AUTONOMOUS VEHICLE: SMART RC CAR

By

INTAN SHAKIRA BINTI AHMAD FAROK

FINAL PROJECT REPORT

Submitted to the Electrical & Electronics Engineering Programme
in Partial Fulfillment of the Requirements
for the Degree
Bachelor of Engineering (Hons)
(Electrical & Electronics Engineering)

Universiti Teknologi Petronas
Bandar Seri Iskandar
31750 Tronoh
Perak Darul Ridzuan

CERTIFICATION OF APPROVAL

AUTONOMOUS VEHICLE: SMART RC CAR

by

Intan Shakira binti Ahmad Farok

A project dissertation submitted to the
Electrical & Electronics Engineering Programme
Universiti Teknologi PETRONAS
in partial fulfilment of the requirement for the
Bachelor of Engineering (Hons)
(Electrical & Electronics Engineering)

Approved:

____

Mr. Abu Bakar Sayuti Mohd Saman
Project Supervisor

UNIVERSITI TEKNOLOGI PETRONAS
TRONOH, PERAK

December 2005

CERTIFICATION OF ORIGINALITY

This is to certify that I am responsible for the work submitted in this project, that the original work is my own except as specified in the references and acknowledgements, and that the original work contained herein have not been undertaken or done by unspecified sources or persons.

Intan Shakira binti Ahmad Farok

ABSTRACT

The objective of this report is to build a prototype of an autonomous vehicle that can operate mostly on its own to get from a designated starting point to a designated destination. The basis of the prototype was a remote controlled (RC) connected to a personal computer. Communication between the RC car and the PC were achieved via two FM transmitter-receiver pairs. The intelligent part (car moving autonomously) was programmed in the Visual Basic program with a fundamental usage of fuzzy logic. The software can control the basic movement of the car such as moving forward, turning and stopping. Obstacle avoidance was coded with the aid of obstacle sensors fitted to the front end of the car. The software was able to compute the distance traveled by counting the revolution of one of the wheels with the use of an Infrared (IR) sensor. The car was able to move from point A to point B through a pre-determined route while avoiding obstacles placed randomly along the route. This project is an alternative for inventing a 'smart' car which can travel on the road without the need of human control. An application from the project is a "Mobile Security Car" where the car runs on a pre-determine route to monitor an area with a camera attach on it. At the end of this report, conclusion and recommendations are made to summarize up the overall project progress.

ACKNOWLEDGEMENTS

Alhamdulillah, with my greatest gratitude to the Almighty Allah for his gracious blessings throughout this project was commenced. I would like to thank my supervisor Mr. Abu Bakar Sayuti whom given me the insightful guidance, valuable suggestions, timely encouragement and fully support for me in completing the project. I would also like to thank him for taking the time to discuss problems encountered. Discussions with him have always been filled with inspirations and stimulations. My personal thanks are also dedicated to all EE technicians, lecturers and colleagues in the Electrical and Electronics department, for their continuous support during the period of my research. I would also like to express special thanks to Miss Hawa for the precious friendship and for always being fun to work with. My sincere appreciation to Ong Kee Keong for his support and the good advices he has given me during my hard time. All the researches and experiments done with him concerning on the project are very valuable to me in carrying out this thesis. I would also like to express my gratitude to Azuan Azizan for providing me useful information and guided me throughout my research. Last but not least, I would like to express a deepest gratitude to my parents for sharing all my pains and joys, for encouraging and supporting me all the time. I would not have been successful without their wishes and blessings. I really hope with all the efforts taken throughout the year on the project will eventually bring benefits to the humanity.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

EE	Electrical Electronic
FYP	Final Year Project
RC	Remote-controlled
PC	Personal computer
IR	Infrared
VB	Visual Basic
IC	Integrated circuit
RF	Radio Frequency
FM	Frequency Modulation
VR	Variable Resistor (rheostat)
LED	Light Emitter Diode
V	Voltage
R	Resistor
Tx	Transmitter
Rx	Receiver

CHAPTER 1
INTRODUCTION

An autonomous vehicle is a vehicle that can get from point A to point B without human intervention. Therefore it has to rely on its own onboard "intelligence" to make decisions. The intelligence is normally implemented as a piece of software for flexibility. The vehicle also has various sensors that can give feedback to the system. One of the most important sensors is the obstacle detection. If the vehicle were to encounter an object either front left, front right, or exactly in front of its route it would be able to re-plan the mission accordingly.

The project is implemented to create an autonomous remote controlled (RC) car which can move on its own and avoid obstacles. For the first stage of the project, the RC car was programmed to move in a pre-determined route within its control speed. By utilizing basic fuzzy logic technique in the software on a PC, an 'intelligent' car was able to be implemented. The RC car also had been interfaced with the programmed route through the PC's parallel port. For the second stage of the project, the intelligent part of the RC car is implemented. This is done by adding two sensors to provide obstacle avoidance and distance measurement of the RC car. Moreover, enhancement on the visual basic coding on the pre-determined route was done to suit the obstacle avoidance of the RC car. All modification was done to make the car able to move autonomously around the route and avoid any obstacles in front. Figure 1 shows the cycle diagram of the Smart RC car.

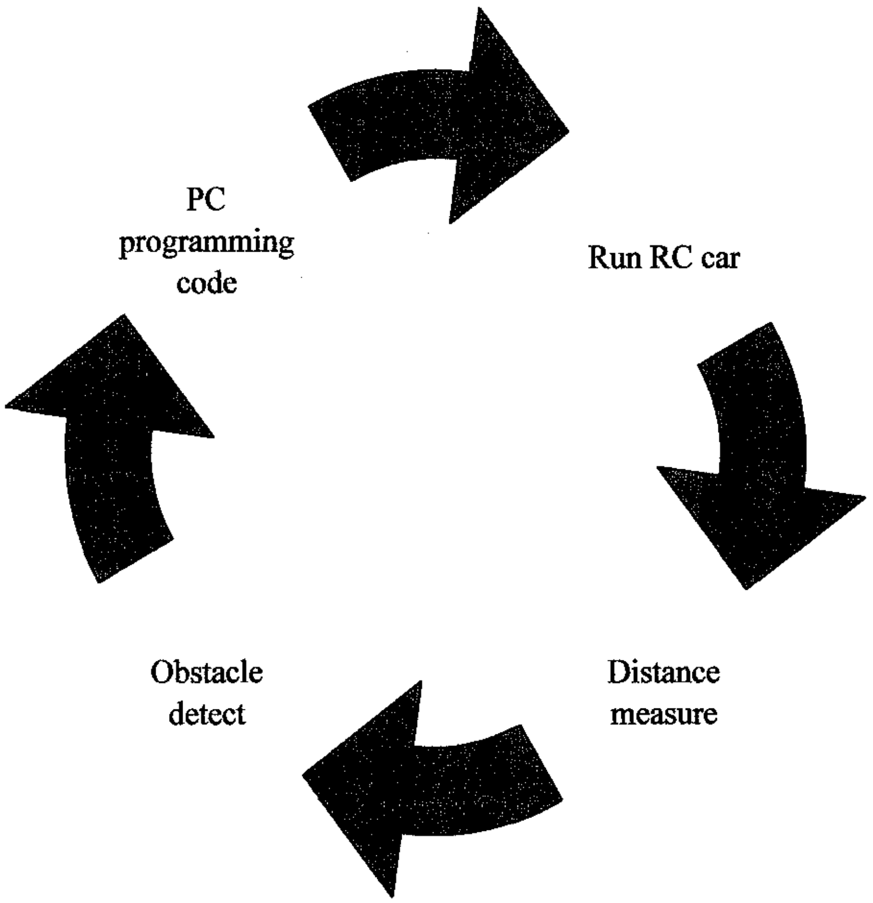


Figure 1 Cycle diagram of the Smart RC car.

1.1 Background of study

There are a few studies done to complete the objectives of the project;-

- Programming code on the Visual basic for pre-determine route.
- Ultrasonic sensor specification and datasheet.
- Voltage regulator circuit diagram.
- Infrared sensor specification and datasheet.
- Controllable speed of the RC car.
- Enhancement on the graphics programming in Visual Basic

1.2 Problem Statement

An RC car needs a driver to control its movement by using the RC transmitter. The aim of this project is to replace the function of the driver with software on a computer. The objective is to enable the computer to autonomously drive the RC car around a pre-determined route and be able to avoid obstacles. The velocity of the RC car cannot be controlled in current models. Therefore, research was also conducted in providing the controllable speed from the programming code of the computer. To provide a distance measurement and obstacle avoidance path, two sensors were fitted on the car: Infrared sensor (IR) and Ultrasonic sensor. These sensors were used to send signals from the RC car to the software to respond on the circumstances.

[illegible]	[illegible]	[illegible]
Runs autonomously on the pre-determined route	NO	YES
Avoid obstacles autonomously	NO	YES
Autonomously control speed	NO	YES
Distance measure	NO	YES

Table 1 Comparison between Normal RC car with the Smart RC car

1.3 Objectives and Scope of Study

The project consists of four main objectives which are described below;-

- Design the pre-determined route by using Visual Basic Programming.
- Examine the method to control the speed of RC car.
- Design and build the circuit for the distance sensor to measure the distance of the RC car.
- Design and build the obstacle sensor to provide an obstacle avoidance path.

There are also five main scopes which will be covered throughout the study as stated below;-

- Visual Basic 6.0 Programming
- Parallel port programming in Visual basic 6.0
- Designing and implementing infrared and ultrasonic sensors
- Designing and implementing speed controller
- Integrating two RF transmitters and two RF receivers.

Next, Chapter 2 explains on the theory and literature review that were used throughout the study period of the project. Chapter 3 discusses the methodology of the project starting from the designing process until the end of the implemented hardware. Chapter 4 covers results of the project and discussion on the results obtained. And lastly, Chapter 5 concludes the overall project and recommendations plan for the future works and improvements.

CHAPTER 2
THEORY AND LITERATURE REVIEW

2.1 Theory

The fundamental research was conducted before implementing the project. This chapter explained on the research of the related theory and literature review throughout the study period of the project

2.1.1 Fuzzy logic application

2.1.1.1 Introduction to fuzzy logic

The concept of Fuzzy Logic (FL) was conceived by Lotfi Zadeh, a professor at the University of California at Berkley, and presented not as a control methodology, but as a way of processing data by allowing partial set membership rather than crisp set membership or non-membership. [7] Fuzzy logic is used in system control and analysis design, because it shortens the time for engineering development and sometimes, in the case of highly complex systems, is the only way to solve the problem. Fuzzy Logic has been gaining increasing acceptance during the past few years. There are over two thousand commercially available products using Fuzzy Logic, ranging from washing machines to high speed trains. Nearly every application can potentially realize some of the benefits of Fuzzy Logic, such as performance, simplicity, lower cost, and productivity. [9]

2.1.1.2 The Fuzzy Logic Method

The fuzzy logic analysis and control method consist of three main parts which is explained below;-

- Receiving of one, or a large number, of measurement or other assessment of conditions existing in some system we wish to analyze or control.
- Processing all these inputs according to human based, fuzzy "If-Then" rules, which can be expressed in plain language words, in combination with traditional non-fuzzy processing.
- Averaging and weighting the resulting outputs from all the individual rules into one single output decision or signal which decides what to do or tells a controlled system what to do. The output signal eventually arrived at is a precise appearing, defuzzified, "crisp" value. Figure 2 shows the following Fuzzy Logic Control/Analysis Method diagram. [7]

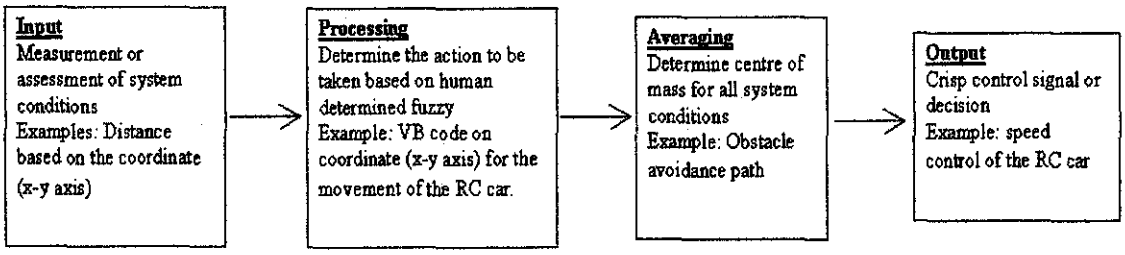

Input
Measurement or assessment of system conditions
Examples: Distance based on the coordinate (x-y axis)
Processing
Determine the action to be taken based on human determined fuzzy
Example: VB code on coordinate (x-y axis) for the movement of the RC car.
Averaging
Determine centre of mass for all system conditions
Example: Obstacle avoidance path
Output
Crisp control signal or decision
Example: speed control of the RC car

Figure 2 The Fuzzy Logic Control-Analysis Method [7]

2.1.1.3 Fuzzy logic terms

- **Fuzzy** - The degree of fuzziness of a system analysis rule can vary between being very precise, in which case we would not call it "fuzzy", to being based on an opinion held by a human, which would be "fuzzy." Being fuzzy or not fuzzy, therefore, has to do with the degree of precision of a system analysis rule. [7]
- **Fuzzy Sets** - A fuzzy set is almost any condition for which we have words: accelerator setting, speed, weight, spongy, etc., where the condition can be given a value between 0 and 1. [7]
- **Degree of Membership** - The degree of membership is the placement in the transition from 0 to 1 of conditions within a fuzzy set. [7]
- **Fuzzy Variable** - Words like red, blue, etc., are fuzzy and can have many shades and tints. They are just human opinions, not based on precise measurement in angstroms. These words are fuzzy variables. If, for example, speed of a system is the attribute being evaluated by fuzzy, “fuzzy" rules, then "speed" is a fuzzy variable. [7]
- **Linguistic Variable** - Linguistic means relating to language, which known as language words. Examples of linguistic variables are: somewhat fast speed, very high speed, real slow speed, etc. A fuzzy variable becomes a linguistic variable when it is modified with descriptive words, such as somewhat fast, very high, real slow, etc. The main function of linguistic variables is to provide a means of working with the complex systems mentioned above as being too complex to handle by conventional mathematics and engineering formulas. Linguistic variables appear in control systems with feedback loop control and relate to each other with conditional, "if-then" statements. [7] Example: If the speed is too fast, then back off on the high accelerator setting.
- **Fuzzy “Cause Effect”** - Fuzzy cause effect is the final outcomes of the fuzzy logic application. It consists of charting techniques which include variables used as an inputs and outputs of a project. [7]

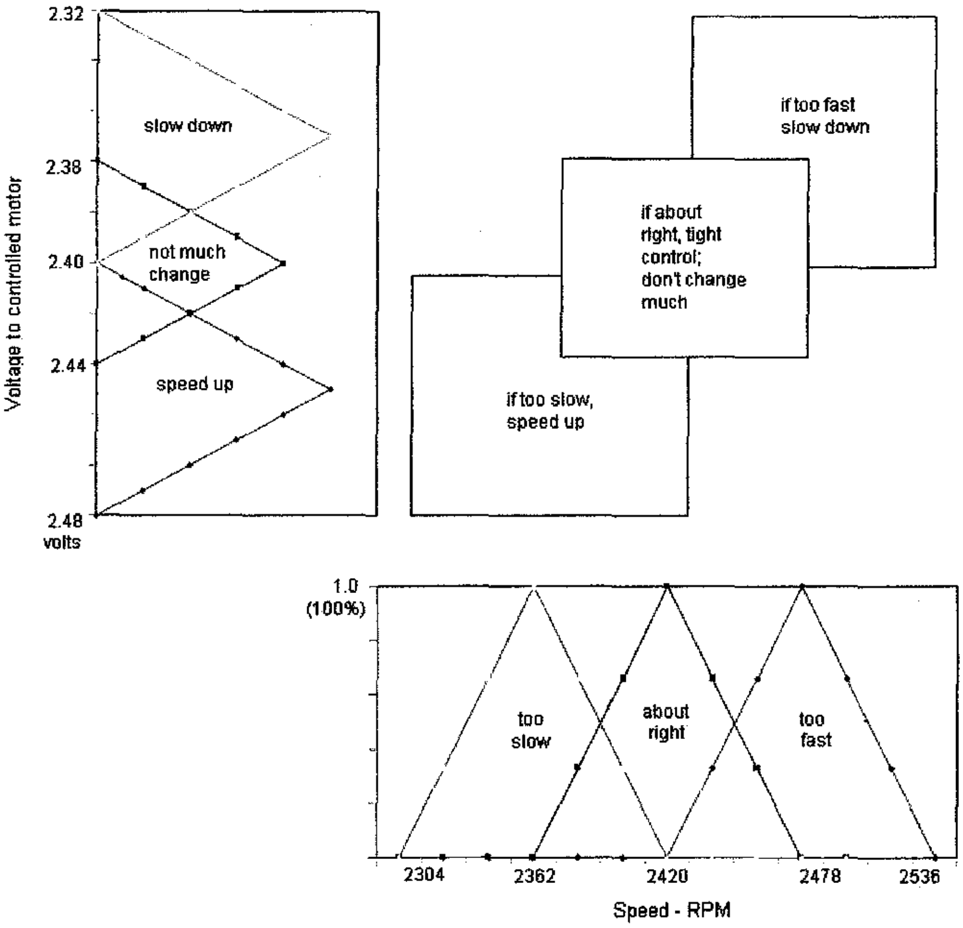


Figure 3 "Cause-Effect" triangle diagram. [7]

Example application of fuzzy "Cause Effect"

Figure 3 shows "Cause Effect" of three "linguistic" rules. The "Cause Effect" is a charting technique which is used to determine the "degree of membership" of the 'Too slow', 'About right' and 'Too fast' triangles, for a given speed. The charting technique helps to make the continuous control feedback loop easier in visualizing the program and fine tuning process. Inputs (Too slow, About right and Too fast) and outputs (Speed up, Not much change needed, Slow down) are associated as causes and effect. Other shapes such as bell curves can also be used to replace the triangles, but triangles are easier to work with. The width of the triangles can vary throughout the scale. Narrow triangles provide tight control when operating conditions are in their area while wide triangles provide looser control. Narrow triangles are usually used in the middle, which is at the set point (the target speed) to give a better speed control. [7]

2.1.2 Infrared IR sensor

Light with wavelengths from 0.7 micron to about 0.1 millimeter is called infrared light. Infrared light contains the least amount of energy per photon of any other band. The range of infrared rays is of thousands times larger than that of visible light. All of the rays are invisible to human eyes. Infrared films and normal video cameras are sensitive to 'very near infrared light' (0.7 to 0.9 microns). [10] Infrared radiation is light just like visible light because it has the same properties as visible light. Infrared can be focused and reflected like visible light. Infrared light can also be "aligned" like regular light and therefore polarized. Infrared light lies between the visible and microwave portions of the electromagnetic spectrum. Infrared light has a range of wavelengths, just like visible light has wavelengths that range from red light to violet. "Near infrared" light is closest in wavelength to visible light and "far infrared" is closer to the microwave region of the electromagnetic spectrum. The longer, far infrared wavelengths are about the size of a pin head and the shorter, near infrared ones are the size of cells, or are microscopic. [12]

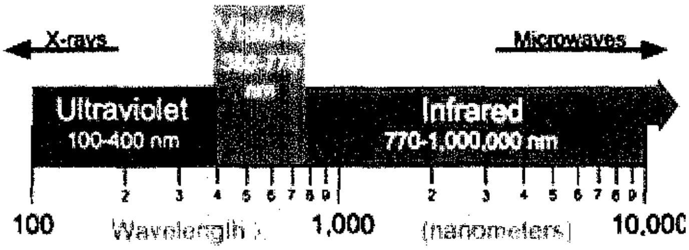


Figure 4 The optical portion of the electromagnetic spectrum [12]

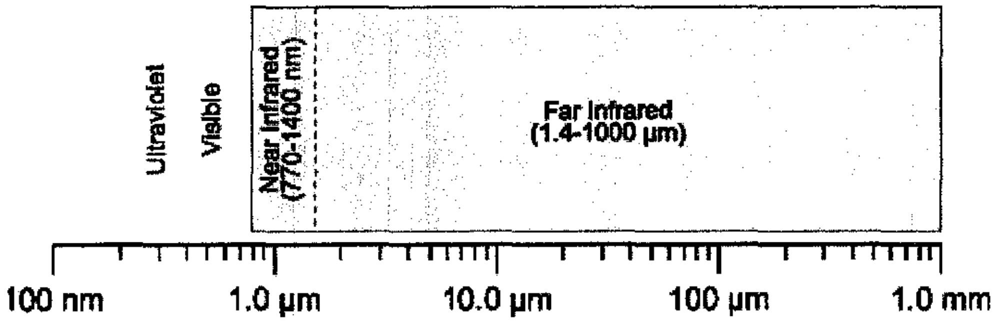


Figure 5 The infrared spectrum [12]

2.1.2.1 IR distance sensor mechanism

The project used one pair of IR transducer sensor which is located at one of the tire at the back. With a power supply of 5 volts, the IR transducer sensor used to detect a change of color on a surface for distance of 1mm. For the project, the IR distance circuit is being placed at one of the tire. According to the reflection theory, white surface will reflect any light and black surface will absorb any light. Therefore, a white tape is used to reflect the spreading of the IR light. Each time the IR transducer detect a white tape, the LED will be OFF (the LED is normally close). A potentiometer (VR) is used to adjust the sensitivity of the sensor.

2.1.3 Ultrasonic Sensor

Ultrasound refers to any sound whose frequency is above the audible range, i.e. above 20 kHz. Ultrasonic range sensors receive or radiate ultrasonic energy in the range of 20 kHz-60 kHz. [13] Ultrasonic sensors were used in this project to detect obstacles. An ultrasonic sensor provides range information based on the time of flight (TOF) principle as given in Equation;-

$$d = vt$$ [14]

where d is the round-trip distance, v is the speed of propagation of the pulse and t is the elapsed time. According to ultrasonic TOF ranging system is today the most commonly used technique employed on mobile robotics systems, primarily due to the following reasons;-

- **Low cost** - Cost is a very important element in developing service robots for everyday use. Ultrasonic sensors are widely available at very low prices. In this aspect, the ultrasonic sensor has great advantage over the laser scanner and other sensors such as the camera. [14]
- **Easy maintenance** - For practical use, maintenance is an important issue. Ultrasonic sensors are compact in design, light in weight and very reliable. It is also easy to interface these sensors with other sub-systems of the robot because the output and input to this module are standard TTL signals.

For example, the ultrasonic sensor module can be directly connected to I/O ports of the microcontroller or PC, which work as the navigation or control subsystem in a robot system. [14]

- **High range detection accuracy** - The range detection from the ultrasonic sensor is very accurate. Ultrasonic sensor is able to provide range information from 0.1524m (6 inches) to 10.668m (35 feet). [14]

2.1.3.1 Ultrasonic sensor mechanism

The project used one pair of ultrasonic sensor transducer which is located in front of the RC car. With a power supply of 9 volts, the 40 kHz ultrasonic transducers detect objects or human bodies up to 10 meters. If there is any obstacle in front, the obstacle will reflect the ultrasonic signal to the receiver. Then, the ultrasonic signal will trigger the relay to energize for 2 seconds.

2.1.4 Parallel port

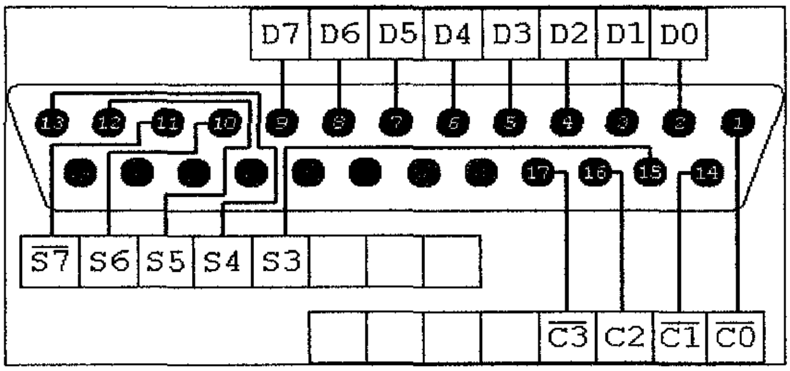


Figure 6 25-way Female D-Type connector. [4]

The PC parallel port adapter is specifically designed to be used as a general input/output port for any device or application that matches its input/output capabilities. It has 12 TTL-buffer output points, which are latched and can be written and read under program control using the processor **Inp** or **Out** instruction. The adapter also has five steady-state input points that may be read using the processor's Inp instruction. As refer to the Figure 6, each of the parallel port pins is defined for certain purpose;- [4]

- 8 output pins accessed via the **DATA Port (D0-D7)**
- 5 input pins (one inverted) accessed via the **STATUS Port (S3-S7)**
- 4 output pins (three inverted) accessed via the **CONTROL Port (C0-C3)**
- The remaining 8 pins are grounded

2.1.4.1 Parallel port mechanism

The project used data port pins as the output pins and status port pins as the input pins. From the PC, the VB programming command will send 5V voltage signals to parallel port and received by the transmitter to activate the command. The voltage signal is then encoded to RF signal to transmit through the air and received by the RF receiver inside the RC car. The RF signal will then being decoded back to the voltage signal before sent to the motors to execute the commands.

2.1.5 Remote Control (RC) car

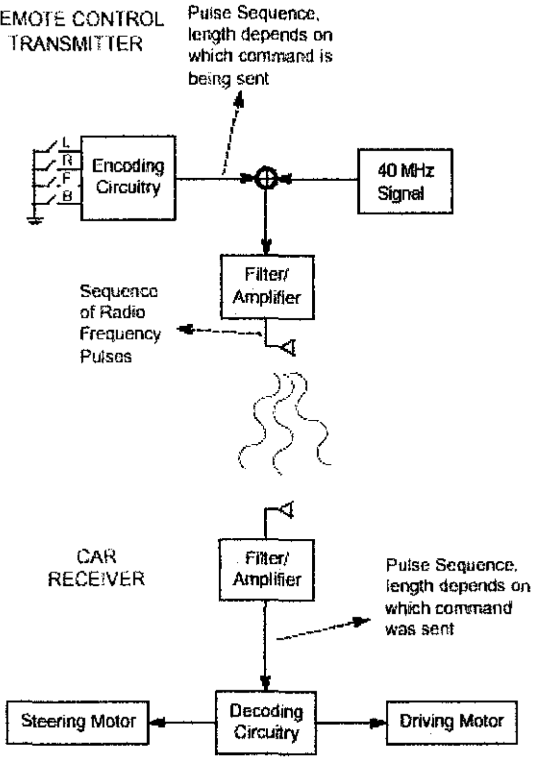

REMOTE CONTROL TRANSMITTER
Pulse Sequence, length depends on which command is being sent
L
R
F
B
Encoding Circuitry
40 MHz Signal
Filter/ Amplifier
Sequence of Radio Frequency Pulses
CAR RECEIVER
Filter/ Amplifier
Pulse Sequence, length depends on which command was sent
Steering Motor
Decoding Circuitry
Driving Motor

Figure 7 Block diagram of Remote Control Car. [11]

From the figure 7, when any of the switches is closed, it will produce pulse sequences. Then the pulses will be modulated with 40 MHz signal. The modulated signal then is amplified by the amplifier. The antenna then converts the electrical energy into radio signal. Then the radio signal is transmitted. When the signal reaches the receiver, the receiver then converts the radio signal into electrical energy. The electrical energy then being decoded by the decoder and decoder decides where to send the signal; either to steering motor or driving motor, or both of them. [11]

2.1.5.1 Transmitter

When the levers in the Remote Control Unit are pushed electrical contacts are made by the switches thus connecting the 7.2V battery power to the transmitter and indicating which commands the user wants sent to the car. Forwards/Backwards and Left/Right commands are controlled by different sets of electrical contacts that are used to encode a sequence of electrical pulses: the number of pulses depends on which command is being sent. In QD Sports RC car model, Left/Right commands are only sent if Forwards/Backwards commands are also being sent, since there is too much friction to turn the wheels unless the car is moving. An electrical circuit that is tuned to a frequency of 40 MHz creates a signal that is sent to the antenna when the pulses are active. The antenna converts this electrical energy into radio energy, creating a stream of radio energy burst, which travel through the air and are picked up by and understood by the radio receiver in the car. For the particular model of RC car used in the project, the pulse sequences are;- [7]

- Forward – 16 pulses
- Backward – 40 pulses
- Right – 18 pulses
- Left – 6 pulses
- Forward Right – 34 pulses
- Forward Left – 28 pulses
- Backward Right – 46 pulses
- Backward Left – 52 pulses

2.1.5.2 Receiver

The car antenna collects radio energy and converts it back into electrical energy. The energy will always be much less than the energy originally applied to the transmitting antenna. If the car is turned on then the radio frequency in the car is continuously monitoring the electrical energy from its antenna. The receiver is basically a filter which is turned to amplify any energy around 40 MHz and block energy the antenna picks up outside this region. If the remote controller transmitter is sending commands then its radio signal will be picked up by the receiver and converted back into the original pulse sequence. Decoding circuitry then determines which commands were sent by the measuring the number of received pulses in the sequence. Signals are then sent to the motors to execute the commands. [7]

2.2 Literature Review

A few reference materials were referred to provide fundamental knowledge on the application theory. The reference materials used are listed below;-

- Sample tutorial from Visual Helper
- Ultrasonic sensor project material
- Electronics components datasheets
- Transmitter and receiver references
- Parallel port datasheet
- Visual Basic Graphic Programming book
- IR sensor datasheet.

CHAPTER 3
METHODOLOGY

This chapter explained on the method used to make sure the project completed based on the specifications and procedures planned.

3.1 Procedure identification

1. Designed on the VB programming code
   - Program the parallel port in the VB code. Coding declaration is essential in initialize each port being used.
   - The pre-determine route is designed using the select case method and timer function and being sent the command out to the RC car through the output pins.
   - The obstacle avoidance path and distance measurement is designed by using the same method with additional input pins declarations.
2. Combination setup for the RC car transmitter and receiver with the additional transmitter and receiver for sensor detection.
   - The transmitter of the RC car (40MHz) is modified to connect with the parallel port.
   - The transmitter for the sensors (35MHz) is modified to connect both ultrasonic and IR sensors.
   - The receiver for the sensors (35MHz) is modified to connect with the parallel port.
   - Output signal from the sensor is being test out with the transmitter and receiver by using oscilloscope.

3. Integrate both VB programming code with the both transmitters and receivers.
    - There are two types of configuration which are one for the PC connection and the others for the RC car connection.
    - The PC connection consists of parallel port, transmitter of the RC car and receiver of the sensors detection.
    - The RC car connection consists of receiver of the RC car, transmitter of the ultrasonic sensor and IR sensor, with both sensors attach at the RC car.
    - VB programming code then being integrated with the RC car through the parallel port and both transmitters and receivers as the medium for transmission the voltage signal by using RF frequencies.

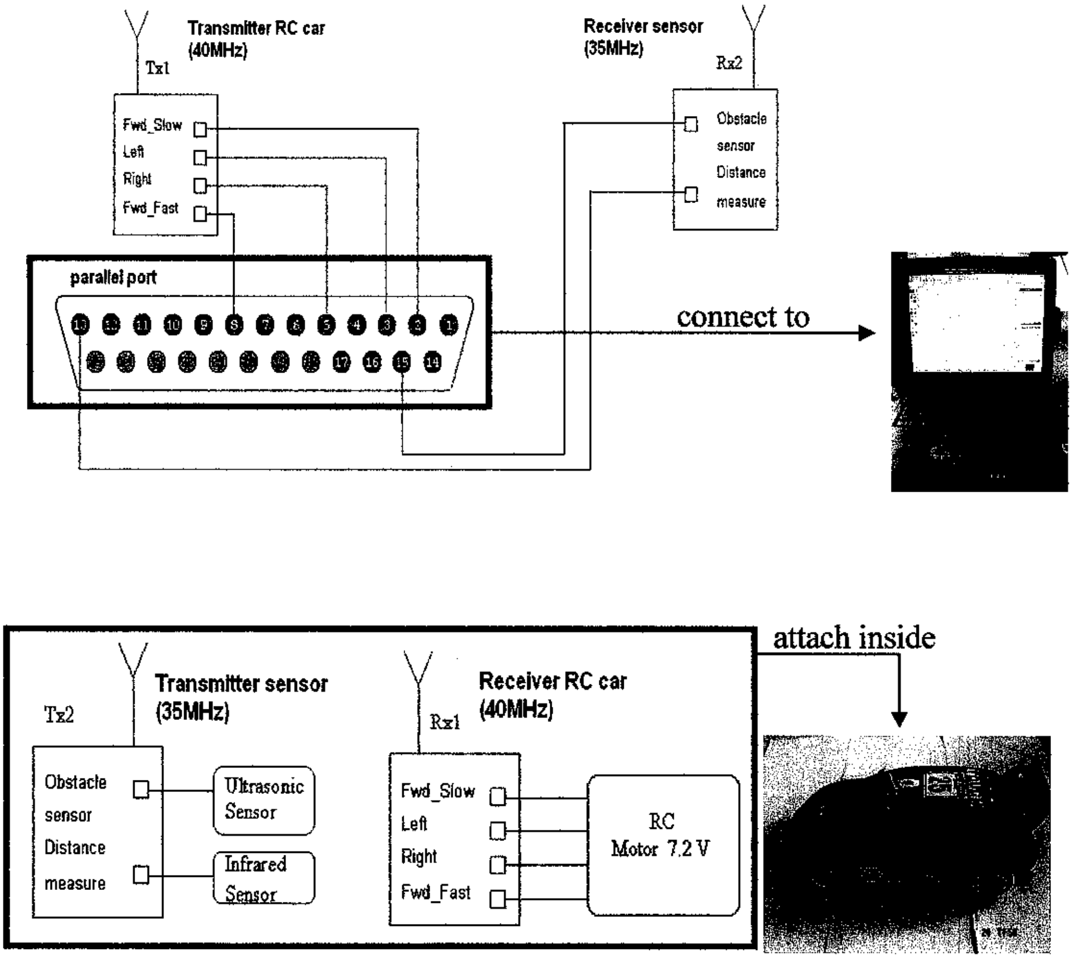


Figure 8 Anatomy of Smart RC car

3.2 Design Diagram

Design method is planned before constructing the project's prototype. Pre-determine route and obstacle avoidance path are created using visual basic programming and the programming command is then being sent to the RC car via output pins and input pins of the parallel port.

3.2.1 Design diagram of the pre-determined route

Fuzzy logic is implemented to assign the speed control within a certain distance of the design pre-determined route. There are two stages of speed control assigned which is slow and fast speed during the movement of the RC through the pre-determined route. The total distance taken by the RC car to move according to the program route depends on a few factors such as condition of the floor surface (whether it is rough or smooth), interval time setup and the type of material use for the tire. Moreover, current weather (dry or wet) also affects the movement and pace of the car. A dry floor surface is setup to have a smaller interval time while a larger interval time is required at a clammy surface.

After taking into account the time and friction of the floor surface, the RC car is programmed to slow down when it moves near to a turning and during the turning while the fast speed is assigned during start up and each time after the car takes a turn. Due to the designed pre-determined route, the transmitter circuit is modified to let the transmitter receives signal of the movement command. Parallel port is use as the medium to interface the transmitter and the programming code. It will get commands from the visual basic and then send voltage signal to transmitter.

3.2.1.1 Fuzzy Cause Effect

As referred to the design diagram, four sections have been divided for the square shape. Each section will have the same length. The same input and output occurred at each section. For the project, three conditions assigned for input based on distance taken by the RC car as stated below:

- **Start Line**
- **Almost Near**
- **Corner**

Each output will sent signal to transmitter at a certain time to change speed according to the input condition. There are three condition assigned for output as stated below:

- **Speed Up (start to accelerate)**
- **Slow Down**
- **Right Slow**

"Linguistic" rules are assigned based on input and output condition. These Rules will appear in the visual basic computer program as "If-Then" statements:

Rule 1: If the distance of the car is on the **start line** then **speed up**.

Rule 2: If the distance of the car is **almost near** to a turning, then **slow down**.

Rule 3: If the distance of the car is at a **corner**, then turn **right** with a **slow** speed.

Based on the input and output conditions as causes and effect with a "linguistic" rules, charting technique is drawn with three triangles each for input and output. The figure of cause effect triangles is shown below:

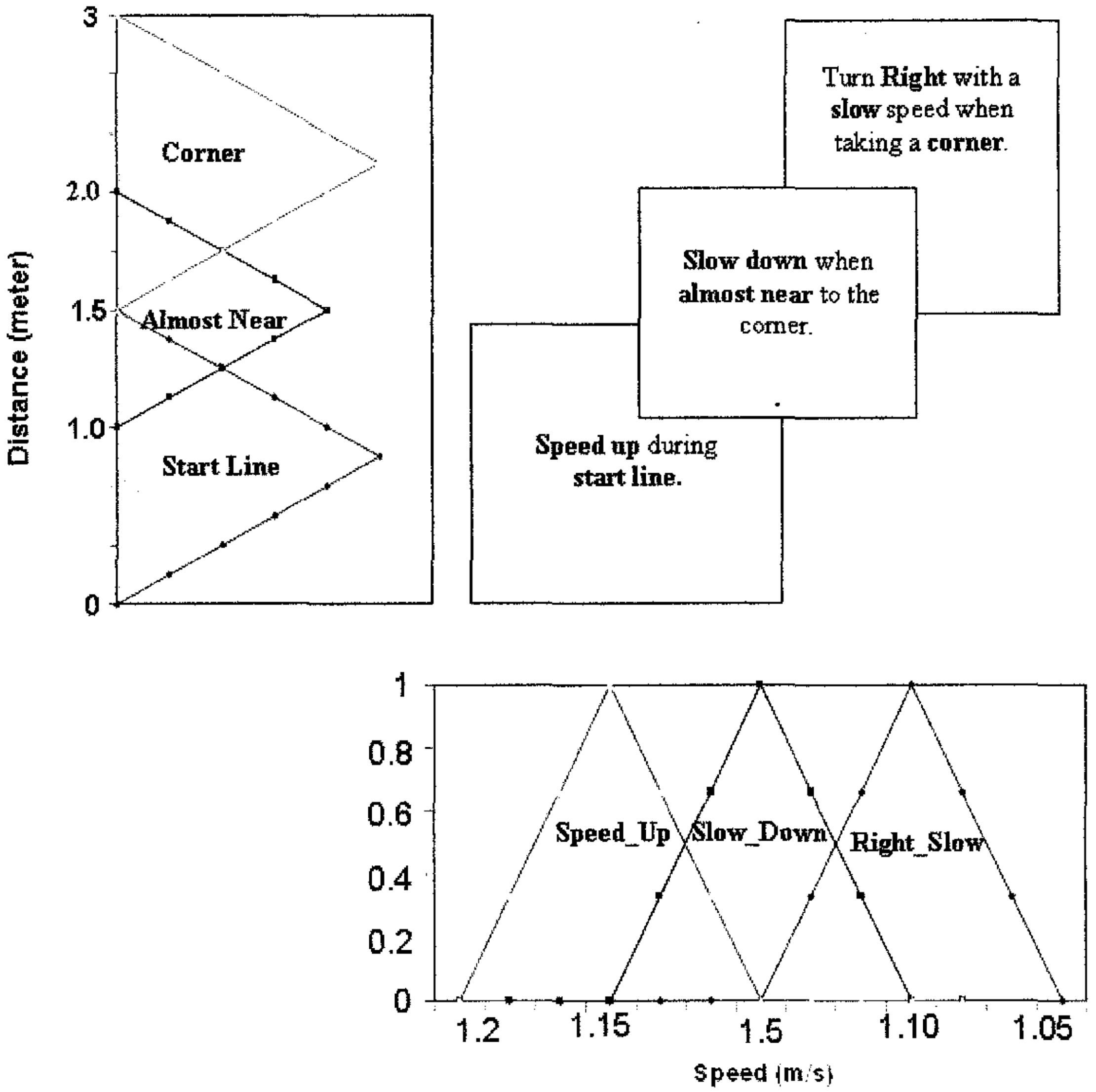


Figure 9 Designed diagram of "Cause effect" triangles for predetermined route

3.2.1.2 Final output from the design diagram of the pre-determine route

From the fuzzy cause effect triangle, three outputs obtained will be set into the VB command. VB command then will send the command to the parallel port as the output to the RC car. The RC car will then be moving accordingly to the VB command.

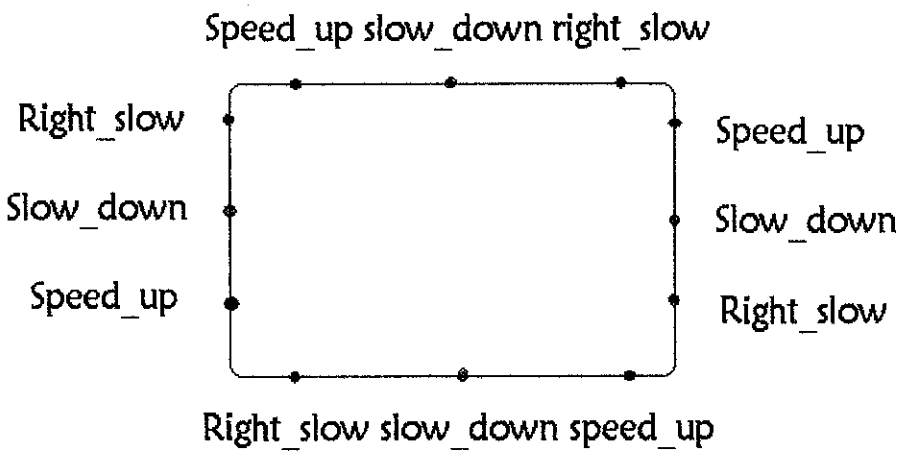


Figure 10 Designed diagram of pre-determined route.

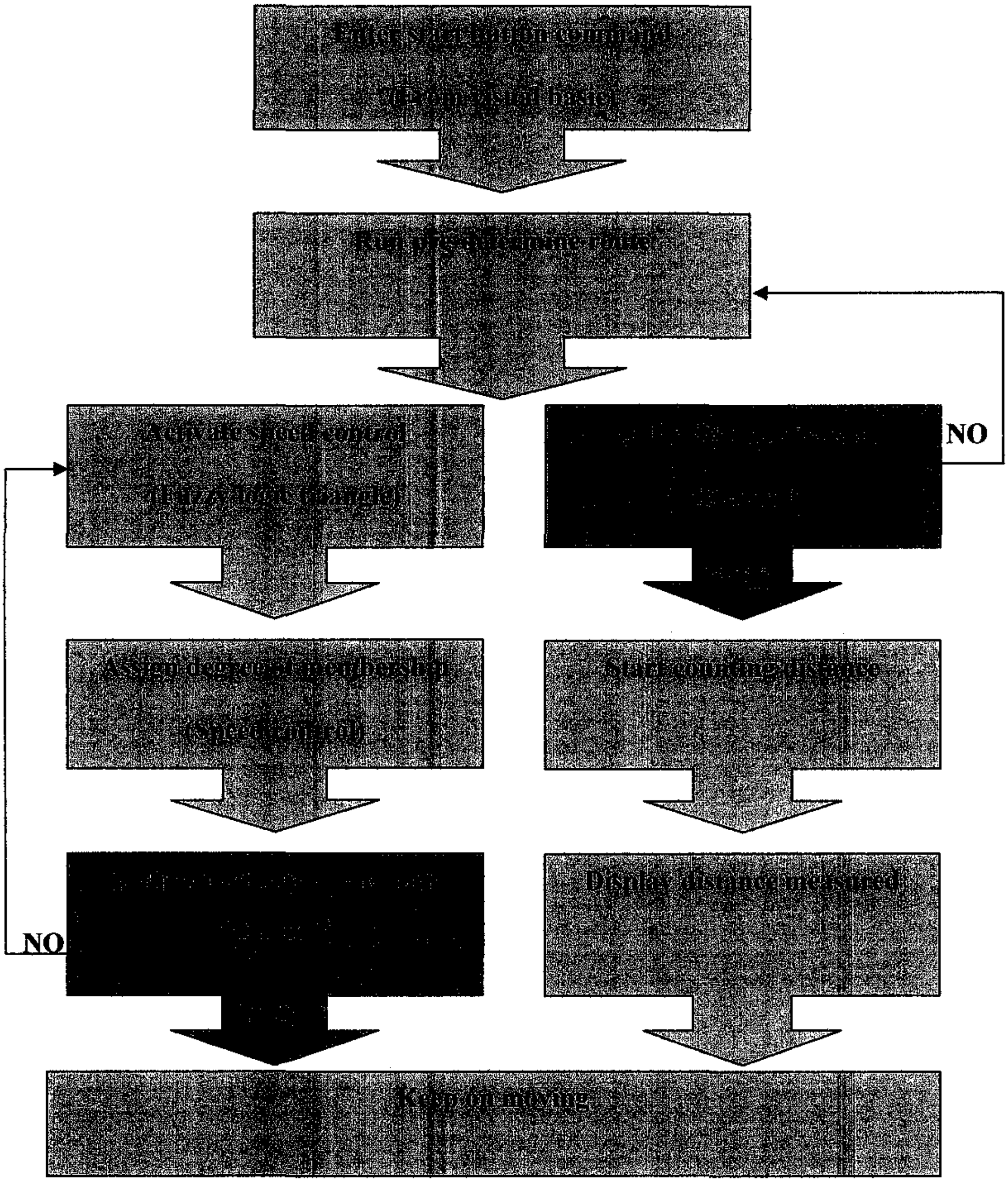


Figure 11 Flow diagram on the pre-determined route program and distance measurement

3.2.2 Design diagram for the obstacle avoidance path

Initially, the program will run on a pre-determined route by the command sent from the VB through the output pins of the parallel port. The obstacle avoidance path is activated when the input pin of the parallel port receive signal from the ultrasonic sensor whenever an obstacle is detected. This is done by the output pins which respond by sending back the signal to the RC car to move on the obstacle avoidance path. The programmed then runs back on the pre-determined route after avoiding the obstacle.

3.2.2.1 Final output from the design diagram of the obstacle avoidance path

The ultrasonic sensor is fitted in front of the RC car. The sensor is connected to the transmitter to send signal to the parallel port. Input pins of the parallel port is used to send signal for the VB to activate the obstacle avoidance path.

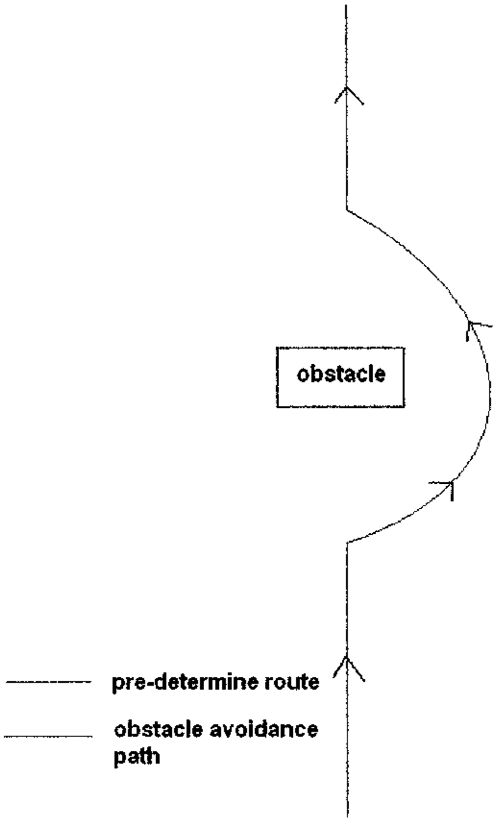


Figure 12 Designed diagram of obstacle avoidance path.

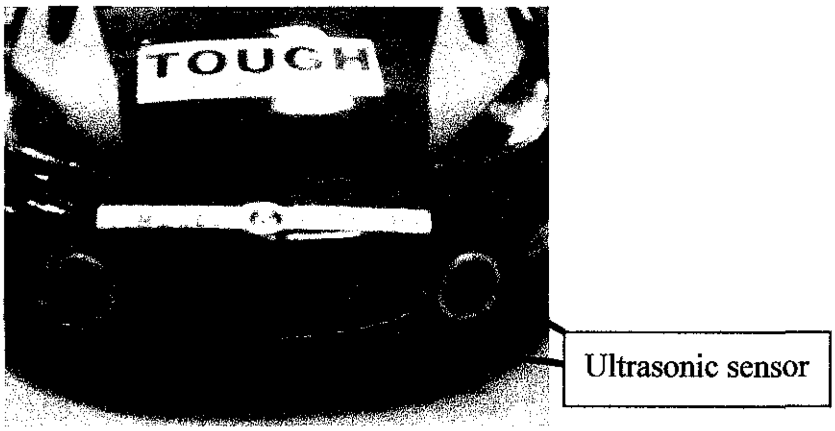


Figure 13 A front view from the modified RC car.

3.2.3 Design diagram of the distance measurement.

The IR distance sensor will activate when the sensor detect a white surface at one of the wheel. Each time the RC car wheel rotates, whenever it detects a white surface, the IR signal will then being reflected and the signal then being encoded into radio energy and transmitted by transmitter to the receiver on the 35 MHz which is then decoded back to the voltage signal to trigger the input pin of the parallel port.

3.2.3.1 Final output from the design diagram of the distance measurement

The distance of the moving RC car is defined by placing a white tape at one of the wheel. Each time the IR sensor sends the reflected of white color, VB will count on the number. The final value of distance is obtained after the wheel stop moving at the finish line.

Total distance measured = n x d

where n = number of times IR sensor detect white tape

d = diameter of the wheel, 18.5 cm

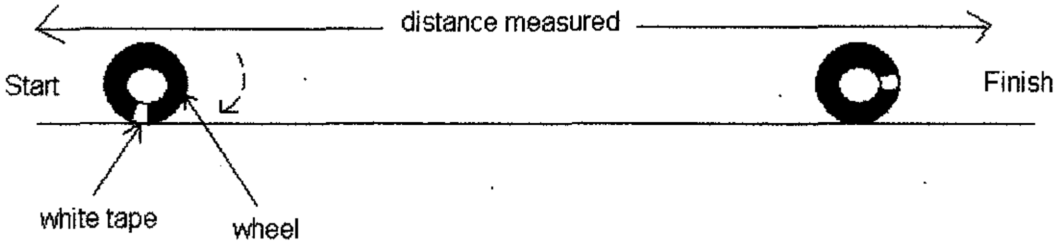


Figure 14 Designed diagram of distance measurement.

Figure 15 A back view from the RC car tire.

3.2.4 Design diagram for the parallel port.

There are two ports used in the parallel port to integrate the PC with VB. One port is used to send command from the PC to the RC car (data port) and the other is used to send a signal from the RC car to the PC (status port). From figure 16, the status port and data port in yellow box shows the input pins and output pins used in the project;

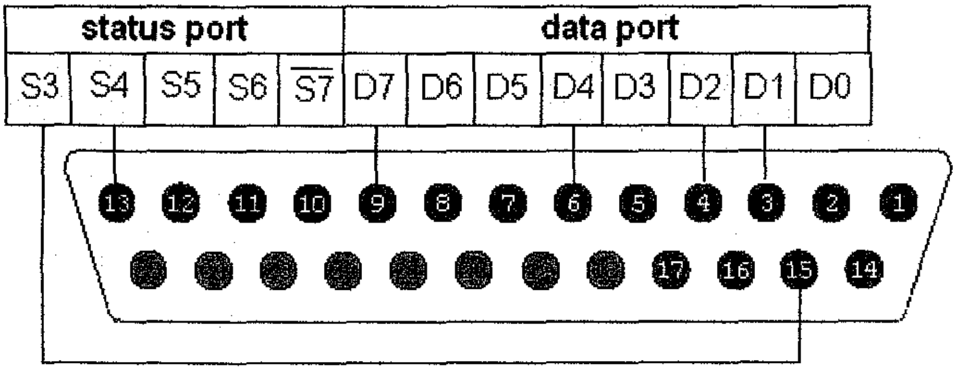


Figure 16 Parallel port, 25-way Female D-Type connector

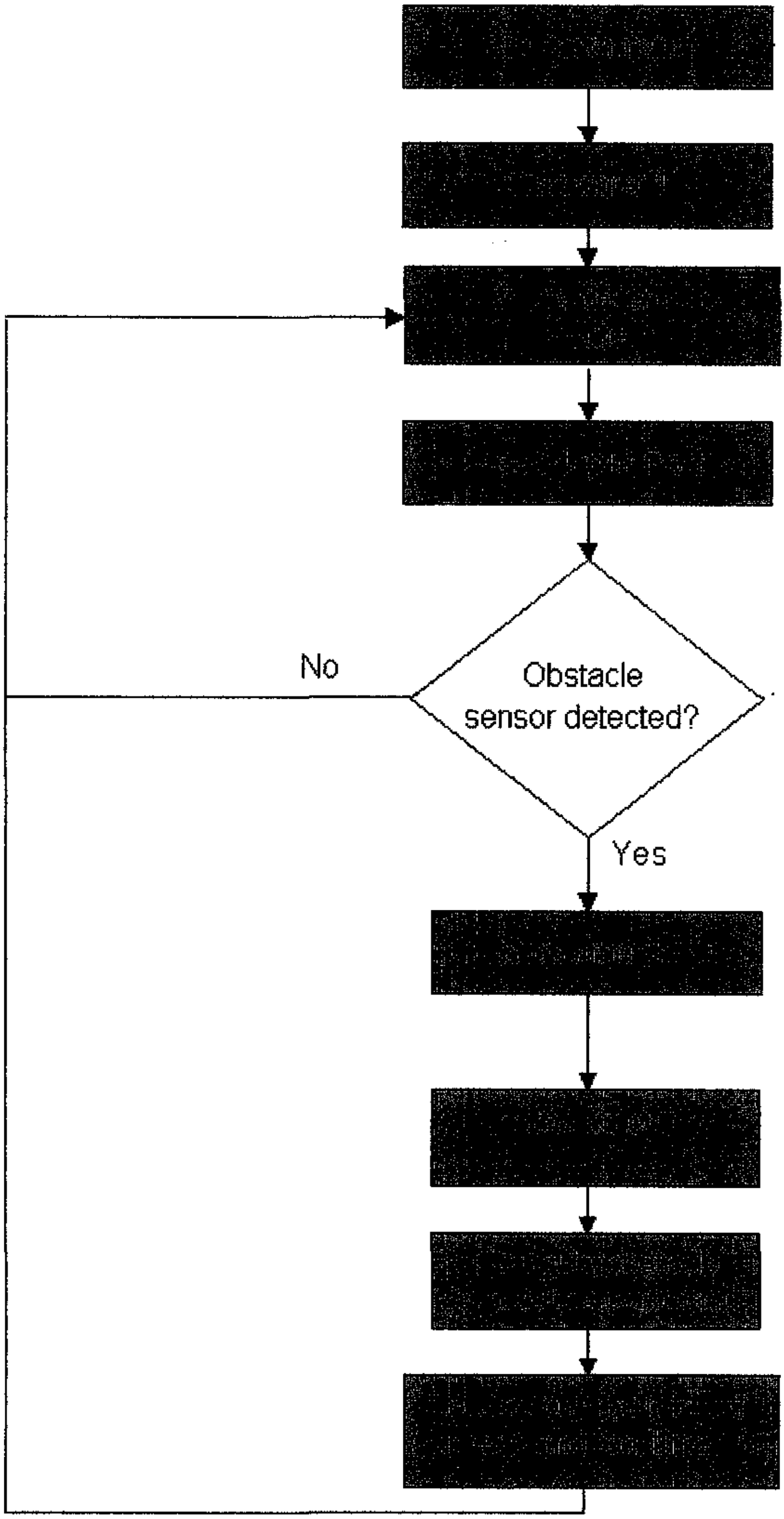


Figure 17 Flow Diagram on the obstacle avoidance path.

3.3 Tools and equipment

The project consists of both software and hardware equipment. Below are the listed equipment used in the project;

3.3.1 Software equipment

The first part of the project related to design the programming code. The programming code must be able to send and receive any command from the hardware part by using parallel port. Therefore, the chosen programming software is;

- Visual Basic 6.0 programming

3.3.2 Hardware equipment

The second part of the project is basically hardware building. So, most of the activity is about soldering and connection-based process. The hardware then integrated with the programming software. Below are the listed equipment used;

- Personal Computer
- RC car
- Parallel port receptacle
- Ribbon cable for 25 pins
- Oscilloscope
- Multimeter
- Soldering station
- Power supply
- Electronic components
- Ultrasonic sensor
- Infrared (IR) sensor
- Small hand tools
- Additional transmitter and receiver set

CHAPTER 4
RESULTS AND DISCUSSIONS

4.1 Results

4.1.1 Parallel port configurations in visual basic programming code

To initialize the parallel port in the VB programming, the address ports need to be defined first. There are two address ports used in this project which are data port address and status port address. Each port identified by its own number of port. The number can be expressed in hex, binary or decimal. The project used decimal value to identify the ports. [3]

Port	Address (in decimal)
Data port	888
Status port	889

Table 4 Address port used in VB programmed code

4.1.1.1 Parallel port configurations for pre-determined route and distance measurement.

For the designed diagram of the pre-determine route, data port is used to send signal to the receiver of the RC car. Three outputs pins D1, D2 AND D4 are set for the program. The connectors of RC transmitter levers are ACTIVE LOW. So the output voltage of the parallel port must be 0 volt. Therefore, output pins are set to 0 as active low pins. For the distance measurement, status port used to receive signal from the IR sensor. The input pin used to detect the signal is S4. The pin which is ACTIVE HIGH will trigger 0 volts each time it receiver signal from the sensor. Then, VB will receive the command and start counting the distance.

4.1.1.2 Parallel port configurations for obstacle detection path

For the designed diagram of the obstacle detection path, both input pins and output pins is initialized too. The input pin used to send signal from the ultrasonic sensor is pin S3. Whenever VB detect signal from pin S3, the obstacle avoidance path will respond by sending command to the output pins to avoid the obstacle.

Pins / Directions	Binary values								Decimal values
	D0	D1	D2	D3	D4	D5	D6	D7	
Fwd_slow	1	■	1	1	1	1	1	1	253
Fwd_fast	1	1	1	1	1	1	1	■	127
Right_slow	1	■	1	1	■	1	1	1	237
Left_slow	1	■	■	1	1	1	1	1	249
Reset	1	1	1	1	1	1	1	1	255

Table 5 Output pins (data port) used for pre-determine route and obstacle avoidance path

Pins / Input sensor	Binary values								Decimal values
	S0	S1	S2	S3	S4	S5	S6	S7	
Obstacle detection	0	0	0	■	1	1	1	0	112
Distance measurement	0	0	0	1	■	1	1	0	104

Table 6 Input pins (status port) used both for obstacle detection and distance measurement.

4.1.2 Final design on the Visual basic programming code

4.1.2.1 VB design object for the pre-determined route and distance measurement.

The designed diagram starts from the Form window of the VB interface. Form window is the place to put objects used in the project programmed code that is showed in the center of the screen. In this form, a picture box of is place at the start line. The picture will move to show the moving car through a pre-determined route. Two labels is placed at one corner of the form. One label is used to show the changing speed according to the caption of the output condition during the movement of the RC car throughout the route. The other label is used to count the distance measurement of the RC car.

Two timers are added up in the form to display a clock that performs events at specified time intervals. To set up the time, the value of the timer control in the timer function need to be changed. The value is the number of milliseconds between calls to a timer control's timer event. An interval time of 625 milliseconds is chosen for the movement of the car from one counter to another counter. An interval time of 1 millisecond is chosen for the distance measurement. Command button is used to start the program when it is clicked.

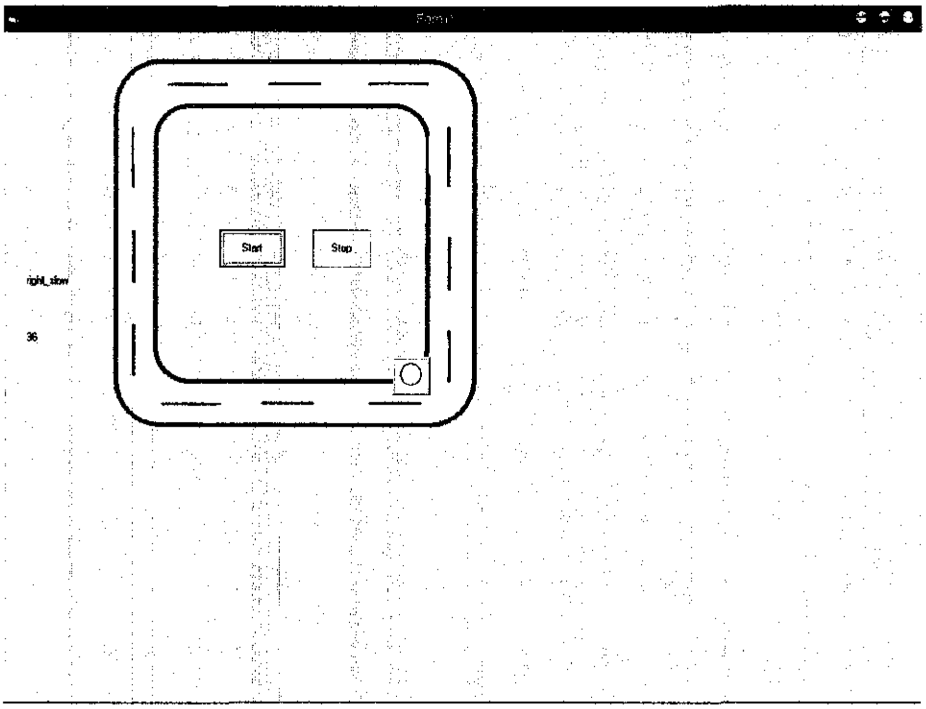


Figure 18 Visual Basic Interface Form for pre-determined route and distance measurement.

4.1.2.2 VB design code for pre-determine route and distance measurement.

Code window is shown when double clicking the visual form window. Below is the command for the pre-determine route. 12 counters are assigned to control the movements of the car to complete one loop of the program route. The angle of displacement for cornering angle of the car is chosen to be 45°. From the pre-determine route, active outputs are assigned for forward slow which is the output for slow down, forward fast which is the output for speed up and slow right which is the output for right slow. Captions are also added up to show label for the speed change during the movement of the RC car throughout the route. The total time taken for complete one loop of the pre-determine route is approximately 8 seconds. Below is the explanation on the programmed code used in the program;-

- Coding declaration is made before using any of the other functions contained within .DLL file. These declarations are to be placed in any module in the program under *General_Declarations* section. Once the declaration has been made to the functions, two new commands are available. These are *Inp* and *Out*. *Out* is a statement and is used to send a bit to a port.

```
Private Declare Function Inp Lib "inpout32.dll" _
Alias "Inp32" (ByVal PortAddress As Integer) As Integer
Private Declare Sub Out Lib "inpout32.dll" _
Alias "Out32" (ByVal PortAddress As Integer, ByVal Value As Integer)
Private Declare Sub Sleep Lib "Kernel32" (ByVal dwMilliseconds As Long)
```

- A Command Box is used to start the program by enabling function of a timer to be activated 'ON' and starts to count a clock.

```
Private Sub start_Click()
tmrTimer.Enabled = True
Label1.Caption = "0"
End Sub
```

- A Command Box is also used to stop the program manually by disabling the timer function of the program.

```
Private Sub stop_Click()
tmrTimer.Enabled = False
Dim stopcar As Boolean
stopcar = True
End Sub
```

- A Timer is used to count a clock that performs events at specified time intervals and to enable the changing of sub function of control speed.

```
Private Sub tmrTimer_Timer()

Picture1.AutoRedraw = True
Picture1.ScaleMode = vbPixels
Picture1.Cls

checkspeed

End Sub
```

- A sub function of speed control is the looping operation to move the Picture Box around the pre-determined route and send a decimal bit to a port through OUT command. All input and output variables is defined in the sub function. The project uses decimal value as the outputs. For example, to reset the port, the 8 data lines are set to 11111111, where decimal value of 255 is sent.

```
Private Sub checkspeed()

Dim port1 As Integer
Dim stopcar As Boolean
Dim badgeCount As Integer
Dim y As Integer
Dim x As Integer
Dim detect As Integer

y = Val(Picture1.Top)
x = Val(Picture1.Left)

stopcar = False

port1 = 888
Out port1, 255
```

- A Select Case function is used to select a command for input condition of the distance. This resembles the fuzzy logic condition. There are two cases that have to be selected that are the case for moving in vertical coordinate (y-axis) and the other one for moving in horizontal coordinate (x-axis). "If-then" statement is used to choose the appropriate distance and proceed with the "do-while" function and end with sending a decimal bit to the port. The bits will determine the direction and speed of the RC car.

```
Select Case y

Case Is = "4560"
badgeCount = 10000
If Picture1.Top = 4560 And Picture1.Left = 1800 Then
lblslow_down.Caption = "speed_up"
Do While badgeCount > 0 And Picture1.Top > 2880
DoEvents
  If stopcar Then
tmrTimer.Enabled = False
Exit Sub
tmrTimer.Enabled = True
End If
   badgeCount = badgeCount - 1
   Picture1.Top = Picture1.Top - 1
   Loop
   port1 = 888
    Out port1, 127
```

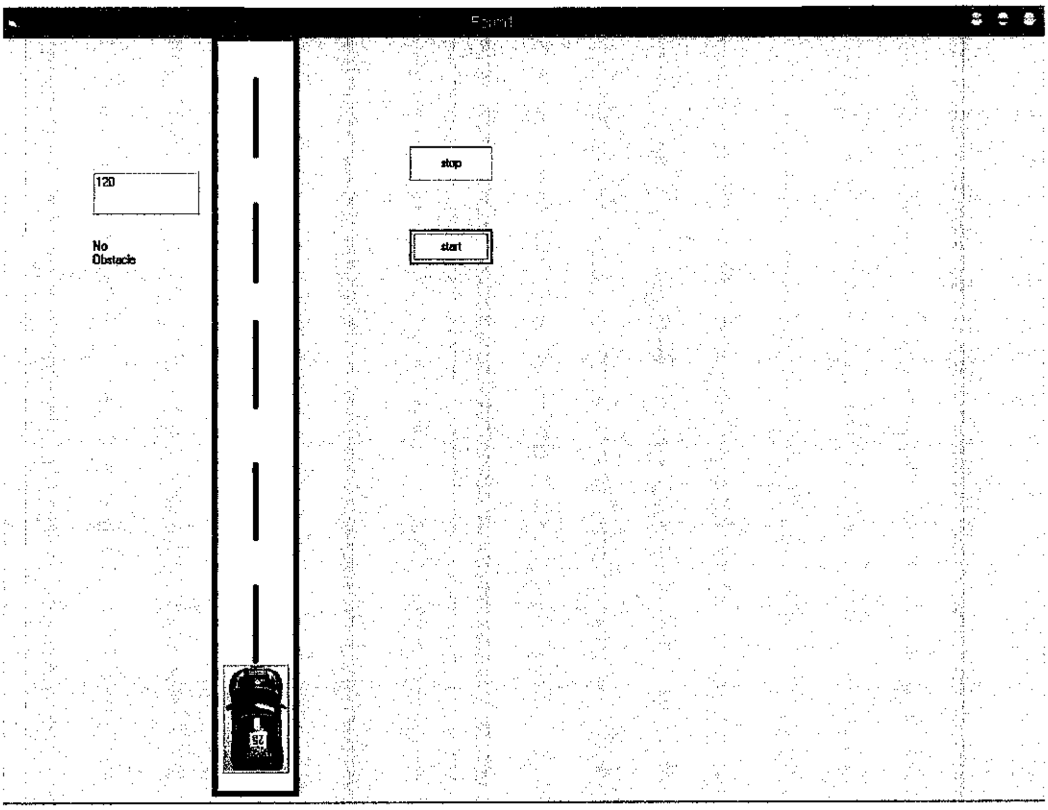
- One timer is used to detect the status port S4 which sense the IR sensor. This timer will always run to count the distance unless the program ends and the RC stop moving.

```
Private Sub Timer1_Timer()
Dim port2 As Integer
Dim detect As Integer

port2 = 889
detect = Inp(port2)

If (detect = 104) Then
Label1.Caption = Label1.Caption + 1
End If
End Sub
```

4.1.2.3 VB designed object for obstacle avoidance path.

The form window is designed to a straight line path for the original path (normal condition). The designed consists of a picture box located at the bottom of the form. The picture will move on a straight line when the start button is entered. One text box is used to show the number of input port that is detected. One label is used to show the current condition of the road whether there is an obstacle in front or not. Three timers are used to count the original path, obstacle avoidance path and distance measured along the way.

To set up the time, the value of the timer control in the timer function need to be changed. The value is the number of milliseconds between calls to a timer control's timer event. An interval time of 1000 milliseconds is chosen for the movement of the car from one counter to another counter. Command button is used to start the program when it is clicked.



120

No Obstacle

Figure 19 Visual Basic Interface Form for obstacle avoidance path.

4.1.2.4 VB designed code for obstacle avoidance path.

For the original path, one counter is assigned for a straight line path. 3 counters are assigned for the obstacle avoidance path to let the car move out of the original path. Below is the explanation on the programmed code used in the program:

- A Command Box is used to start the program by enabling timers to be activated 'ON' and starts to count a clock. A label is located at the start command to count the distance measured when the car starts to run.

```
Private Sub start_Click()
Tmr1.Enabled = True
Tmr2.Enabled = False
Tmr3.Enabled = True

Dim distance As Integer
Dim port2 As Integer
Dim port1 As Integer

port2 = 889
port1 = 888

Label1.Caption = "0"

End Sub
```

- One timer function is being used to run the original path which is the straight line path. A default value of status port is declared in the original path. If the status port detects others port, this timer will be 'OFF'.

```
Private Sub tmr1_Timer()
Out port1, 0
port2 = 889
detect = Inp(port2)

If (detect = 120) Then
label2.Caption = "No Obstacle"
    checkspeed
Else
    detect_obstacle
    Text1.Text = detect
End If
End Sub
```

- One timer is used for the obstacle avoidance path. If status port detect S3 (port used for obstacle detection), the timer will be 'ON' and the RC car will runs on an obstacle avoidance path.

```
Private Sub tmr2_Timer()
port2 = 889
detect = Inp(port2)

If (detect = 112) Then
label2.Caption = "Detect Obstacle"
Tmr1.Enabled = False
    detect_obstacle
Else
    checkspeed
End If

End Sub
```

- A sub function is used to move the RC car on the original path which is the straight line path. The function also essential to ensure the status port always in a default condition. All input and output variables is defined in the sub function. The project uses decimal value as the outputs.

```
Private Sub checkspeed()
Out port1, 255

port2 = 889
detect = Inp(port2)

Text1.Text = detect
Dim Count As Integer

Count = 100
If (detect = 120) Then
Do While Count > 0 And Picture1.Top > 240
    Count = Count - 1
    Picture1.Top = Picture1.Top - 1
    Loop
    Out port1, 253
End If

End Sub
```

- This sub function is used to move the RC car on the obstacle avoidance path. If any obstacle is detected, this function will runs to give command to the output port to let the RC avoiding the obstacle.

```
Private Sub detect_obstacle()

 Select Case Count
 Case Is = 1000
 Do While Count > 0
     Count = Count - 0.7
     Picture1.Top = Picture1.Top - 1
     Picture1.Left = Picture1.Left + 1
     Loop
 Out port1, 237
 End Select
```

4.1.3 Testing software configurations with the hardware connections

4.1.3.1 Experiment on the VB code for the pre-determine route

The program code on the pre-determine route was tested on a basketball court. Based on the experiment, an interval time of 625 milliseconds is selected for the movement of the car from one position to another position. This interval time is set based on the friction of the floor surface. From the observation, a rough surface will need a larger interval times. Therefore the interval time have to be changed according to the ground surface condition. The material of the tire is also crucial in determining the interval time. Inaccurate estimation of the tires' 'slickness' will cause the movement of the car to be distracted. As a result, the wheel will tend to spin at each starting point due to the neglected condition. Therefore, a shorter distance is observed during the start up part, causing the path shape to change. Total distance taken during the final testing is approximately 11.6 meters.

4.1.3.2 Experiment on the speed controller circuit

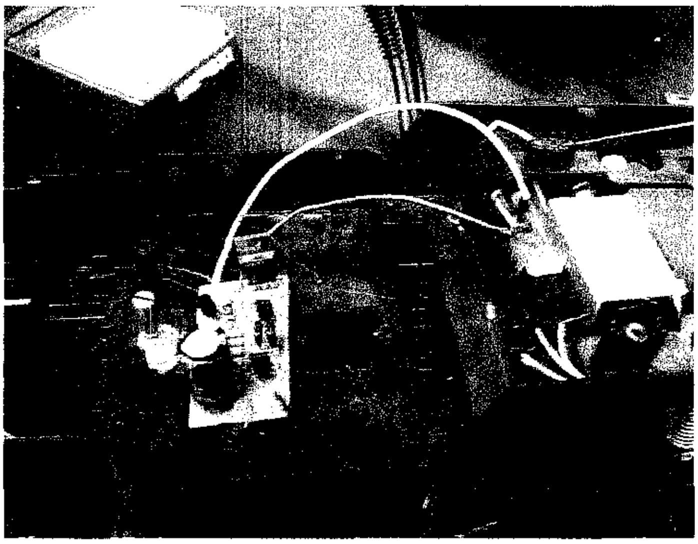


Figure 20 The speed controller kit connected to the motor of the RC car.

Figure 20 shows the speed controller kit. The output of the speed controller was connected to the motor while the receiver of the RC car is used to supply voltage for the speed controller. The speed of the motor was then being able to be adjusted by the VR (potentiometer).

4.1.3.3 Experiment on the Ultrasonic sensor circuit

There are few testing run to make sure all the hardware connections are working in correct order. Firstly, after soldering the ultrasonic sensor kit, testing was done by connecting it to the transmitter of the RC car. A communication test was carried out to make certain the sensor was able to trigger and sent to the receiver of the RC car. This is done as the receiver sensor (35MHz) is now being used to transfer data to the input port of the parallel port which is then sent to the computer. Subsequently, to check the measured distance of the RC car to the obstacle face, oscilloscope was connected to the output of the parallel port to observe the Voltage vs Time graph. If the obstacle moves closer to the RC car, the timing (second) will be longer. Therefore, approximately 43 cm is the safest distance chose by the author to detect the obstacle. The distance allowed for the RC car to turn round the corner.

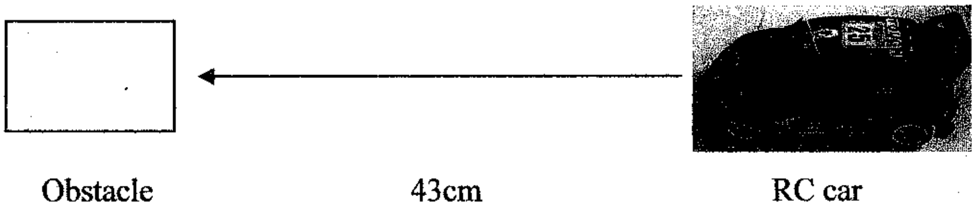


Figure 21 Diagram for obstacle detection of the RC car.

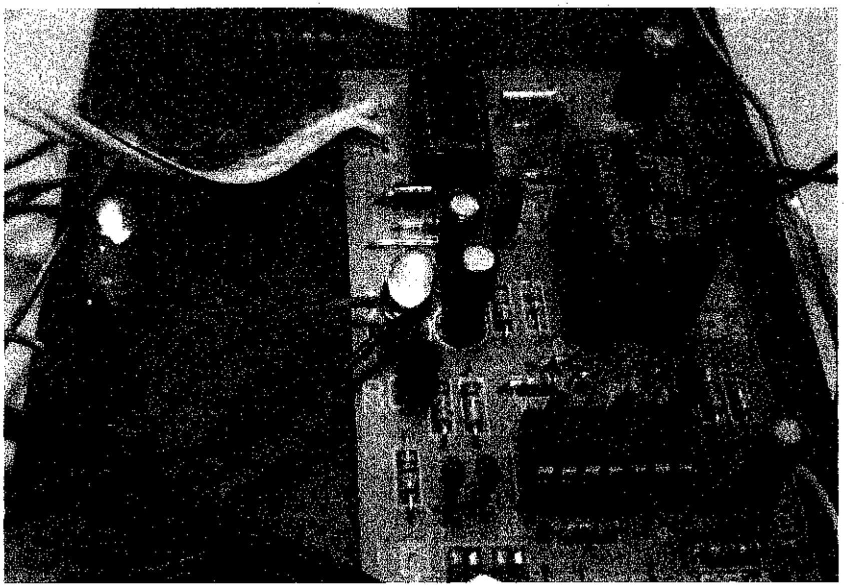


Figure 22 Ultrasonic sensor circuit.

Figure 22 shows the ultrasonic sensor circuit. The circuit is connected to transmitter sensor (35MHz) circuit. Two LEDs lights 'ON' to show both connections were satisfactory.

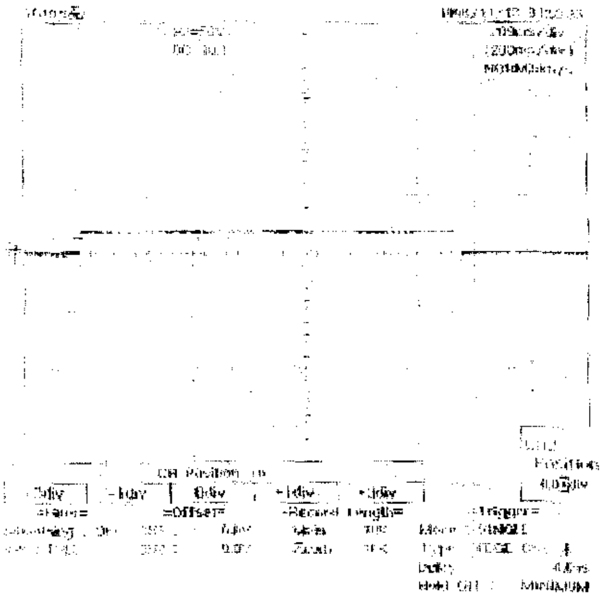
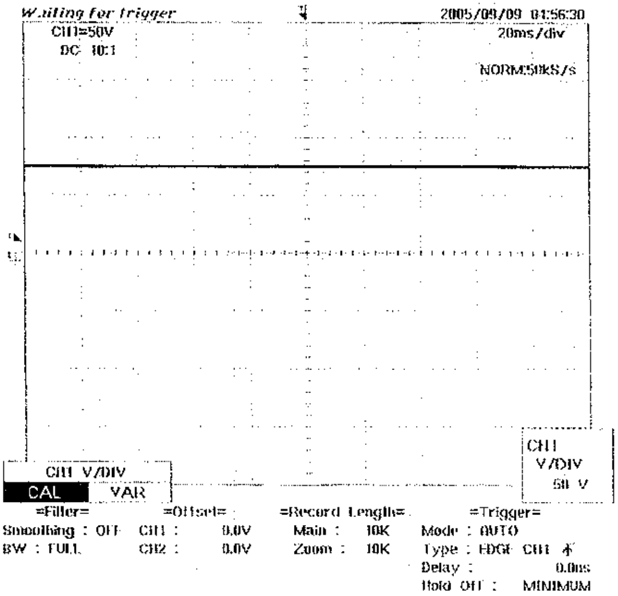


Figure 23 Graph of Voltage (V) vs Time (ms) for output voltage.

Figure 23 shows a reading of impulse period of the output voltage for an obstacle for a distance of 43cm from the RC car. The time varies according to the distance of the obstacle from the RC car.

4.1.3.4 Experiment on the Infrared (IR) distance measurement circuit

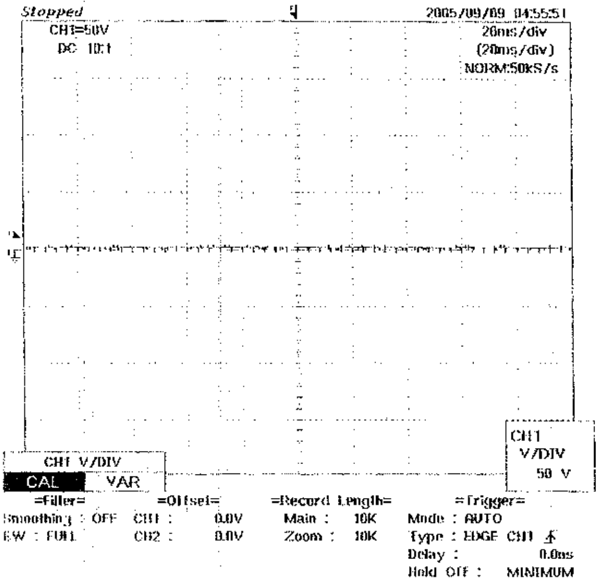
After successfully tested on the Ultrasonic obstacle sensor, the Infrared (IR) sensor was then being observed. The Infrared (IR) sensor was tested by using the oscilloscope and the Visual Basic programming. Oscilloscope was connected to the output of the Infrared (IR) sensor to observe the Voltage vs Time graph. The output voltage from the IR distance sensor circuit read 7.5V when the sensor detect black surface of the wheel. The output voltage change to read 0V each time the IR sensor detect a different color on the surface.



```
Waiting for trigger                    2005/09/09 04:56:30
  CH1=50V                                      20ms/div
   DC  10:1
                                          NORM:50kS/s

                                                   CH1
                                                    V/DIV
     CH1 V/DIV                                      50 V
 CAL        VAR
    =Filter=          =Offset=     =Record Length=        =Trigger=
Smoothing : OFF   CH1 :     0.0V    Main :   10K    Mode : AUTO
BW : FULL         CH2 :     0.0V    Zoom :   10K    Type : EDGE CH1
                                                    Delay :          0.0ns
                                                    Hold OFF :    MINIMUM
```

Figure 24 Graph of Voltage (V) vs Time (ms) for output voltage

Figure 24 shows the output voltage reads 7.5V during normal condition of the IR sensor (when the IR sensor did not detect any changes of color).



```
Stopped                                2005/09/09 04:55:51
  CH1=50V                                      20ms/div
   DC  10:1                                   (20ms/div)
                                          NORM:50kS/s

                                                   CH1
                                                    V/DIV
     CH1 V/DIV                                      50 V
 CAL        VAR
    =Filter=          =Offset=     =Record Length=        =Trigger=
Smoothing : OFF   CH1 :     0.0V    Main :   10K    Mode : AUTO
BW : FULL         CH2 :     0.0V    Zoom :   10K    Type : EDGE CH1
                                                    Delay :          0.0ms
                                                    Hold OFF :    MINIMUM
```

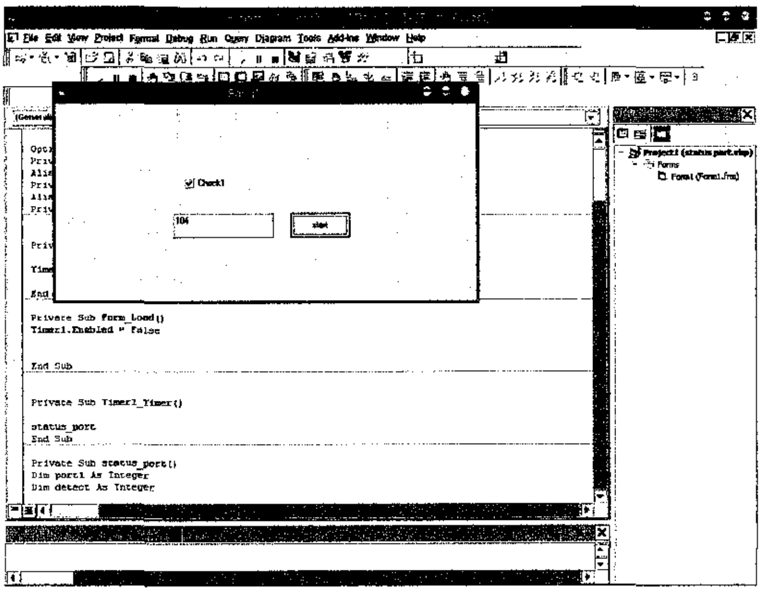
Figure 25 Graph of Voltage (V) vs Time (ms) for output voltage

Figure 25 shows the output voltage reads 0V during interrupted condition of the IR sensor (when the IR sensor sense white tape).

.

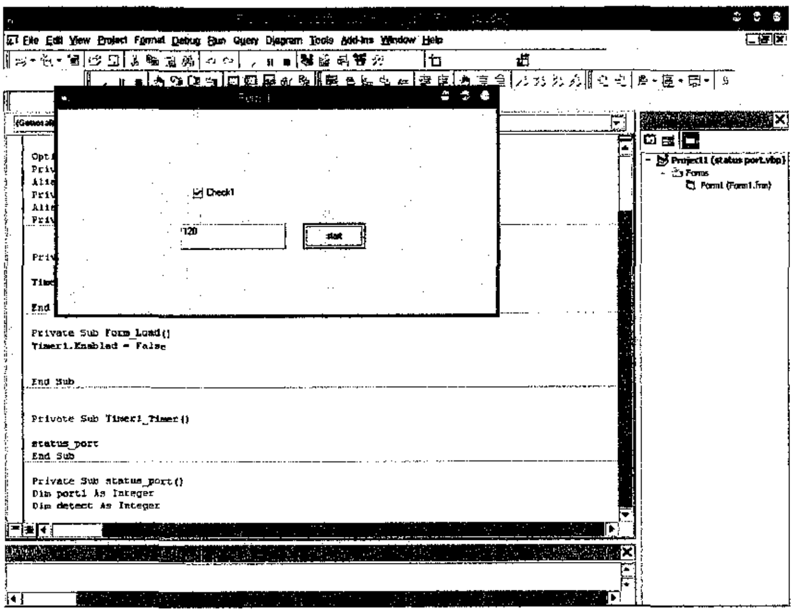
4.1.3.5 Experiment on the parallel port for the VB code

From the binary values, the S7 is ensured to be an inverted pin and S5 is default pin. Pin S4 is choose to be the input pin for the obstacle sensor. While, S3 is choose as the input pin for the distance sensor.


File Edit View Project Format Debug Run Query Diagram Tools Add-Ins Window Help
(General)
☑ Check1
104
start
Project1 (status port.vbp)
Forms
Form1 (Form1.frm)

```
Opt
Priv
Alia
Priv
Alia
Priv

Priv

Time

End

Private Sub Form_Load()
Timer1.Enabled = False


End Sub


Private Sub Timer1_Timer()

status_port
End Sub

Private Sub status_port()
Dim port1 As Integer
Dim detect As Integer
```

Figure 26 Status port, S4 ON during normal condition – 104 in decimal values


File Edit View Project Format Debug Run Query Diagram Tools Add-Ins Window Help
(General)
Form1
☑ Check1
120
start
Project1 (status port.vbp)
Forms
Form1 (Form1.frm)

```
Opt
Priv
Alie
Priv
Alie
Priv

Priv

Time

End

Private Sub Form_Load()
Timer1.Enabled = False


End Sub


Private Sub Timer1_Timer()

status_port
End Sub

Private Sub status_port()
Dim port1 As Integer
Dim detect As Integer
```

Figure 27 Status port, S4 OFF during obstacle detection – 120 in decimal values

As refer to the methodology, the final output of the project is implemented by integrating the PC with the RC car via parallel port. Figure 28 shows the configurations of the project prototype;

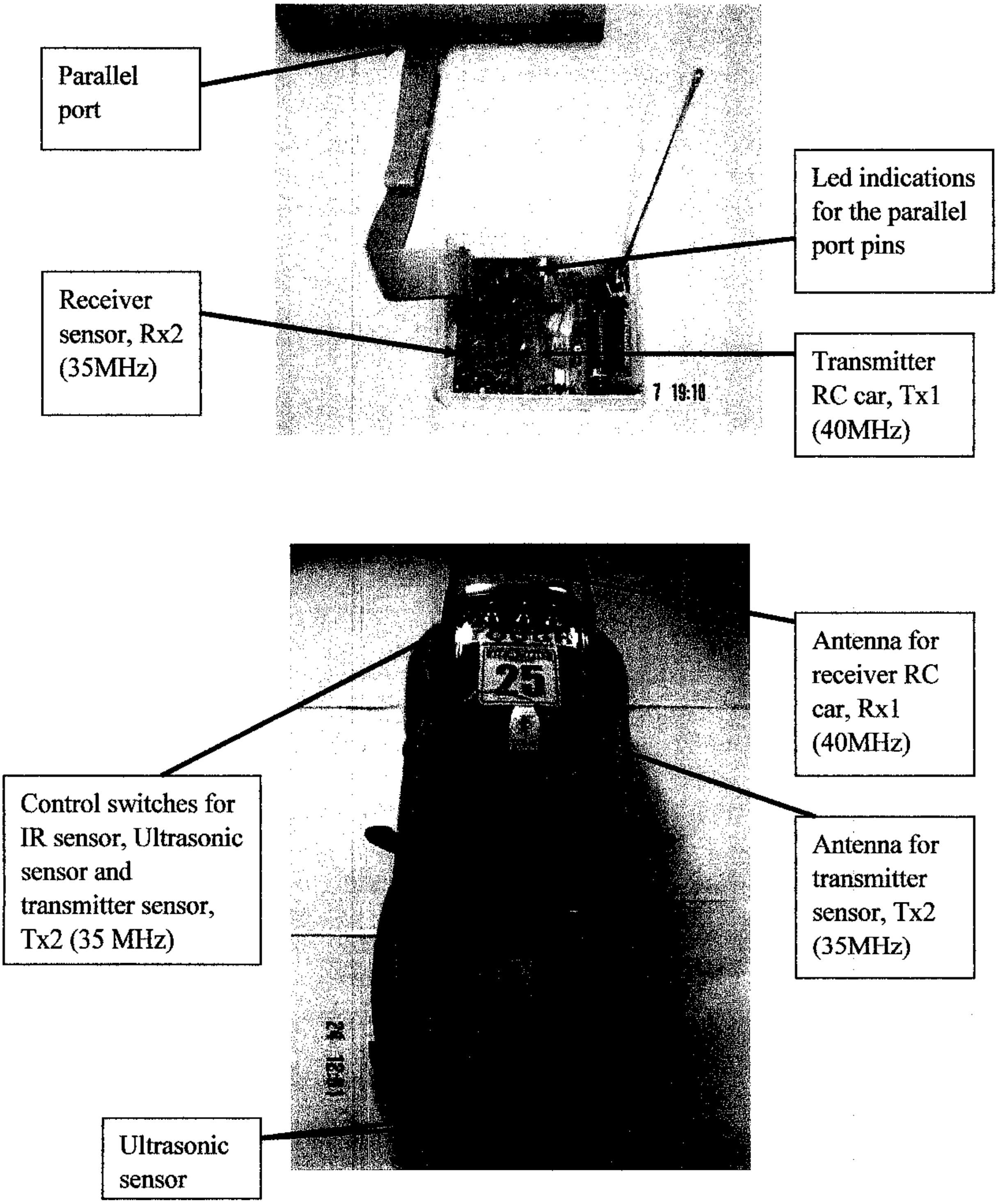

Parallel port
Led indications for the parallel port pins
Receiver sensor, Rx2 (35MHz)
Transmitter RC car, Tx1 (40MHz)
7 19:10
Antenna for receiver RC car, Rx1 (40MHz)
25
Control switches for IR sensor, Ultrasonic sensor and transmitter sensor, Tx2 (35 MHz)
Antenna for transmitter sensor, Tx2 (35MHz)
Ultrasonic sensor

Figure 28 Final configuration for the Smart RC car.

4.2 Discussion

The results show that it is possible to control the speed of the RC car using the Visual Basic 6.0 Programming. The controlled speed is set based on the fuzzy logic "Cause Effect" method. Knowledge on defining the best possible code is essential in developing the interface between the visual basic and the RC car.

Research on the speed of the RC car is done before deciding the suitable controlled speed. Factors such as friction force of the tires on the ground and the 'discharging condition' of the battery are taken into account during the research. All those factors will lead to an inaccurate run. Interval time for the timer control is set during the final testing of the RC car. Based on the final testing, there are tolerance between the visual basic interface and the real movement of the RC car. This is considered negligible as the car's movement can't be sensed totally (only speed is sensed) by the computer.

The RC car will move about one second after the program runs. These is due to the transmission process starting when programmed code is sent to the parallel port and trigger the electrical contact of the transmitter. The signal is then transferred to the radio reception to be decoded before it is finally sent to the motor to be executed.

Moreover, it is confirmed both sensors which are Ultrasonic sensor and IR sensor able to detect obstacle and measured distance covered by the RC car. All the hardware connections were connected properly and able to sense obstacle. The obstacle sensor can integrate successfully with the circuit board of the RC car which consists of transmitter and receiver. Impulse period of the output voltage is useful to determine the timing for the RC car to start cornering.

There are few problems on the timing of the Visual Basic. Enhancement on the Visual Basic code is essential to produce unwavering mode during running program. Overall, it is confirmed that two way communications was successfully being done. The measured distance of obstacle was able to test by using oscilloscope. The hardware connection was successfully connected and the programming code problems are overcome. This is due to the additional timer function and interval needs to be add that interfere the code of the pre-determine route

CHAPTER 5
CONCLUSIONS AND RECOMMENDATIONS

From the observed result, the project was successfully done. This project will enable the computer to control the movement of the RC car using Visual Basic program via the parallel-transmitter connection. The application is implemented on an 'intelligent' RC car which can avoid obstacle detected with the appropriate speed. Through various testing, the accuracy of the program has been increased tremendously along the project period.

Knowledge in visual basic programming is essential to enable the invention of the autonomous movement of the RC car. Beside that, in-depth study on fuzzy logic is vital in obtaining the controlled speed of the RC car. All the experience and knowledge gathered throughout the project is useful as a step to be a good engineer with expertise in critical thinking.

As a conclusion, both sensors were successfully being test out and able to sense obstacle and distance measure of the RC car. The visual basic code was written separately for pre-determine route and obstacle avoidance path. The project's aim is successfully achieved by implementing the pre-determine route and controlling speed of the RC car using the program code from the visual basic programming code. The Smart RC car is finally completed.

Recommendation required in enhancing the Visual Basic code. It is important to implement the reliable programming code for the intelligence part of the RC car. Furthermore, a design using fuzzy logic software is useful to get a better application of 'Smart' RC car.

REFERENCES

1. Rod Stephens, Visual Basic Graphics Programming Book, 2nd edition. Hands-on Applications with a Design Perspective, Wiley Computer Publishing

2. Peter M. Maurer. "Visual Basic Programming" (Mac 2005)
   http://www.csee.usf.edu/~maurer/vbasic/

3. Aaron. "Programming the Parallel Port in Visual Basic" (Mac 2005)
   http://www.aaroncake.net/forum/

4. Ian Harries. "Interfacing to the IBM-PC Parallel Port" (Mac 2005)
   http://www.doc.ic.ac.uk/~ih/doc/par/doc/program.html

5. Elenco Electronics.inc. "Radio-Controlled Car Kit" (Mac 2005)
   http://www.elenco.com

6. Final Year Project Complete Guidelines, Universiti Teknologi Petronas.

7. Fuzzy Logic Jump-Start Publications (April 2005)
   http://www.fuzzy-logic.com/Ch1.htm
   http://www.fuzzy-logic.com/Ch3.htm

8. Ian Harries, "IBM-PC Parallel Printer Port" (Mac 2005)
   http://www.hamhud.net/ka9mva/intro.html

9. Dr. Bart Kosko, "What Is Fuzzy Logic?" (April 2005)
   http://www.aptronix.com/fide/whatfuzzy.htm

REFERENCES

10. J.S. Sweitzer, "What is Infrared Light?" (September 2005)

http://astro.uchicago.edu/cara/about_cara/defn/irlight.html

11. How stuff works. “How RC car works” (Mac 2005)

http://www.howstuffworks.com

12. The LED Light.com “Infrared IR Light” (September 2005)

http://www.theledlight.com/LEDFlashlights.html

13. Madhup Chandra, Electronic Textiles for Autonomous Location Awareness (Mac 2005)

http://www.ccm.ece.vt.edu/etextiles/publications/Thesis_Madhup_Chandra.pdf

14. Zou Yi, Thesis on Multi-Ultrasonic Sensor Fusion for Mobile Robots in Confined Spaces, 2001 (September 2005)

15. “Planet source code”, ***http://www.planet-source-code.com/*** (April 2005)

16. ESCOL Electronics Hobby Kit “Ultrasonic Motion Detector” (August 2005)

http://www.escol.com.my/ES-015-2.html

http://www.escol.com.my/ES-032.html

17. “VB helper”, ***http://www.vb-helper.com/howto_animate_image.html*** (Mac 2005)

APPENDICES

APPENDIX A

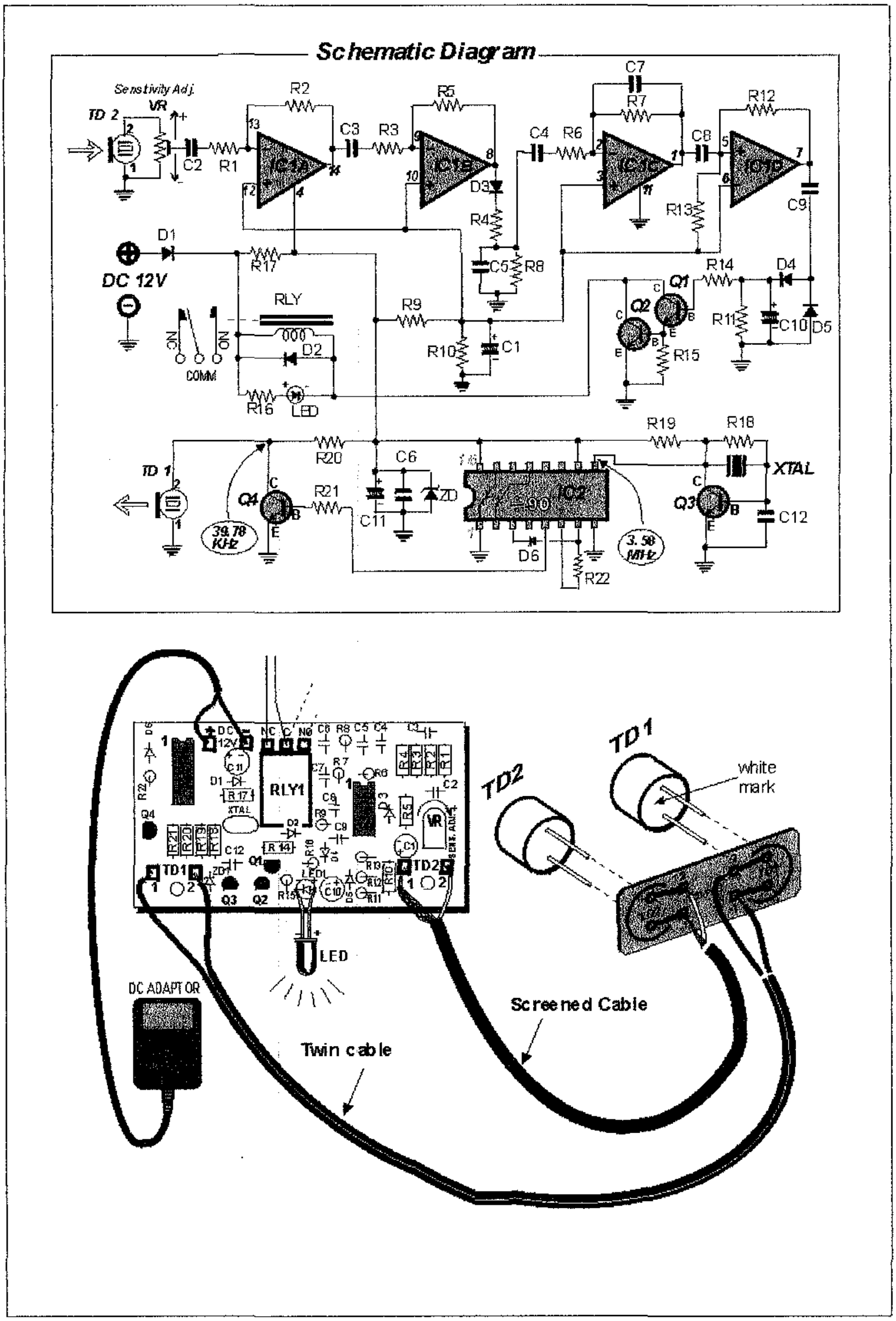
PROJECT PROTOTYPE: RC CAR

Normal RC car

Smart RC car

APPENDIX B

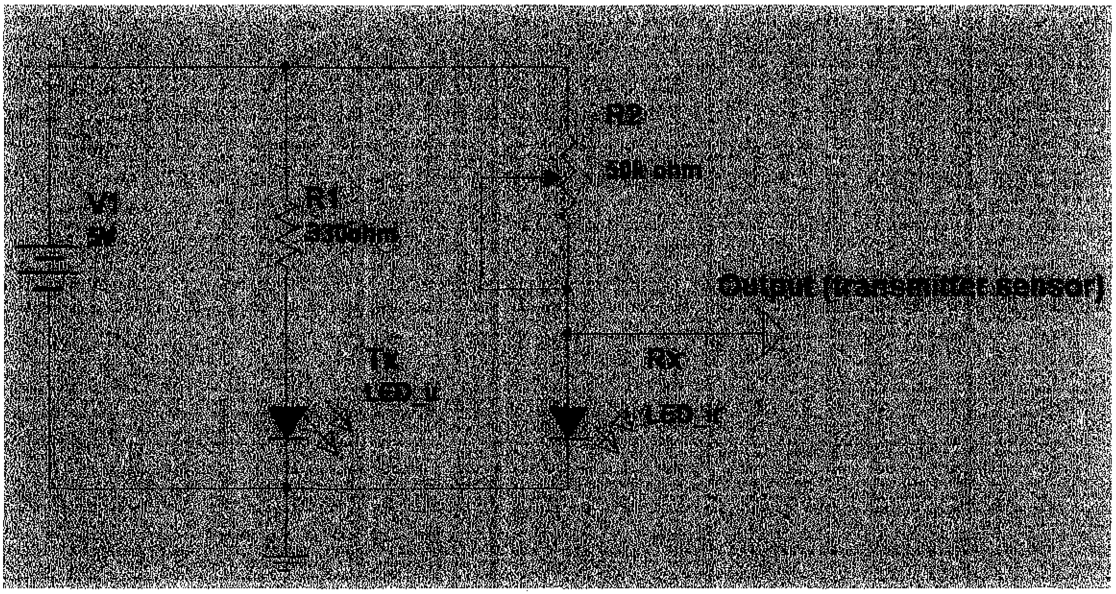
ULTRASONIC SENSOR KIT DIAGRAM



Taken from: ***http://www.escol.com.my/ES-015-2.html***

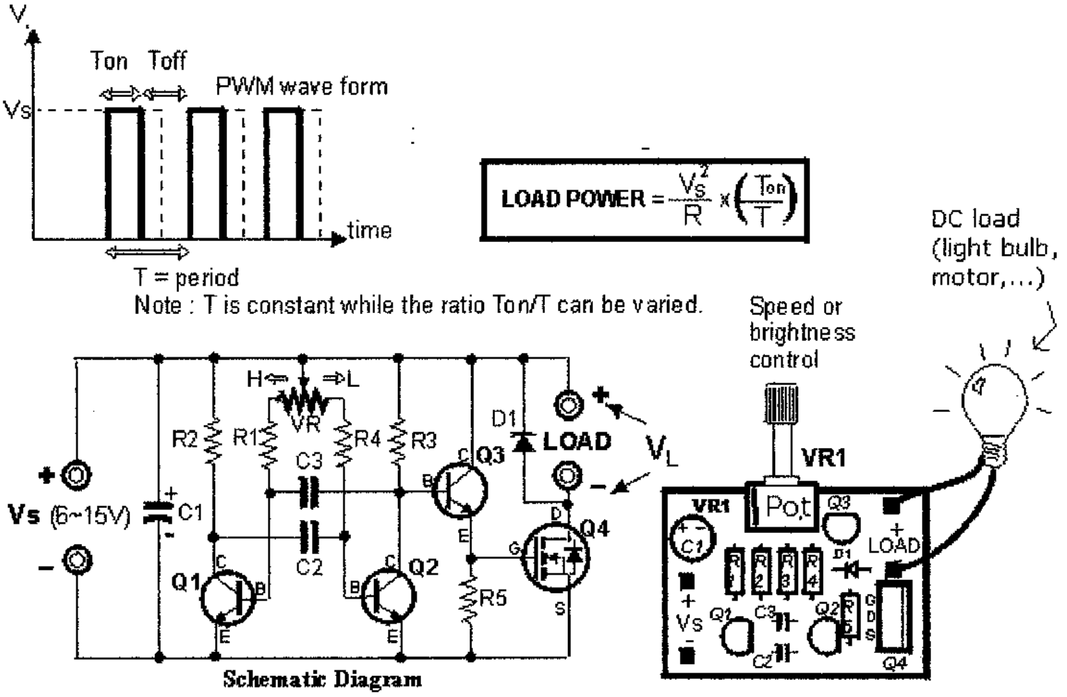
APPENDIX C

IR SENSOR CIRCUIT SCHEMATIC

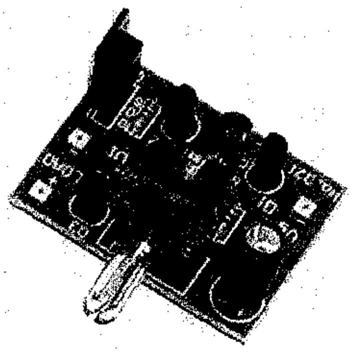


APPENDIX D

MOTOR SPEED CONTROLLER SCHEMATIC DIAGRAM



Schematic Diagram



47 x 35 mm

Taken from: ***http://www.escol.com.my/ES-032.html***

APPENDIX E

VB PROGRAMMING CODE: PRE-DETERMINED ROUTE

```
Option Explicit
'initialize parallel port
Private Declare Function Inp Lib "inpout32.dll" _
Alias "Inp32" (ByVal PortAddress As Integer) As Integer
Private Declare Sub Out Lib "inpout32.dll" _
Alias "Out32" (ByVal PortAddress As Integer, ByVal Value As Integer)
Private Declare Sub Sleep Lib "Kernel32" (ByVal dwMilliseconds As
Long)

'start the program
Private Sub start_Click()
tmrTimer.Enabled = True
Timer1.Enabled = True   'timer for pre-determine route
Label1.Caption = "0"    'label for distance measured
End Sub

Private Sub Form_Load()
tmrTimer.Enabled = False
End Sub

Private Sub stop_Click() 'stop manually the program
tmrTimer.Enabled = False
Timer1.Enabled = False
Dim stopcar As Boolean
stopcar = True
End Sub

Private Sub Timer1_Timer() 'activate timer for distance measurement
Dim port2 As Integer
Dim detect As Integer      'detect status port
Dim stopcar As Boolean

port2 = 889                'address for status port
detect = Inp(port2)
DoEvents
  If stopcar Then
Timer1.Enabled = False
```

```
Exit Sub
Timer1.Enabled = True
End If
If (detect = 104) Then  'input pin, S4 (pin 15)
Label1.Caption = Label1.Caption + 1 'adding up distance measured
End If
End Sub

Private Sub tmrTimer_Timer()'activate timer for  pre-determine route

Picture1.AutoRedraw = True
Picture1.ScaleMode = vbPixels
Picture1.Cls

Checkspeed              'run the pre-determine route
End Sub

Private Sub checkspeed()

Dim port1 As Integer
Dim stopcar As Boolean
Dim badgeCount As Integer
Dim y As Integer
Dim x As Integer
Dim detect As Integer

y = Val(Picture1.Top)   'vertical movement,forward
x = Val(Picture1.Left)  'horizontal movement,left or right turn

stopcar = False

port1 = 888             'address for data port
Out port1, 255          'output pin, D7 (pins 8)

Select Case y           'define coordinate of y-axis

Case Is = "4560"
badgeCount = 10000
If Picture1.Top = 4560 And Picture1.Left = 1800 Then
lblslow_down.Caption= "speed_up" 'fuzzy logic cause effect triangles
Do While badgeCount > 0 And Picture1.Top > 2880
DoEvents                      'manually stop if any emergency
```

```
  If stopcar Then
tmrTimer.Enabled = False
Exit Sub
tmrTimer.Enabled = True
End If
   badgeCount = badgeCount - 1
   Picture1.Top = Picture1.Top - 1  'move RC car forward direction
   Loop
   port1 = 888
    Out port1, 127
Else
badgeCount = 10000
lblslow_down.Caption = "right_slow"
Do While badgeCount > 0 And Picture1.Top < 5880 And Picture1.Left >
6000
DoEvents
  If stopcar Then
tmrTimer.Enabled = False
Exit Sub
tmrTimer.Enabled = True
End If
   badgeCount = badgeCount - 1
   Picture1.Top = Picture1.Top + 1
   Picture1.Left = Picture1.Left - 1 'move RC car right direction
move RC car right direction
   Loop
   port1 = 888
     Out port1, 237
End If

Case Is = "2880"
badgeCount = 10000
If Picture1.Top = 2880 And Picture1.Left = 1800 Then
lblslow_down.Caption = "slow_down"
Do While badgeCount > 0 And Picture1.Top > 1920
DoEvents
  If stopcar Then
tmrTimer.Enabled = False
Exit Sub
tmrTimer.Enabled = True
End If
    badgeCount = badgeCount - 1
    Picture1.Top = Picture1.Top - 1
    Loop
    port1 = 888
    Out port1, 253

Else
badgeCount = 10000
lblslow_down.Caption = "slow_down"
Do While badgeCount > 0 And Picture1.Top < 4560
DoEvents
  If stopcar Then
tmrTimer.Enabled = False
Exit Sub
tmrTimer.Enabled = True
End If
    badgeCount = badgeCount - 1
    Picture1.Top = Picture1.Top + 1
    Loop
    port1 = 888
    Out port1, 253
End If

Case Is = "1920"
badgeCount = 10000
If Picture1.Top = 1920 And Picture1.Left = 1800 Then
lblslow_down.Caption = "right_slow"
Do While badgeCount > 0 And Picture1.Top > 600 And Picture1.Left <
3120
DoEvents
  If stopcar Then
tmrTimer.Enabled = False
Exit Sub
tmrTimer.Enabled = True
End If
    badgeCount = badgeCount - 1
 If Picture1.Picture = 0 Then Exit Sub
    Picture1.Top = Picture1.Top - 1
    Picture1.Left = Picture1.Left + 1
    Loop
    port1 = 888
    Out port1, 237
Else
```

```
badgeCount = 10000
lblslow_down.Caption = "speed_up"
Do While badgeCount > 0 And Picture1.Top < 2880
DoEvents
  If stopcar Then
tmrTimer.Enabled = False
Exit Sub
tmrTimer.Enabled = True
End If
    badgeCount = badgeCount - 1
 If Picture1.Picture = 0 Then Exit Sub
    Picture1.Top = Picture1.Top + 1
    Loop
    port1 = 888
    Out port1, 127
End If
Case Else

Select Case x            'define coordinate of x-axis

Case Is = "3120"
badgeCount = 10000
If Picture1.Top = 600 And Picture1.Left = 3120 Then
lblslow_down.Caption = "speed_up"
Do While badgeCount > 0 And Picture1.Left < 4800
DoEvents
  If stopcar Then
tmrTimer.Enabled = False
Exit Sub
tmrTimer.Enabled = True
End If
    badgeCount = badgeCount - 1
If Picture1.Picture = 0 Then Exit Sub
    Picture1.Left = Picture1.Left + 1
    Loop
    port1 = 888
    Out port1, 127
Else
badgeCount = 10000
lblslow_down.Caption = "right_slow"
Do While badgeCount > 0 And Picture1.Top >= 4560 And Picture1.Left
>= 1800
```

```
DoEvents
  If stopcar Then
tmrTimer.Enabled = False
Exit Sub
tmrTimer.Enabled = True
End If
    badgeCount = badgeCount - 1
If Picture1.Picture = 0 Then Exit Sub
    Picture1.Top = Picture1.Top - 1
    Picture1.Left = Picture1.Left - 1
    Loop
    port1 = 888
    Out port1, 237
End If

Case Is = "4800"
badgeCount = 10000
If Picture1.Top = 600 And Picture1.Left = 4800 Then
lblslow_down.Caption = "slow_down"
Do While badgeCount > 0 And Picture1.Left < 6000
DoEvents
  If stopcar Then
tmrTimer.Enabled = False
Exit Sub
tmrTimer.Enabled = True
End If
    badgeCount = badgeCount - 1
 If Picture1.Picture = 0 Then Exit Sub
    Picture1.Left = Picture1.Left + 1
    Loop
    port1 = 888
    Out port1, 253
Else
badgeCount = 10000
lblslow_down.Caption = "slow_down"
Do While badgeCount > 0 And Picture1.Left > 3120
DoEvents
  If stopcar Then
tmrTimer.Enabled = False
Exit Sub
tmrTimer.Enabled = True
End If
```

```
    badgeCount = badgeCount - 1
 If Picture1.Picture = 0 Then Exit Sub
    Picture1.Left = Picture1.Left - 1
    Loop
    port1 = 888
    Out port1, 253
End If

Case Is = "6000"
badgeCount = 10000
If Picture1.Top = 600 And Picture1.Left = 6000 Then
lblslow_down.Caption = "right_slow"
Do While badgeCount > 0 And Picture1.Left < 7320 And Picture1.Top <
1920
DoEvents
  If stopcar Then
tmrTimer.Enabled = False
Exit Sub
tmrTimer.Enabled = True
End If
    badgeCount = badgeCount - 1
 If Picture1.Picture = 0 Then Exit Sub
    Picture1.Left = Picture1.Left + 1
    Picture1.Top = Picture1.Top + 1
    Loop
    port1 = 888
    Out port1, 237
Else
badgeCount = 10000
lblslow_down.Caption = "speed_up"
Do While badgeCount > 0 And Picture1.Left > 4800
DoEvents
  If stopcar Then
tmrTimer.Enabled = False
Exit Sub
tmrTimer.Enabled = True
End If
    badgeCount = badgeCount - 1
 If Picture1.Picture = 0 Then Exit Sub
    Picture1.Left = Picture1.Left - 1
    Loop
    port1 = 888
    Out port1, 127
End If
Case Else

End Select
End Select

End Sub
```

APPENDIX F

VB PROGRAMMING CODE: OBSTACLE DETECTION

```
Option Explicit
'initialize parallel port
Private Declare Function Inp Lib "inpout32.dll" _
Alias "Inp32" (ByVal PortAddress As Integer) As Integer
Private Declare Sub Out Lib "inpout32.dll" _
Alias "Out32" (ByVal PortAddress As Integer, ByVal Value As Integer)
Private Declare Sub Sleep Lib "Kernel32" (ByVal dwMilliseconds As
Long)

'start the program
Private Sub start_Click()
Tmr1.Enabled = True        'timer for pre-determine route
Tmr2.Enabled = False       'timer for obstacle detection

Dim distance As Integer
Dim port2 As Integer
Dim port1 As Integer

port2 = 889               'address for status port
port1 = 888               'address for data port

End Sub

Private Sub Form_Load()
Tmr1.Enabled = False
Tmr2.Enabled = False
Dim port1 As Integer

Out port1, 255            'output pin, D7 (pin 8)
End Sub

Private Sub stop_Click()   'manually stop the program
Tmr1.Enabled = False
Tmr2.Enabled = False
Dim port1 As Integer

port1 = 888
```

```
Out port1, 255
End Sub

Private Sub tmr1_Timer()

Picture1.AutoRedraw = True
Picture1.ScaleMode = vbPixels
Picture1.Cls
Dim detect As Integer
Dim port2 As Integer
Dim port1 As Integer

Out port1, 255
port2 = 889
detect = Inp(port2)

If (detect = 120) Then          'default value of input port
label2.Caption = "No Obstacle"
    checkspeed                  'runs the pre-determine route
Else
    detect_obstacle             'runs the obstacle avoidance path
    Text1.Text = detect         'show the input pins detect
End If
End Sub

Private Sub tmr2_Timer() 'activate timer for obstacle avoidance path

Picture1.AutoRedraw = True
Picture1.ScaleMode = vbPixels
Picture1.Cls
Dim detect As Integer
Dim port2 As Integer

port2 = 889
detect = Inp(port2)          'initialize input pins

If (detect = 112) Then       'input port, S4 (pin 15)
label2.Caption = "Detect Obstacle"
Tmr1.Enabled = False
    detect_obstacle
Else
    checkspeed
```

```
End If
End Sub

Private Sub checkspeed()
Dim port1 As Integer
Dim port2 As Integer
Dim stopcar As Boolean
Dim pos As Integer
Dim pos2 As Integer
Dim detect As Integer
Dim step As Integer

pos = Val(Picture1.Top)
pos2 = Val(Picture1.Left)
port1 = 888

Out port1, 255

port2 = 889
detect = Inp(port2)

Text1.Text = detect

Dim Count As Integer

Count = 100
If (detect = 120) Then
Do While Count > 0 And Picture1.Top > 240
    Count = Count - 1
    Picture1.Top = Picture1.Top - 1
    Loop
    Out port1, 253
End If
End Sub

Private Sub detect_obstacle()

Dim port1 As Integer
Dim port2 As Integer
Dim stopcar As Boolean
Dim Count As Double
Dim count2 As Double
Dim count3 As Double
Dim count4 As Double
Dim pos As Integer
Dim pos2 As Integer
Dim detect As Integer

pos = Val(Picture1.Top)
pos2 = Val(Picture1.Left)
Count = 1000
count2 = 1000
count3 = 1000
count4 = 1000
stopcar = False

Tmr2.Enabled = True
Tmr1.Enabled = False

port1 = 888
Out port1, 255

port2 = 889
Inp port2

detect = Inp(port2)
port1 = 888

Select Case Count
Case Is = 1000

Do While Count > 0
    Count = Count - 1
    Picture1.Top = Picture1.Top - 1
    Picture1.Left = Picture1.Left + 1
    Loop
Out port1, 237
End Select

Select Case Count
Case Is = 1000

Do While Count > 0
    Count = Count - 1
    Picture1.Top = Picture1.Top - 1
    Picture1.Left = Picture1.Left + 1
    Loop
Out port1, 237
End Select

Select Case Count
Case Is = 1000

Do While Count > 0
    Count = Count - 1
    Picture1.Top = Picture1.Top - 1
    Picture1.Left = Picture1.Left + 1
    Loop
Out port1, 237
End Select

Select Case count2
Case Is = 1000

Do While count2 > 0
    count2 = count2 - 1
    Picture1.Top = Picture1.Top - 1
    Loop
Out port1, 253
End Select

Select Case count2
Case Is = 1000

Do While count2 > 0
    count2 = count2 - 1
    Picture1.Top = Picture1.Top - 1
    Loop
Out port1, 253
End Select

Select Case count3
Case Is = 1000

Do While count3 > 0
    count3 = count3 - 1
    Picture1.Top = Picture1.Top - 1
    Picture1.Left = Picture1.Left - 1 ‘move RC car left direction
    Loop
Out port1, 249

Select Case count4
Case Is = 1000

Do While count4 > 0
    count4 = count4 - 1
    Picture1.Top = Picture1.Top - 1
    Picture1.Left = Picture1.Left - 1
    Loop
Out port1, 249

Select Case count4
Case Is = 1000

Do While count4 > 0
    count4 = count4 - 1
    Picture1.Top = Picture1.Top - 1
    Picture1.Left = Picture1.Left - 1
    Loop
Out port1, 249
End Select

Tmr1.Enabled = True
checkspeed

End Select
End Select
End Sub
```

APPENDIX G

VB PROGRAMMING CODE: DISTANCE MEASUREMENT

```
Option Explicit
'initialize parallel port
Private Declare Function Inp Lib "inpout32.dll" _
Alias "Inp32" (ByVal PortAddress As Integer) As Integer
Private Declare Sub Out Lib "inpout32.dll" _
Alias "Out32" (ByVal PortAddress As Integer, ByVal Value As Integer)
Private Declare Sub Sleep Lib "Kernel32" (ByVal dwMilliseconds As
Long)
'start the program
Private Sub start_Click()
tmrTimer.Enabled = True
Timer1.Enabled = True
Label1.Caption = "0"           'initialize the distance
End Sub

Private Sub Timer1_Timer()     'activate distance measurement
Dim port2 As Integer
Dim detect As Integer
Dim stopcar As Boolean

port2 = 889                    'address for status port
detect = Inp(port2)            'initialize for input pins
DoEvents
  If stopcar Then
Timer1.Enabled = False
Exit Sub
Timer1.Enabled = True
End If
If (detect = 104) Then         'input port, S4 (pin 15)
Label1.Caption = Label1.Caption + 1 'show the distance measured
End If
End Sub
```