Intelligent Parallel Parking and Obstacle Avoidance System

Spring 2011 Senior Design Project

Ajit Gauli

Department of Electrical Engineering

University of New Orleans

New Orleans, LA USA

agauli@uno.edu

***Abstract—*** **The objective of this project is to design and build the Intelligent Parallel Parking and Obstacle Avoidance System. The system is comprised of mechanical, navigation, microprocessing and power systems installed on a model car. The system measures an available parking space, checks whether the space is large enough for parking, aligns the car correctly in the traffic lane and then performs the necessary parallel parking maneuvers into the parking spot. In addition to parallel parking, the system can detect static obstacles, like road blocks and dynamic obstacles, like pedestrians, in the vehicle’s path. If dynamic obstacles are detected, the vehicle stops until the obstacle moves out of the way. If static obstacles are detected, the vehicle maneuvers around the obstacle ensuring a collision free trajectory. The product provides an economic solution to autonomous parallel parking and can be added to any regular vehicle thus encompassing a wider customer base. This paper primarily focuses on sensor system and motor system while giving general idea about other systems.**

Keywords- IPPOAS(Intelligent Parallel Parking and Obstacle Avoidance System), Infra Red (IR),Ultrasonic Range Finder (URF) sensors; motor control; dc motor; servo motor

# Introduction

With the advent of smarter technology, more car manufactures such as Toyota, Ford and BMW  are upgrading their product by adding autonomous features. One of the important additions to the new age car is the function of autonomous parallel parking. Due to the need of precisely calculated maneuvers in a risky situation, parallel parking is the task even experienced drivers find very difficult to perform. Therefore, autonomous parallel parking feature could be a sigh of relief to the metro drivers who can’t avoid parallel parking due to the limited availability of parking space in the cities.

IPPOAS has made car-parking a fully autonomous job. Unlike competitors such as Ford whose system only performs the steering maneuvers and requires the driver to manually operate the gas pedal or hit the brake as described in [1], our system completely relieves the drivers from the parking hassles .It is flexible enough to be installed in any four-wheeled vehicle, regardless of make and model.. It is safe as it can detect obstacles in the traffic lane, maneuver around static obstacles like road blocks, stop for dynamic obstacles like pedestrians crossing the traffic lane, detect parking space, check whether the parking space is large enough for parallel parking and perform the necessary parking maneuver into the parking spot.

The design and implementation of our autonomous parallel parking system was completed in two semesters starting Fall 2010. During the first semester master schedule, initial design, and estimated budget were prepared. At the end of the first semester necessary components were procured after careful market research. During the second (current) semester, the individual components were assembled together, and the complete system was programmed to perform autonomous parallel parking avoiding obstacles if any.

IPPOAS will have a positive impact on the society by incorporating smart technologies to improve the comfort and safety of drivers, and decrease the parking related accidents.

# SYSTEM DESIGN

## Initial Design

The tentative design of our project was sketched during our Senior Design I class. After several team meeting an initial design was finalized with the approval from my team members. We decided to use a DC motor to drive the back wheel, a stepper motor to steer the vehicle, four infrared sensors for obstacle detection (two on the front and two on the rear), two ultrasonic sensors for distance measurement (attached to the right hand side of the chassis), a microprocessor with at least 6 analog pins, a 7-8 V battery to power the dc motor and a 5 V USB battery to power all other components. Based on these criteria we ordered the parts, and started to build our product.

## Final Design

Once we received these parts, we started to test them individually and finalize our design. The final design phase introduced a few modifications from our initial design. First of all, the DC motor provided with the chassis required high current and power which didn’t meet our design specification. It required a high current of 2.5 A as seen in [2]; whereas, the H-bridge providing the forward and backward drive logic to the DC motor could not handle current over 1 A as seen in [3]. Also, the speed of the original DC motor was too high for the purpose of parallel parking. Therefore, we used the back-up DC motor which was ordered separately. As the back-up DC motor was considerably smaller in size, we had to design a new motor mount to secure the new DC motor onto the DC motor casing designed for a larger motor. To solve this problem, we used a plastic housing that would hold the small DC motor into the larger space. It was fastened to the housing using screws and washers to eliminate any unwanted wobbling of the motor body.

Next, we finalized the design of the steering motor. While testing the steering motor, we realized that the stepper motor that we initially bought was too bulky and using a servo motor instead was actually sufficient for our project precision-wise. Therefore, we replaced the stepper with a servo motor bought from a local hobby store.

The range of digital infrared sensors (24cm default) initially bought could not be modified through the code. Thus we mechanically modified them to make the range shorter. While doing so the IR sensors in the back were damaged. So three analog sensors were purchased later whose range could be changed through the code. Two of them were used in the back, one was kept for backup.

After, the completion of the final design, we were ready to assemble these parts and interface them with each other. Figure 1 shows the configuration of individual parts mounted on the chassis.

**2**

**1**

**3**

**3**

**4**

**4**

**5**

**5**

**S. No Components**

**1** DC Motor for Rear-Wheel Drive

**2** Servo Steering Motor

**3** Cross-tanked Front IR Sensors

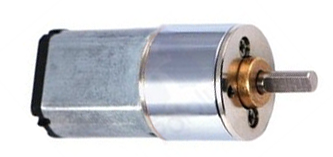
**4** Cross-tanked Back IR Sensors

**5** Distance sensors

Figure 1 – Strategic Placement of Sensors and Motors

# MOTOR SYSTEM

We needed two motors for this project mainly to drive the vehicle forward or backward and to steer the front wheels to the left and right as required. Each motor is described below.



**Figure 2: DC motor on left, Servo motor on right**

## Driving Motor (DC Motor)

For driving the vehicle DC motor was chosen because it can give continuous rotation and can be simply controlled by a motorcontroller. Pololulu DC gear motor was selected after its fair price, compatibility to the system, and small size. More on its features is given below.

**Features:**

•Nominal Voltage: 6V,Stall Current: 800mA, FreeRPM:140  
•StallTorque: 15oz-in(1.0kg-cm)Reduction:100:1

•Size(mm):1.66"x0.46"x0.60"  
•Weight: 0.80oz / 27g

•Price: $14.95

Before installing the motor into the chassis, it was tested using this piece of code.

digitalWrite(motor1Pin, LOW); //set leg 1 of the H-bridge low

digitalWrite(motor2Pin,HIGH);//set leg 2 of the H-bridge high

After successful testing it was then mounted into the DC motor housing.

## Steering Motor( Servo Motor)

Servo Motor was chosen over stepper motor for the ease of use. This particular model of Servo Motor (Futaba 3003S) was selected because it can be directly controlled by the microprocessor, low price, and fits perfectly onto the chassis servo mount.

**Features:**

•Speed: 0.23 sec/60° @ 4.8V, 0.19 sec/60° @ 6V

•Torque: 3.2 kg-cm) @ 4.8V, 4.1 kg-cm) @ 6V

•Dimensions: 40 x 20 x 36mm

•Weight: 37g

•Connector: "J" type with approx. 5" lead

•Price: $14.00

Following code that turns the front wheel from straight to 55 degree to the right was used to test it.

for(pos = 90; pos <= 145; pos += 1) // turn 55 degree right

{ // in steps of 1 degree

myservo.write(pos);//servo position = variable 'pos'

delay(15); // waits 15ms for the servo

}

After successfully testing the servo motor it was then mounted onto the servo mount on the chassis.

## Motor Interfacing

**Figure 3: Interfacing of the motor system**

As shown in Figure 3,servo steering motor is controlled and powered by the microprocessor. Back wheel driving motor (DC Motor) is controlled by the motorcontroller through the logic provided by the microprocessor and powered by the 7.2 V Ni-Cd battery.

# IV. SENSOR SYSTEM

Sensor system consists of two kind of sensors as shown in Figure 4 below.

## 

## **Figure 4: Obstacle Sensor on left, distance sensor on right**

## A. Obstacle Sensor

Sharp GP2D15 Infrared Sensor

This sensor was chosen because of the following features.

* Infrared distance measuring sensor.
* Digital Output, high when an object is closer than 24cm.
* Input Voltage 4.5 to 5.5 V, Input current: 33 mA
* Immune to interference from ambient light
* Size: 60 x 35 x 20 mm
* Price: $14

Four of these sensors a pair in front and another pair in back are used to detect the obstacles.

For maximum area coverage and for checking the tire fronts efficiently we mounted the sensors in a cross-tanked style as shown in Figure 5.

Cross Tank Configuration



**Figure 5: Obstacle Sensors mounted in cross-tanked style**

The analog voltage output from the sensors was scaled to the value range between 10-1024. Then these sensors were tested using the following codes.

int distanced1 = analogRead(0);

int distanced2 = analogRead(1);

if ( distanced1 > 900 || distanced2 > 900)

{ //stop when obstacle present

digitalWrite(motor1Pin,LOW); digitalWrite(motor2Pin, LOW);

}

**Theory**

These sensors use triangulation and a small linear CCD array to compute the distance and/or presence of objects in the field of view.  A pulse of IR light is emitted by the emitter.  This light travels out in the field of view and either hits an object or just keeps on going.  In the case of no object, the light is never reflected and the reading shows no object.  If the light reflects off an object, it returns to the detector and creates a triangle between the point of reflection, the emitter, and the detector. The angles in this triangle vary based on the distance to the object.  The receiver portion of these new detectors is actually a precision lens that transmits the reflected light onto various portions of the enclosed linear CCD array based on the angle of the triangle described above.  The CCD array can then determine what angle the reflected light came back at and therefore, it can calculate the distance to the object.

It is therefore these sensors are highly immune to interference from ambient light and offer amazing indifference to the color of object being detected.

## B. Distance Sensor

Devantech SFR05 Ultrasonic Sensor

Servo Motor

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

This particular sensor was chosen for the following reasons.

•Voltage: 5V, Current: 4mA Typ.

•Frequency: 40KHz

•Max Range: 4m, Min Range: 1cm

•Modes: Single pin for trig/echo

•Input Trigger: 10uS Min. TTL level pulse

•Echo Pulse: Positive TTL signal, width proportional range

•Small Size: 43mm x 20mm x 17mm height

Following code was used to test the distance sensors.

digitalWrite(LED, LOW);

pinMode(pingPin, OUTPUT);

digitalWrite(pingPin, LOW);

delayMicroseconds(2);

digitalWrite(pingPin, HIGH);

delayMicroseconds(5);

digitalWrite(pingPin, LOW);

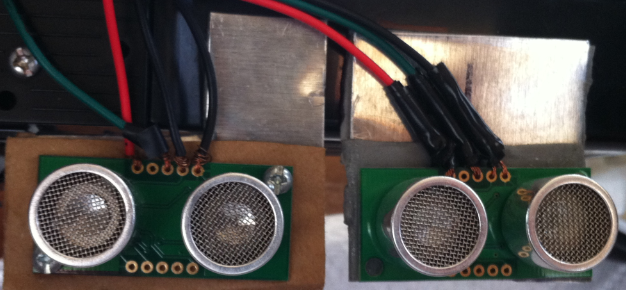
pinMode(pingPin, INPUT);

duration = pulseIn(pingPin, HIGH);

// convert the time into a distance

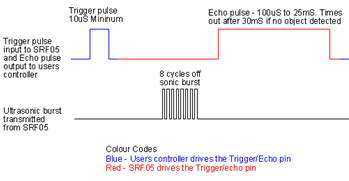
mm = microsecondsToCentimeters(duration);

After testing them and determining that they work correctly we mounted them side by side onto the right side of the chassis as shown in Figure 6. These were used for distance calculation and the reading of the two was compared to align the vehicle along the traffic lane.

****

**Figure 6: Distance Sensors Placement**

## C. Theory



**Figure 7: SFR05 Timing Diagram**

The theory of operation in detail can be viewed in [3]. As shown in the timing diagram above after a short 10uS trigger pulse from the microprocessor, the SRF05 starts the ranging. The SRF05 will send out an 8 cycle burst of ultrasound at 40khz and raise its echo line high. It then listens for an echo, and as soon as it detects one it lowers the echo line again. The echo line is therefore a pulse whose width is proportional to the distance to the object. By timing the pulse it is possible to calculate the distance as given by the formula below.

Distance (in cm) = width of the pulse is in uS/ 58

If nothing is detected then the SRF05 will lower its echo line anyway after about 30mS.The SRF05 can be triggered as fast as every 50mS, or 20 times each second.

## D. Sensor Interfacing

Figure 8 below shows the interfacing of the distance and obstacle sensors.

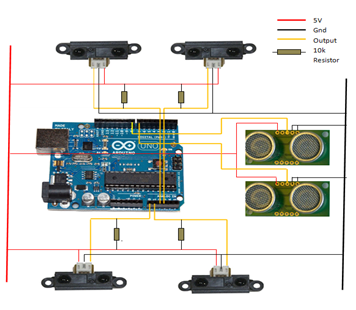
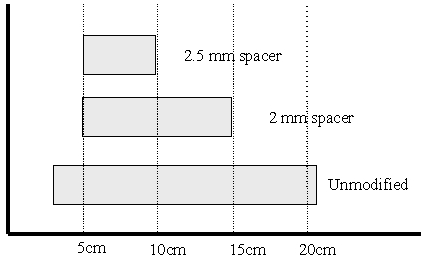


Figure 8: Sensor Interfacing

# V. Errors And Remidies

GP2D15 has a factory set 24 cm default detection range which was too large for our design. In order to solve this issue we manually modified the sensors to have a shorter range.



**Figure 9: Spacer width and range modified**

By following the modification technique as described in [4] we were able to overcome this issue by placing a spacer in between the lens and the IR board. Using a 2mm spacer between the lenses and the IR board we could reduce the detection range of the GP2D15 down to 15 cm. The spacer width and the distance modified can be seen in Figure 9.

##### Acknowledgment

I would like to thank Dr. Parviz Rastgoufard for his insightful guidance, consistent motivation, and valuable feedback which were crucial to the success of this project. I would also like to thank Dr. Jeffrey Frank Gray for guiding us through the initial phase of project selection, schedule preparation, and initial design. Next, I would like to thank the Department of Electrical Engineering at The University of New Orleans for partially funding our project.

The credit for the success of this project also goes to my sincere and hard-working team-mates. I would like to thank Abhishek Yadav for managing the team efficiently and designing the mechanical system. I would like to thank Anil Pandey for the design of the power system. I would like to thank Nischal Basnet for the his laudable endeavors on coding and debugging the system. Without such valuable efforts from all of them this project would not be completed. .

Last but not the least, I would like to extend my gratitude to all my friends and family members, who had constructive interest in this project and provided the moral support to help me keep going.

##### References

1. Ford Motor Company. <http://media.ford.com/article_display.cfm?article_id=29625> [Online]
2. Mabuchi Motor. <http://www.mabuchi-motor.co.jp/en_US/cat_files/rs_540rhsh.pdf> [Online]
3. Robot Electronics

<http://www.robot-electronics.co.uk/htm/srf05tech.htm> [Online]

1. Acroname Robotics

<http://www.acroname.com/robotics/info/ideas/adjust15/adjust15.html> [Online]

The template will number citations consecutively within brackets [1]. The sentence punctuation follows the bracket [2]. Refer simply to the reference number, as in [3]—do not use “Ref. [3]” or “reference [3]” except at the beginning of a sentence: “Reference [3] was the first . . .”

The template will number citations consecutively within brackets [1]. The sentence punctuation follows the bracket [2]. Refer simply to the reference number, as in [3]—do not use “Ref. [3]” or “reference [3]” except at the beginning of a sentence: “Reference [3] was the first . . .”

The template will number citations consecutively within brackets [1]. The sentence —se “Ref. [3]” or “reference [3]” except at the of a sentence: “Reference [3] was the first reference [3except at the beginning of a sentence: “Reference [3] was the first . . .”

The template will number citations consecutively within brackets [1]. The sentence punctuation follows the bracket [2]. Refer simply to the reference number, as in [3]—do not use “Ref. [3]” or “reference [3]” except at the b