sides. Sensors have the capability of measuring the distances between 10 and 80 cm. This constraint will be obtained in simulation also. In the simulation part, the data about direction will be associated with the rotary sensor, on the other hand it will be associated with the servo on the manual controlling part. The functions of motion will be arranged by the help of the kinematics equations.

Planning of actions will be implemented on the central computer part entirely. The next motion command will be sent to all of the robots.

In the simulation application, the user will place the robots by determining the coordinates, compose special maps, can save the desired maps on the interface in robot simulation part, then after some options are assigned controller will start to listen to the serial port. At the same time, to make the application be close to the real time system, option of adding noise to the sensors will be inserted to the project. In the manual control part, it is intended to giving information about the risk of the collusion between robots will be determined by considering their existing coordinates and the directions. Also the collusion of wall warnings will be inserted to the project to help the user while controlling them.

1. INTRODUCTION

1.1. Robot Systems

Robots are comprised of several systems working together as a whole. The type of job the robot does dictate what system elements it needs. The general categories of robot systems are controller, body, mobility, power, sensors and tools.

The controller is the robot's brain and controls the robot's movements. It's usually a computer of some type which is used to store information about the robot and the work environment and to store and execute programs which operate the robot. The control system contains programs, data algorithms, logic analysis and various other processing activities which enable the robot to perform.

The body of a robot is related to the job it must perform. Industrial robots often take the shape of a bodiless arm since its job requires it to remain stationary relative to its task. Space robots have many different body shapes such as a sphere, a platform with wheels or legs, or a ballon, depending on its job.

Robots movement depends on the job they have to do and the environment they operate in. In the water, conventional unmanned, submersible robots are used in science and industry throughout the oceans of the world. Land based rovers can move around on legs, tracks or wheels. In the Air (Space), robots that operate in the air use engines and thrusters to get around.

Power for industrial robots can be electric, pneumatic or hydraulic. Electric motors are efficient, require little maintenance, and aren't very noisy. Pneumatic robots use compressed air and come in a wide variety of sizes. A pneumatic robot requires another source of energy such as electricity, propane or gasoline to provide the compressed air. Hydraulic robots use oil under pressure and generally perform heavy duty jobs. This power type is noisy, large and heavier than the other power sources. A hydraulic robot also needs another source of energy to move the fluids through its components.

Pneumatic and hydraulic robots require maintenance of the tubes, fittings and hoses that connect the components and distribute the energy.
Sensors are the perceptual system of a robot and measure physical quantities like contact, distance, light, sound, strain, rotation, magnetism, smell, temperature, inclination, pressure, or altitude. Sensors provide the raw information or signals that must be processed through the robot's computer brain to provide meaningful information. Robots are equipped with sensors so they can have an understanding of their surrounding environment and make changes in their behavior based on the information they have gathered.

As working machines, robots have defined job duties and carry all the tools they need to accomplish their tasks onboard their bodies. Many robots carry their tools at the end of a manipulator. The manipulator contains a series of segments, jointed or sliding relative to one another for the purpose of moving objects. The manipulator includes the arm, wrist and end-effector. An end-effector is a tool or gripping mechanism attached to the end of a robot arm to accomplish some task. It often encompasses a motor or a driven mechanical device. An end-effector can be a sensor, a gripping device, a paint gun, a drill, an arc welding device, etc.

2. SYSTEM ANALYSIS AND FEASIBILITY

2.1. Requirements Analysis

Requirement analysis shows the system's requirements for this project. First of all, we need a robot which has programmed on it's microcontroller and has all electronic staff on it. This project will be performed to help collecting data and testing the robot routing algorithms. The basic thing is a computer that we will be able to write and run java codes in a good performance. To test the algorithms and control the robots movements manually this project is being planned to be used.

2.2. Feasibility Study

2.2.1. Technical Feasibility

Technical feasibility shows the system's hardware and software requirements for this project.

2.2.1.1. Hardware Requirements

We need a robot which is programmed with some functions. Driving the robots required direction by preferred velocity value, we need a central computer to command it.

2.2.1.2. Software Requirements

Microsoft XP Home Edition is used as operating system and Netbeans IDE in order to write programs with java.

2.2.2. Legal Feasibility

In this project all the software and hardware are licensed to Yıldız Technical University Computer Engineering Department. One can use any part of this project without permission.

2.2.3. Economical Feasibility

Economical feasibility shows the financial reports for this project. Software and hardware requirements' all expenses are shown with the cost table.

Table 2.1. Cost Analyses Table

Robot	3000 YTL
Windows XP Professional	260 YTL
Net Beans IDE 6.0.1	0 YTL (Free)
J2SE 1.5 Java Development Kit	0 YTL (Free)
Work Force	2000 YTL
TOTAL	5260 YTL

2.2.4. Usability Feasibility

This project will be developed using latest technology available that meet all requirements. Everybody will be able to use easily this project when they meet with it first time by the help of its user-friendly interface design.

3. SYSTEM ARCHITECTURE

In this project, there will be two main parts; a simulation part in which a shortest path algorithm for autonomous robots (three robots) will be tested, and a manual control part to collect data for future studies. Robots have 6 infrared sensors, a potentiometer, and an encoder. In this section, robots' architecture will be explained and some tools will be shown.

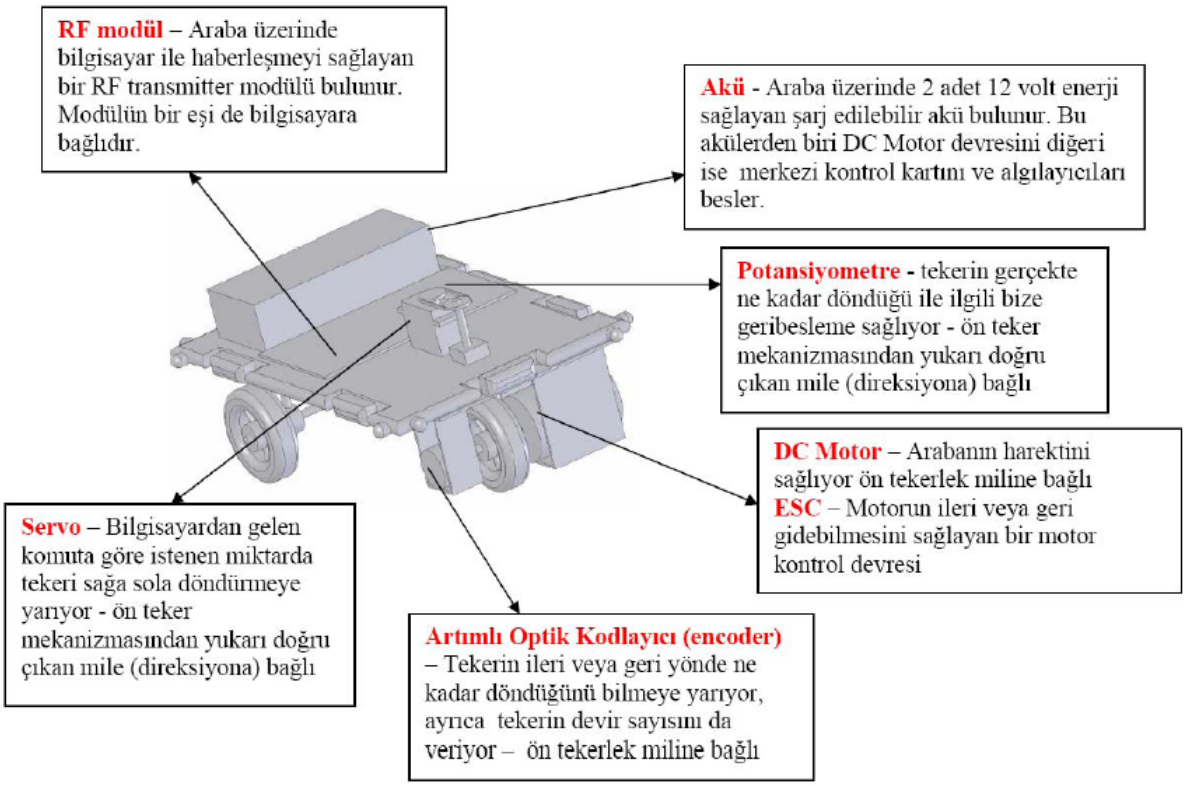


Figure 3.1 Robot Architecture

3.1. Microcontroller

A microcontroller could be likened to the "brains" of the robot. It can be programmed by the designer to accomplish the task at hand. It is responsible for sending commands to other individual systems in the robot, receiving data from external devices, and coordinating activities.

3.2. RF Communication

The RF part of the project has been greatly simplified because both of side computer and pc use the same RF module. It is a 433.92 MHz RF receiver WIZ-SML-IA from Aurel. This is transceivers for point-to-point data transfer in half-duplex mode and its card integrates a 100kbps XTR transceiver. This device will be used to receive the data in serially, and should integrate perfectly into our current system through the USART on the MMC. The MMC listens the RF Module for incoming data by using USART interrupts.

Table 3.1 IR Sensor Byte Values for Specific Distances

CM	BYTE
80	216
75	220
70	224
65	231
60	236
55	244
50	256
45	269
40	280
35	300
30	324
25	360
20	411
15	488
10	634

The model that is constituted by the help of the samples shown in Figure 3.1

x: byte value

y: cm value

$$y(cm) = ae^{bx} + ce^{dx}$$

a=3733

b=-0,02049

c=62,76

d=-0,002876

x: cm value

y: byte value

$$y(byte) = ae^{bx} + ce^{dx}$$

a= 894,5

b= -0,1052

c= 338,2

d= -0,005858

Figure 3.1 Equation of IR Value Conversion to cm and byte

3.3. Servo

A servo motor is comprised of a DC motor, a gear train, a potentiometer connected to the output shaft and an integrated circuit for positional control. All this hardware is packaged in a black casing. A servo motor is used because it is a low power device that has precise control in the angular position of the front wheel. Its specification is shown below.

Table 3.2. Servo Motor Specifications

Speed	*0.23 sec/60 degrees at 4.8 v*
Torque	*44.4 oz/in (3.2 kg/cm) at 4.8 v*
Size	*1.59"L x o.78"W x 1.42"H w/o output shaft*
Weight	*1.31 oz (37.2g)*
Connector	*"J" type with approx. 5"*

3.4. Infrared Sensors

These units report the distance to a given target as an analog voltage. Using an analog to digital converter on the microcontroller, one can determine the range to a given target. Infrared sensors function by emitting a special wavelength of light invisible to the human eye unaided, infrared, and then calculating the angle of return of the light. This method uses simple trigonometry to determine the distance the sensor is from an object. However, if the light is blocked or reflected away from the sensor, it will not be able to detect the object. The method of measuring distance is shown below in Figure 3.2. Infrared Distance Measurement.

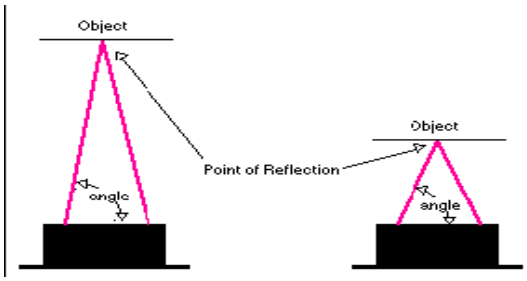


Figure 3.2. IR Distances Masurement

It has a range of 10 cm to 80 cm, which should have sufficient range for our purposes, but it was mainly chosen for the cost. The minimum range of the sensor is incurred because the angle of return becomes too wide for the width of the sensor, causing the emitted infrared beam to be unable to be sensed.

3.5. Dc Motor

DC motors are more common and easily available. Dc motor was used to move the robot. They are generally more powerful, easier to interface to, and allow the robot to move faster. However, despite these advantages, they also have a serious disadvantage. They have no method for exact control of the distance to travel. They simply turn on and drive and slowly turn off. It would be possible to implement our own control device to determine the number of rotations of the wheel by having an encoder get clock pulse each time the wheel has rotated. However, this measurement is not very precise and therefore the DC motor must be examined critically in order to interface it properly with the rest of the mobile platform and achieve the desired goals.

3.6. Encoder

Incremental encoders are sensors capable of generating signals in response to rotary movement. In conjunction with mechanical conversion devices, such as measuring wheels or spindles, incremental shaft encoders can also be used to measure linear movement. The shaft encoder generates a signal for each incremental change in position. Its resolution is 1024. It has 6 leads. Yellow leads generate one clock pulse in a cycle. Green and white leads generate 1024 pulses in a cycle. But between green and white there is 90 degrees phase difference. This difference make possible to realize the direction of the encoder.

3.7. Potentiometer

Potentiometer is an instrument for measuring electrical potential, on the robot, a potentiometer was attached on the front wheel in order to obtain direction information. Because servo motors are reliable but some times they fails because of obstacles.

4. MOTION MECHANISM OF THE ROBOT

DC motor, ESC and Servo are some of the components that help to move the robot. The values that would be sent to the robots to determine the velocity should be between 0-255 decimal values (8 bits).

Some decimal values that can be sent to the robot by computer for special states of robot's motion is shown below.

Table 4.1. Decimal Values for Robot's Velocity

Decimal Value	Effect on motion
127	Stops the motor
0-127	Makes robot go backwards
127-255	Makes robot go forwards
135-145	The reference range for reliable results

To calculate the speed of the robot as form of m/s or cm/s, some samples are taken for determining the ways that the robot taken in predefined time period. These measurements are shown in below.

Table 4.2. Relation between Motor Byte Values and Taken Path Quantity

Duration (s)	Encoder	#Tour (Encoder/512)	Wheel Mesure(cm)	Taken Path(cm/s)	Velocity (cm/s)	Motor byte
45	6296	12.29688	19	245.9375	5.465278	135 (frward)
28	7621	14.88477	19	297.6953	10.63198	145 (frward)
43	9254	18.07422	19	361.4844	8.406613	140 frward)
27	8945	17.4707	19	349.4141	12.94126	120 (frward)

Servo achieves to rotate the front wheel in required angle value. In theory servo motor has the capacity of rotating 180 degree. But because of some technological constraints,

the valid angle range is not the same practically. That's why the decimal values that are sent to the servo are limited in range of 61-131 for reliable results.

Table 4.3. Relation between Servo Decimal Values and Effect on Direction

Decimal Value (Servo)	Direction
97	Straight
97-131	Turning left
61-97	Turning right

In this project and the previous ones, it has to be thought that robot motion like vectorel not like a point to calculate correct coordinates. In my project it is assumed there are two motion type calculation: linear motion and circular motion.

In linear motion, robot follows a direct way to the chosen point. Also robot's slope doesn't change. We get the distance (d) value from robot's encoder and the calculations which are shown below we find new coordinates.

Usually it is needed to change direction of robot, so it has to be found new slope with circular motion.

5. KINEMATICS EQUATION AND DETERMINING THE ROBOT'S POSITION

In the current system, by the help of the encoder the taken path length (d) and by the help of the potentiometer the angle between the front wheel and the robot's body (Q1) can be calculated. By the help of this variables and a simple calculation the angle of the back wheels can be extracted.

If the command of "rotation by angle of α" is ordered to the robot in a specific t time while it is moving on direction of | OA| its new location can be defined by the help of direct kinematics equation as shown in figure 5.1.

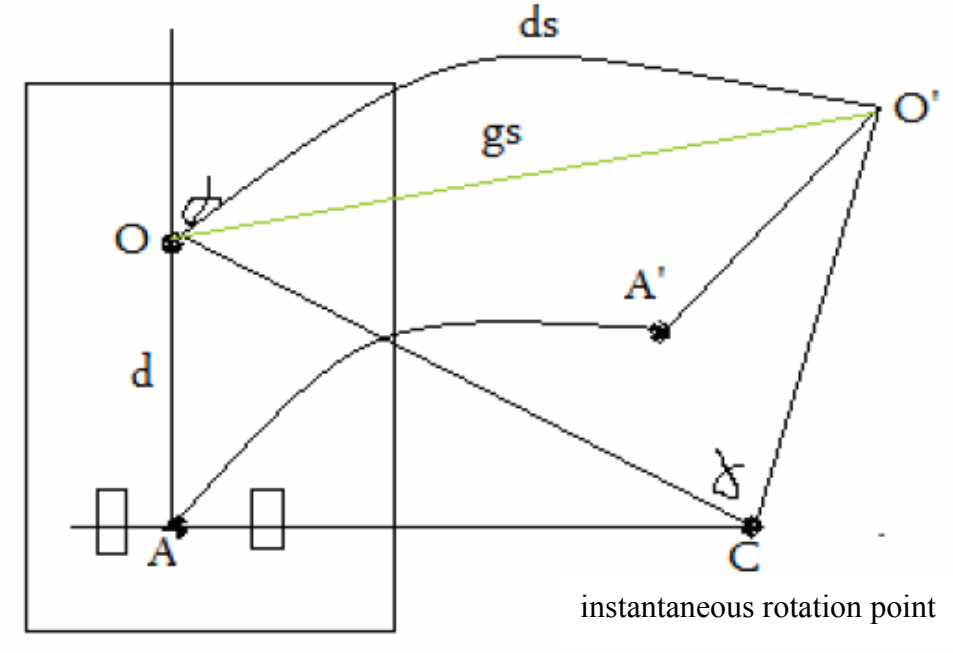


Figure 5.1. Kinematic Equations Graphics

$$|OC| = r_{ön} = \frac{|OA|}{\sin\alpha} = \frac{d}{\sin\alpha} = \frac{15}{\sin\alpha}$$

The difference between the value of encoder in time t and t+1 gives us the opportunity of calculating the path has been covered.

$$ds = \frac{Encoder(t+1) - Encoder(t)}{512} x Teker\ çevresi(cm) = \frac{Encoderdaki\ değişim}{512} x19cm$$

By the help of this equation, the angle that faces the ds curve can be estimated as the formulation.

$$\frac{ds}{2\pi r_{ön}} = \frac{\gamma}{360}$$

According to the start gradient of robot the angles of θ_1, θ_2 that are used to determine the next position of the robot can be assigned as following equations.

$$\theta_1 = 90 - \alpha - \frac{\gamma}{2} = mAraba - \alpha - \frac{\gamma}{2}$$

$$\theta_2 = 90 - \frac{\gamma}{2} = mAraba - \frac{\gamma}{2}$$

When $O(x_1(t),y_1(t))$ and $A(x_2(t),y_2(t))$ are asummepted as the first coordinates of the robot, the equation for calculating the next coordination of the front wheel in t+1 time when the coordination of t specific time is known, is showed in equation system (5), and the equation for back wheels is indicated in equation system (6) below.

$$\begin{bmatrix} x_{1(t+1)} \\ y_{1(t+1)} \\ d \end{bmatrix} = \begin{bmatrix} 1 & 0 & -\cos\theta_1 \\ 0 & 1 & -\sin\theta_1 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{1(t)} \\ y_{1(t)} \\ d \end{bmatrix} \qquad (5)$$

$$\begin{bmatrix} x_{2(t+1)} \\ y_{2(t+1)} \\ d \end{bmatrix} = \begin{bmatrix} 1 & 0 & -\cos\theta_2 \\ 0 & 1 & -\sin\theta_2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_{2(t)} \\ y_{2(t)} \\ d \end{bmatrix} \qquad (6)$$

For calculating the front wheel's coordination of the robots the following equations are used:

$X_1 = X_1 + gs * \cos(\upsilon^1)$

$Y_1 = Y_1 + gs * \sin(\upsilon^1)$

For calculating the back wheels' coordination of the robots the following equations are used:

$X_2 = X_2 + gsArka * \cos(\upsilon^2)$

$Y_2 = Y_2 + gsArka * \sin(\upsilon^2)$

6. COMMUNICATION PROTOCOL AND PACKET STRUCTURE

The communication between computer and robot(s) is performed by RF (Radio Frequency) (or Bluetooth in the future). There are Full Duplex RF modules both computer and robot.

Serial communication can be implemented by several of data transmission baud rates (115200, 57600,…,9600) ; 10 bits strings with one start bit and 8 data bits. Processing and evaluating of start and stop bits are implemented by serial communication devices.

The data and command transmission between robot(s) and computer, can performed by sending at least 7 hello packets and in computer side, maximum 19 (all IR, potentiometer and encoder data), in robot side maximum 23 byte. (This data is valid for 8 bits-ADC, if 10 bits-ADC is used then one byte should be added to entire values).

Command data structure is shown as follows;

SYNCH	SFD	LENGTH	ADDRESS	CONTROL	PAYLOAD	CRC

Figure 6.1. Packet Structure

6.1. SYNCH

It indicates the beginning of the packet. The synchronization byte comes first. This is the flag of the data, if it comes, next byte (SFD) is expected. The pattern of synch byte is 01010101 (0x55h) because this is the most difficult byte that can be send by RF due to the nature of RF communication.

6.2. SFD (Start Frame Delimiter)

The second byte is start frame delimiter; this byte is determiner of boundaries. The pattern of SFD byte is 01111110 (0x7Eh). If this byte is correct, receiver expects the length byte; else it goes back and waits for synch byte again.

6.3. LENGTH

The third byte is length of the bytes which follow this byte. In other words, this declares the sum of the length of address byte, control byte, payload bytes and CRC

bytes. It can be up to a maximum of 255. Specifying the packet length is necessary because each packet can contain a different amount of data. For example, a packet containing all devices information will have the maximum amount of payload. However, a simple HELLO packet has just 7 bytes. Therefore, specifying the packet length is essential to coordinating messages properly.

6.4. ADDRESS

The fourth byte shows the sender and receiver addresses. First fourth bits of this byte show sender address and the next four bits show receiver address. For instance, if sender address is 0xFh and the receiver address is 0x0h then address byte must be 0xF0h.

6.5. CONTROL

The fifth byte is control byte. This byte contains control data. It is possible to add a new control byte because of new situations. But these conrol bytes are enough to handle robot control for now. These bytes make difference sense for robot and computer. For example collision packet can only be sent by robot to computer.

Table 6.1. Control Bytes

Control Byte	**Meaning**
ACK (0x00)	Last packet is send successfully
NACK (0x01)	Time out
COLLISION (0x02)	Collusion is perceived
ERROR (0x03)	Error on byte
HELLO (0x04)	Handshaking

Connection handshaking is ensured by the following processes;

“Hello” packet is sent by computer after the port is opened then, the robot gets the hello package and the connection is ensured.

ACK means the last packet has been received by computer. When computer sends a packet and receives its response on time , next packet's control byte will be ACK. NACK means that robot's response to the last packet has not been received on time. COLLISION means robot collided with something, ERROR means data is received by one of the side but there is error on data. HELLO is a connection establishment byte. At the beginning of connection HELLO packet sends to robot, if robot takes this packet, it resends the same packet to the sender.

6.6. PAYLOAD

The values of the devices that are on the robot can be two state as; read-only, and read-write. Therefore, all the devices have a unique address. Address of the read-only state devices most significant bit is 1, the others is 0.

The reason of giving a unique address to all devices is to ensure standardization of the data read and write structure from the devices that are specialized for a task. By the help of this, new fixed devices can be adapted the system easily by not implying changes in the packet structure.

To read a device from computer, it's the only requirement to send the address of the device whose value is necessary to read. For writing process; the data byte that is required to be written, should be sent with the address of the device.

Table 6.2. Device Adresses

Device Address	Device Name
0x00	IR sensor on front
0x01	IR sensor on front left side
0x02	IR sensor on back left side
0x03	IR sensor on back
0x04	IR sensor on back right side
0x05	IR sensor on front right side
0x06	Potentiometer
0x07	Encoder
0x08	Switches (Every bit of this byte shows one of the switch respectively)

0x80	DC motor
0x81	Servo

Payload part of packet is the main data that want to be transmitted. In this project a flexible packet structure was used. On the robot a unique address was assigned for every device, and devices are separated into main two groups. In the first group there are readable devices. A sensor is an example of these devices. Computer only sends readable devices' addresses and robot reads the value of these devices and sends to computer. In the second group there are writable devices. Servo is an example of these devices. If computer wants to set the value of servo, it must send the address of these devices and in the following byte value of these devices must be sent. Devices and their addresses are shown in the table 7. Readable and writeable devices can be distinguished from each other by checking the most significant bits of address byte. If it is a readable device, the most significant bit of address byte is 0, if it is a writable device the most significant bit of address byte is 1.
These packet structures also give an opportunity to create different size of payload. We can only send one of the device information or all of the devices information. Devices' sequence in the payload is not important.

In some environment EMI can decrease the performance of RF communication. If packet size increases, at the same time the number of corrupted packet increases. In this circumstance we can reduce the packet size by demanding devices' value one by one, this can increase the performance.

6.7. CRC (Cyclic Redundancy Check)

This is used as a control data of 16 bits. CRC bytes are two bytes shows 16 bit CRC checksum. A CRC (cyclic redundancy check) is a type of hash function used to produce a checksum – a small, fixed number of bits – against a block of data, such as a packet of network traffic or a block of a computer file. A CRC "checksum" is the remainder of a binary division with no bit carry (XOR used instead of subtraction), of the message bit stream, by a predefined (short) bit stream of length n, which represents

the coefficients of a polynomial. The checksum is used to detect errors after transmission or storage. CRC is computed and appended before transmission or storage, and verified afterwards by the receiver to confirm that no changes occurred on transit. CRCs are popular because they are simple to implement in binary hardware, are easy to analyze mathematically, and are particularly good at detecting common errors caused by noise in transmission channels. For substantiality, we used the 16-bit CRC-16-CCITT polynomial $x^{16} + x^{12} + x^5 + 1$ (0x1021h). While being sent, the most significant byte of CRC is sent firstly.

An example can be given to constitute CRC code:
Code string = 11010011101100
Polynomial = 1011

1	1	0	1	0	0	1	1	1	0	1	1	0	0
1	0	1	1										
0	1	1	0	0	0	1	1	1	0	1	1	0	0
	1	0	1	1									
0	0	1	1	1	0	1	1	1	0	1	1	0	0
		1	0	1	1								
0	0	0	1	0	1	1	1	1	0	1	1	0	0
			1	0	1	1							
0	0	0	0	0	0	0	1	1	0	1	1	0	0
							1	0	1	1			
0	0	0	0	0	0	0	0	1	1	0	1	0	0
								1	0	1	1		
0	0	0	0	0	0	0	0	0	1	1	0	0	0
									1	0	1	1	
0	0	0	0	0	0	0	0	0	0	1	1	1	0
										1	0	1	1
0	0	0	0	0	0	0	0	0	0	0	1	0	1

The process of extracting output stream is implemented by performing arithmetic operator XOR between input string and CRC polynomial. CRC polynomial is applied by starting from left hand side of the input stream on 1 bit values.

The Algorithm of the communication can be shown in figure:

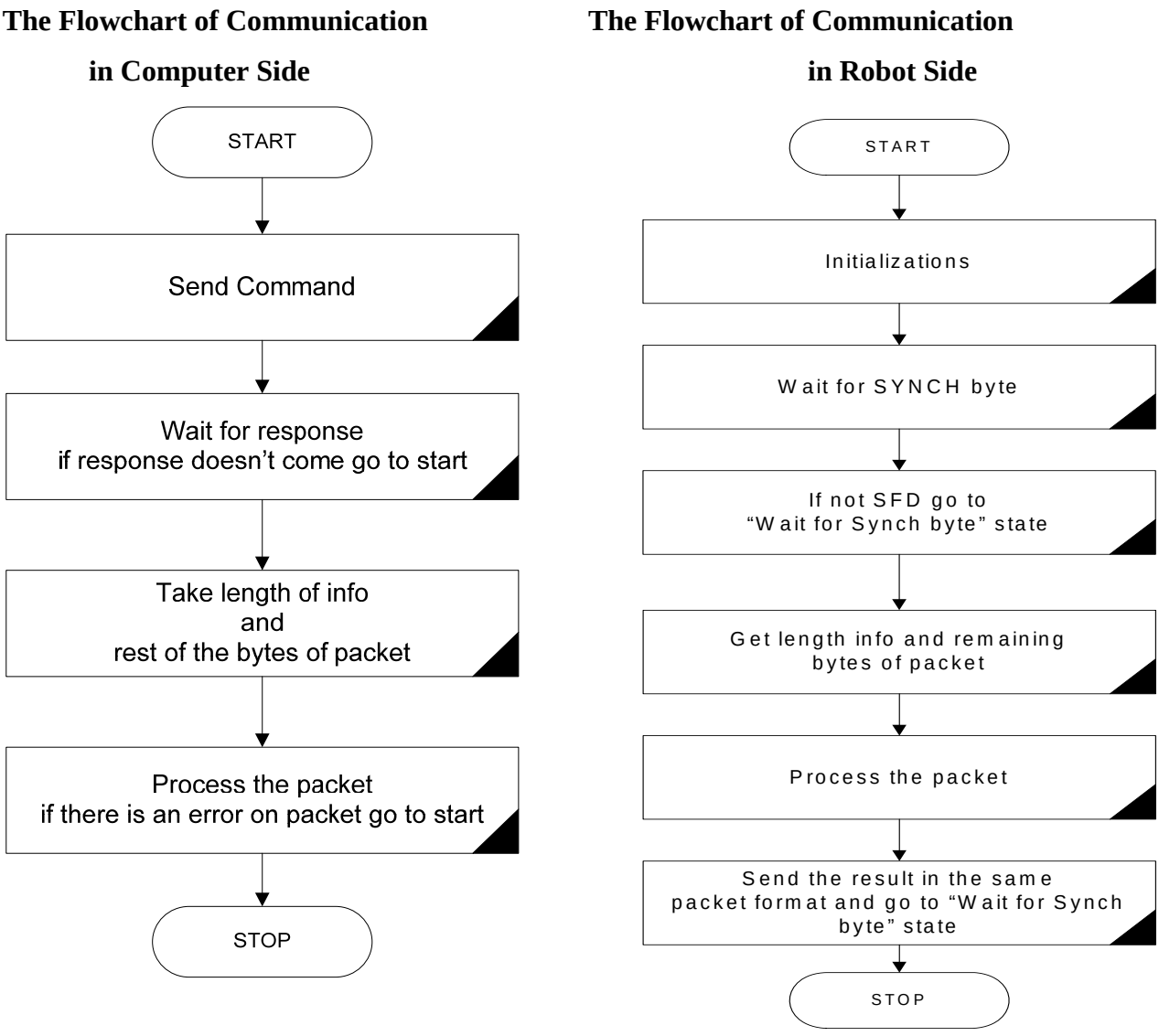


Figure 6.2. Communication Flow for Both Partlies

6.8. Process Steps:

1. Hello packet is send by computer (computer address = f, robot address =0)

0x55h	0x7Eh	0x04h	0xF0h	0x04h	0x4Eh	0x4Ah

2. Direction and velocity values are send to robot by computer.

0x55h	0x7Eh	0x04h	0xF0h	0x00h	0x80h	0x68h	0x81h	0x76h	0x46h	0xA6h

(DC motor) Address — 0x80h

Velocity value — 0x68h

Servo Address — 0x81h

Direction value — 0x76h

3. All the IR sensors, potentiometer and encoder values are read robot by computer.

0x55h	0x7Eh	0x04h	0xF0h	0x00h	0x00h	0x01h	0x02h	0x03h	0x04h
0x05h	0x06h	0x07h	0x80h	0x68h	0x81h	0x76h	0xE0h	0xE1h	

Potentiometer — 0x06h

IR sensor — 0x05h

encoder — 0x07h

6.9. Using javax.com Package for Communication

6.9.1. Class Hierarchy

- Class java.lang.Object
  - Class javax.comm.CommPort
    - Class javax.comm.ParallelPort
    - Class javax.comm.SerialPort
- Class javax.comm.CommPortIdentifier

- Class java.util.EventObject (Implements java.io.Serializable)
  - Class javax.comm.ParallelPortEvent
  - Class javax.comm.SerialPortEvent
- Class java.lang.Throwable (Implements java.io.Serializable)
  - Class java.lang.Exception
    - Class javax.comm.NoSuchPortException
    - Class javax.comm.PortınUseException
    - Class javax.comm.Unsupported CommOperationException

6.9.2. Interface Hierarchy

- interface javax.comm.CommDriver
- interface java.util.EventListener
  - interface javax.comm.CommPortOwnershipListener
  - interface javax.comm.ParallelPortEventListener
  - interface javax.comm.SerialPortEventListener

For the process of opening port by the help of the javax.comm library, the structure of that is given before, can be implemented as an example given following.

Port Name = COM5
Number of Data Bits = 8 bits
Number of Stop Bits = 1 bit
Parity = No parity
Baud Rate = 115200 bps

```
import javax.comm.CommPortIdentifier;
import javax.comm.NoSuchPortException;
import javax.comm.PortInUseException;
 import javax.comm.SerialPort;
import javax.comm.UnsupportedCommOperationException;
```

```
CommPortIdentifier cpi=new CommPortIdentifier.getPortIdentifier(“COM5”);
SerialPort sp=new SerialPort(SerialPort) cpi.open(“COM5”, 115200);
Sp.setSerialPortParams (115200,
                        SerialPort.DATABITS_8,
                        SerialPort.STOPBITS_1,
                        SerialPort.PARITY_NONE);
```

To be used for reading from port and writing on port, the variables of “os”,”is” should be created;

```
InputStream is = sp.getInputStream();
OutputStream  os = sp.getOutputStream();

byte b = (byte) is.read();  // reading byte from serial port

os.write(b); //writing to serial port
os.flush();
```

7. BACKGROUND ON MULTI-ROBOT SIMULATION

7.1. Behavior Based Formation Control Systems for Multi-Robot Simulation [9]

In Bench and Arkin's article, new reactive behaviors that implement formations in multi-robot teams are presented and evaluated. The formation behaviors are integrated with other navigational behaviors to enable a robotic team to reach navigational goals, avoid hazards and simultaneously remain in formation. The behaviors are implemented in simulation, on robots in the laboratory and aboard DARPA's HMMWV-based Unmanned Ground Vehicles. The technique has been integrated with the Autonomous Robot Architecture (AuRA) and the UGV Demo II architecture. The results demonstrate the value of various types of formations in autonomous, human-led and communications-restricted applications, and their appropriateness in different types of task environments.

. 1. A team of four robotic scout vehicles manufactured for DARPA's Demo II project. The formation techniques reported in this article were implemented on these robots. Photograph courtesy of Lockheed-Martin.

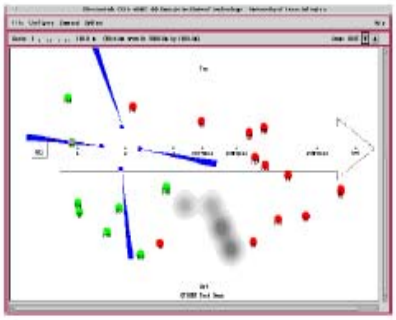


Fig. 2. An example of how scouts in formation focus their sensor assets so as to ensure complete coverage. Four robot scouts sweep from left to right in a *diamond* formation. The wedges indicate the sensor focus for each scout. Figure courtesy of Diane Cook

Figure 7.1. Behavior Based Control System Interface [9]

Figure 7.2. Formation Types [9]

Several formations for a team of four robots are considered (Figure 7.2):

- line - where the robots travel line-abreast.
- column - where the robots travel one after the other.
- diamond - where the robots travel in a diamond.
- wedge - where the robots travel in a V

These formations are used by U.S. Army mechanized scout platoons on the battlefield. For each formation, each robot has a specific position based on its identification number (ID). Figure 3 shows the formations and robots' positions within them.

Each robot computes its proper position in the formation based on the locations of the other robots. Three techniques for formation position determination have been identified [9]:

Unit-center-referenced: A unit-center is computed independently by each robot by averaging the x and y positions of all the robots involved in the formation. Each robot determines its own formation position relative to that center.

Leader-referenced: Each robot determines its formation position in relation to the lead robot (Robot1). The leader does not attempt to maintain formation; the other robots are responsible for formation maintenance.

Neighbor-referenced: Each robot maintains a position relative to one other predetermined robot.

7.1.1. Conclusion of the Study

Separate experiments in simulation evaluated the utility of the various formation types and references in turns and across obstacle fields. For 90º turns, the diamond formation performs best when the unit-center-reference for formation position is used, while wedge and line formations work best when the leader-reference is used. For travel across an obstacle field, the column formation works best for both unit-center- and leader-referenced formations. In most cases, unit-center-referenced formations perform

better than leader-referenced formations. Even so, some applications probably rule out the use of unit-center-referenced formations [9]:

Human leader: A human serving as team leader can not be reasonably expected to compute a formation's unit-center on the y, especially while simultaneously avoiding obstacles. A leader-referenced formation is most appropriate for this application.

Communications restricted applications: The unit-center approach requires a transmitter and receiver for each robot and a protocol for exchanging position information. Conversely, the leader-referenced approach only requires one transmitter for the leader, and one receiver for each following robot. Bandwidth requirements are cut by 75% in a four robot formation.

Passive sensors for formation maintenance: Unit-center-referenced formations place a great demand on passive sensor systems (e.g. vision). In a four robot visual formation for instance, each robot would have to track three other robots which may spread across a 180º field of view. Leader- and neighbor-referenced formations only call for tracking one other robot.

7.2. Nathan Koenig, Andrew Howard's Multi-Robot Simulator [10]

Simulators have played a critical role in robotics research as tools for quick and efficient testing of new concepts, strategies, and algorithms. To date, most simulators have been restricted to 2D worlds, and few have matured to the point where they are both highly capable and easily adaptable. Gazebo is designed to fill this niche by creating a 3D dynamic multi-robot environment capable of recreating the complex worlds that will be encountered by the next generation of mobile robots [10]. Its open source status, fine grained control, and high fidelity place Gazebo in a unique position to become more than just a stepping stone between the drawing board and real hardware: data visualization, simulation of remote environments, and even reverse engineering of black box systems are all possible applications.

This study, explains the design principles of Gazebo and its applicability to the development process of real world robotics. The software that Nathan Koenig, Andrew

Howard are developed has maintained a simple powerful interface to the underlying physics engine and rendering capabilities while retaining compatibility with the Player and Stage initiatives. This has resulted in a quick and easy adoption of Gazebo by many people, and has the potential to be used in ways never before seen in a simulator.

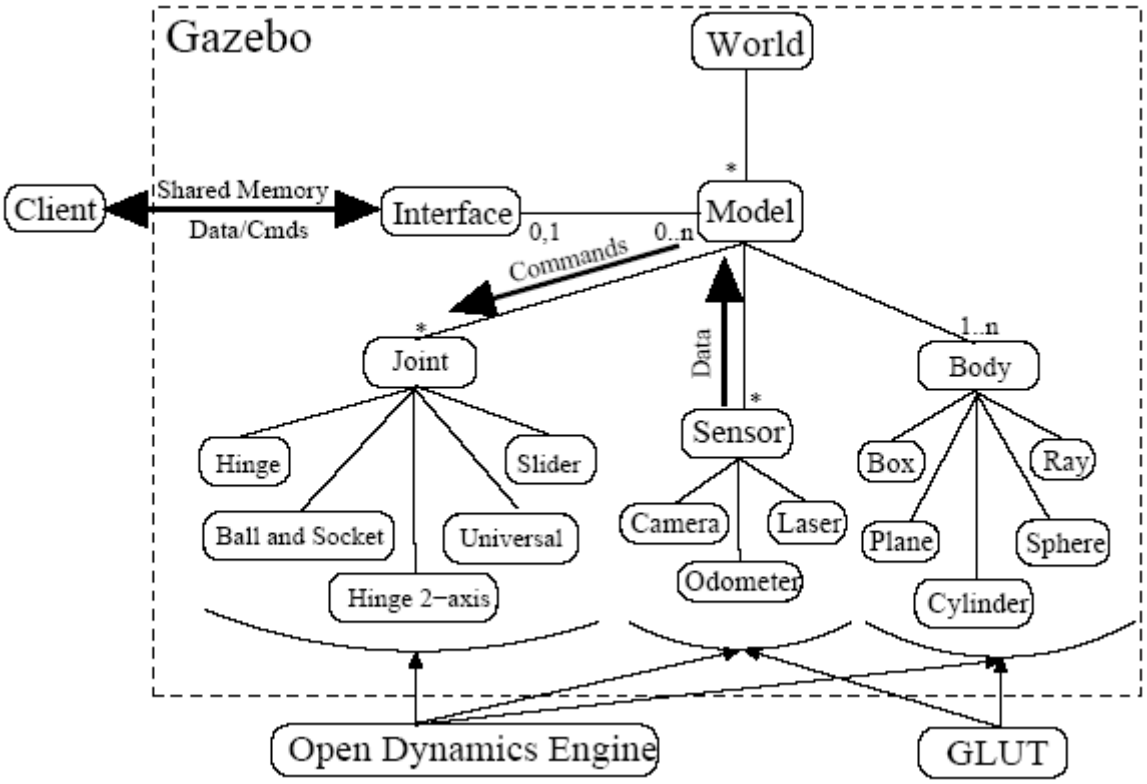


Figure 7.3. general Structure of Garazbo Components

8. RECOMENDED SYSTEM STRUCTURE

8.1. Designed User Interface

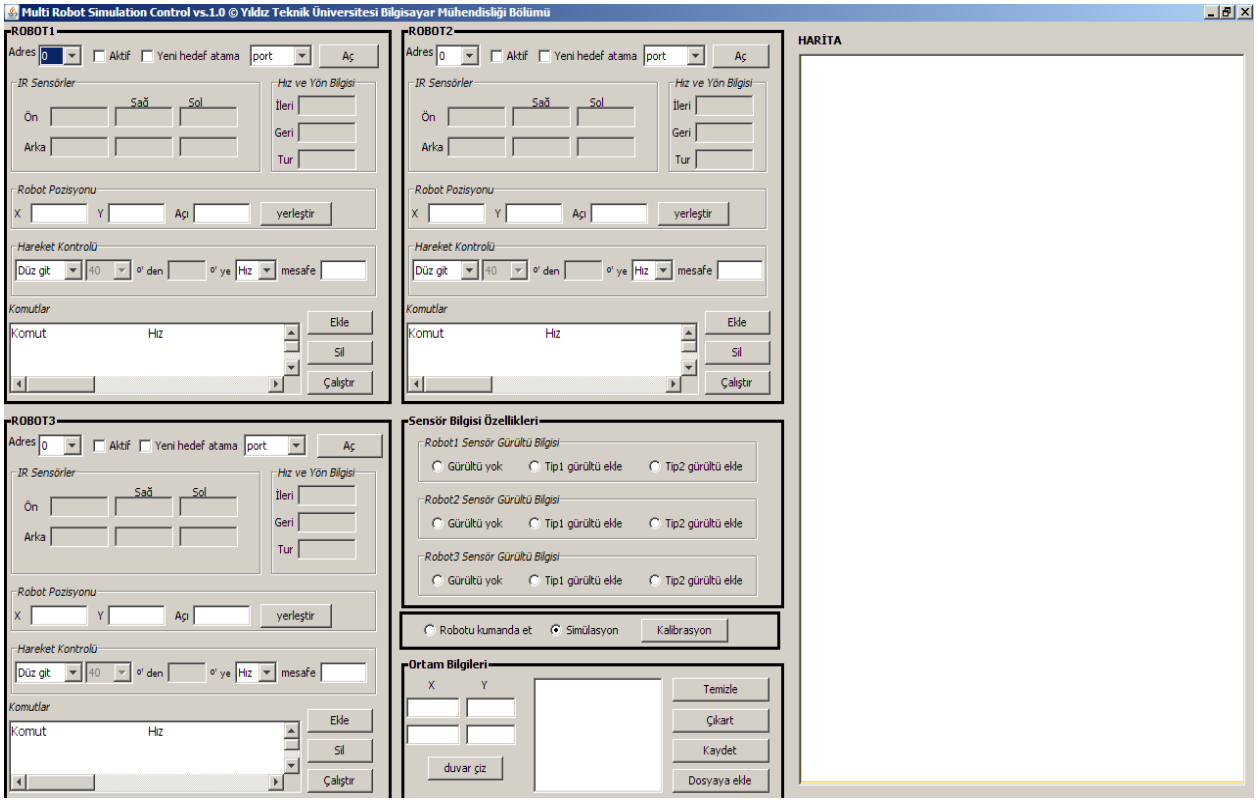


Figure 8.1. Designed User Interface

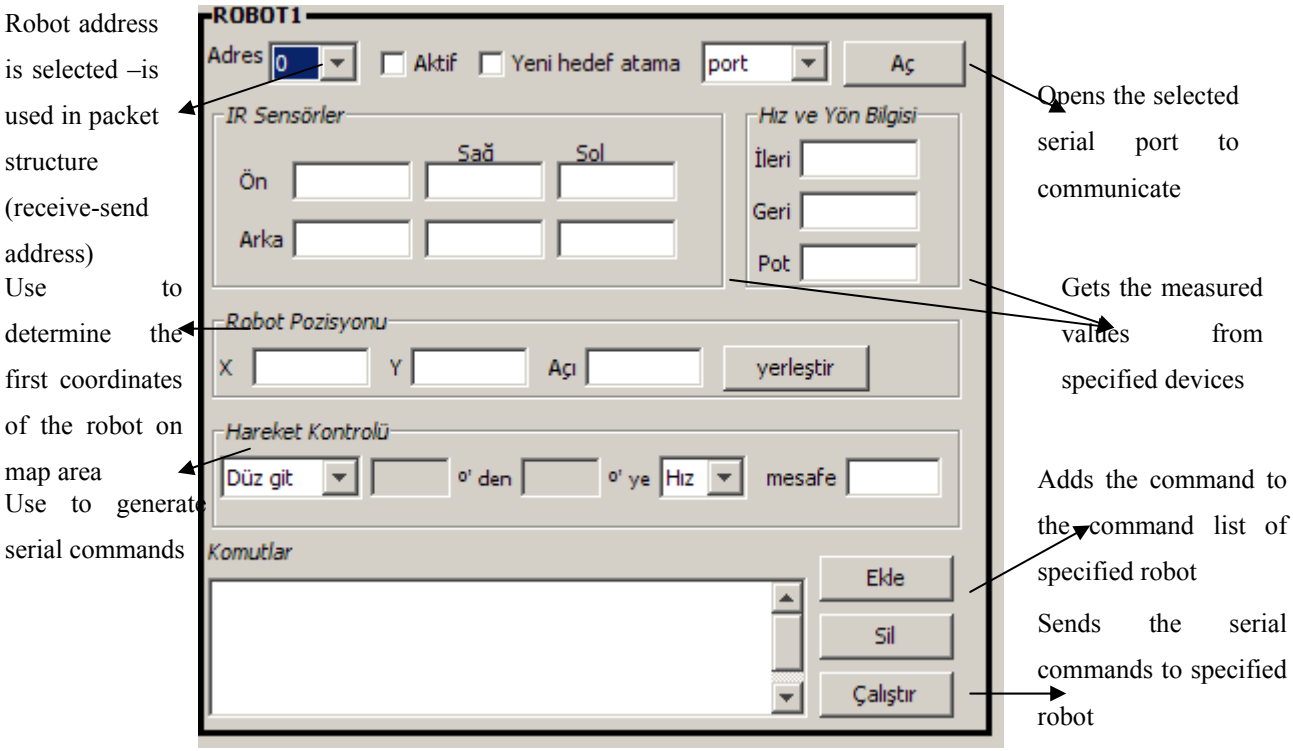
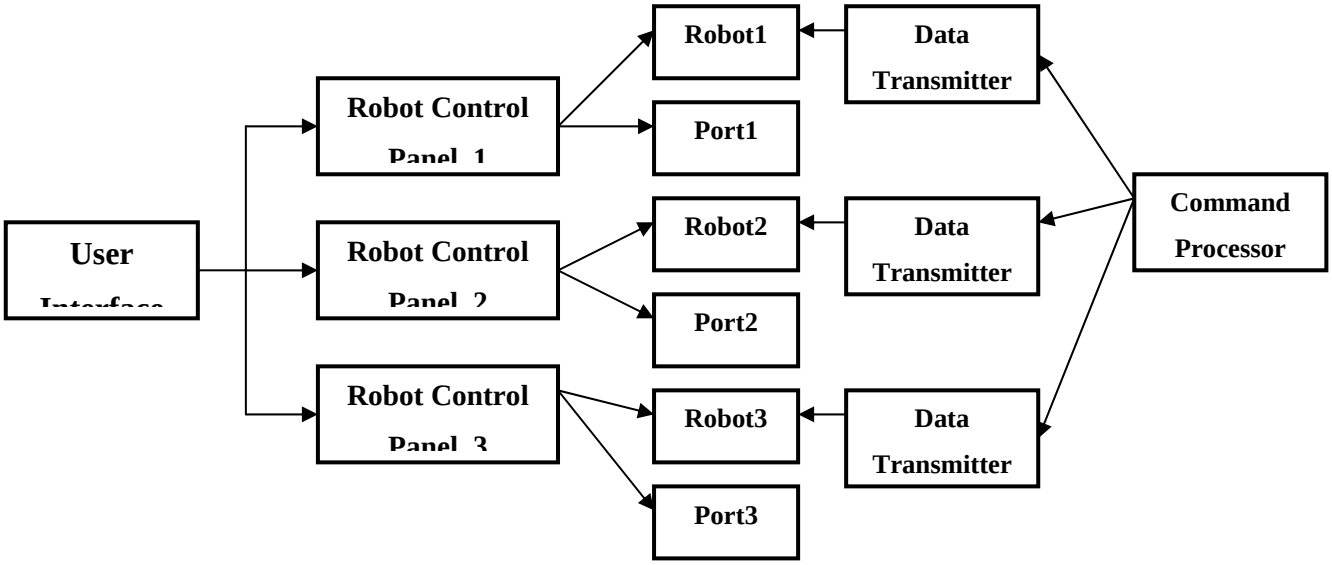


Figure 8.2. User Interface Robot Control

8.2. Designed System Structure and Object Relationship



In planned structure, the options and command series will be taken from user interface and then control panels will be formed. Control objects will be used to create objects for robots and control its behaviors by opening ports for its communications. Then after running action is triggered, control object will call the methods for converting the user command expressions to packets to send the specified robot and at the same time listen-port is opened in simulation side. After the packet transmissions, the awakened (by control panel) command processor that has kinematics inside, determines the next coordinates of the robot and the values of the required specified devices (IR, encoder…) on it through the data transmitters the measured (if real robot / calculated if simulation) data are transmitted to specified robot back.

In this designed structure, the specified byte values for the devices on the robot can be calibrated if requested. The “kalibrasyon” button helps the user to display the second interface of the program for calibration on which the default values are printed. If it is requested the values can be changed before simulation or commanding the robot.

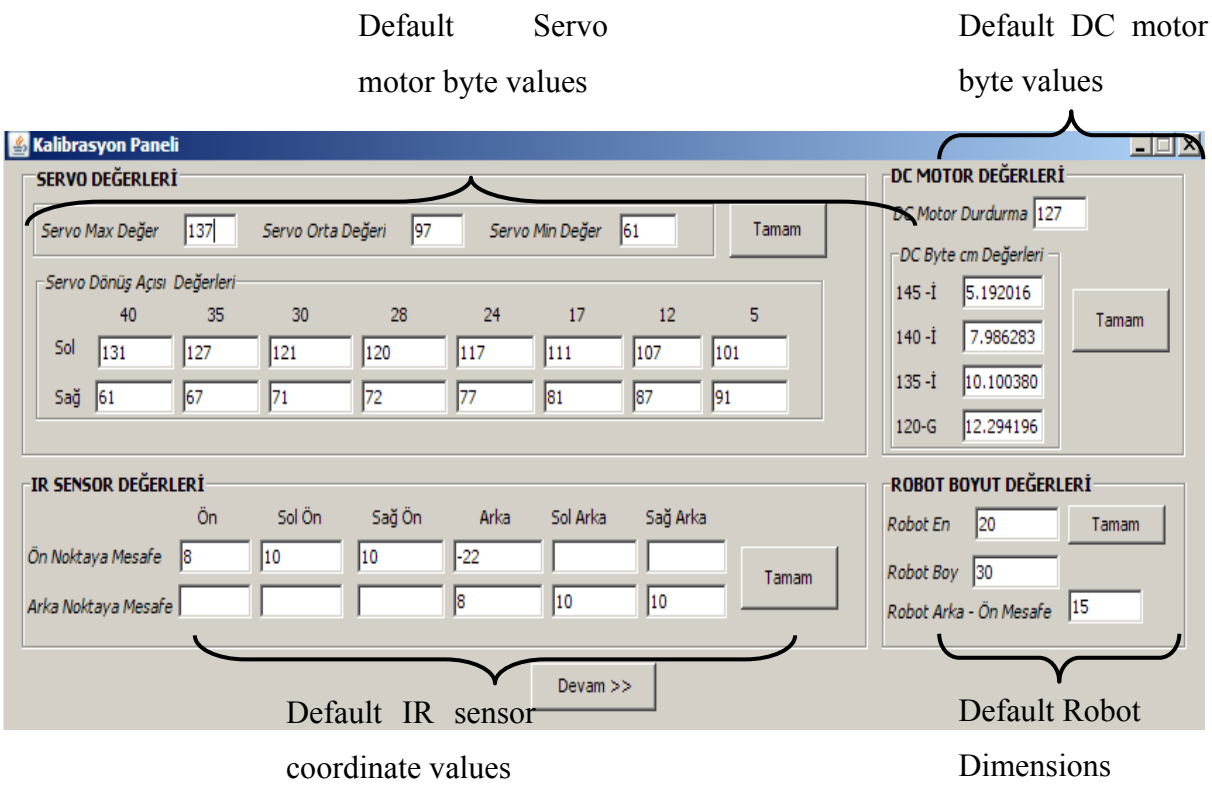


Figure 8.3. User Interface Calibration Panel

The above panel named “Kalibrasyon Paneli”, must be called before starting the simulation. Otherwise simulation will use the default values that are shown in above table. If calibration is applied than the window will disappear after the “Devam” button is clicked. Otherwise, until the event of click, the window states on screen and doesn’t allow the user to use simulation main interface.

Another section on the main menu is named “Ortam Bilgileri ” that is used to give the opportunity of preparing the environment that the robots will be circulated. By using this panel the user can prepare his own environment by drawing barriers by giving its start and stop x, y coordinates and clicking the button named “duvar çiz”.

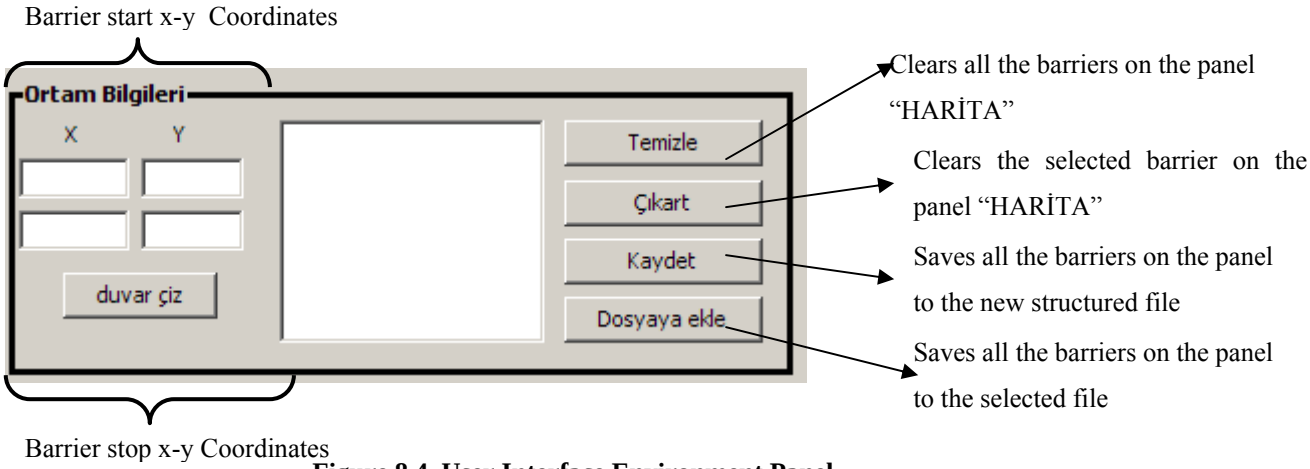


Figure 8.4. User Interface Environment Panel

Another important panel on the main menu interface is about the values that are taken from robot side depending of the environment and the device values. The values read from the IR sensors are affected from the environment conditions. But in simulation side, the values that are read from IR sensors are calculated mathematically inside of the algorithm and so they don't reflect the real ones because of the noise factor. To obtain more realistic values, it is decided to add noise factor to the mathematically calculated data. This is optional, if the user wants to add noise to the results of the calculations, one can click the related button by giving the type of the noise factor. Different two types of noise distribution can be optionally added.

Figure 8.5. User Interface Sensor Panel

In the simulation application, the user can activate single, double or three robots at the same time. The application can be used either as a single robot application or as a multi robot application.

8.3. Designed System Structural Mechanism

Designed system structure has four main tasks.

Control Side
1. Converting linguistic expressions to the command packets (encoding operation) that the robot gets the commands (decoding operation)
2. Sending the packets to the robot side sequentially by serial port operations

Robot Side
3. Listening the port in robot side and controlling if a new command is received
4. Processing the command that is sent from the central computer and getting the values from the robot side (in simulation calculates the values)

To implement that mentioned four main task three main thread is used. One of them is in the control side and others are in the robot side.

Task of the thread in control side: Sending the command packets sequentially to the robot, in this sequence, packets are sent after the prior command is processed.

Task of the first thread in robot side: Listening the port and evaluating the received bytes and determining the command packets to transmit it to the robot motion processor.

Task of the second thread in robot side: Making the robot perform the command that transmitted from port listener, calculating the values of the devices (IR, encoder…) according to the motion of the robot.

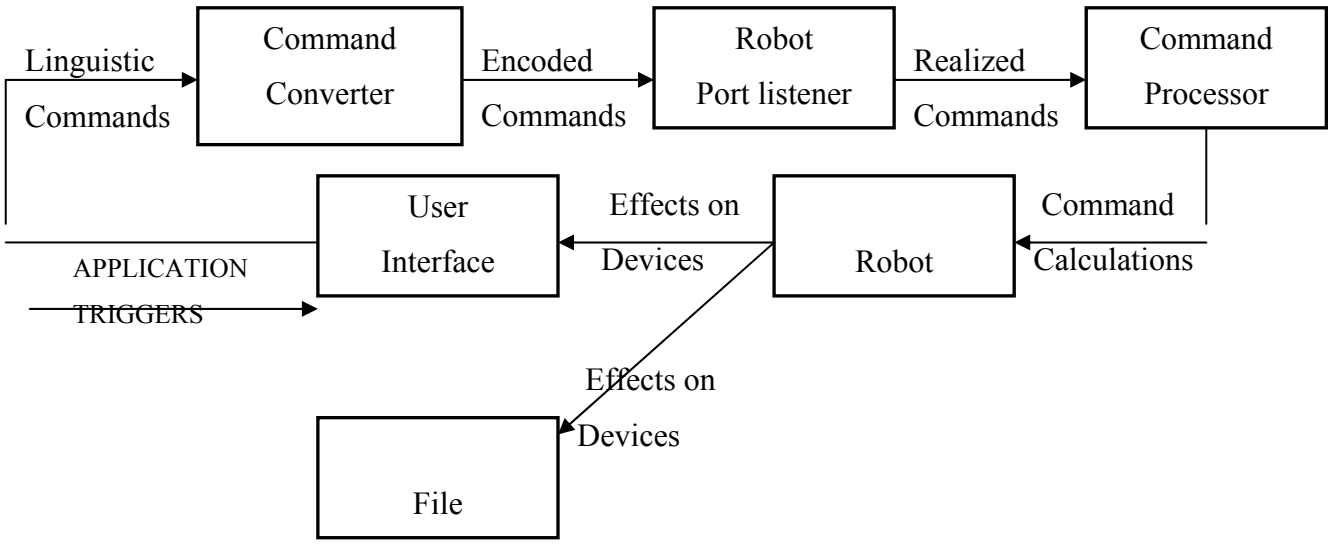


Figure 8.6. System Structural Mechanism Diagram

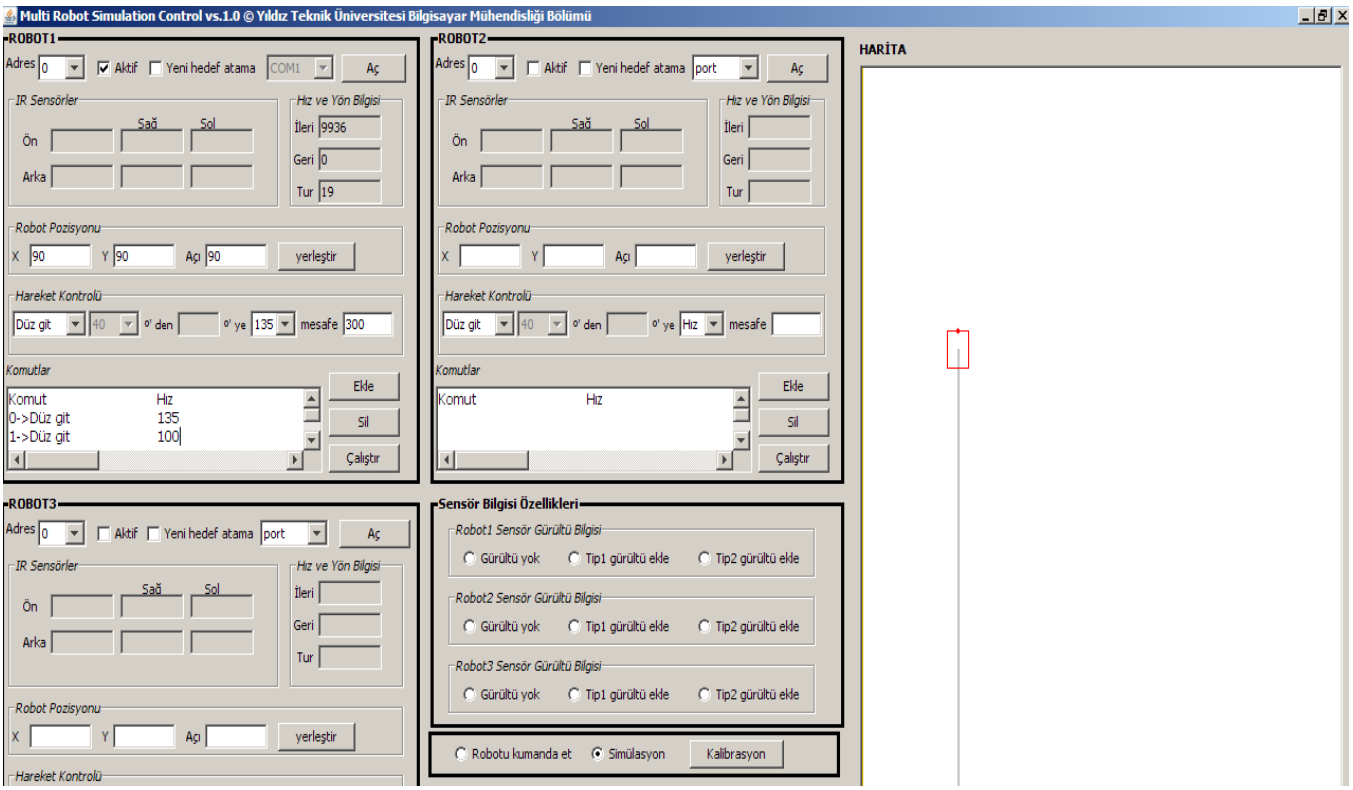


Figure 8.7. Program View

9. RESULT

With this project I had a large amount of information on robotics and modelling a system by using kinematics equations.

Java is an important object oriented programming language in the information technology. I used advanced java apis and learned very much about java. Robot's communication system, movement modelling as simulation and robot control system's codes is written with this technology. So I used different classes which have different duties, on the same framework.

Communication type used in project, Rf communication, is a good way for robotic communication system. I had a large amount of information about RF communication that is an effective way for robotic communication. Between robot and computer, communication with RF played a basic role. I had to learn all things about communication.

The important thing is, I learned how to work with a project group and to organize project schedule. I have had excellent experience with this project.

REFERENCES

[1] “Java Uygulamaları”, David Flanagan, Pusula Yayıncılık, 2004 İstanbul

[2] “Java Examples”, http://www.java2s.com/

[3] “Java Tutorials”, http://java.sun.com/docs/books/tutorial/

[4] http://www.ercim.org/publication/Ercim_News/enw42/rives.html

[5] http://www.inria.fr/recherche/equipes/icare.en.html

[6] http://robotica.isa.upv.es/virtualrobot/

[7] Yıldız Teknik Üniversitesi, Elektrik-Elektronik Fakültesi, Bilgisayar Mühendisliği Bölümü, Robotik Grubu, Şubat 2008 “ Eş Zamanlı Konum Belirleme Ve HaritaOluşturma Amaçlı Otonom Robot Sistemi Projesi Kılavuzu”, guide

[8] Azarnasap Ehsan, 2007, “Robot in the Loop Simulation to Support Multi-Robot System Development: A Dynamic Team Formation Example”, Master Thesis Georgia State University

[9] Balch T., Alkin C.,1999, “Behavior Based Formation Control for Multi Robot Teams”, IEEE Transactions on Robotics and Automation

[10] Koenig N., Howard A., 2004,” Design and Use Paradigms for Gazebo, An Open-Source Multi-Robot Simulator”, IEEE/RSJ International Conference on Intelligent Robots and Systems Robotics Research Labs, University of Southern California Los Angeles, CA 90089-0721, USA

RESUME

Name Surname	: Belgin TAŞDELEN
Birthdate	: 23–01–1985
BirthPlace	: İstanbul
High School	: İstanbul Köy Hizmetleri Anatolian High School
Internship	: BELBIM AŞ.
	Mavili Elektronik Ticaret AŞ.
	Garanti Teknoloji