



Robotic Arm

With
Image Processing
&
Deep Learning

BCA 3rd Year Major Project

DECLARATION

“I declare that this report entitled **“Robotic Arm With Image Processing & Deep Learning”** is the result of my own research except as cited in the references. The report has not been accepted for any degree and is not concurrently submitted in candidature of any other degree.”

Signature: _____

Name: ANKAN POLLEY _____

EJAZ AHMED _____

SOURAV DAS _____

VISHAL SAMANTA _____

Date:

CERTIFICATE OF TRAINING

This is to certify that Industrial Training Report entitled
**“ROBOTIC ARM WITH
IMAGE PROCESSING & NEURAL NETWORK”**

Is a record of bona fide work done by

ANKAN POLLEY

EJAZ AHMED

SOURAV DAS

VISHAL SAMANTA

under my supervision and guidance. This project is submitted to from

ARDENT COMPUTECH PVT.LTD



in partial fulfillment of the requirement for the award of the Completion
Certificate in “MAJOR PROJECT” during the year 2017-2018.

The matter embodied in thesis is original and has not been submitted for the
award of any other course.

ACKNOWLEDGEMENT

The sincerest thanks and gratitude to the Al-Mighty who has eased the difficulty and cleared the ambiguity for me to fulfill this achievement.

First of all, I would like to say that it has been my honor to be a student of and also, I would like to thank Mr. Kaustav Chatterjee, my final year project supervisor, who is for the guidance, support and encouragement he provided to me throughout the entire duration while I was doing and preparing my biggest project yet.

I also would like to take a moment thank all my academic supervisors for the support and guidance, they showed us throughout my three years as a Bachelor of Computer Application student whenever I needed them.

Last but definitely not least, I would like to thank my Mother, Father, family member and friends for the constant encouragement and constant support they showed me throughout my entire period as a university student which helped me to keep going and never give up.

From the bottom of my heart THANK you all very much.

ABSTRACT

This paper presents implementation of **“Robotic Arm With Image Processing & Deep Learning”**. This framework is intended to give help to industries which require human labors to do dangerous jobs like bomb disposal, Glass manufacturing industry, acid & chemical industry etc. Additionally, the idea of using a robotic arm to replicate a human arm using image processing to replicate human eyes. The fundamental control system uses Servo motors, 3D printed robotic arm and Image processing & machine learning. The system design does not use any manual control but uses the image processing to automate all the task in hand. The robotic arm status is totally automatic using the program in the Arduino board connected to it as well as the image processing. The only human level commands required are high level ones.

INDEX

	Topic	Page
1.	Acknowledgement	4
2.	Abstract	5
3.	Introduction	7
4.	Sequence Diagram	8
5.	Components used in our projects	9
6.	Arduino UNO Board	10-11
7.	SG-90 Servo Motors	12-13
8.	3D Printed Robot Arm	14-15
9.	Electrical Equipment's	16-17
10.	Circuits	18
11.	Hardware - Mechanics	20-22
12.	Software - Work Flow	24
13.	Software - Processing Java	25-26
14.	Software - Communication	27-28
15.	Software - Arduino in C++	29-30
16.	Inverse Kinematics	31-32
17.	Inverse Kinematic Engine	33
18.	Image Processing	34
19.	Deep Neural Network	35-36
20.	Bringing it all together	37
21.	Snaps of our project	38
22.	Future scope and Application	39
23.	Costs	40
24.	Conclusion	41
25.	Bibliography	42

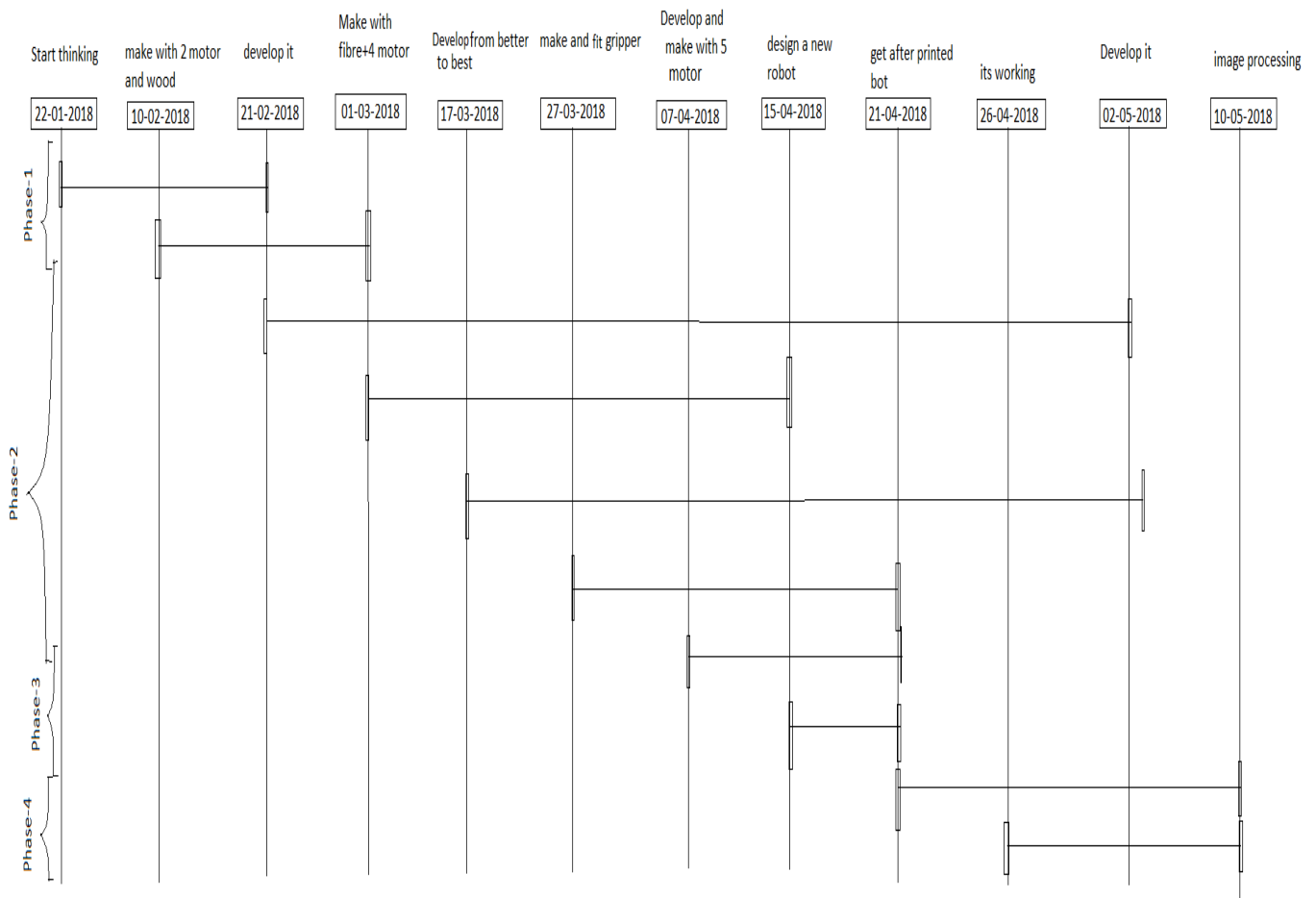
INTRODUCTION

A robotic arm is a device, which enables the holding of an object to be manipulated. Just like a human hand, a gripper enables holding, tightening, handling and releasing of an object. With the use of intelligent mechanisms, an underactuated robot hand can be developed. Passive elements are used to kinematically constrain the finger and ensure the shape adaptation of the finger to the object grasped.

In our research we found that robotic arms are simply used as slaves and needs a human counterpart to do their work.

Thus, in this project of ours we have tried to gift the robotic arms with their own eyes and brains i.e., machine language. We made it possible by using inverse kinematics on the movement of the gripper, image processing to help it detect a certain object that it needs to pick up or help with a certain work, and deep neural networking to help it think what to pick and what to do and what not to do.

SEQUENCE DIAGRAM



COMPONENTS USED IN OUR PROJECT

➤ **Hardware**

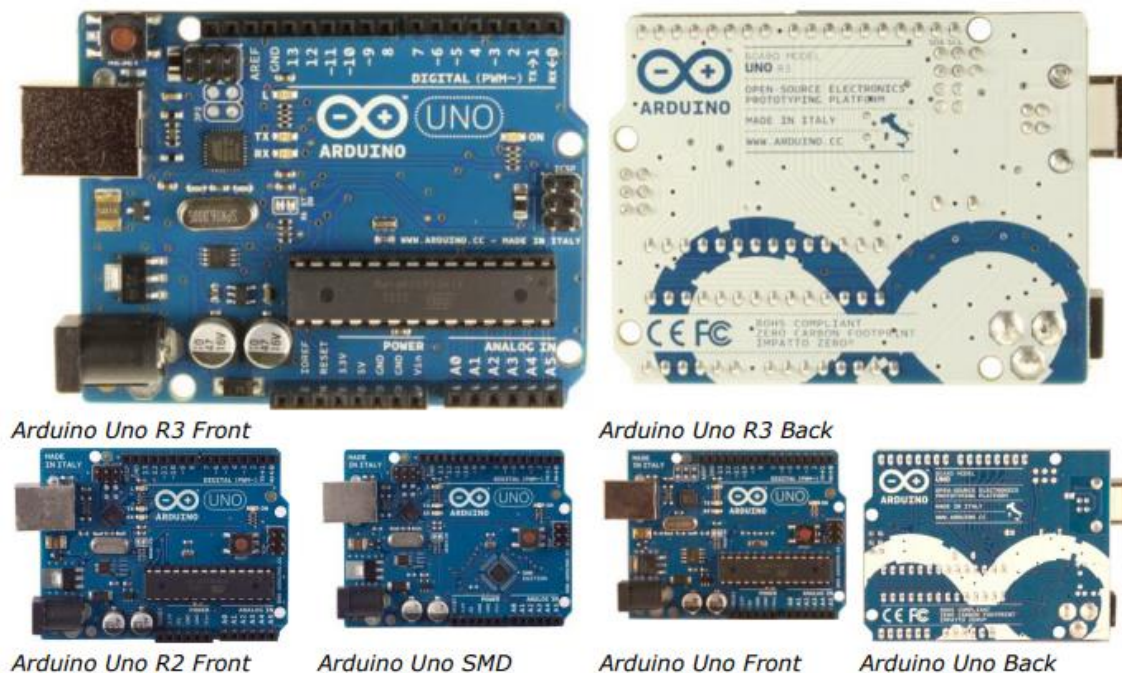
- 1 - Arduino Uno Board (Fake one)
- 4 – SG90 Servo Motors
- 1 - Robotic Arm
- 1 - Bread Board
- 1 - Camera
- Many Utilities
 - Jumper Cable
 - Resistance
 - Led

➤ **Software**

- Arduino IDE
- Processing IDE
- OPEN CV
- Talker

ARDUINO UNO BOARD

The Arduino Uno is a microcontroller board based on the ATmega328. It has 14 digital input/output pins (of which 6 can be used as PWM outputs), 6 analog inputs, a 16 MHz ceramic resonator, a USB connection, a power jack, and a reset button. The Uno differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega16U2 (Atmega8U2 up to version R2) programmed as a USB-to-serial converter.



- 1.0 pinout: added SDA and SCL pins that are near to the AREF pin and two other new pins placed near to the RESET pin, the IOREF that allow the shields to adapt to the voltage provided from the board. In future, shields will be compatible both with the board that use the AVR, which operate with 5V and with the Arduino Due that operate with 3.3V. The second one is a not connected pin, that is reserved for future purposes.

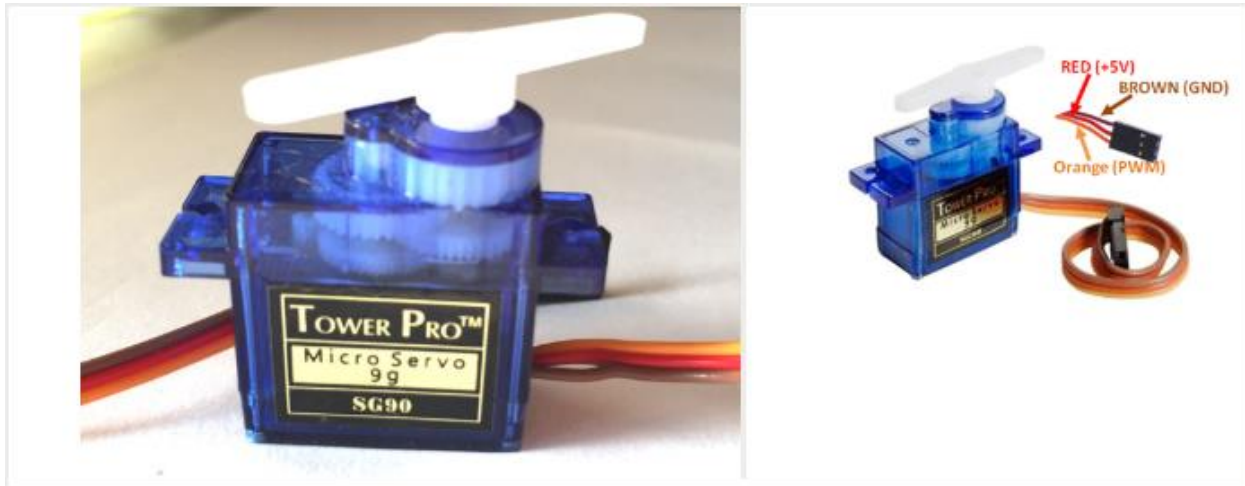
- Stronger RESET circuit.
- Atmega 16U2 replace the 8U2.

Specification of Arduino:

Arduino Parts	Specifications
Microcontroller	ATmega328
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	6
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)
Clock Speed	16 MHz

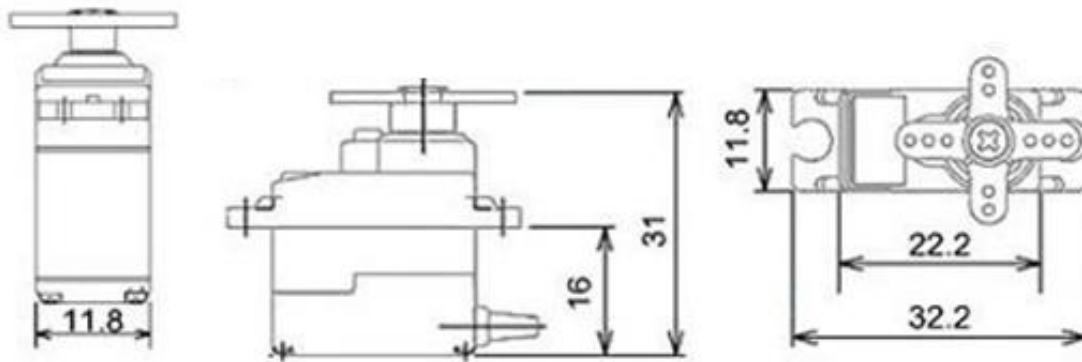
SERVO MOTOR SG-90

There are lots of servo motors available in the market and each one has its own speciality and applications.



Most of the Servo motors operates from 4.8V to 6.5V, the higher the voltage higher the torque we can achieve, but most commonly they are operated at +5V. Almost all hobby servo motors can rotate only from 0° to 180° due to their gear arrangement so make sure you project can live with the half circle if no, you can prefer for a 0° to 360° motor or modify the motor to make a full circle. The gears in the motors are easily subjected to wear and tear, so if your application requires stronger and long running motors you can go with metal gears or just stick with normal plastic gear.

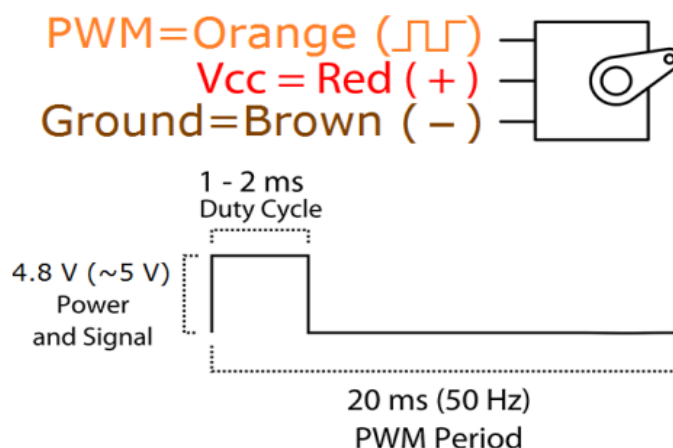
Next comes the most important parameter, which is the **torque** at which the motor operates. Again, there are many choices here but the commonly available one is the 2.5kg/cm torque which comes with the Towerpro SG90 Motor. This 2.5kg/cm torque means that the motor can pull a weight of 2.5kg when it is suspended at a distance of 1cm. So, if you suspend the load at 0.5cm then the motor can pull a load of 5kg similarly if you suspend the load at 2cm then can pull only 1.25.



Tiny and lightweight with high output power. Servo can rotate approximately 180 degrees (90 in each direction) **and** works just like the standard kinds but smaller. You can use any servo code, hardware or library to control these servos. Good for beginners who want to make stuff move without building a motor controller with feedback & gear box, especially since it will fit in small places. It comes with a 3 horns (arms) and hardware.

Specifications:

Weight	9g
Dimension	22.2 x 11.8 x 31 mm approx.
Stall torque	1.8 kg/cm
Operating speed	0.1 s/60 degree
Operating voltage	4.8 V (~5V)
Dead band width	10 μ s
Temperature range	0 °C – 55 °C



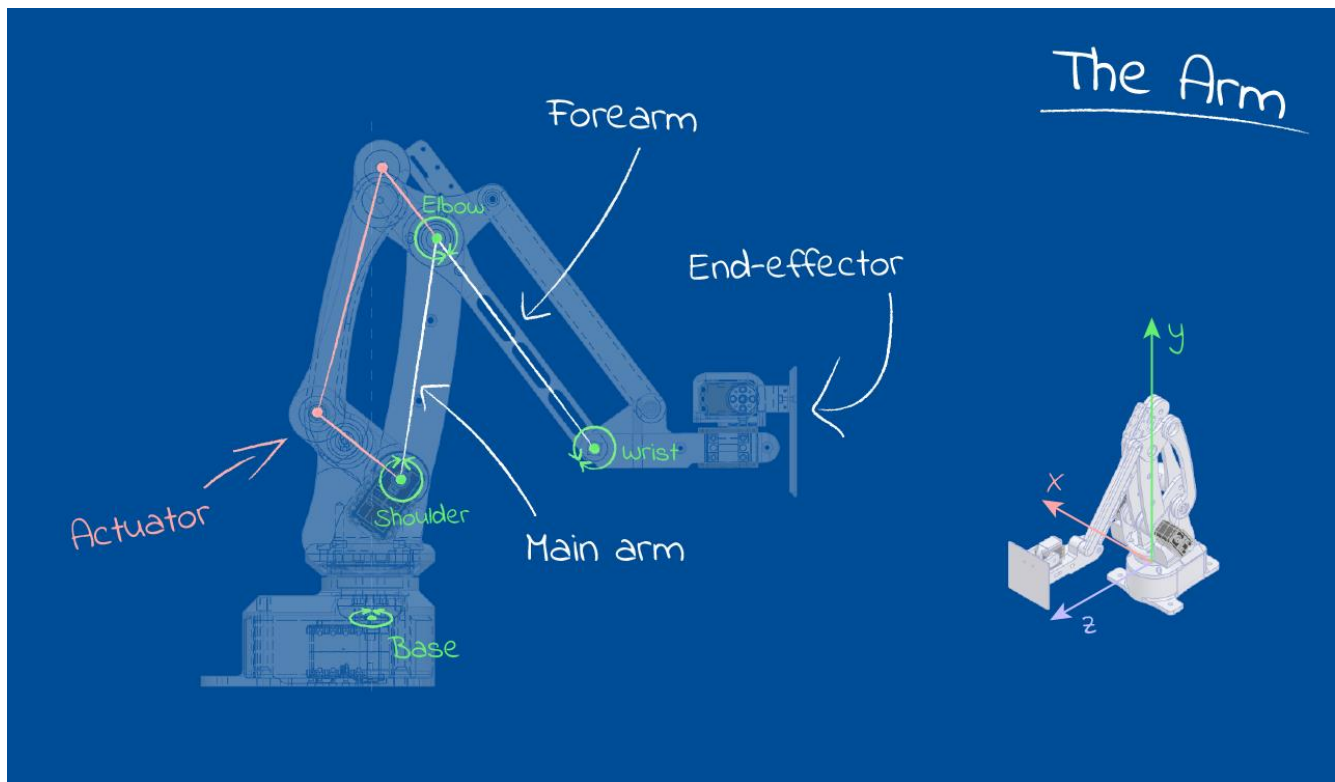
3D PRINTED ROBOT ARM







Part list:

1. 20 - 3D printed parts
2. 1 - Vertical Drive Arm
3. 1 - Forward Drive Arm
4. 1 - Horizontal Arm
 - 1 - Triangular
 - 2 - Servo Plate
1. - Basement
2. - Round Plate
3. - Claw Support

- 4. - Right Finger
- 5. - Left Finger
- 6. - Drive Gear
- 7. - Driven Gear
- 8. - Ramp
- 9. - Maestro Holder
- 10.- Ball
 - 3 - SG90 servo (gripper)
- 5. 15 - M4 washers
- 6. 7 - M3 nuts
 - 1 - M3 x 30 screw
 - 2 - M3 washers
 - 3 - M3 x 12 hex screw
- 1 - M3 x 12 TCEI screw
- 2 - M3 x 20 TCEI screw
- 3 - M4 x 20 round head hex recess screw
- 7. - Brass pipe 4 x 3 x 22 + n°1 4 x 3 x 26



ELECTRICAL EQUIPMENTS

Equipment	Image
Bread Board	
Resistance	
Jumper Wires	
Camera	

LED



Arduino Data Cable



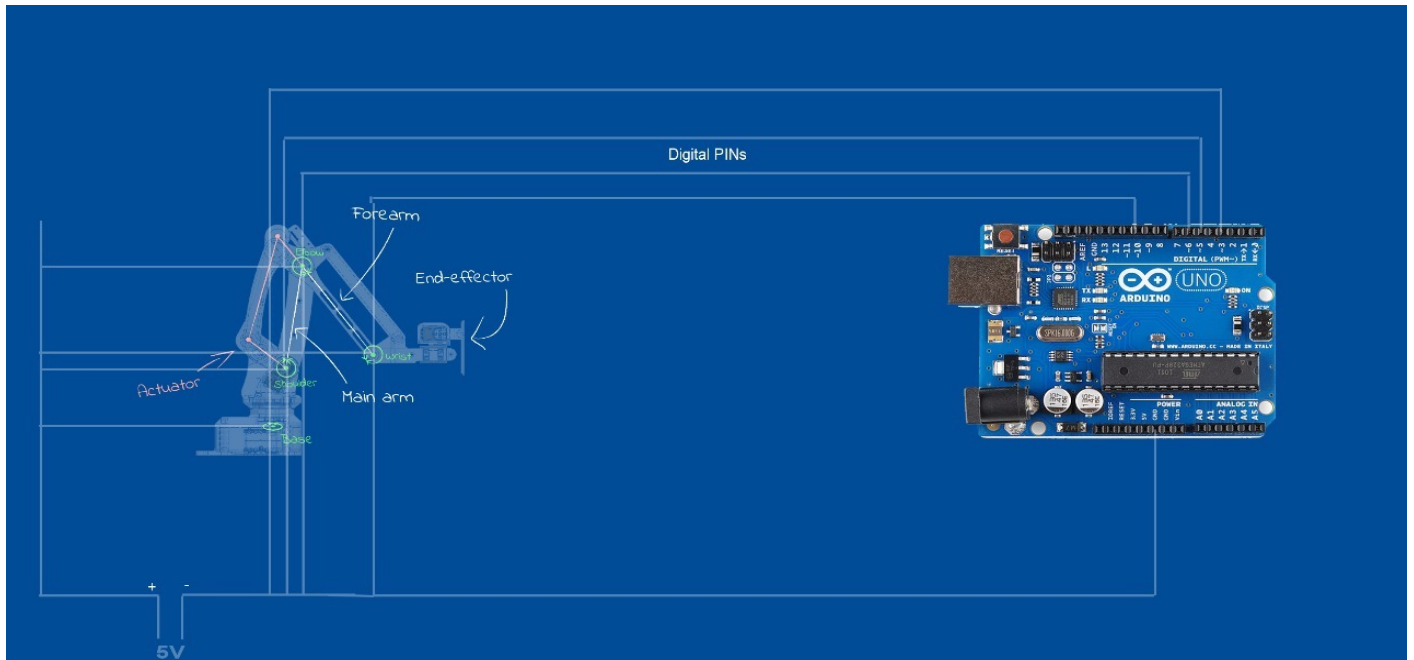
Simple data cable



Adapter



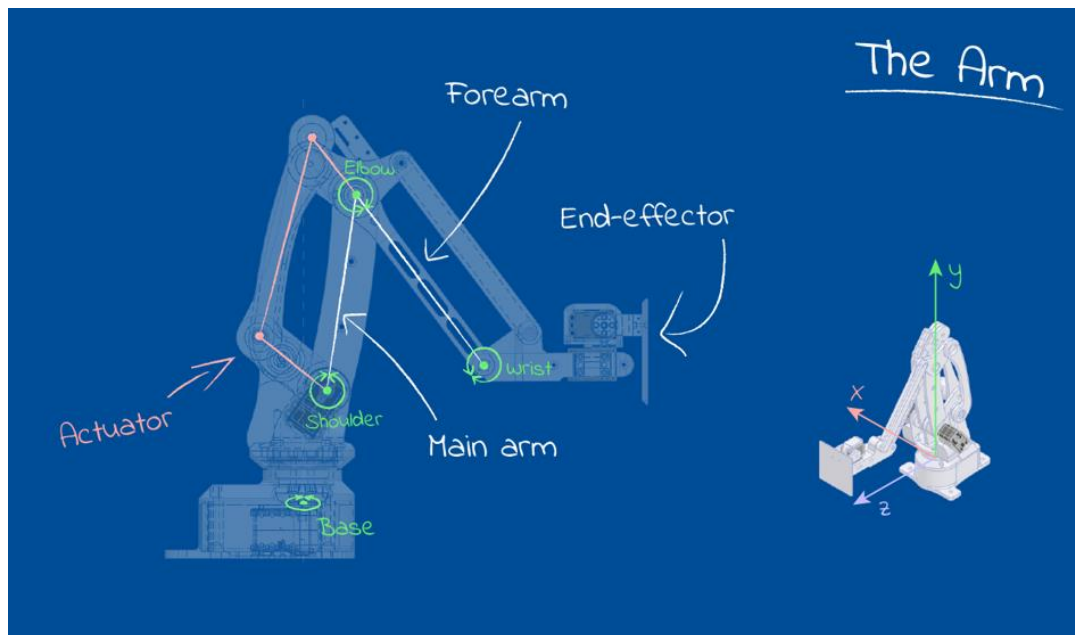
CIRCUITS





HARDWARE

M ECHANICS

➤ Designing:



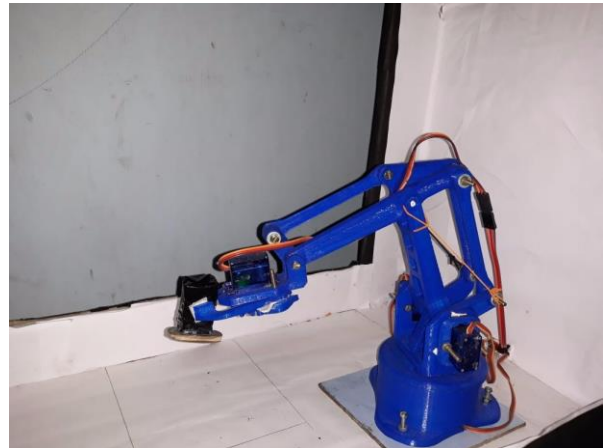
Mechanical Assembly -

Phase	Description	Drawbacks	
Phase 1 (Robotic arm made by using wooden Ice-cream sticks)	In this phase, we made a robotic arm by assembling parts made up of wooden Ice-cream sticks. This prototype used 4 servo motors to move.	This prototype looked good and could move, but it couldn't lift much load and increase in numbers of ice cream sticks to increase its reach made it hard to move.	
Phase 2 (Robot arm made by using fiber plates)	In this phase, the prototype was made by using fiber plates.	Same as the previous model it wasn't able to cover the desired length.	
Phase 3 (An extra servo was added to the fiber prototype of the robotic arm)	In this phase, an extra servo and extra plate of fiber was used by us to try and increase the area of coverage by the robotic arm.	But the increase of an extra servo and extra fiber made it heavier and couldn't move its own body.	

Phase 4
(We 3D
printed a
robotic
arm)

We had to start
from the
scratch and
went back to
using 4 servo
motors for less
power
consumptions
and replaced
the 5th servo
with an
Actuator and
made the 3D
printed robotic
arm as light as
possible.

We printed
the whole
Arm with 3D
printer , but it
came with
some printing
mistake as
well as it is
very hard to
find its
metallic parts
in any market.



But, successfully we have solved many errors and are using it with minor errors

SOFTWARE

WORK FLOW

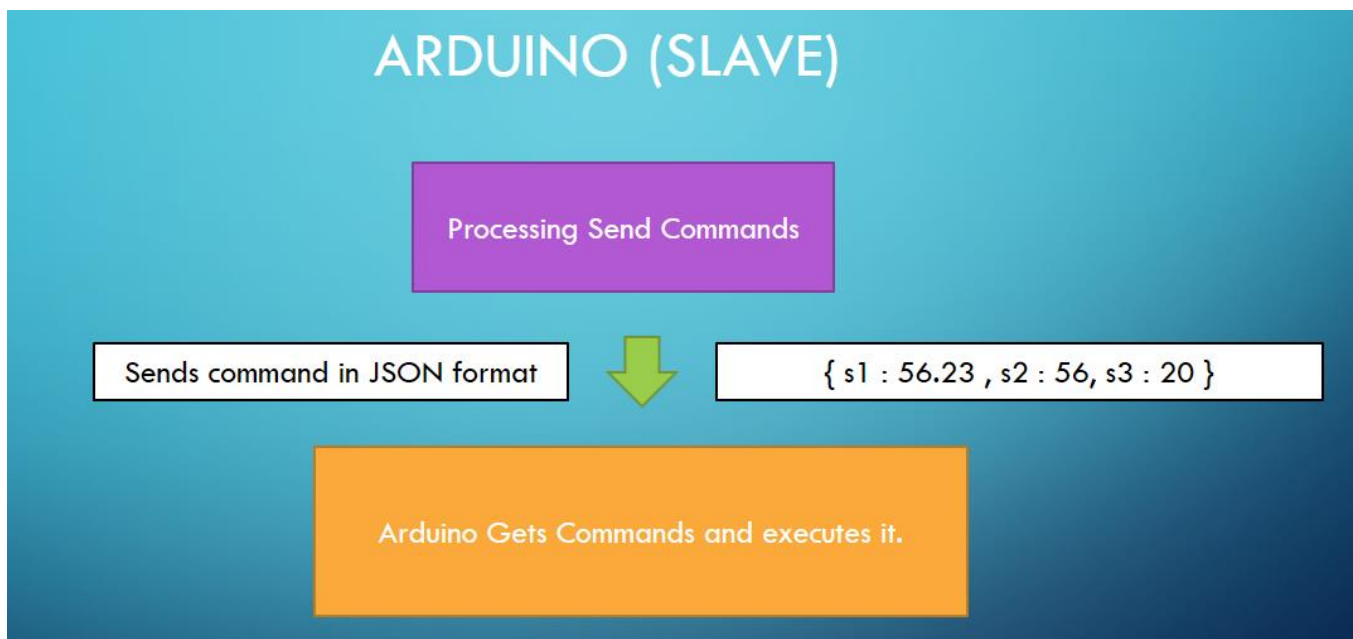
The entire system of processing and Arduino works on the Master and Slave architecture. Processing 3 being the Master gives commands to Arduino with the board mindlessly carries out.

Arduino can only receive servo inputs and validate them and relay them to the servo motors.

Processing 3 on the other hand can a lot more

e.g.

- Image processing
- Inverse Kinematics
- Coordinate mapping
- Error reduction
- Error smoothing



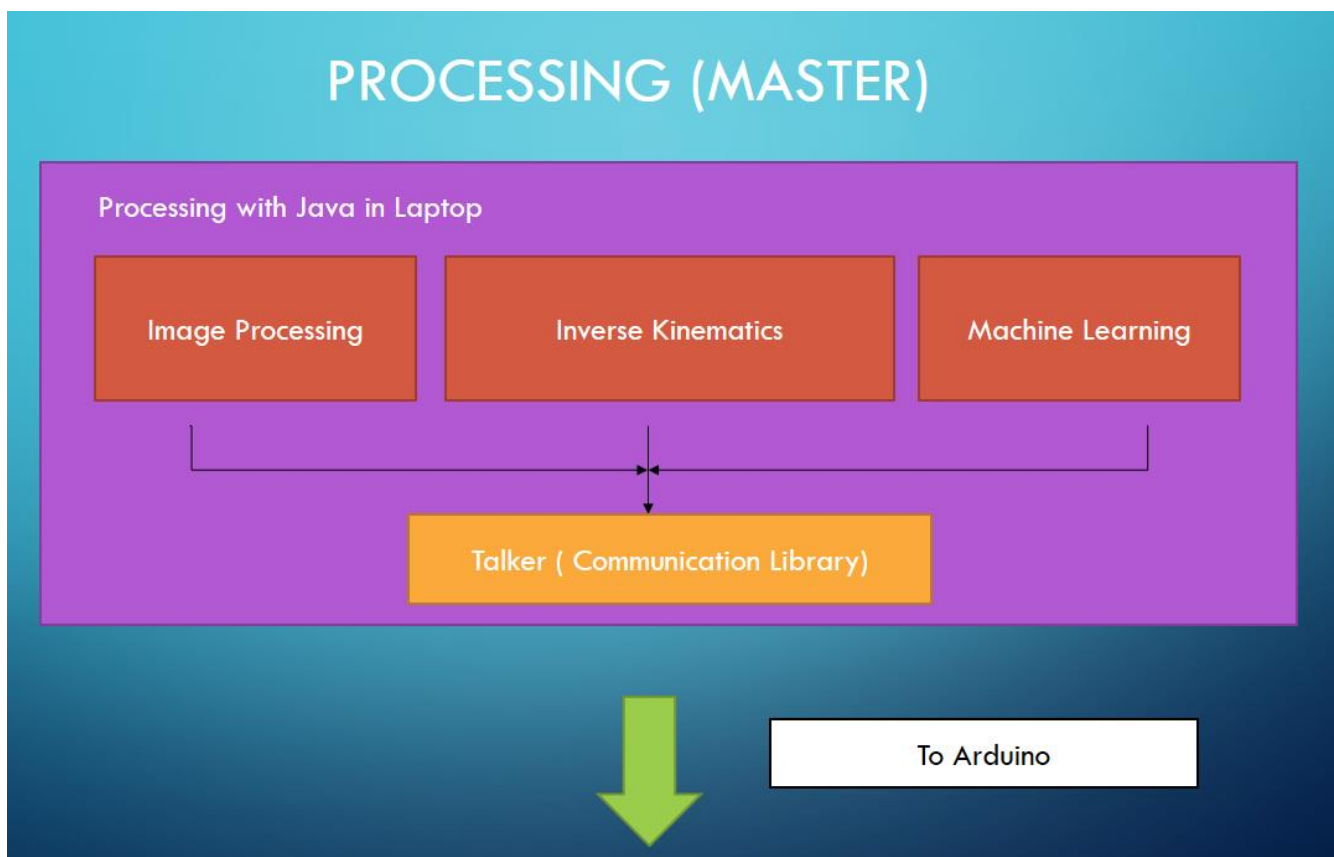
PROCESSING JAVA

Master (Processing in Java)

The processing Programing is used to compute and collaborate the working of the robotic arm.

For Example:

- The calculation of the servo angle from the coordinate provided by the user.
- For processing of image to find out the coordinates of the object.



CODE EXAMPLES

```
void setup()
{
  size(700,400);
  talker = new Talker(this,28800,"COM8");
  talker.waitUntilConnect();
  arm = new Arm();
  createGUI();
  initIP();
}
```

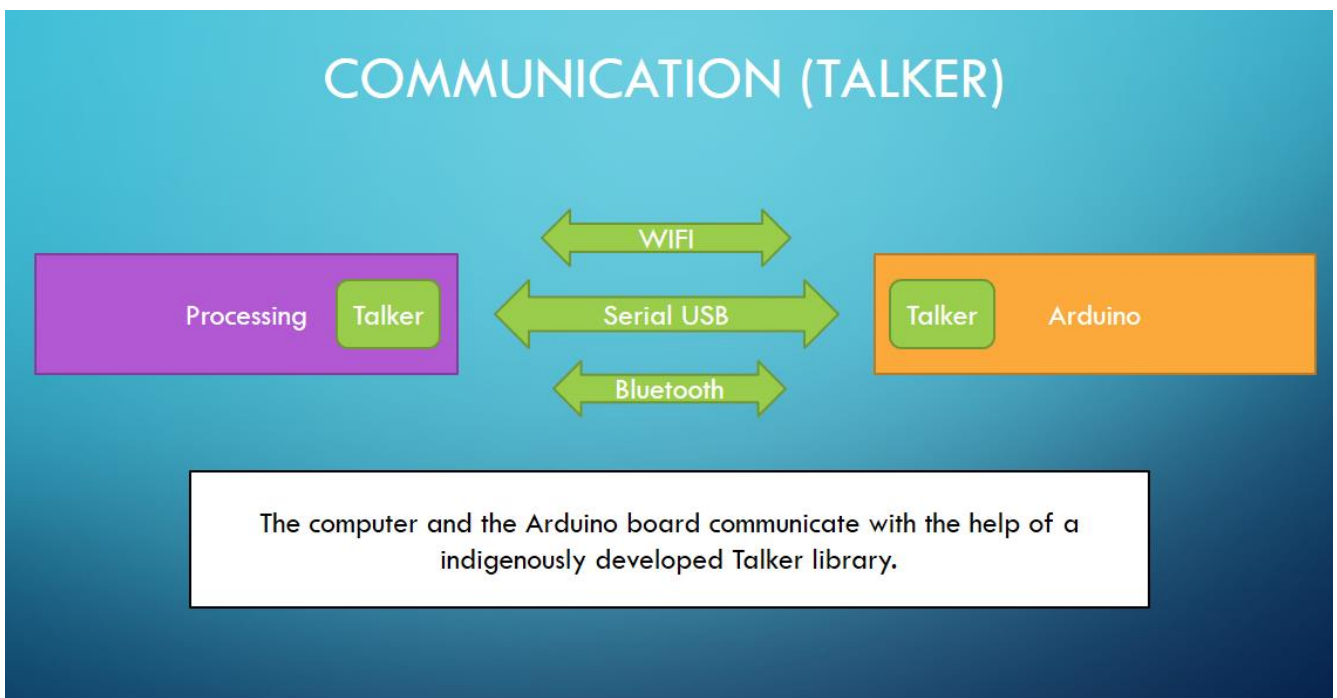
```
void draw()
{
  background(128);
  if( talker.available() > 0)
  {
    val = talker.readStringUntil('\n');
    if (val != null )
      print ("Received: “, val);
  }
}
```

```
float[] solve(float x,float y ,float z)
{
  //x = x + 70;
  if(x == 0 || y == 0 || z == 0)
  {
    return null;
  }
  double s1 = segments[0];
  double s2 = segments[1];
  double s3 = segments[2];
  double theta1 = atan(y/x);
  double theta3 = Math.acos(( Math.pow(s2,2) +
    Math.pow(s3,2) - pow(x,2) - Math.pow(y,2) - Math.pow(s1 - z,2))
    /(2*s2*s3));
  double alpha = Math.acos(( Math.pow(s1,2) -
    Math.pow(z,2) + Math.pow(s1 - z,2))/(2*s1*Math.sqrt(Math.pow(x,2) +
    Math.pow(y,2) + Math.pow(s1 - z,2))));
  double beta = Math.acos(( Math.pow(s2,2) - Math.pow(s3,2) + Math.pow(x,2) +
    Math.pow(y,2) + Math.pow(s1 - z,2))/(2*s2*Math.sqrt(Math.pow(x,2) +
    Math.pow(y,2) + Math.pow(s1 - z,2))));
  double theta2 = alpha + beta;
  theta1 = (double)Math.toDegrees(theta1);
  theta2 = (double)Math.toDegrees(theta2);
  theta3 = (double)Math.toDegrees(theta3);

  float[] t1 = {(float)theta1,(float)theta2,(float)theta3};
  return t1;
}
```

COMMUNICATION

The communication between the Computer (Processing Programming) and the Arduino Board is done via the **TALKER** library. This library is indigenously made by our team and works with different communication devices.



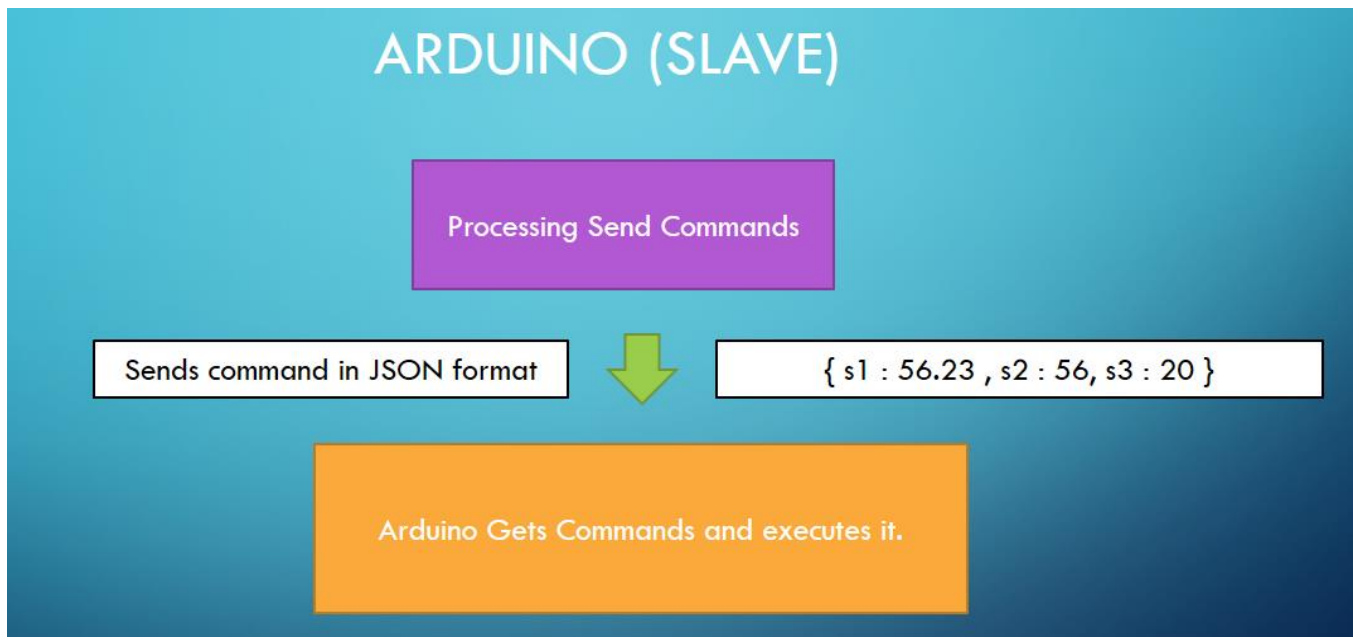
CODE EXAMPLES

```
Talker talker;  
talker = new Talker(this,28800,"COM8");  
talker.waitUntilConnect();  
  
if(talker.available() > 0)  
{  
    val = talker.readStringUntil('\n');  
    if(val != null )  
        print("Received : ",val);  
}  
  
msg = "Hi";  
talker.send(msg);
```

ARDUINO C++

➤ **Slave** (Arduino in C++)

The Arduino board and programming functions as a slave where the receives commands from its master (Processing) to executes it tasks.



Code Example

```
#include "Arm.h"
#include "Talker.h"

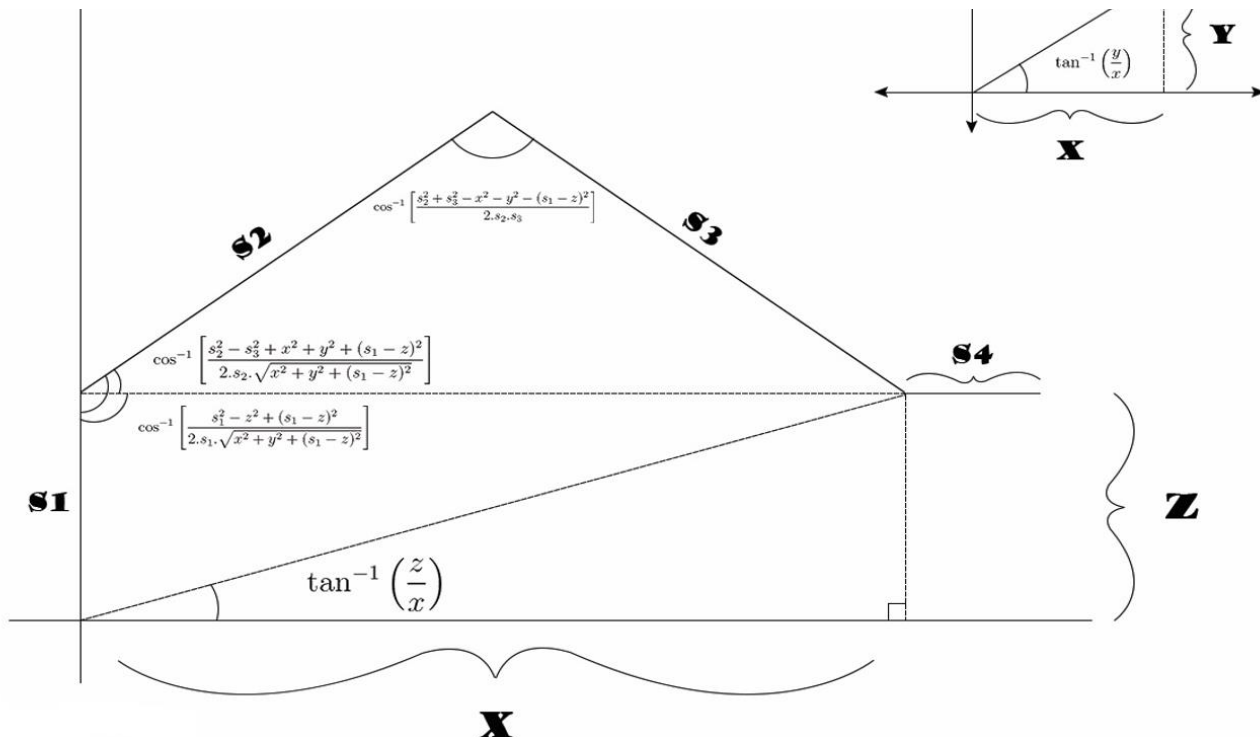
Arm arm;
Talker talker;

void setup() {
    talker.begin(28800);
    arm.init();
    arm.restPos();
    if( talker.connect() ) talker.send("Connected From Board.");
}

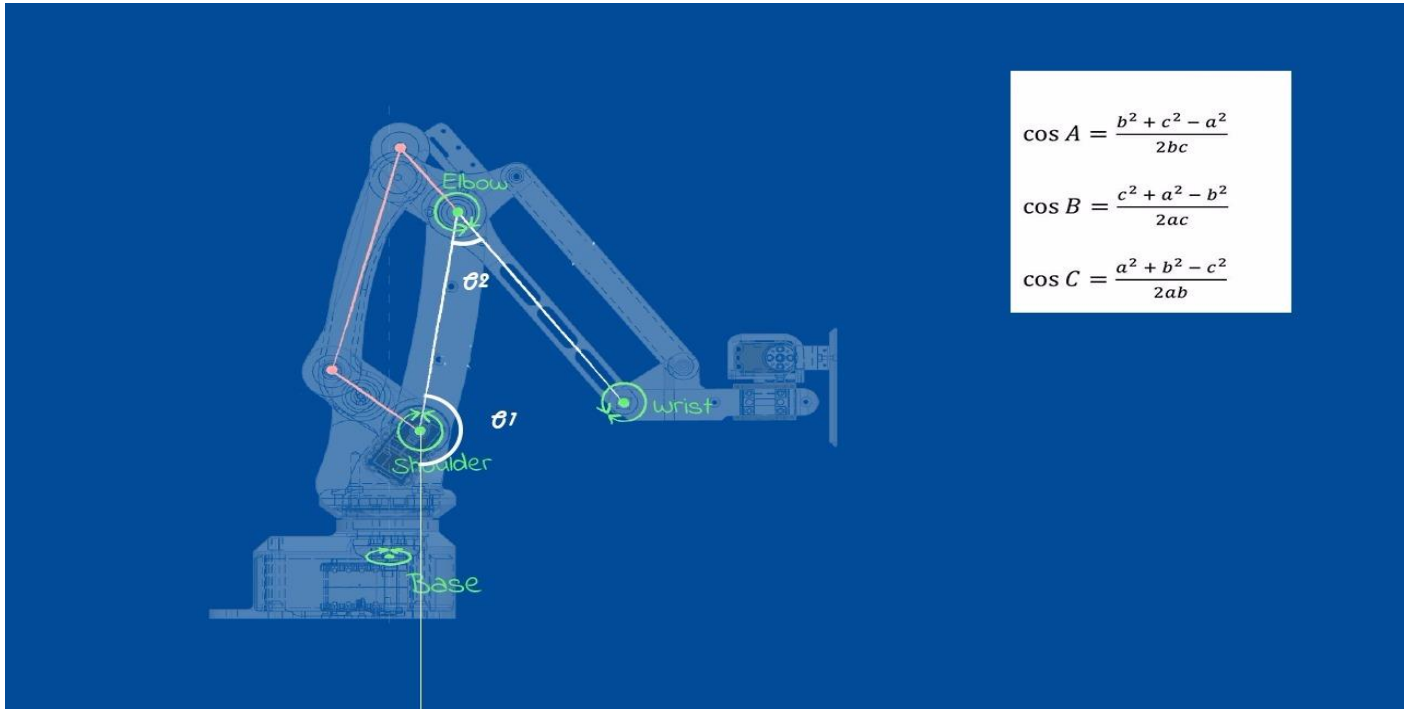
void loop()
{
    if( talker.available() )
    {
        talker.send(" ");
        String s = talker.getStringUntil('\n');
        arm.execute(s);
        talker.send(s + "\n" +arm.repr() );
    }
    //arm.refresh();
    //delay(2000);
}
```

INVERSE KINEMATICS

Inverse kinematics (IK) is a very powerful tool in game development. Its origins come from Robotics, but it's found a place in many computer and science fields. The basic concept behind IK, which we'll go over in more depth later, is to calculate positions for a joint system so that it will reach a certain end goal. This has immense use in things such as animation, body rigging, and so forth.

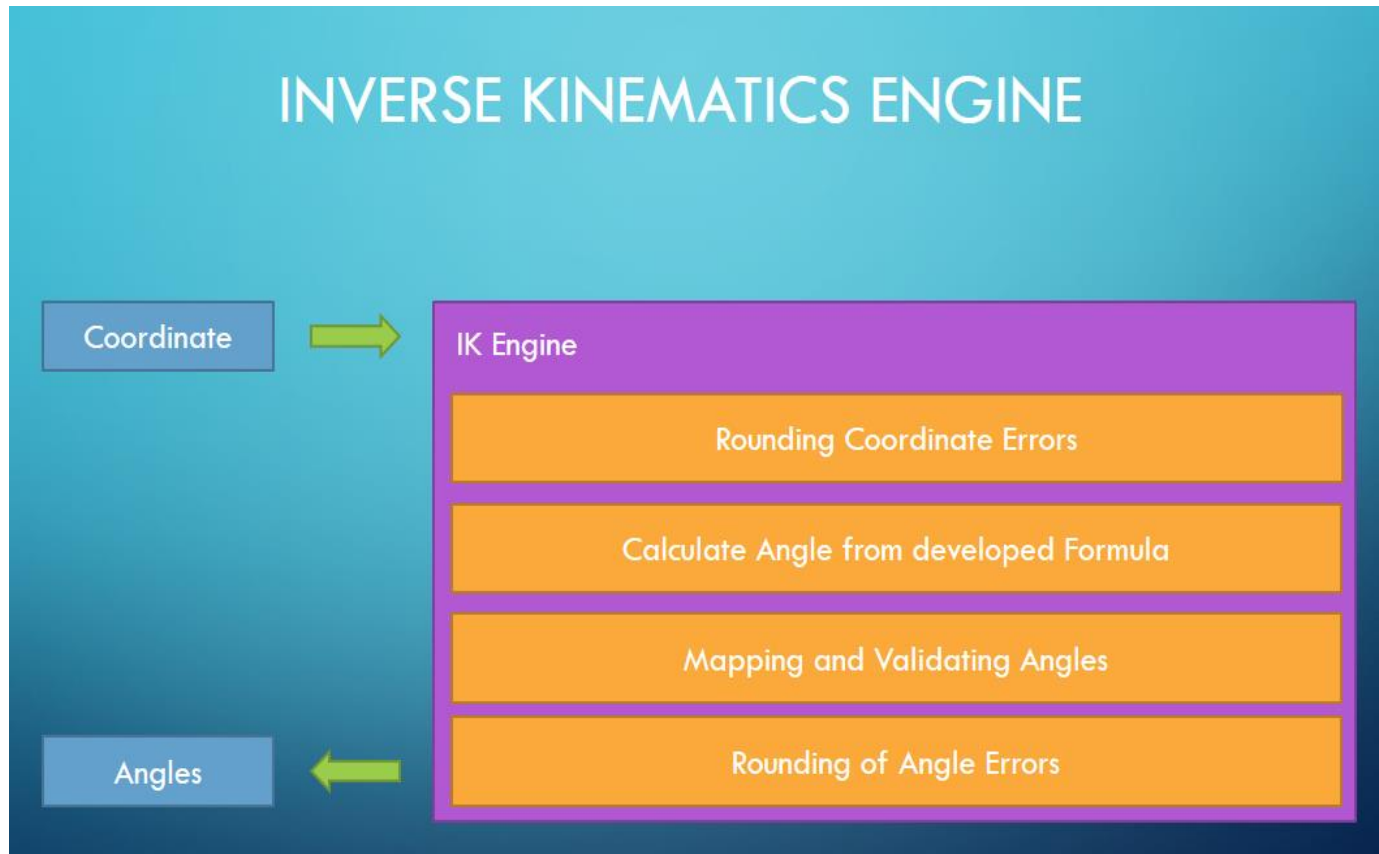


There are many ways to calculate inverse kinematics, but in order to keep things simple, this article focuses on a method known as FABRIK because it's relatively easy to use and implement compared to other IK methods.



The above diagram depicts the inverse kinetics used by us in our project.

INVERSE KINEMATICS ENGINE



From the above calculations we can calculate the angles for the servos to reach the provided co-ordinates to accurately implement the system in our robotic arm we did this.

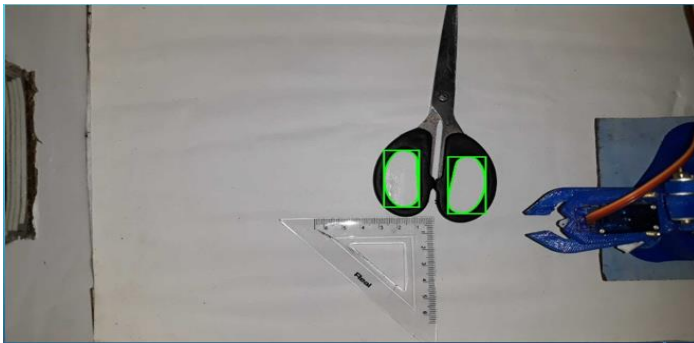
- Fetching the co-ordinates
- Rounding of floating point errors of the coordinates
- Calculating angle from the developed formulae.
- Mapping the angles to physical robotic arm with angle constraints because the robotic arm cannot move freely through all access.
- Validating all angles to make sure that all angles are understandable by the servo motors.
- Rounding up floating point servo angle errors.

IMAGE PROCESSING

Image processing is a method to perform some operations on an image, in order to get an enhanced image or to extract some useful information from it. It is a type of signal processing in which input is an image and output may be image or characteristics/features associated with that image. Nowadays, image processing is among rapidly growing technologies. It forms core research area within engineering and computer science disciplines too.

Image processing basically includes the following three steps:

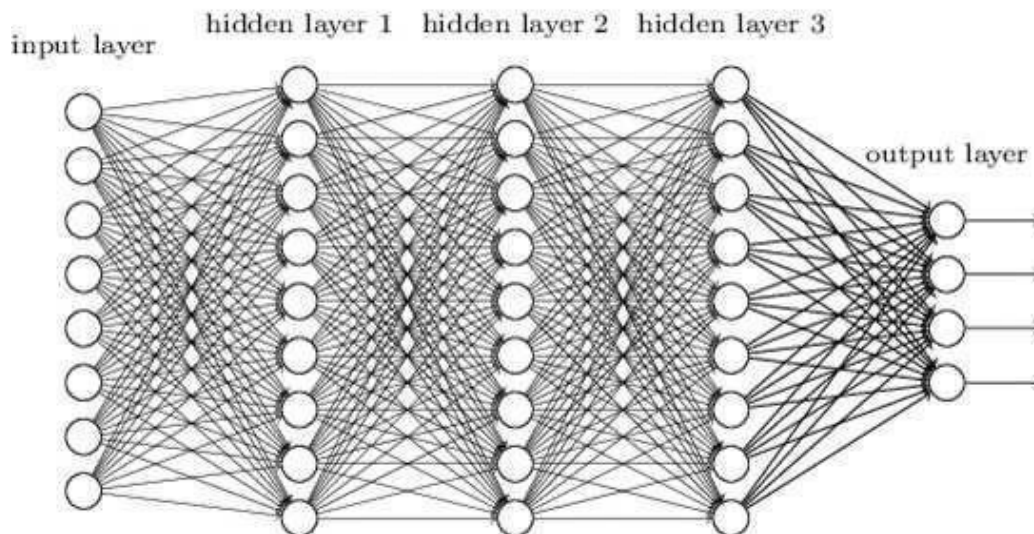
- Importing the image via image acquisition tools;
- The image is passed through basic filters like greyscale, blur, threshold to reduce edge & smooth out edge;
- Then all contours are extracted from the image and fed to DNN to detection.



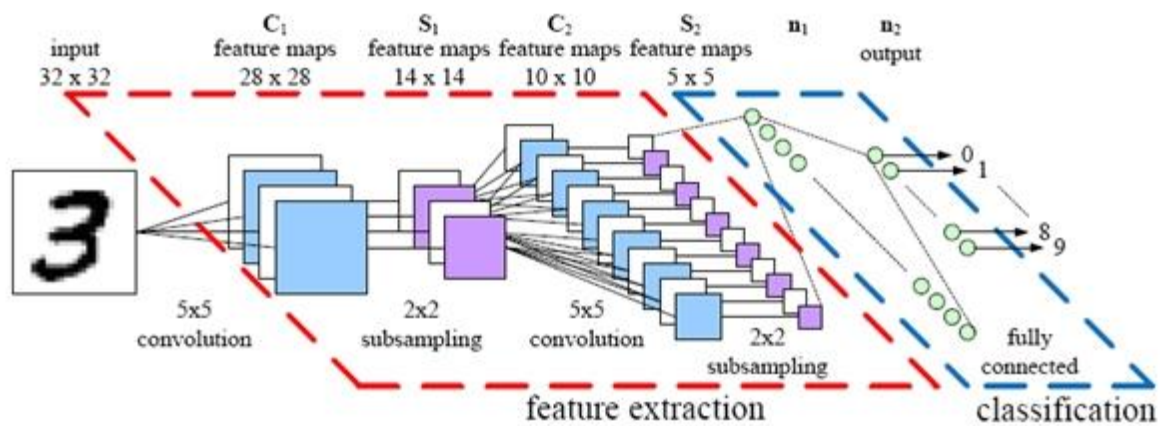
DEEP NEURAL NETWORK

Neural networks are a set of algorithms, modelled loosely after the human brain, that are designed to recognize patterns. They interpret sensory data through a kind of machine perception, labelling or clustering raw input. The patterns they recognize are numerical, contained in vectors, into which all real-world data, be it images, sound, text or time series, must be translated.

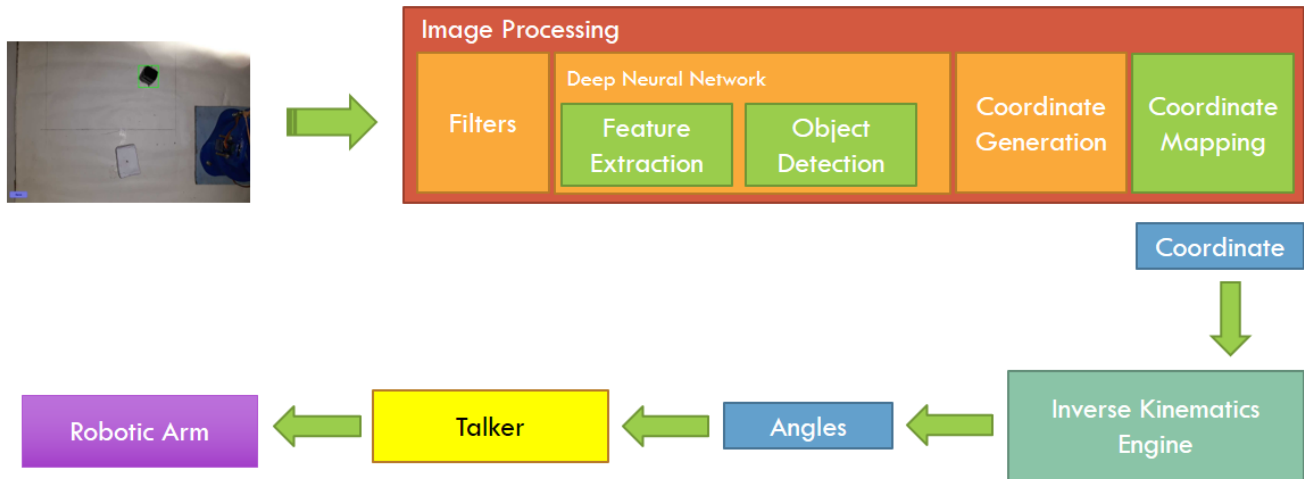
Neural networks help us cluster and classify. You can think of them as a clustering and classification layer on top of the data you store and manage. They help to group unlabelled data according to similarities among the example inputs, and they classify data when they have a labelled dataset to train on. (Neural networks can also extract features that are fed to other algorithms for clustering and classification; so you can think of deep neural networks as components of larger machine-learning applications involving algorithms for reinforcement learning, classification and regression.



- We used Open CV deep neural network for detection of the object.
- The model is trained with around 1600 images.
- The model can detect the particular object around good lighting conditions.



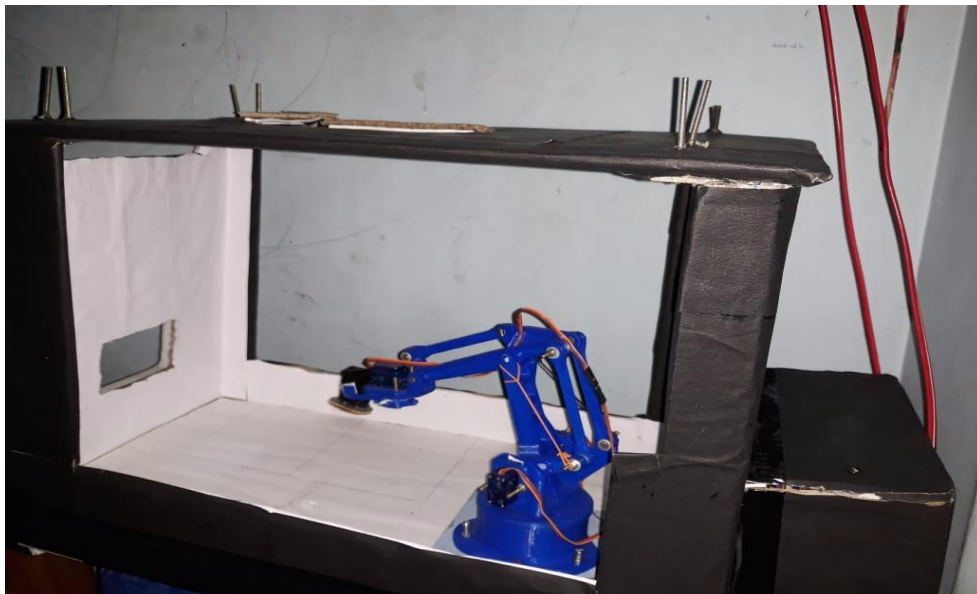
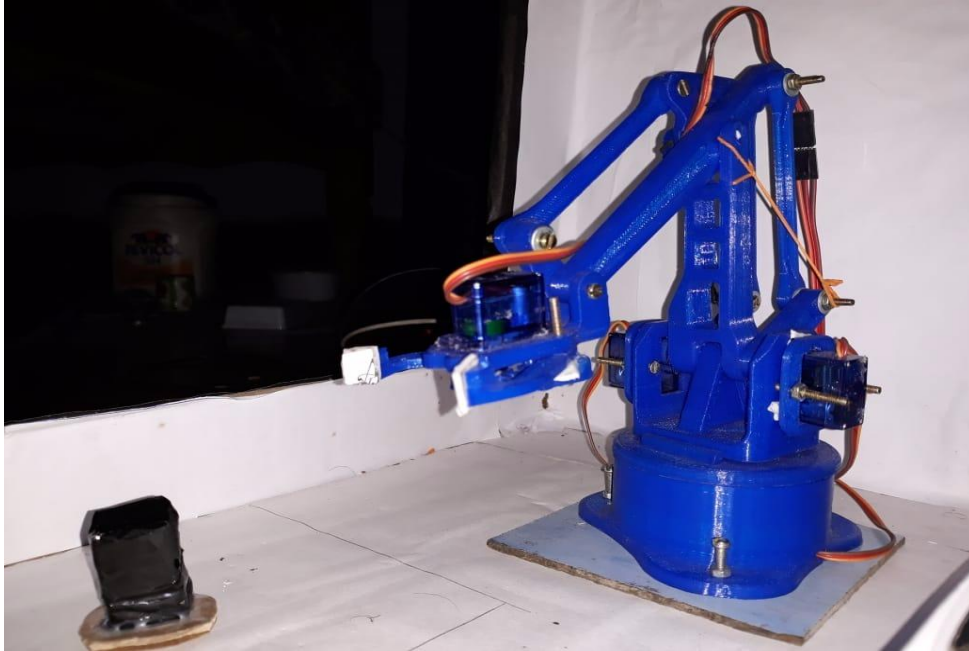
BRINGING IT ALL TOGETHER



This system implements all the different modules into a single unit for them to work together, the entire system works as follows:

- When an incoming image is seen by the camera the image is directly send to the image processing module. The image processing modules works as follows:
 - At first the image is passed through different filters like grey scale, blur and threshold to reduce noise and to smooth out the edges.
- Then the image is send through the deep neural network of the open CV module where the image is broken into different contours of different shapes and colors.
- These contours are one by one fed to the perceptron data structure or neural network for object detection if a positive response is found the detection closes and bounding rectangle of the object is returned.
- Now the center of the rectangle is processed and the object coordinate is generated.
- This generated co-ordinate has to be mapped to the robotic arm co-ordinates that is it has to be converted from pixel co-ordinate to millimeter co-ordinates, after all the co-ordinates have been mapped it is fed to the inverse kinematic engine which generates the angle which is in turn fed to the Arduino board via talker.

SNAPS OF OUR PROJECT



FUTURE SCOPE AND APPLICATION

- Bomb Disposal
- Industry Automation
- Printing
- Cutting
- Engraving
- Etching
- Garbage Collections

COSTS

- The effective cost for this project till now is INR 10,000.
- This cost includes all parts only needed by the hardware support.

CONCLUSION

It took us a lot of time to complete this project, we went through a lot of hardships. Gathering knowledge, researching about topics we do not know is all a part of this project.

We hope our project is liked by all our faculties and juniors and it gives our juniors some inspiration to make more and enhanced projects in future. And we hope our project gets renowned as an inspiring idea for many.

BIBLIOGRAPHY

There are many websites that helped us look for the solution of our problems but we would like to thank:

- Wikipedia
- YouTube
- EEZYBOT
- Arduino website
- Google
- Gmail

And a very special thanks to our mentor Mr. Kaustav Chatterjee to guide us.