POLITECNICO DI MILANO

Final Year Bachelor Project

PIXYBOT

Mentor:

Prof.Andrea Bonarini

Author:

Rohit Prakash

Contents

List of Figures

Abstract

The objective of the project is to develop a simple robogame using Pixy Cam mounted on a 4wd Arduino compatible mobile robot. The Pixy Cam developed by Charmedlabs is fast vision sensor and can be easily interfaced with arduino. The Robot is mounted with IR sensor vision to detect the obstacles using mapping technique and Pixy vision to see the colored target in a room of obstacles. The robot will also have a simple gripper to capture the target and Arduino Mega board as the controller.

1. Introduction

The Project aims to develop a simple Robogame. In this game, a robot is given the task to search for an object in a room of obstacles and capture it. Once captured by the robot, the object should to be taken to another location in the room and dropped. The Robot should also be capable to avoid any obstacles in its path and reach its final destination. The Robot in itself is a four wheel drive with Arduino compatibility and a motor driver. It is equipped with IR sensors to sense distance of obstacles in its path and most importantly PIXY cam to search for the target object. The target is of a particular hue well saturated in light, for the camera to detect among other objects in the room. The Robot is also equipped with a gripper to capture the target. The whole journey of the Robot and its vision to see the targets is the main focus of the project. The robot has to be carefully maneuvered through the obstacles by using precise control algorithms without any flaw.

2. Components

2.1 4WD Arduino compatible mobile platform

This is a well-designed mobile platform, very simple to program and control. It comes with four high quality micro-speed motors to each wheel, giving it enough power. Its metallic aluminum alloy body makes it sturdy and flexibility in rapid motions. It also has variety of options to place the sensors wherever required.

2.2 Micro-Speed Motors

In this project four motors are used , attached to each wheel.

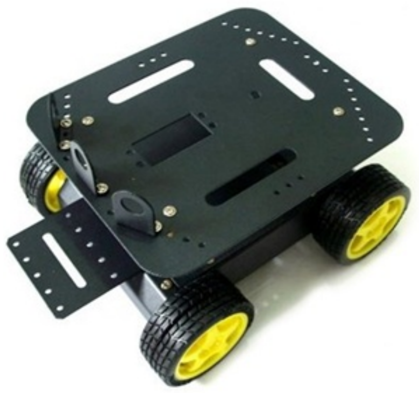

Figure 1: 4WD Arduino compatible mobile platform

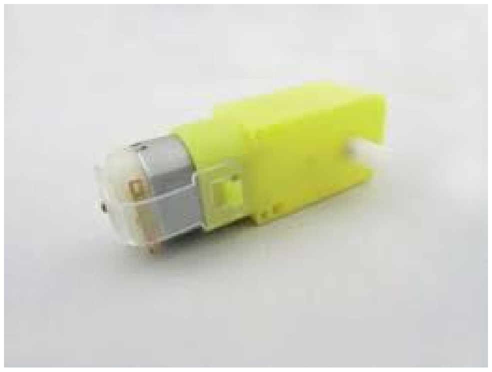

Figure 2: Micro-Speed Motors

Specifications:

- Gear Ratio 1:120
- No-load speed(3V):100RPM
- No-load speed(6V):200RPM
- No-load current(3V):60mA
- No-load current(6V):71mA
- Stall current(3V):260mA

- Stall current(6V):470mA
- Torgue (3V): 1.2Kgcm
- Torque (6V): 1.92Kgcm
- Size: 55mm x 48.3mm x 23mm
- Weight:45g

2.3 Sharp IR sensors GP2D120

Figure 3: IR GP2D120

It is a distance measuring sensor with integrated signal processing and analog voltage output. Features:

- Analog output
- Effective range: 4 to 30 cm
- Typical response time: 39 ms
- Typical start up delay: 44 ms
- Average Current Consumption: 33 mA

The analog voltage output is inverse to the distance measure. The analogue voltage received is converted to digital format and using a simple mathematical formula, the voltage can be converted into distance.

2.4 Servo Motor

Figure 4: Servo Motor

There are two servo motors used in the project. One for the Mapping of obstacles and other for the Gripper. Servo motors are motors which allow precise control of its angular position and speed. The angular position of motor can be controlled by PWM. The PWM range can vary from 1ms to 2ms and each value determines a particular angular position of the motor. The servomotor has three wires- Red, Black and Yellow/White. Red is the power supply, Black is the ground and Yellow/White is the PWM input to control the angular position and speed of the Motor.

2.5 Arduino Mega

The controller used is Arduino Mega 2560. It is based on the ATmega2560. It has 54 digital input/output pins which includes 15 PWM output pins. In addition, there are 16 analog inputs,4 UARTs, a 16MHz crystal oscillator, a USB connection, a power jack, an ICSP header and a reset button.

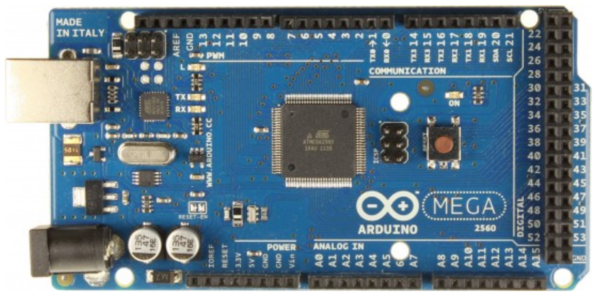

Figure 5: Arduino Mega

Features:

- Microcontroller ATmega2560
- Operating Voltage 5V
- Input Voltage (recommended) 7-12V
- Input Voltage (limits) 6-20V
- Digital I/O Pins 54 (of which 14 provide PWM output)
- Analog Input Pins 16
- DC Current per I/O Pin 40 mA
- DC Current for 3.3V Pin 50 mA
- Flash Memory 256 KB of which 8 KB used by bootloader
- SRAM 8 KB
- EEPROM 4 KB
- Clock Speed 16 MHz

The Arduino Mega can be powered via the USB connection or with an external power supply. The optimum range is between 7 to 12 Volts. The power pins are as follows:

- VIN - It is the input voltage to Arduino when it is using an external power source .
- 5V -This pin of Arduino outputs 5V from a regulator.
- 3.3V- This pin of Arduino outputs 3.3 V/50mA
- GND - These are the ground pins.

It provides control logic for the functioning of the Robot. It is mounted over with a Motor shield to increase its capabilities.

2.6 Motor shield with Motor Driver TB6612FNG

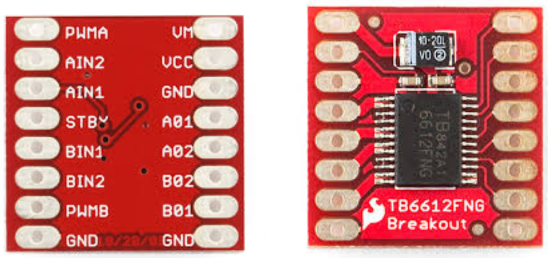

Figure 6: TB6612FNG

Features:

- Power supply voltage: VM=15V max, VCC=2.7-5.5V
- Output current: Iout=1.2A(average) / 3.2A (peak)
- Standby control to save power
- CW/CCW/short brake/stop motor control modes
- Built-in thermal shutdown circuit and low voltage detecting circuit

- All pins of the TB6612FNG broken out to 0.1” spaced pins
- Filtering capacitors on both supply lines

There are two dual motor drivers used for control mechanism. It is driver IC for DC motor with output transistor in LD MOS structure with low ON-resistor and is capable of supplying up to 13V and 1.2A. It is basically a dual H bridge. H bridge is a setup of transistors such that it allows to switch the direction of current. When connected to a motor, it allows spinning in both directions.

Pin functions:

No:	Pin Name	Function	Corresponding Arduino Pins used
1	STBY	Stand By	Pin number 10.
2	PWMA	Speed control for MotorA	Pin number 3
3	AIN1	Direction Control	Pin Number 9
4	AIN2	Direction Control	Pin Number 8
5	PWMB	Speed Control for Motor B	Pin Number 5
6	BIN1	Direction Control	Pin Number 11
7	BIN2	Direction Control	Pin Number 12

2.7 Gripper

The Robot is attached with an indigenous soft metallic gripper with required leverage powered by a servo motor. The gripper is used to grip the target and carry it. It opens to drop the target when final destination is reached.

2.8 PixyCam

Pixy cam is used in the project to detect the targets. The camera is fast and is a very efficient sensor to detect hues. It can detect

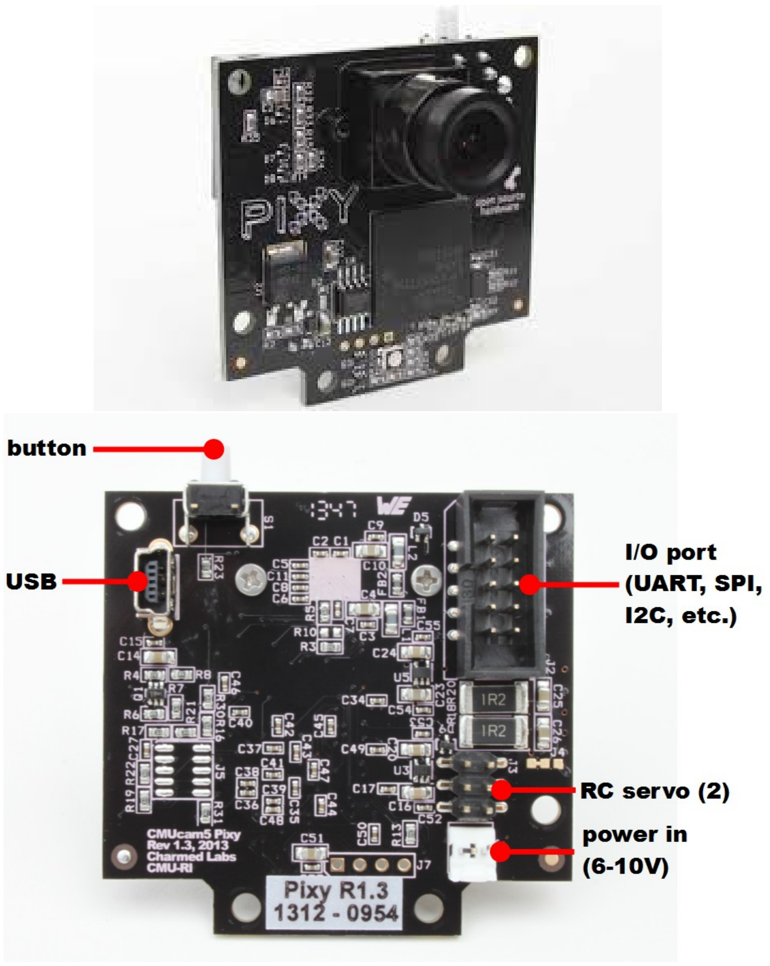

Figure 7: PixyCam

hundreds of different objects at a time and can remember up to 7 different color signatures. In this project only two color signatures are required Yellow and Red. The yellow is the color of the target and Red is color of the final destination. These colors can be easily thought to the Pixy Cam.

Technical Features:

- Processor: NXP LPC4330, 204 MHz, dual core
- Image sensor: Omnivision OV9715, 1/4”, 1280x800
- Lens field-of-view: 75 degrees horizontal, 47 degrees vertical
- Lens type: standard M12 (several different types available)
- Power consumption: 140 mA typical
- Power input: USB input (5V) or unregulated input (6V to 10V)
- RAM: 264K bytes
- Flash: 1M bytes
- Available data outputs: UART serial, SPI, I2C, USB, digital, analog
- Dimensions: 2.1” x 2.0” x 1.4
- Weight: 27gms

3. Interfacing of Motor driver with Arduino

The Motor Driver TB6612FNG can be easily interfaced with arduino. Since it is present in a motor shield, it can be directly mounted over the board. The control algorithm is very simple. For each motor there are three control pins, two for direction and one for speed When one direction pin is HIGH and other is LOW, the motor will rotate in one direction and vice-versa in opposite direction. The speed of the motor is controlled with PWM. The controller generates PWM for a specific speed and this is sent to the driver. If it is zero, the motor stops while it is 255 , it goes in full speed.

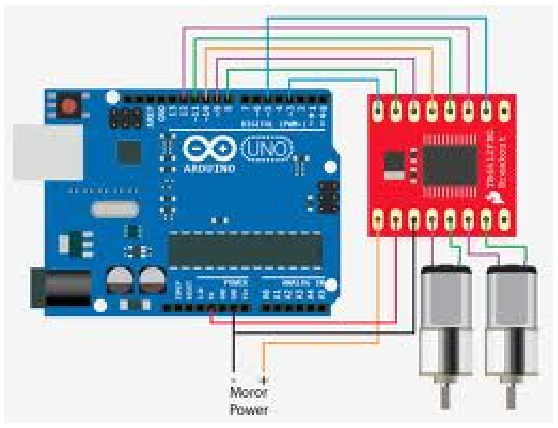

Figure 8: Motor Driver with Arduino

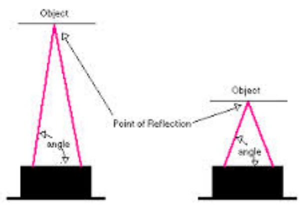

Figure 9: Distance calculation with GP2D120

4. Interfacing of GP2D120 IR Distance Sensors

IR distance sensors can simply be interfaced with Arduino. The data output of the sensors are pinned to the analog inputs of the board. The sensor has an infrared emitting LED and a lens side by side. The light emitted by the LED is sensed when it returns from an object. Based on the angle , the distance is detected. The sensor outputs analog voltages depending on the distance(but it is not linear). The Arduino reads these analog values of voltage and convert these values into actual distances using a simple formula.

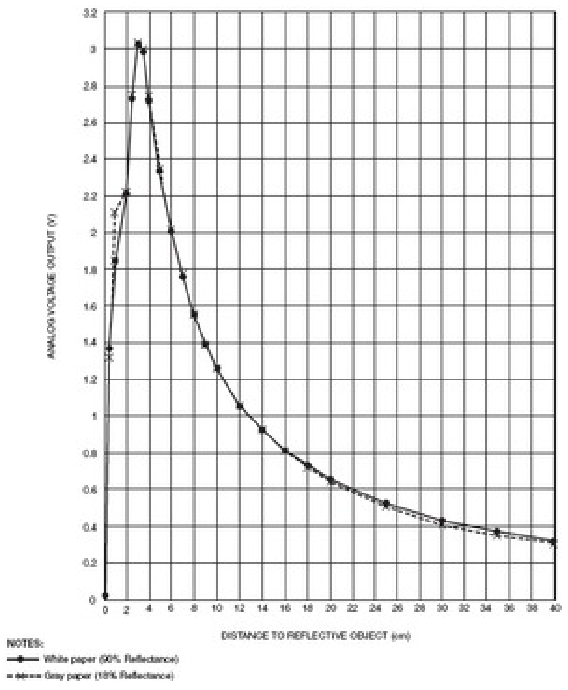

Figure 10: Distance vs Analog voltage

5. Obstacle Avoidance using IR sensors

The Obstacle avoidance is a very important part of the project. The bot has to move safely around the room without colliding with its surroundings. Hence three IR sensors are used. These sensors were mounted on three sides of the bot - Front sensor (Sensor 1), Right Sensor (sensor 2) and Left Sensor (Sensor 3).
IR sensors detects the obstacle by calculating distance of the objects around the bot. It senses distances of the objects in its range and sends it to the Arduino. The Arduino analyses the data and compares it with the limit of 20cm. If an object happens to be present at a distance less than the limit, arduino identifies it as an obstacle.The three sensors work simultaneously and the Arduino checks for the best possibility of its motion. If the front sensor happens

to detect an obstacle, it stops and sensor 2 and 3 checks for the maximum free space in either sides. The bot is given instructions to move in the direction of maximum free space.The bot continues to go forward until another obstacle is encountered.

6. Mapping of Obstacles

It is imperative for the robot to map its surrounding obstacles. The previous section was just a basic obstacle avoidance and it is not enough to give the whole picture of its surrounding. For this project, an IR sensor mounted on a servomotor is used to map front 180 degree of the robot. The IR sensor on the servo motor is used to measure the distances of any object in its view of 180 degrees. The servo motor is given necessary pwm to move to and fro 180 degrees. These values are recorded and decided for an obstacle or not. The minimum obstacle free limit is taken about 20cm of proximity. The mapped distance values are compared with the limit and put in an array as binary to have a picture of the free path The array has a size of 18 to store the obstacles distances.
The frontal view is divided into 18 parts of 10 degree precision each. After every to and fro motion , an array with a binary picture of obstacles in its front is obtained. The controller sweeps through the array and searches for the largest gap between the obstacles. Thus the most free path is determined and the commands are given such that the bot moves precisely towards that angle. The amount of turn with respect to an angle has to be determined to make exact turns. Therefore the turning has to be calibrated.
The bot will continue to move forward unless another obstacle blocks its path. Then again the IR sensor is used to map frontal 180 degrees and controller decides its next path. This goes on and vehicle continues to move without colliding with any obstacle.

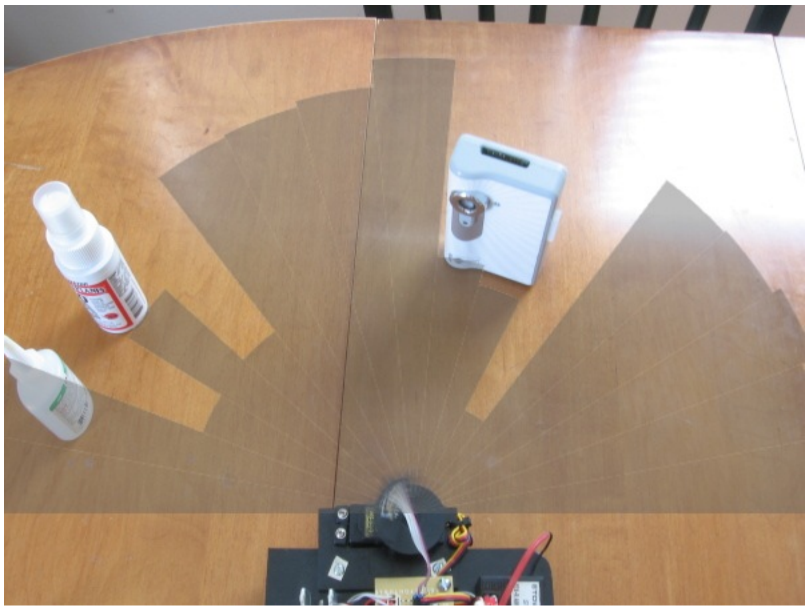

Figure 11: Mapping of Obstacles

7. Interfacing of PixyCam with Arduino

The PixyCam is an easily compatible peripheral with Arduino. In this project, the mode of communication used between the arduino and the sensor is SPI. The pin 50 (MISO),51 (MOSI), 52 (SCK), 53 (SS) of the board are the ones that support SPI communication. These pins are also broken into ICSP header. The cam is connected to ICSP pins using a connecting cable.

For this project, the pixy is thought Yellow and Red and it responds only to these colors alone. The objects can be thought very easily to the pixy. First choose an object to be thought to pixy and power it. Initially, it goes through a series of LED flashes and goes off. Hold down the button on the top and after short while led will turn on, first white, red and other colors. But when you release the

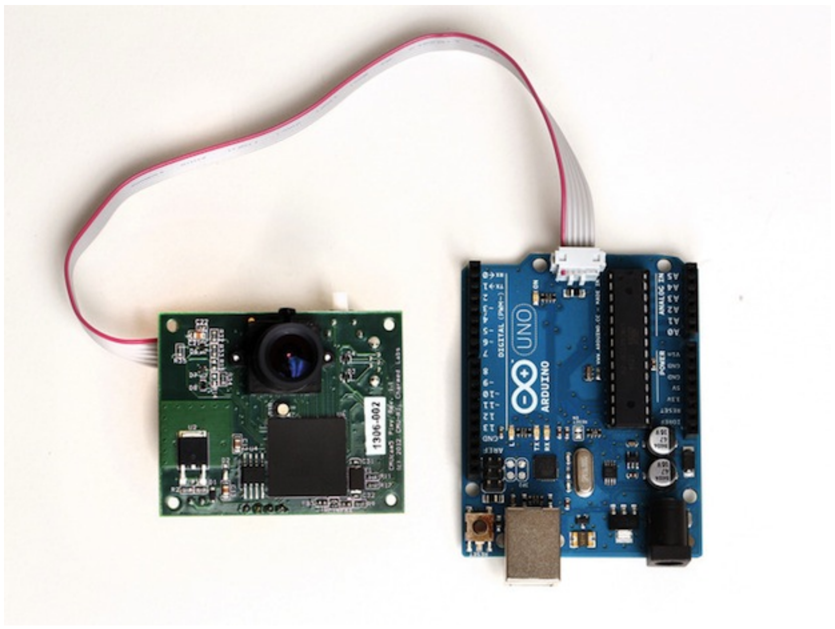

Figure 12: Pixy with Arduino

button during the process, pixy will enter ”light pipe” mode . The Led will take the color of center pixels of pixy’s image frame. The led color is like a feedback if a particular color is detected. When the led glows the required color, push and release the button at the top. If pixy realizes the hue of the color is well saturated, led will flash and the color is learned by the cam.

The colors can also be thought to pixy by connecting the camera to the computer using a mini usb cable. The Pixymon software allows us to view what pixy sees. By placing the required object of a particular hue in front of it and choosing the option to set signature, we can trace on the color to be thought. Pixy remembers the color signature and keeps track of it within its viewing range. Care must be taken to make sure that there are no other objects except those targets with similar hue in vicinity. Else the pixy might detect that too. Such situation can hinder the movement of the bot by confusing the controller during the operation.

There are other modes of communication used as well. If needed, I2C and Uart type of communication can also be available. These modes can be selected in "Data out port" parameter in the "Interface" tab in "Configure Parameters" option in Pixymon software. A particular value in "Data out port" determines the type of communication.These values are:

- 0: SPI- It is configured by default and uses 3 wires(pin 1,3 and 4 of the i/o connector)
- 1: I2C - this is a multi-drop 2-wire port (pins 5 and 9 of i/o connector), which allows single master to have control over its slaves.
- 2: UART-It is the common serial port. Pixy receives data via pin 1 and transmits via pin 4.
- 3: analog/digital x- In this mode , x value of the largest detected object is outputted as an analog value between 0 and 3.3V.
- 4: analog/digital y- In this mode , y value of the largest detected object is outputted as an analog value between o and 3.3V.

7.1 Pixy commands

While programming, the header files SPI.h and Pixy.h has to be included in the program.These header files includes the necessary commands to obtain required values from the camera sensor. The camera provides the position in X and Y; and also the height and width of the target with their color signatures.

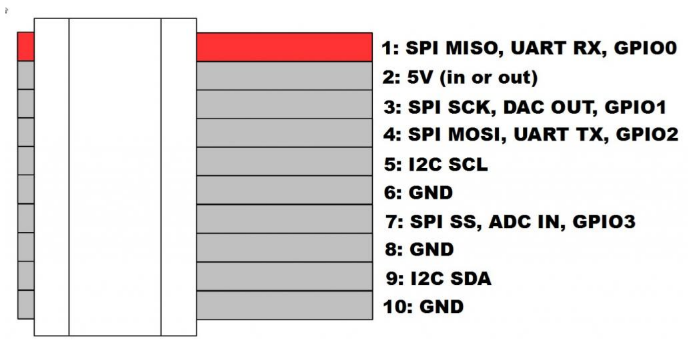
1: SPI MISO, UART RX, GPIO0
2: 5V (in or out)
3: SPI SCK, DAC OUT, GPIO1
4: SPI MOSI, UART TX, GPIO2
5: I2C SCL
6: GND
7: SPI SS, ADC IN, GPIO3
8: GND
9: I2C SDA
10: GND

Figure 13: 10 pin i/o connector

To access the information a global instance of Pixy should be made outside loop() and setup() functions. Arduino API consists of one call : getblocks() ,which returns the number of objects Pixy has detected. The pixy.blocks[] array will have information about each detected object (one array member for each detected object.) Each array member contains the following fields:

- pixy.blocks[i].signature The signature number of the detected object (1-7)
- pixy.blocks[i].x The x location of the center of the detected object (0 to 319)
- pixy.blocks[i].y The y location of the center of the detected object (0 to 199)
- pixy.blocks[i].width The width of the detected object (1 to 320)
- pixy.blocks[i].height The height of the detected object (1 to 200)
- pixy.blocks[i].print() A member function that prints the detected object information to the serial port

8. Interfacing Pixy and the Motor Driver

After Pixy was thought colors to it, attempts were done to simply maneuver the bot to follow a colored object. A red ball was taken as the colored target. Pixy has a range of 0 to 319 in x axis. So the frontal plane was divided into three equal sections- left, center and right.Hence ,if the target's x position is in first section, it moves left; second section, it moves forward; third section, it moves right. However, pixy cannot be used with Arduino UNO , if the motor driver Tb6612fng is also interfaced simultaneously. This is because of conflict in pin numbers 3 and 10 when the two are used together. But individually, both work fine with Arduino UNO. So therefore for this project Arduino Mega was a good option. With availability of more pins, the conflict was corrected.

9. Interfacing Gripper

A simple gripper with enough leverage is designed for serving the purpose of the project. The gripper is powered by a servomotor fitted mechanically at the front of the bot. The servomotor is attached to the legs of the gripper mechanically such that there is freedom of movement.

The gripper closes if the servo turns by 90 degree clockwise and opens up when it turns back to its normal position. The gripper remains closed when a target is captured and it opens only at a particular distance from the final destination to drop the target. The gripper is reliable upto a certain weight but it is best to use a soft target with enough grip.

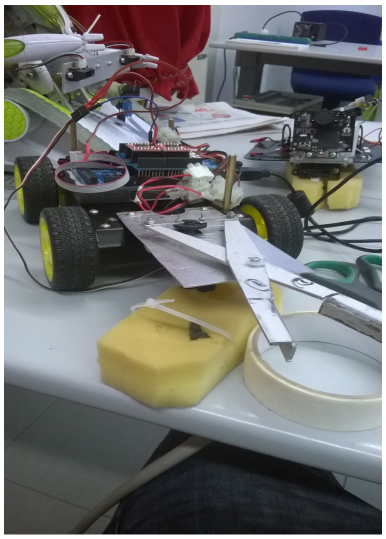

Figure 14: Indigenous gripper

10. Integration of Obstacle Mapping and Pixy Cam

The Pixy helps to detect the colored target, but it is not enough to avoid obstacles. The Cam sensor cannot detect edges or measure the distance between the bot and other obstacles. Therefore it becomes a necessity to include Obstacle Mapping along with Pixy cam. During the operation, the IR sensor on a servo motor makes a note of obstacles in its path in front , spanning 180 degrees. Until the target (yellow) object comes in sight of the cam, IR radar is the main vision and pixy works in background looking for the target. But when the target is seen, the pixy becomes the main vision of the bot. The bot moves in the direction of the target following the inputs from the pixy.

The capturing of the target is a bit tricky process since the gripper has to correctly align with the target object. For this purpose, an IR sensor is placed right in front below the joint of the gripper.

The pixy after detecting the target, leads the bot towards it . With precise movements, the object can come between the mouth of the gripper. The gripper sensor detects the object up till a particular distance and sends information to close the mouth. Hence capturing the object.
After the object is captured, IR radar leads the bot in its path. When the Final hued destination (red) is visible, again the pixy takes over as the main vision and guides towards it. At a particular distance, the target is dropped and bot turns around. All throughout the process, both the IR radar and Pixy works throughout but the priority changes as each step is achieved.

11. PID Algorithm and Low Pass Filter

To make the whole capturing process more precise, a PID algorithm and a Low Pass filter was incorporated into the software. The X position values coming from the pixy is passed through this filter to reduce jitters and to create delayed response of the fast moving data.The filter uses a buffer variable and limits the new data coming from the sensor each time. Finally the values don't vary much and stable response is observed from the sensor.
These values are fed to a PID algorithm which calculates precise amount of time to keep moving left or right with constant speed such that the target object remains in the center of the pixy frame. The output of PID is basically a measure of time. The Proportional term finds the error of the actual postion to the desired error. The Integral term adds up the error overtime and it helps the system to converge to a set point faster. It eliminates the residual error developed from the proportional term. The derivative term reduces the oscillations about the set point. It improves the settling time and over all system behavior.
Therefore the PID output along with stable sensor values helps in

precise capturing of the target. This drastically alters the motion of the bot for good.

12. The Software Description

Basically ,the whole code is mainly divided into functions for IR sensor scanner , deciding the Direction to move ,Pixy cam scan ,PID algorithm, Low Pass Filter, Distance calculation for IR sensors and various motor control mechanism functions to move forward, backward, left and right. In addition to these functions , there are functions for servomotor control for IR scanner and control mechanisms for Gripper.

12.1 Obstacle Scanning

Initially when powered, the arduino sends the instructions for an IR scan. The IR sensor on servo motor makes a scan of 180 degrees to and fro and looks for obstacles in front of it. An array of size 18 has the information of obstacles in form of binary.This array is passed to the function that decides the direction to move after a scan. After a sweep of the array, the largest gap is determined and its middle position is found. This mid position will correspond to a degree with respect to the current position of the bot. Suppose if the gap is between bit 3 and 11 (bit 4,5,6,7,8,9,10 are zeros) , the mid position correspond to bit 7 and its angular position,ie,if 18 bits together is equal to 180 degrees, bit 7 will correspond to 70 degrees. But always the bot is at 90 degrees. So therefore the bot just have to move just 20 degrees to the left. Thus a calibration for the time for movement by 10 degree(minimum angle) has to be determined. After this value is known, the total time for the whole motion will be the product of " the angle to move" with the "time for moving 10 degrees" for a constant speed. This time is passed on to the motor control functions for the required motion. Here time is taken as the

parameter to decide for the turns. It is not a reliable means since the speed changes as the batteries discharge reducing the voltage, the calibrated values will also change eventually. Since the reference also changes with motion , it becomes difficult to keep track of the original angle. Therefore, time remains the only parameter in this case though not reliable always but care must be taken to have constant voltage.
After this the bot checks for obstacles in exact front and goes on moving forward until it detects.Once detected , it again scans and the process is repeated again.

12.2 Pixy Scan

The Pixyscan function is called each time Forward function is iterated. This is done so as to look if the target is present in front view. If not present, it returns nothing and bot continues to move under IR vision. If a target is found, the vision switches to Pixycam. The x position of the target is sent to the arduino. These values are passed through a low pass filter to have a stable response and a simple PID algorithm for precision of motion. The Motor control functions used here are same as that used for Obstacle scanning so therefore a parameter of "time" is required here too for the motion. Hence the PID output was scaled to have this measure of time for a particular constant speed. The PID output could be converted to have PWM value too but in this case the time for the PWM run is not defined so having the output as a measure of time was a better choice. This output was passed to the motion control functions for the necessary motion.Eventually, the bot is led towards the target such that gripper mouth happens to be around the target. There is an IR sensor fitted below the gripper to continuously measure the distance of the target to capture. If the required limit is reached the gripper closes and the target is captured. Again it goes back to

obstacle scanning until the Red destination is seen. If it sees Red destination, it moves towards it similarly and drops the target.
The calibration of time is quite tricky part in this case. Since the x position ranges from 0 to 319, 160 was defined as the center. But to know how much time does the bot need to reach a particular position for the object to be at its center with the constant speed, a target was fixed at a position in front of the bot and its position was noted via serial monitor and an another program was written where just pixy vision was used with simple logic to move just left or right. If the object happens to be at x position less than 160, then left or vice versa. Hence the position of the object was noted, suppose 50.The time parameter in the motor control function was changed until the desired alignment with object was obtained. After that, the PID algorithm was included to this program and its output was viewed.The objective was to find a scale that has to be multiplied with the original PID output such that it equals the required "time". By some trial and error, a value of "2 "for the scale was found to sufficient. Initially, only P term was used and Kp was taken as 1. After getting the scale, I and D terms were introduced and the constants Ki and Kd was found as 0.5 and 0.7 respectively with some more trials.

13. Limitations

Since both the scanning methods involves dependency on the "Time" parameter, some limitations were found during testing of the bot. It was found that as the battery discharges and the voltage falls, the motor tends to move with less pace. Hence the initial calibrated value of time changes with the falling voltage and the system starts responding non ideally. Care must be taken to ensure that the voltage to the entire system remains constant throughout.Also since the arduino board powers all the peripherals, enough voltage should be supplied for it. Typically in the range of 7-12 V. Otherwise, it wont be able to distribute the power properly to all the sensors and this may affect the overall system behavior.

14. Conclusion

The PixyBot is able to achieve its objective quite satisfactorily. There can be many add ons to this project and can be taken to a further level. The Pixy cam is very efficient in its functioning and has numerous possibilities in the field of Robotics. This new product is new bench mark in image processing and its ability to interface with Arduino is a great leap. The PixyBot can be improved further and put into different applications. This project which deals with its application in Robogames is only a beginning to many more yet to come.

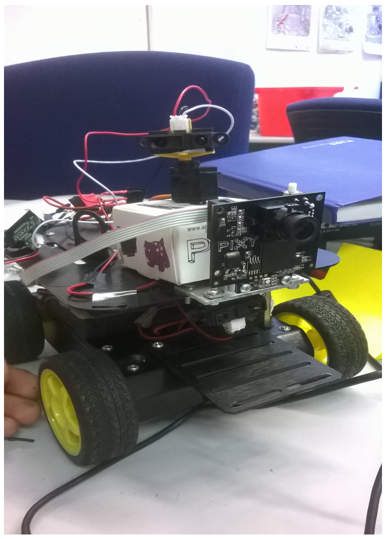

Figure 15: PixyBot