ROBOTIC ARM BOT
MINIPROJECT REPORT

Submittedby

KOVELAZHAGAN.V **REGISTER NO:21TN0010**
MATHIVANAN.T **REGISTER NO:21TN0012**
AVINASH.K **REGISTERNO:21TN0002**

Undertheguidanceof

Mr. G. KEERTHIRAAJ M.E.
Assistant Professor, DEPT OF CSE (IoT&CS)

In partial fulfillment for the award of the degree

of

BACHELOROFTECHNOLOGY

in

DEPARTMENTOFARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

MANAKULAVINAYAGARINSTITUTEOFTECHNOLOGY,

KALITHEERTHALKUPPAM, PONDICHERRY
PONDICHERRY UNIVERSITY, INDIA.
MAY2023

MANAKULA VINAYAGAR INSTITUTE OF TECHNOLOGY
PONDICHERRY

DEPARTMENTOFARTIFICIAL INTELLIGENCE AND MACHINE LEARNING

BONAFIDECERTIFICATE

This is to certify that the mini project work entitled **"ROBOTIC ARM BOT"** is a bonafide workdone by KOVELAZHAGAN.V [REGISTERNO: 20TN0010], MATHIVANAN.T [REGISTERNO: 20TN0012], AVINASH.K [REGISTER NO: 20TN0002] in partial fulfillment of the requirement for the award of B.Tech Degree in DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING by Pondicherry University during the academic year2 022-23.

PROJECTGUIDE

Mr. G. KEERTHIRAAJ M.E
ASSISTANT PROFESSOR.

HEADOFTHEDEPARTMENT

Dr.N.POONGUZHALI,Ph.D.
ASSOCIATEPROFESSO&HOD

ACKNOWLEDGEMENT

We are personally indebted to a number of persons that a complete acknowledgement would be encyclopedic .First of all, We love to record our deepest gratitude tothe AlmightyLordand our family.

We express our sincere thanks to our Chairman and Managing Director **Shri. M. DHANASEKARAN** for all his encouragement and moral support. We thank our Vice Chairman **Shri.S.V.SUGUMARAN** and Secretary **Dr.K.NARAYANASAMY** for his support and encouragement.

It gives us great ecstasy of pleasure to convey our deep and sincere thanks to our Principal **Dr. S.MALARKKAN**, for giving constant motivation in succeeding our goal.

With profoundness we would like to express our sincere thanks to **Dr.N. POONGUZHALI** , Head of the Department, Artificial Intelligence and Machine Learning, for her kindness in extending the infrastructural facilities to carry out our project work successfully.

We extend our sincere and heartfelt thanks to our guide Mr.G.KEERTHIRAAJ, Assistant Professor, Department of Artificial Intelligence and Machine Learningfor providing the right ambience for carrying out this workand her valuable guidance and suggestions for our project work. I thank her for the continuousencouragement and the interest shown onus to bring outour project work at this stage and alsoforprovidingthefreedomthat weneededandwanted.

We would like to express our gratitude to all teaching and non-teaching staff members of ourDepartmentforthe supportthey extended during thecourseofthisproject.

TABLEOFCONTENTS

PROBLEM DEFINITION

Robotic arms are fast, accurate and reliable, and can collectively be programmed to perform an almost infiniterange of different operations. Robotic arms can be used for all manner of industrial production, processing and manufacturing roles - any task in which extremely precise, fast and repeatable movements arerequired at times humans may tend to error or get tired or may not be competent to work at certain levels and hencethe use of Robotic Arm becomes Mandatory.Robot arms are ideal for operations which are repetitive, consistent and require a very high degree of accuracy, as well as for applications in which a human worker might struggle to perform safely.

ABSTRACT

In this paper the design, construction, technical characteristics, functionality and control scheme of a new modular hand-arm robot system are presented. This system is designed for applications in robotic and rehabilitation medicine. The development of novel, compact modules is based on flexible fluidic actuators which are equipped with measurement and control elements for a robust control and, hence, precise positioning of the modules. Modules with pneumatic flexible driving elements are characterized by a mechanical modularity that allows for various types of robot kinematics and in particular for lightweight and low-cost concepts. Using the module, medium-sized enterprises will be able to rapidly and reliably adapt this system to various working environments and tasks and develop innovative robot systems for new and existing markets.

INTRODUCTION

Robotic Arm is a mechanical arm like structured robot that is capable of lifting a object and keeping it in desired Place within the limit of arms As manual labour is being reduced at big scale industries and factories to increase efficiency and gainprofit by installing robots that can do repetitive works. A simple robotic arm is one of the most commonly installed machines. We are introducing the basic
concepts of an Arduino controlled robotic arm project. The structure of a robotic arm bot typically consists of multiple linked segments called joints, which allow for smooth and precise movements. These joints are driven by powerful motors or actuators, enabling the arm bot to navigate and interact with its environment with great accuracy

One of the defining features of robotic arm bots is their ability to be programmed and controlled. With the help of cutting-edge software and algorithms, these machines can execute complex tasks autonomously or under human supervision. By inputting commands, coordinates, or sequences of actions, operators can instruct the robotic arm bot to perform a wide array of functions, such as assembly, pick-and-place operations, welding, painting, and much more to assisting in hazardous environments and exploring distant planets, these intelligent The versatility of robotic arm bots is further enhanced by their adaptability and modularity. Depending on the specific needs of a task, different end-effectors or specialized tools can be attached to the arm bot, allowing it to handle diverse objects and materials. This flexibility enables robotic arm bots to be seamlessly integrated into existing workflows, increasing efficiency and productivity in various industries.

Moreover, robotic arm bots are equipped with sensors and vision systems that provide them with feedback about their surroundings. This sensory information and reliable operations.As technology continues to advance, we can expect robotic arm bots to play an even more significant role in shaping our future. From revolutionizing manufacturing processes and aiding in delicate surgical procedures machines are poised to make remarkable contributions to numerous fields.

EXISTING SYSTEM

The existing methods of robotic arm bots encompass a range of advanced technologies and features that enable these machines to perform complex tasks with precision and efficiency. Some notable components include:

Joints and actuators: Robotic arm bots employ various types of joints, such as revolute, prismatic, or spherical joints, coupled with powerful actuators like motors or hydraulics. These mechanisms enable the arm bot to articulate and move with high accuracy and strength.Control systems: Sophisticated control systems, often utilizing algorithms such as inverse kinematics and motion planning, are employed to govern the movements of the robotic arm. These systems ensure smooth and coordinated motions while optimizing trajectories and avoiding collisions.

Sensors and vision systems: Robotic arm bots incorporate sensors, such as encoders, force/torque sensors, and proximity sensors, to provide feedback on position, force, and proximity to objects. Vision systems, including cameras and depth sensors, enable the arm bot to perceive its environment and accurately locate and manipulate objects.

End-effectors and grippers: Different types of end-effectors and grippers are utilized to interact with objects effectively. These can range from simple grippers for picking and placing objects to specialized tools for tasks like welding, cutting, or even surgical procedures.Human-machine interface: Interfaces, including graphical user interfaces (GUIs) or haptic devices, allow humans to interact and control the robotic arm bot. These interfaces facilitate programming, monitoring, and adjusting parameters, ensuring seamless collaboration between humans and machines.

Force/Torque Sensors: Install force and torque sensors at various points along the robotic arm to measure the external forces and torques acting on it. These sensors provide feedback to the control system, allowing it to adjust the arm's movements based on the resistance encountered. This enables the robotic arm to detect and respond to changes in the environment or unexpected forces

Impedance Control: Impedance control focuses on controlling the interaction between the robotic arm and the environment. It involves specifying desired impedance (stiffness, damping, and inertia) for the arm, allowing it to adapt to external forces and resistances. By adjusting the impedance parameters, the arm can actively respond to different levels of resistance encountered during its tasks.

PROPOSED SYSTEM

In the proposed system for robotic arm bots, several advancements and enhancements can be introduced to further improve their capabilities and performance. Here are some key elements that can be included in a proposed system: Enhanced sensory perception: Integrate advanced sensing technologies such as 3D cameras, LiDAR, or depth sensors to enhance the robotic arm bot's perception of its environment. This enables more precise object recognition, improved collision avoidance, and better adaptability to changing surroundings.

Artificial intelligence and machine learning: Incorporate AI and machine learning algorithms to enable the robotic arm bot to learn and adapt to new tasks and scenarios. This allows for autonomous decision-making, intelligent grasping strategies, and the ability to optimize movements based on real-time feedback and data analysis.

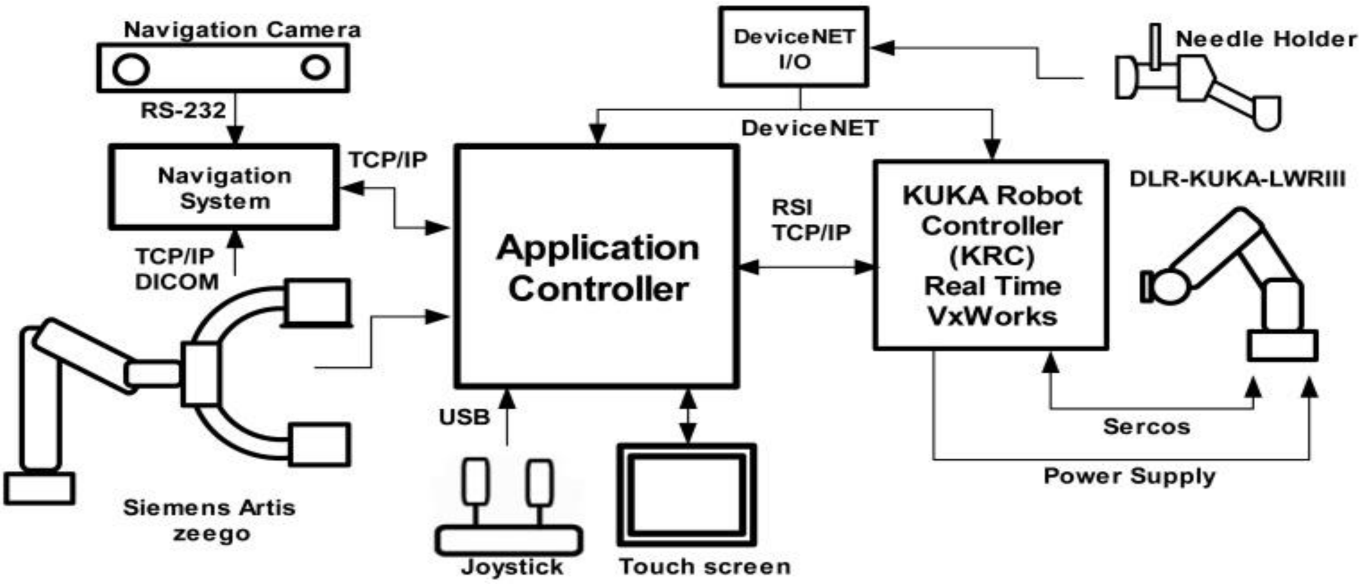
Collaborative capabilities: Enable collaborative functionality, where the robotic arm bot can work alongside human operators or other robots in a coordinated and cooperative manner. This involves incorporating safety measures, intuitive interfaces, and communication protocols to facilitate seamless interaction and teamwork.

Object Recognition and Localization: To enhance the robotic arm's autonomy, integrating object recognition and localization algorithms can enable it to identify and locate specific objects in its environment. This can be done using machine learning techniques or computer vision algorithms, which analyze the input from the vision system and provide object information to the control system.

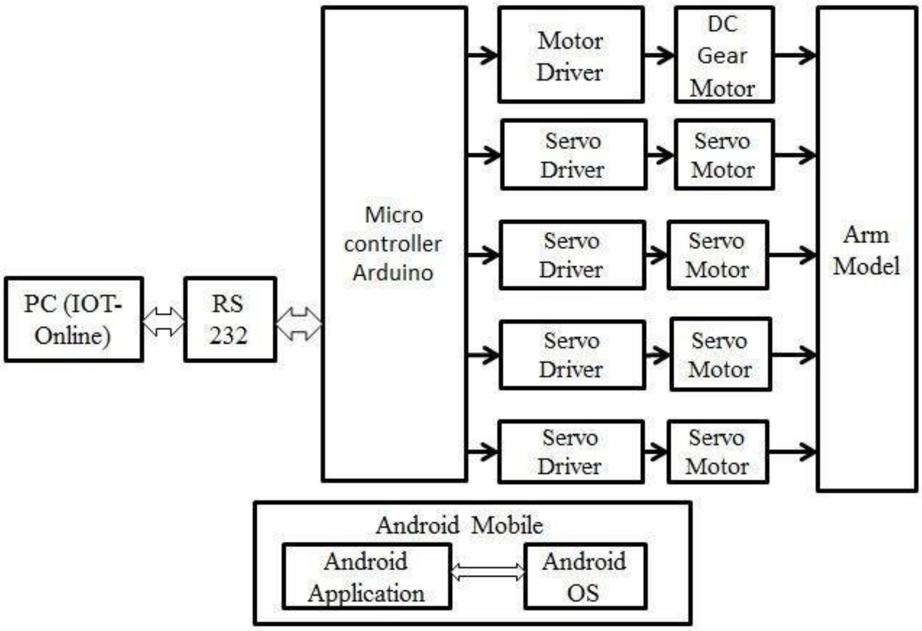
Path Planning and Trajectory Generation: The system should include path planning algorithms that generate optimal paths and trajectories for the robotic arm to follow while avoiding obstacles. These algorithms take into account the arm's kinematics, environmental constraints, and task requirements to plan efficient and collision-free movements.

Safety Features: Safety is crucial in any robotic system. Implement safety features such as emergency stop buttons, collision detection sensors, or force/torque sensors to ensure safe operation. These features can trigger the arm to stop or adjust its movements when it detects potential hazards or excessive forces. Autonomous Operation: To enhance the autonomy of the robotic arm bot, consider integrating autonomous capabilities. This can involve developing algorithms that enable the arm to perform tasks independently, adapt to changing environments, or learn from previous experiences. Maintenance and Diagnostics: The system should include mechanisms for monitoring the arm's health, diagnosing issues, and performing maintenance tasks. This can involve self-diagnostic routines, predictive maintenance algorithms, or remote monitoring capabilities.

ARCHITECTURE DIAGRAM



BLOCK DIAGRAM



SOURCE CODE

```
#include VarSpeedServo.h

/* Servo control for AL5D arm */

/* Arm dimensions( mm ) */

#define BASE_HGT 90 //base height

#define HUMERUS 100 //shoulder-to-elbow "bone"

#define ULNA 135 //elbow-to-wrist "bone"

#define GRIPPER 200 //gripper (incl.heavy duty wrist rotate mechanism) length "

#define ftl(x) ((x)>=0?(long)((x)+0.5):(long)((x)-0.5)) //float to long conversion

/* Servo names/numbers *

* Base servo HS-485HB */

#define BAS_SERVO 4

/* Shoulder Servo HS-5745-MG */

#define SHL_SERVO 5

/* Elbow Servo HS-5745-MG */

#define ELB_SERVO 6

/* Wrist servo HS-645MG */

#define WRI_SERVO 7

/* Wrist rotate servo HS-485HB */

#define WRO_SERVO 8

/* Gripper servo HS-422 */

#define GRI_SERVO 9

/* pre-calculations */
```

```
float hum_sq = HUMERUS*HUMERUS;
float uln_sq = ULNA*ULNA;
int servoSPeed = 10;
//ServoShield servos; //ServoShield object
VarSpeedServo servo1,servo2,servo3,servo4,servo5,servo6;
int loopCounter=0;
int pulseWidth = 6.6;
int microsecondsToDegrees;
void setup()
{
servo1.attach( BAS_SERVO, 544, 2400 );
servo2.attach( SHL_SERVO, 544, 2400 );
servo3.attach( ELB_SERVO, 544, 2400 );
servo4.attach( WRI_SERVO, 544, 2400 );
servo5.attach( WRO_SERVO, 544, 2400 );
servo6.attach( GRI_SERVO, 544, 2400 );
delay( 5500 );
//servos.start(); //Start the servo shield
servo_park();
delay(4000);
Serial.begin( 9600 );
Serial.println("Start");
}
```

```
void loop()

{

loopCounter +=1;

//set_arm( -300, 0, 100, 0 ,10); //

//delay(7000);

//zero_x();

//line();

//circle();

delay(4000);

if (loopCounter> 1) {

servo_park();

//set_arm( 0, 0, 0, 0 ,10); // park

delay(5000);

exit(0); }//pause program - hit reset to continue

//exit(0);

}

/* arm positioning routine utilizing inverse kinematics */

/* z is height, y is distance from base center out, x is side to side. y,z can only be
positive */

//void set_arm( uint16_t x, uint16_t y, uint16_t z, uint16_t grip_angle )

void set_arm( float x, float y, float z, float grip_angle_d, int servoSpeed )

{

float grip_angle_r = radians( grip_angle_d ); //grip angle in radians for use in
calculations
```

```
/* Base angle and radial distance from x,y coordinates */

float bas_angle_r = atan2( x, y );

float rdist = sqrt(( x * x ) + ( y * y ));

/* rdist is y coordinate for the arm */

y = rdist;

/* Grip offsets calculated based on grip angle */

float grip_off_z = ( sin( grip_angle_r )) * GRIPPER;

float grip_off_y = ( cos( grip_angle_r )) * GRIPPER;

/* Wrist position */

float wrist_z = ( z - grip_off_z ) - BASE_HGT;

float wrist_y = y - grip_off_y;

/* Shoulder to wrist distance ( AKA sw ) */

float s_w = ( wrist_z * wrist_z ) + ( wrist_y * wrist_y );

float s_w_sqrt = sqrt( s_w );

/* s_w angle to ground */

float a1 = atan2( wrist_z, wrist_y );

/* s_w angle to humerus */

float a2 = acos((( hum_sq - uln_sq ) + s_w ) / ( 2 * HUMERUS * s_w_sqrt ));

/* shoulder angle */

float shl_angle_r = a1 + a2;

float shl_angle_d = degrees( shl_angle_r );

/* elbow angle */

float elb_angle_r = acos(( hum_sq + uln_sq - s_w ) / ( 2 * HUMERUS * ULNA ));
```

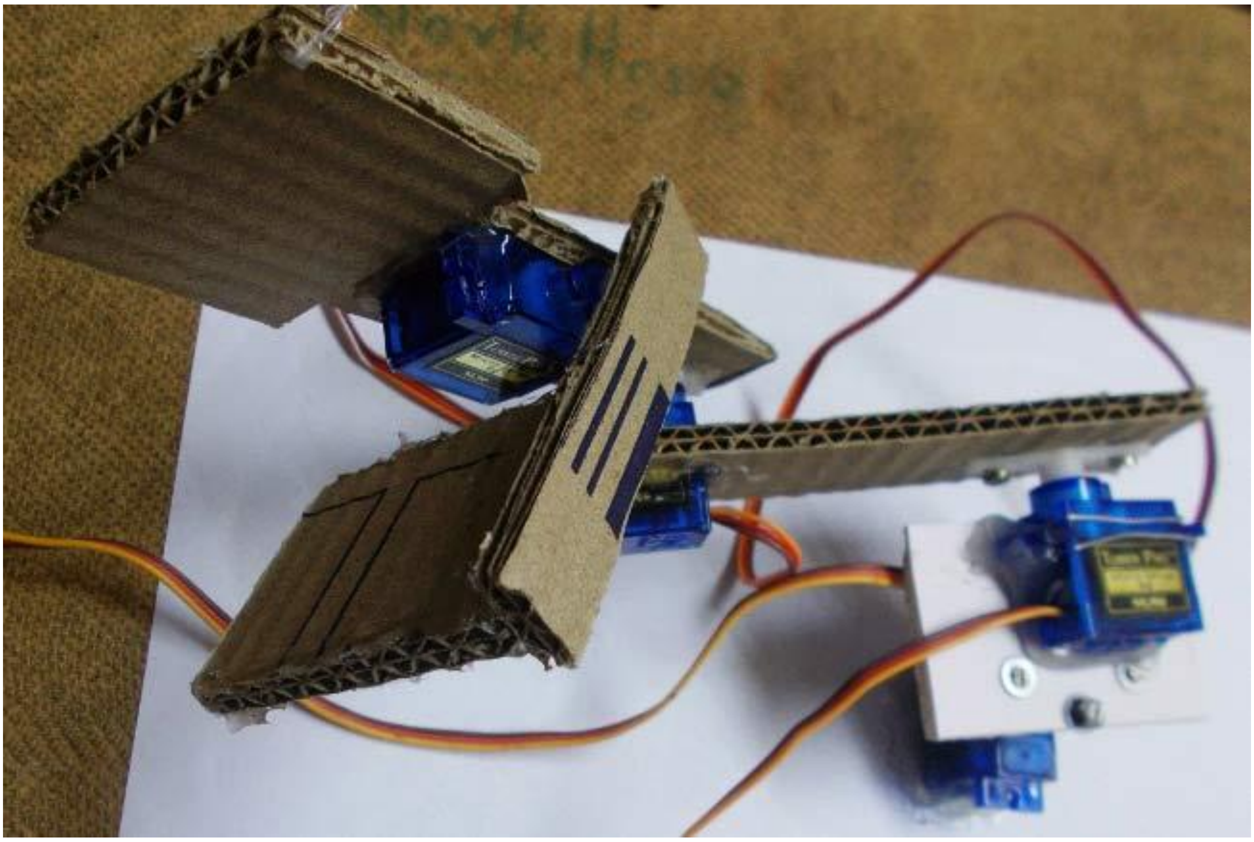
```
float elb_angle_d = degrees( elb_angle_r );

float elb_angle_dn = -( 180.0 - elb_angle_d );

/* wrist angle */

float wri_angle_d = ( grip_angle_d - elb_angle_dn ) - shl_angle_d;

/* Servo pulses */

float bas_servopulse = 1500.0 - (( degrees( bas_angle_r )) * pulseWidth );

float shl_servopulse = 1500.0 + (( shl_angle_d - 90.0 ) * pulseWidth );

float elb_servopulse = 1500.0 - (( elb_angle_d - 90.0 ) * pulseWidth );

//float wri_servopulse = 1500 + ( wri_angle_d * pulseWidth );

//float wri_servopulse = 1500 + ( wri_angle_d * pulseWidth );

float wri_servopulse = 1500 - ( wri_angle_d * pulseWidth );

// updated 2018/2/11 by jimrd - I changed the plus to a minus - not sure how this code worked for anyone before. Could be that the elbow servo was mounted with 0 degrees facing down rather than up.

/* Set servos */

//servos.setposition( BAS_SERVO, ftl( bas_servopulse ));

MicrosecondsToDegrees = map(ftl(bas_servopulse),544,2400,0,180);

servo1.write(microsecondsToDegrees,servoSpeed);

// use this function so that you can set servo speed //

//servos.setposition( SHL_SERVO, ftl( shl_servopulse ));

microsecondsToDegrees = map(ftl(shl_servopulse),544,2400,0,180);

servo2.write(microsecondsToDegrees,servoSpeed);

//servos.setposition( ELB_SERVO, ftl( elb_servopulse ));

microsecondsToDegrees = map(ftl(elb_servopulse),544,2400,0,180);
```

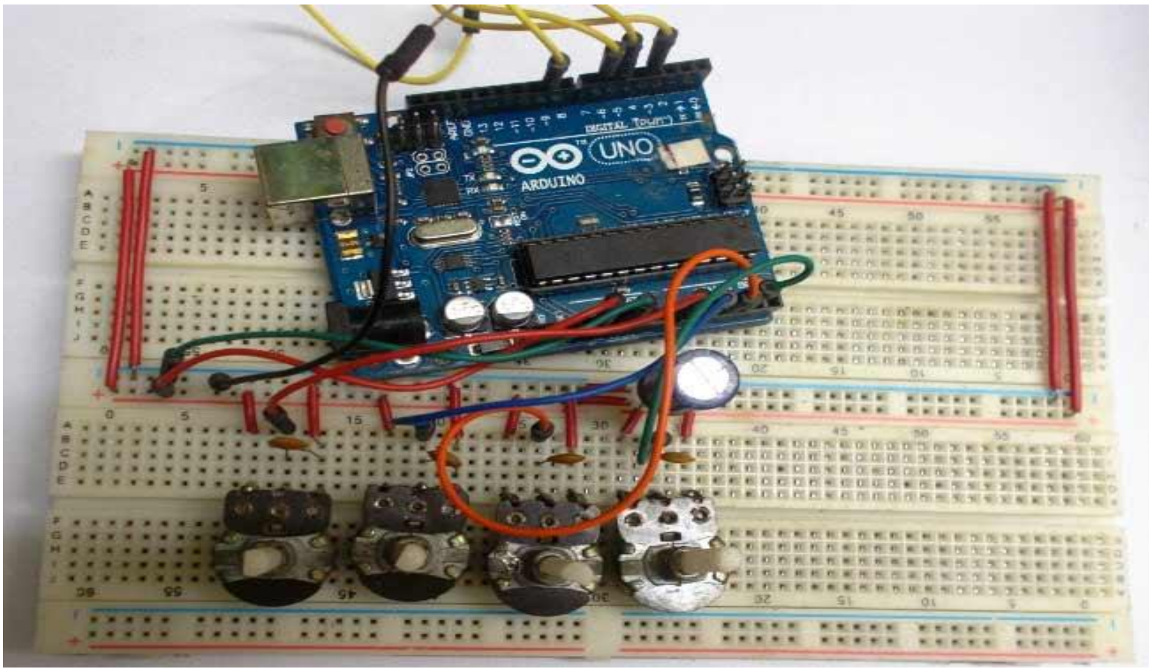
```
servo3.write(microsecondsToDegrees,servoSpeed);
//servos.setposition( WRI_SERVO, ftl( wri_servopulse ));
microsecondsToDegrees = map(ftl(wri_servopulse),544,2400,0,180);
servo4.write(microsecondsToDegrees,servoSpeed);
}
/* move servos to parking position */
void servo_park()
{
//servos.setposition( BAS_SERVO, 1500 );
servo1.write(90,10);
//servos.setposition( SHL_SERVO, 2100 );
servo2.write(90,10);
//servos.setposition( ELB_SERVO, 2100 );
servo3.write(90,10);
//servos.setposition( WRI_SERVO, 1800 );
servo4.write(90,10);
//servos.setposition( WRO_SERVO, 600 );
servo5.write(90,10);
//servos.setposition( GRI_SERVO, 900 );
servo6.write(80,10);
return;
}
void zero_x()
```

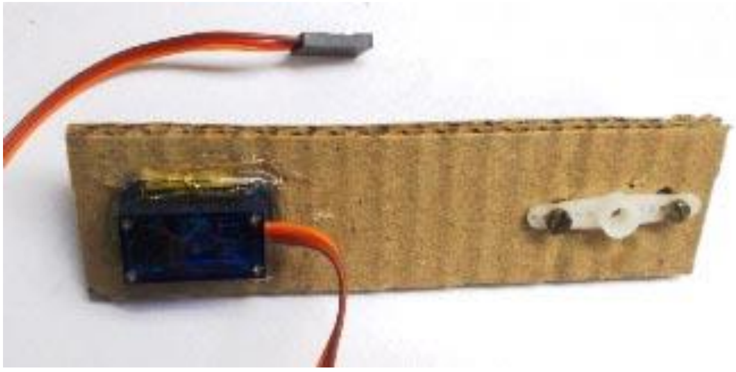
```
{
for( double yaxis = 250.0; yaxis< 400.0; yaxis += 1 ) {
Serial.print(" yaxis= : ");Serial.println(yaxis);
set_arm( 0, yaxis, 200.0, 0 ,10);
delay( 10 );
}
for( double yaxis = 400.0; yaxis> 250.0; yaxis -= 1 ) {
set_arm( 0, yaxis, 200.0, 0 ,10);
delay( 10 );
}
}
/* moves arm in a straight line */
void line()
{
for( double xaxis = -100.0; xaxis< 100.0; xaxis += 0.5 ) {
set_arm( xaxis, 250, 120, 0 ,10);
delay( 10 );
for( float xaxis = 100.0; xaxis> -100.0; xaxis -= 0.5 ) {
set_arm( xaxis, 250, 120, 0 ,10);
delay( 10 );
}
}
void circle()
```

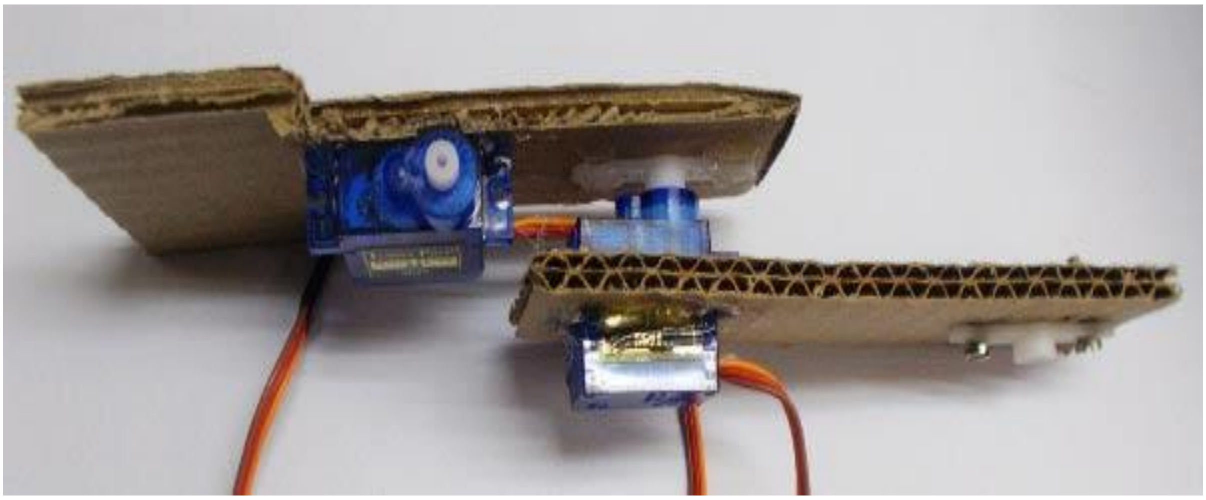
```
{
#define RADIUS 50.0
//float angle = 0;
float zaxis,yaxis;
for( float angle = 0.0; angle < 360.0; angle += 1.0 ) {
yaxis = RADIUS * sin( radians( angle )) + 300;
zaxis = RADIUS * cos( radians( angle )) + 200;
set_arm( 0, yaxis, zaxis, 0 ,50);
delay( 10 );
}
}
```

SAMPLE PICTURES










TOWER PRO™
Micro Servo
9g
SG90