**A Mini Project Report**

**On**

**SEMI - AUTO SPIDDY**

**Submitted**

**By**

**Urvesh - Saifi**

**(Roll No:2102220310030)**

**{ 2021-2025 }**

**Course Code : KEC554**

**Under the Supervision of**

**Ms. Pragati Tripathi**

**Head of Department**

**Dr. Monika Jain**

**DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING**

**5th semester (0ctober)**

**2023-2024**

**Certificate**

Certified that this mini project report **“ Semi - Auto Spiddy ”** is the bonafide work of “ **Urvesh Saifi ”** who carried out this project work under my supervision. Certified further that to the best of my knowledge the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this by I.T.S Engineering College.

Signature of the Mentor Signature of the Head of the Department

(Electronics and Communication Engineering)

###### 

###### 

‘ Submitted for Semester Mini-Project examination held in year 2023-24 (5th semester) ‘

**Acknowledgement**

"I **Urvesh - Saifi**  has taken efforts in this project. However, it would not have been possible without the kind support and help of many individuals and college organizations. I would like to extend my sincere thanks to all of them.

I am highly indebted to **I.T.S ENGINEERING COLLEGE** for their guidance and constant supervision as well as for providing necessary information regarding the project & also for their support in completing the project.

I would like to express my gratitude towards my **parents** & **friends** for their kind co-operation and encouragement which help me in completion of this project.

I would like to express my special gratitude and thanks to **NEW GEN IDEC** members for giving me such support and time. My special thanks and appreciations also go to batch mates in developing the project and people who have willingly helped me out with their abilities."

**SIGNATURES**

**SEMI-AUTO SPIDDY**

The spider robot is a device used in monitoring surroundings wirelessly. This review work will assist in solving the weak adaptive ability of ordinary existing robots. The spider robots function without interfacing, it can easily adapt to the new situation or obstacles due to its legs of locomotion on like the ordinary two legs robots. A spider robotic system will help in monitoring of toxic or nuclear environments and moving in an environment that ordinary robots cannot do like; climbing of rough surfaces, trace and locating of missing items. The user of this spider robotic system poses a perfect resolution to such problems, eradicating the demand for humankind to access such places and regularly providing information on the state of such environment that would not otherwise be available for humanity.

A spider robot is a compact robotic creation inspired by the anatomy and movement of spiders. Constructed with lightweight materials and driven by servo motors, it emulates the agility of its biological counterparts. Utilizing technologies like Arduino Nano microcontrollers, these robots can be programmed for intricate leg movements, enabling them to traverse diverse environments with ease. Additionally, spider robots often integrate wireless communication modules, such as Bluetooth, allowing users to control them remotely. While primarily designed for educational purposes, these robots showcase the potential of biomimicry in robotics, demonstrating how nature-inspired designs can contribute to innovative technological advancements.

**Table of Content :-**

|  |  |  |
| --- | --- | --- |
| **Serial Number** | **Topic Name** | **Page No.** |
| Chapter - 1 | Project Outline | 1 |
| Chapter - 2  2.1 | List of components and their cost | 2 |
| 2.2 | Circuit Diagram of the Project | 3 |
| Chapter-3 | Real Picture of the Project | 4 |
| 3.1 | Application of each component of the project | 5 |
| 3.2 | Application of the Project formed | 6 |
| 3.3 | Advantages & Disadvantages of the Project | 7 |
| 3.4 | Challenges during the Project | 8 |
| Chapter 4 | Conclusion & Future Scope | 9 |

**Project Outline**

**Project Name:** SEMI-AUTO SPIDDY

**Current Date:** October 20th, 2023

**Document Author:** Urvesh Saifi

**Executive Summary**

This is a semi-auto robot which look and work like spider .

**Project Background**

This project is made for mini project based on IOT. The purpose of this project is to make a robot that can easily adapt in local/ rural areas and surfaces.

**Statement Of Work**

This project have a huge scope in future for multipurpose work specially in defence and security.

**Project Objectives**

1. Entertainment purpose
2. Defence purpose (Upgraded version)
3. Espionage (Upgraded version)

**Contact Person :-**

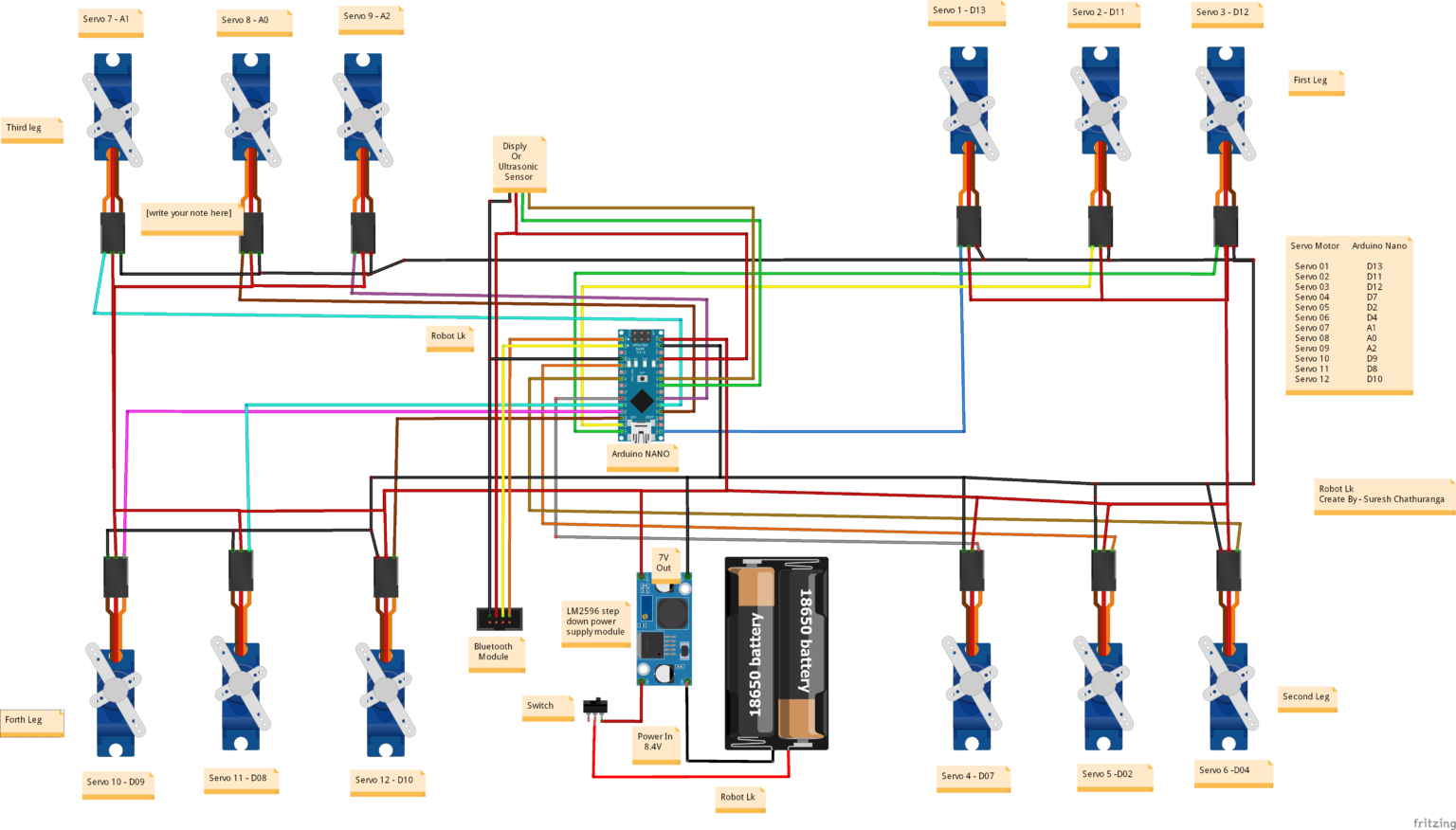
Urvesh Saifi (9917922432)

**Project Budget -** Rs 3,000

**List of components and their cost**

|  |  |  |
| --- | --- | --- |
| 1. No | Component | Price |
|  | Arduino Nano | Rs. 300 |
|  | Nano 328P Expansion Adapter  Breakout Board IO Shield | Rs. 150 |
|  | HC05 Bluetooth Module | Rs. 300 |
|  | SG90 Mini Servo 12 pieces | Rs. 1200 |
|  | Buck Converter Lm2596 | Rs. 60 |
|  | Lithium Ion Battery 2 pieces | Rs. 240 |
|  | Leds \* 2 piece | Rs. 10 |
|  | Jumper Wires | Rs. 60 |

**Circuit Diagram of the Project**



**Real Picture of the Project**



**Application of each component of the project**

1. **Arduino Nano**

The Arduino Nano is a microcontroller - based device with 16 digital pins that can be used for various purposes. It can be used for almost every task, from minor to massive industrial-scale projects. It can also be used for prototyping and developing new applications.

1. **Nano 328P Expansion Adapter Breakout Board IO Shield**

To use the shield, mount it on top of an Arduino board (e.g. the Uno). To upload sketches to the board, connect it to your computer with a USB cable as you normally would. Once the sketch has been uploaded, you can disconnect the board from your computer and power it with an external power supply

1. **Buck Converter Lm2596**

A buck converter is used to step down voltage of the given input in order to achieve required output. Buck converters are mostly used for USB on the go, point of load converters for PCs and laptops, Battery Chargers, Quad Copters, Solar Chargers, and power audio amplifiers.

1. **Jumper Wires**

Jumper wires are simply wires that have connector pins at each end, allowing them to be used to connect two points to each other without soldering. Jumper wires are typically used with breadboards and other prototyping tools in order to make it easy to change a circuit as needed.

1. **SG90 Mini Servo**

A wide range of applications for servo motors exists, including cameras, telescopes, antennas, industrial automation, and robots. A motor rotates from 0 to 180 degrees at each position of 90 degrees so that names it SG90.

1. **Lithium Ion Battery**

 the main application of rechargeable Li-ion batteries is in portable electronic devices, such as cellular phones, digital cameras, global positioning system devices, tablets, and laptop computers.

1. **Leds**

The on-state voltage of LED approximately ranges from 1.2 to 2.0 V. The diode is used in voltage rectifiers, clipping & clamping circuits, voltage multipliers. The applications of LED are traffic signals, automotive headlamps, in medical devices, camera flashes, etc.

1. **HC05 Bluetooth Module**

Designed to replace cable connections HC-05 uses serial communication to communicate with the electronics. Usually, it is used to connect small devices like mobile phones using a short-range wireless connection to exchange files. It uses the 2.45GHz frequency band.

**Application of this ROBOT**

### 1. Security / Military

Imagine if all the security guards are robots? Even thieves would be scared! That’s why robots are being proposed as security agents as they can protect humans, and they wouldn’t be in danger like human security guards would be. Currently, robotics companies are working on pairing robot guards with human security consultants. A very famous company in this field is Knight scope in the United States that has autonomous security robots capable of assisting human security guards with real-time, actionable intelligence. These robots can help with crimes such as armed robberies, burglaries, domestic violence, fraud, hit, and runs, etc.

### 2. Space Exploration

There are many things in space that are very dangerous for astronauts to do. Humans can’t roam on Mars all day to collect soil samples or work on repairing a spaceship from the outside while it’s in deep space! In these situations, robots are a great choice because there are no chances for the loss of human life then. So space institutions like NASA frequently use robots and autonomous vehicles to do things that humans can’t. For example, Mars Rover is an autonomous robot that travels on Mars and takes pictures of Martian rock formations that are interesting or important and then sends them back on Earth for the NASA scientists to study.

### 3. Entertainment

Robots are also a big draw in the entertainment industry. While they cannot exactly become actors and actresses, they can be used behind the sets in movies and serials to manage the camera, provide special effects, etc. They can be used for boring repetitive tasks that are not suitable for a human as cinema is, after all, a creative industry. Robots can also be used to do stunt work that is very dangerous for humans but looks pretty cool in an action movie. Theme parks like Disney World are also using autonomous robots to enhance the magical experience of their customers.

### 4. Underwater Exploration

Robots are a great option for exploring places that humans cannot reach easily, like the depths of the ocean! There is a lot of water pressure deep in the ocean which means humans cannot go that down and machines such as submarines can only go to a certain depth as well. A deep underwater is a mysterious place that can finally be explored using specially designed robots. These robots are remote-controlled, and they can go into depths of the ocean to collect data and images about the aquatic plant and animal life.

**Advantages of the Project**

* In many situations this robot can increase productivity, efficiency, quality and consistency of products:
* Unlike humans, robots don’t get bored
* Until they wear out, they can do the same thing again and again
* They can be very accurate – to fractions of an inch (taking step)
* This Robot can work in environments which are unsafe for humans – in the nuclear or chemical industries for example
* This Robot don’t have the same environmental requirements that humans do – such as lighting, air conditioning or noise protection
* This Robot have some sensors/actuators which are more capable than humans

**Disadvantages of the Project**

* This Robot can only do what they are told to do – they can’t improvise.
* This means that safety procedures are needed to protect humans and other robots.
* Although this robot can be superior to humans in some ways, this is less dextrous than humans, this robot do not have such powerful brains, and cannot compete with a human’s ability to understand what they can see.

**Challenges during the Project**

* **Cost:** Personal service robots can be expensive, which can be a barrier to adoption for some consumers.
* **Limited capabilities:** Personal service robots are limited by the capabilities of their programming and may not be able to perform tasks that require creativity or complex problem-solving.
* **User acceptance:** Some people may be hesitant to accept personal service robots, either due to concerns about privacy or because they prefer human interaction.
* **Safety:** Personal service robots must be designed with safety in mind and be able to operate safely around humans and pets.
* **Usability:** Personal service robots must be easy to use and understand, with intuitive controls and user interfaces.
* **Adaptability:** Personal service robots must be able to adapt to different environments and tasks, and be able to learn and improve over time.
* **Social acceptance:** Personal service robots must be socially acceptable and not be perceived as a threat or a replacement for human interaction.
* **Security:** Personal service robots must protect the privacy of their users and be secure against hacking and other types of cyber attacks.

**Conclusion**

In conclusion, this paper reviewed some studies relating to spider robots, highlighted its principle of operation, it also summarizes how it is made or constructed and the areas of applications in real life in this world. With the help of advancement in technology, the spider robot system will be able to monitor every important environment also analyzes the situation of such environment in which one can have full access due to the complication of such places and implement the proper action needs to be executed in such areas

**Future Scope**

1. We can use this spider robot in discovering dangerous or rough areas in which humankind can have full access easily. For example, searching for survivors after a terrible nuclear tragedy, also exploring in war zones, for inspecting unstable buildings after a natural tragedy such as earthquake, tsunami or a volcanic eruption.
2. We can also use spider robots in defusing bombs such as land mines.
3. We can also equip the spider robots with sensors and weapons; such robot is used in a crisis or war to avoid risking human lives on the battlefield.
4. We can also use this spider robot in guarding our properties or areas of high importance.

**References**

**<https://robotlk.com/how-to-make-spider-robot-12dof-sg90-servo-arduino-nano/>**

**<https://www.youtube.com/watch?v=CxfBYyjs4FY&t=2290s>**

**<https://www.tutorialspoint.com/arduino/arduino_servo_motor.htm#:~:text=Working%20of%20a%20Servo%20Motor&text=If%20the%20shaft%20is%20at,traveling%20somewhere%20around%20180%20degrees.>**

**<https://www.instructables.com/ARDUINO-SPIDER-ROBOT-QUADRUPED/>**

**CODE**

/\* -----------------------------------------------------------------------------

- Project: Semi auto spiddy

- Author: Urvesh Saifi

- Date: 20th oct’23

-----------------------------------------------------------------------------

- Overview

- This project was written for the Crawling robot designed by Urvesh Saifi.

This version of the robot has 4 legs, and each leg is driven by 3 servos.

This robot is driven by a Ardunio Nano Board with an expansion Board.

- Request

- This project requires some library files, which you can find in the head of

this file. Make sure you have installed these files.

---------------------------------------------------------------------------\*/

/\* Includes ------------------------------------------------------------------\*/

#include <Servo.h> //to define and control servos

#include "FlexiTimer2.h"//to set a timer to manage all servos

/\* Servos --------------------------------------------------------------------\*/

//define 12 servos for 4 legs

char data = 0;

Servo servo[4][3];

//define servos' ports

const int servo\_pin[4][3] = { {2, 3, 4}, {5, 6, 7}, {8, 9, 10}, {11, 12, 13} };

/\* Size of the robot ---------------------------------------------------------\*/

const float length\_a = 55;

const float length\_b = 77.5;

const float length\_c = 27.5;

const float length\_side = 71;

const float z\_absolute = -28;

/\* Constants for movement ----------------------------------------------------\*/

const float z\_default = -50, z\_up = -30, z\_boot = z\_absolute;

const float x\_default = 62, x\_offset = 0;

const float y\_start = 0, y\_step = 40;

const float y\_default = x\_default;

/\* variables for movement ----------------------------------------------------\*/

volatile float site\_now[4][3]; //real-time coordinates of the end of each leg

volatile float site\_expect[4][3]; //expected coordinates of the end of each leg

float temp\_speed[4][3]; //each axis' speed, needs to be recalculated before each movement

float move\_speed; //movement speed

float speed\_multiple = 1; //movement speed multiple

const float spot\_turn\_speed = 4;

const float leg\_move\_speed = 8;

const float body\_move\_speed = 3;

const float stand\_seat\_speed = 1;

volatile int rest\_counter; //+1/0.02s, for automatic rest

//functions' parameter

const float KEEP = 255;

//define PI for calculation

const float pi = 3.1415926;

/\* Constants for turn --------------------------------------------------------\*/

//temp length

const float temp\_a = sqrt(pow(2 \* x\_default + length\_side, 2) + pow(y\_step, 2));

const float temp\_b = 2 \* (y\_start + y\_step) + length\_side;

const float temp\_c = sqrt(pow(2 \* x\_default + length\_side, 2) + pow(2 \* y\_start + y\_step + length\_side, 2));

const float temp\_alpha = acos((pow(temp\_a, 2) + pow(temp\_b, 2) - pow(temp\_c, 2)) / 2 / temp\_a / temp\_b);

//site for turn

const float turn\_x1 = (temp\_a - length\_side) / 2;

const float turn\_y1 = y\_start + y\_step / 2;

const float turn\_x0 = turn\_x1 - temp\_b \* cos(temp\_alpha);

const float turn\_y0 = temp\_b \* sin(temp\_alpha) - turn\_y1 - length\_side;

/\* ---------------------------------------------------------------------------\*/

/\*

- setup function

---------------------------------------------------------------------------\*/

void setup()

{

//start serial for debug

Serial.begin(9600);

Serial.println("Robot starts initialization");

//initialize default parameter

pinMode(14, OUTPUT);

set\_site(0, x\_default - x\_offset, y\_start + y\_step, z\_boot);

set\_site(1, x\_default - x\_offset, y\_start + y\_step, z\_boot);

set\_site(2, x\_default + x\_offset, y\_start, z\_boot);

set\_site(3, x\_default + x\_offset, y\_start, z\_boot);

for (int i = 0; i < 4; i++)

{

for (int j = 0; j < 3; j++)

{

site\_now[i][j] = site\_expect[i][j];

}

}

//start servo service

FlexiTimer2::set(20, servo\_service);

FlexiTimer2::start();

Serial.println("Servo service started");

//initialize servos

servo\_attach();

Serial.println("Servos initialized");

Serial.println("Robot initialization Complete");

}

void servo\_attach(void)

{

for (int i = 0; i < 4; i++)

{

for (int j = 0; j < 3; j++)

{

servo[i][j].attach(servo\_pin[i][j]);

delay(100);

}

}

}

void servo\_detach(void)

{

for (int i = 0; i < 4; i++)

{

for (int j = 0; j < 3; j++)

{

servo[i][j].detach();

delay(100);

}

}

}

/\*

- loop function

---------------------------------------------------------------------------\*/

void loop()

{

if(Serial.available() > 0)

{

data = Serial.read();

Serial.print(data);

Serial.print("\n");

if(data == 'F')

{

Serial.println("Step forward");

step\_forward();

}

else if(data == 'B')

{

Serial.println("Step back");

step\_back();

}

else if(data == 'L')

{

Serial.println("Turn left");

turn\_left();

}

else if(data == 'R')

{

Serial.println("Turn right");

turn\_right();

}

else if(data == 'X')

{

Serial.println("Stand");

stand();

}

else if(data == 'x')

{

Serial.println("Sit");

sit();

}

else if(data == 'S' ||data == 'D' )

{

}

else if(data == 'W')

{

digitalWrite(14, HIGH);

}

else if(data == 'w')

{

digitalWrite(14, LOW);

}

else if(data == 'V')

{

Serial.println("Hand wave");

hand\_shake(3);

}

else if(data == 'v')

{

Serial.println("Hand wave");

hand\_shake(3);

}

else if(data == 'U')

{

Serial.println("Body dance");

body\_dance(10);

}

else if(data == 'u')

{

Serial.println("Body dance");

body\_dance(10);

}

while(Serial.available()) {Serial.read();}

}

}

/\*

- sit

- blocking function

---------------------------------------------------------------------------\*/

void sit(void)

{

move\_speed = stand\_seat\_speed;

for (int leg = 0; leg < 4; leg++)

{

set\_site(leg, KEEP, KEEP, z\_boot);

}

wait\_all\_reach();

}

/\*

- stand

- blocking function

---------------------------------------------------------------------------\*/

void stand(void)

{

move\_speed = stand\_seat\_speed;

for (int leg = 0; leg < 4; leg++)

{

set\_site(leg, KEEP, KEEP, z\_default);

}

wait\_all\_reach();

}

/\*

- spot turn to left

- blocking function

- parameter step steps wanted to turn

---------------------------------------------------------------------------\*/

void turn\_left()

{

move\_speed = spot\_turn\_speed;

if (site\_now[3][1] == y\_start)

{

//leg 3&1 move

set\_site(3, x\_default + x\_offset, y\_start, z\_up);

wait\_all\_reach();

set\_site(0, turn\_x1 - x\_offset, turn\_y1, z\_default);

set\_site(1, turn\_x0 - x\_offset, turn\_y0, z\_default);

set\_site(2, turn\_x1 + x\_offset, turn\_y1, z\_default);

set\_site(3, turn\_x0 + x\_offset, turn\_y0, z\_up);

wait\_all\_reach();

set\_site(3, turn\_x0 + x\_offset, turn\_y0, z\_default);

wait\_all\_reach();

set\_site(0, turn\_x1 + x\_offset, turn\_y1, z\_default);

set\_site(1, turn\_x0 + x\_offset, turn\_y0, z\_default);

set\_site(2, turn\_x1 - x\_offset, turn\_y1, z\_default);

set\_site(3, turn\_x0 - x\_offset, turn\_y0, z\_default);

wait\_all\_reach();

set\_site(1, turn\_x0 + x\_offset, turn\_y0, z\_up);

wait\_all\_reach();

set\_site(0, x\_default + x\_offset, y\_start, z\_default);

set\_site(1, x\_default + x\_offset, y\_start, z\_up);

set\_site(2, x\_default - x\_offset, y\_start + y\_step, z\_default);

set\_site(3, x\_default - x\_offset, y\_start + y\_step, z\_default);

wait\_all\_reach();

set\_site(1, x\_default + x\_offset, y\_start, z\_default);

wait\_all\_reach();

}

else

{

//leg 0&2 move

set\_site(0, x\_default + x\_offset, y\_start, z\_up);

wait\_all\_reach();

set\_site(0, turn\_x0 + x\_offset, turn\_y0, z\_up);

set\_site(1, turn\_x1 + x\_offset, turn\_y1, z\_default);

set\_site(2, turn\_x0 - x\_offset, turn\_y0, z\_default);

set\_site(3, turn\_x1 - x\_offset, turn\_y1, z\_default);

wait\_all\_reach();

set\_site(0, turn\_x0 + x\_offset, turn\_y0, z\_default);

wait\_all\_reach();

set\_site(0, turn\_x0 - x\_offset, turn\_y0, z\_default);

set\_site(1, turn\_x1 - x\_offset, turn\_y1, z\_default);

set\_site(2, turn\_x0 + x\_offset, turn\_y0, z\_default);

set\_site(3, turn\_x1 + x\_offset, turn\_y1, z\_default);

wait\_all\_reach();

set\_site(2, turn\_x0 + x\_offset, turn\_y0, z\_up);

wait\_all\_reach();

set\_site(0, x\_default - x\_offset, y\_start + y\_step, z\_default);

set\_site(1, x\_default - x\_offset, y\_start + y\_step, z\_default);

set\_site(2, x\_default + x\_offset, y\_start, z\_up);

set\_site(3, x\_default + x\_offset, y\_start, z\_default);

wait\_all\_reach();

set\_site(2, x\_default + x\_offset, y\_start, z\_default);

wait\_all\_reach();

}

}

/\*

- spot turn to right

- blocking function

- parameter step steps wanted to turn

---------------------------------------------------------------------------\*/

void turn\_right()

{

move\_speed = spot\_turn\_speed;

if (site\_now[2][1] == y\_start)

{

//leg 2&0 move

set\_site(2, x\_default + x\_offset, y\_start, z\_up);

wait\_all\_reach();

set\_site(0, turn\_x0 - x\_offset, turn\_y0, z\_default);

set\_site(1, turn\_x1 - x\_offset, turn\_y1, z\_default);

set\_site(2, turn\_x0 + x\_offset, turn\_y0, z\_up);

set\_site(3, turn\_x1 + x\_offset, turn\_y1, z\_default);

wait\_all\_reach();

set\_site(2, turn\_x0 + x\_offset, turn\_y0, z\_default);

wait\_all\_reach();

set\_site(0, turn\_x0 + x\_offset, turn\_y0, z\_default);

set\_site(1, turn\_x1 + x\_offset, turn\_y1, z\_default);

set\_site(2, turn\_x0 - x\_offset, turn\_y0, z\_default);

set\_site(3, turn\_x1 - x\_offset, turn\_y1, z\_default);

wait\_all\_reach();

set\_site(0, turn\_x0 + x\_offset, turn\_y0, z\_up);

wait\_all\_reach();

set\_site(0, x\_default + x\_offset, y\_start, z\_up);

set\_site(1, x\_default + x\_offset, y\_start, z\_default);

set\_site(2, x\_default - x\_offset, y\_start + y\_step, z\_default);

set\_site(3, x\_default - x\_offset, y\_start + y\_step, z\_default);

wait\_all\_reach();

set\_site(0, x\_default + x\_offset, y\_start, z\_default);

wait\_all\_reach();

}

else

{

//leg 1&3 move

set\_site(1, x\_default + x\_offset, y\_start, z\_up);

wait\_all\_reach();

set\_site(0, turn\_x1 + x\_offset, turn\_y1, z\_default);

set\_site(1, turn\_x0 + x\_offset, turn\_y0, z\_up);

set\_site(2, turn\_x1 - x\_offset, turn\_y1, z\_default);

set\_site(3, turn\_x0 - x\_offset, turn\_y0, z\_default);

wait\_all\_reach();

set\_site(1, turn\_x0 + x\_offset, turn\_y0, z\_default);

wait\_all\_reach();

set\_site(0, turn\_x1 - x\_offset, turn\_y1, z\_default);

set\_site(1, turn\_x0 - x\_offset, turn\_y0, z\_default);

set\_site(2, turn\_x1 + x\_offset, turn\_y1, z\_default);

set\_site(3, turn\_x0 + x\_offset, turn\_y0, z\_default);

wait\_all\_reach();

set\_site(3, turn\_x0 + x\_offset, turn\_y0, z\_up);

wait\_all\_reach();

set\_site(0, x\_default - x\_offset, y\_start + y\_step, z\_default);

set\_site(1, x\_default - x\_offset, y\_start + y\_step, z\_default);

set\_site(2, x\_default + x\_offset, y\_start, z\_default);

set\_site(3, x\_default + x\_offset, y\_start, z\_up);

wait\_all\_reach();

set\_site(3, x\_default + x\_offset, y\_start, z\_default);

wait\_all\_reach();

}

}

/\*

- go forward

- blocking function

- parameter step steps wanted to go

---------------------------------------------------------------------------\*/

void step\_forward()

{

move\_speed = leg\_move\_speed;

if (site\_now[2][1] == y\_start)

{

//leg 2&1 move

set\_site(2, x\_default + x\_offset, y\_start, z\_up);

wait\_all\_reach();

set\_site(2, x\_default + x\_offset, y\_start + 2 \* y\_step, z\_up);

wait\_all\_reach();

set\_site(2, x\_default + x\_offset, y\_start + 2 \* y\_step, z\_default);

wait\_all\_reach();

move\_speed = body\_move\_speed;

set\_site(0, x\_default + x\_offset, y\_start, z\_default);

set\_site(1, x\_default + x\_offset, y\_start + 2 \* y\_step, z\_default);

set\_site(2, x\_default - x\_offset, y\_start + y\_step, z\_default);

set\_site(3, x\_default - x\_offset, y\_start + y\_step, z\_default);

wait\_all\_reach();

move\_speed = leg\_move\_speed;

set\_site(1, x\_default + x\_offset, y\_start + 2z\_up);

wait\_all\_reach();

set\_site(1, x\_default + x\_offset, y\_start, z\_up);

wait\_all\_reach();

set\_site(1, x\_default + x\_offset, y\_start, z\_default);

wait\_all\_reach();

}

else

{

//leg 0&3 move

set\_site(0, x\_default + x\_offset, y\_start, z\_up);

wait\_all\_reach();

set\_site(0, x\_default + x\_offset, y\_start + 2 \* y\_step, z\_up);

wait\_all\_reach();

set\_site(0, x\_default + x\_offset, y\_start + 2 \* y\_step, z\_default);

wait\_all\_reach();

move\_speed = body\_move\_speed;

set\_site(0, x\_default - x\_offset, y\_start + y\_step, z\_default);

set\_site(1, x\_default - x\_offset, y\_start + y\_step, z\_default);

set\_site(2, x\_default + x\_offset, y\_start, z\_default);

set\_site(3, x\_default + x\_offset, y\_start + 2 \* y\_step, z\_default);

wait\_all\_reach();

move\_speed = leg\_move\_speed;

set\_site(3, x\_default + x\_offset, y\_start + 2 \* y\_step, z\_up);

wait\_all\_reach();

set\_site(3, x\_default + x\_offset, y\_start, z\_up);

wait\_all\_reach();

set\_site(3, x\_default + x\_offset, y\_start, z\_default);

wait\_all\_reach();

}

}

/\*

- go back

- blocking function

- parameter step steps wanted to go

---------------------------------------------------------------------------\*/

void step\_back()

{

move\_speed = leg\_move\_speed;

if (site\_now[3][1] == y\_start)

{

//leg 3&0 move

set\_site(3, x\_default + x\_offset, y\_start, z\_up);

wait\_all\_reach();

set\_site(3, x\_default + x\_offset, y\_start + 2 \* y\_step, z\_up);

wait\_all\_reach();

set\_site(3, x\_default + x\_offset, y\_start + 2 \* y\_step, z\_default);

wait\_all\_reach();

move\_speed = body\_move\_speed;

set\_site(0, x\_default + x\_offset, y\_start + 2 \* y\_step, z\_default);

set\_site(1, x\_default + x\_offset, y\_start, z\_default);

set\_site(2, x\_default - x\_offset, y\_start + y\_step, z\_default);

set\_site(3, x\_default - x\_offset, y\_start + y\_step, z\_default);

wait\_all\_reach();

move\_speed = leg\_move\_speed;

set\_site(0, x\_default + x\_offset, y\_start + 2 \* y\_step, z\_up);

wait\_all\_reach();

set\_site(0, x\_default + x\_offset, y\_start, z\_up);

wait\_all\_reach();

set\_site(0, x\_default + x\_offset, y\_start, z\_default);

wait\_all\_reach();

}

else

{

//leg 1&2 move

set\_site(1, x\_default + x\_offset, y\_start, z\_up);

wait\_all\_reach();

set\_site(1, x\_default + x\_offset, y\_start + 2 \* y\_step, z\_up);

wait\_all\_reach();

set\_site(1, x\_default + x\_offset, y\_start + 2 \* y\_step, z\_default);

wait\_all\_reach();

move\_speed = body\_move\_speed;

set\_site(0, x\_default - x\_offset, y\_start + y\_step, z\_default);

set\_site(1, x\_default - x\_offset, y\_start + y\_step, z\_default);

set\_site(2, x\_default + x\_offset, y\_start + 2 \* y\_step, z\_default);

set\_site(3, x\_default + x\_offset, y\_start, z\_default);

wait\_all\_reach();

move\_speed = leg\_move\_speed;

set\_site(2, x\_default + x\_offset, y\_start + 2 \* y\_step, z\_up);

wait\_all\_reach();

set\_site(2, x\_default + x\_offset, y\_start, z\_up);

wait\_all\_reach();

set\_site(2, x\_default + x\_offset, y\_start, z\_default);

wait\_all\_reach();

}

}

// add by RegisHsu

void body\_left(int i)

{

set\_site(0, site\_now[0][0] + i, KEEP, KEEP);

set\_site(1, site\_now[1][0] + i, KEEP, KEEP);

set\_site(2, site\_now[2][0] - i, KEEP, KEEP);

set\_site(3, site\_now[3][0] - i, KEEP, KEEP);

wait\_all\_reach();

}

void body\_right(int i)

{

set\_site(0, site\_now[0][0] - i, KEEP, KEEP);

set\_site(1, site\_now[1][0] - i, KEEP, KEEP);

set\_site(2, site\_now[2][0] + i, KEEP, KEEP);

set\_site(3, site\_now[3][0] + i, KEEP, KEEP);

wait\_all\_reach();

}

void hand\_wave(int i)

{

float x\_tmp;

float y\_tmp;

float z\_tmp;

move\_speed = 1;

if (site\_now[3][1] == y\_start)

{

body\_right(15);

x\_tmp = site\_now[2][0];

y\_tmp = site\_now[2][1];

z\_tmp = site\_now[2][2];

move\_speed = body\_move\_speed;

for (int j = 0; j < i; j++)

{

set\_site(2, turn\_x1, turn\_y1, 50);

wait\_all\_reach();

set\_site(2, turn\_x0, turn\_y0, 50);

wait\_all\_reach();

}

set\_site(2, x\_tmp, y\_tmp, z\_tmp);

wait\_all\_reach();

move\_speed = 1;

body\_left(15);

}

else

{

body\_left(15);

x\_tmp = site\_now[0][0];

y\_tmp = site\_now[0][1];

z\_tmp = site\_now[0][2];

move\_speed = body\_move\_speed;

for (int j = 0; j < i; j++)

{

set\_site(0, turn\_x1, turn\_y1, 50);

wait\_all\_reach();

set\_site(0, turn\_x0, turn\_y0, 50);

wait\_all\_reach();

}

set\_site(0, x\_tmp, y\_tmp, z\_tmp);

wait\_all\_reach();

move\_speed = 1;

body\_right(15);

}

}

void hand\_shake(int i)

{

float x\_tmp;

float y\_tmp;

float z\_tmp;

move\_speed = 1;

if (site\_now[3][1] == y\_start)

{

body\_right(15);

x\_tmp = site\_now[2][0];

y\_tmp = site\_now[2][1];

z\_tmp = site\_now[2][2];

move\_speed = body\_move\_speed;

for (int j = 0; j < i; j++)

{

set\_site(2, x\_default - 30, y\_start + 2 \* y\_step, 55);

wait\_all\_reach();

set\_site(2, x\_default - 30, y\_start + 2 \* y\_step, 10);

wait\_all\_reach();

}

set\_site(2, x\_tmp, y\_tmp, z\_tmp);

wait\_all\_reach();

move\_speed = 1;

body\_left(15);

}

else

{

body\_left(15);

x\_tmp = site\_now[0][0];

y\_tmp = site\_now[0][1];

z\_tmp = site\_now[0][2];

move\_speed = body\_move\_speed;

for (int j = 0; j < i; j++)

{

set\_site(0, x\_default - 30, y\_start + 2 \* y\_step, 55);

wait\_all\_reach();

set\_site(0, x\_default - 30, y\_start + 2 \* y\_step, 10);

wait\_all\_reach();

}

set\_site(0, x\_tmp, y\_tmp, z\_tmp);

wait\_all\_reach();

move\_speed = 1;

body\_right(15);

}

}

void head\_up(int i)

{

set\_site(0, KEEP, KEEP, site\_now[0][2] - i);

set\_site(1, KEEP, KEEP, site\_now[1][2] + i);

set\_site(2, KEEP, KEEP, site\_now[2][2] - i);

set\_site(3, KEEP, KEEP, site\_now[3][2] + i);

wait\_all\_reach();

}

void head\_down(int i)

{

set\_site(0, KEEP, KEEP, site\_now[0][2] + i);

set\_site(1, KEEP, KEEP, site\_now[1][2] - i);

set\_site(2, KEEP, KEEP, site\_now[2][2] + i);

set\_site(3, KEEP, KEEP, site\_now[3][2] - i);

wait\_all\_reach();

}

void body\_dance(int i)

{

float x\_tmp;

float y\_tmp;

float z\_tmp;

float body\_dance\_speed = 2;

sit();

move\_speed = 1;

set\_site(0, x\_default, y\_default, KEEP);

set\_site(1, x\_default, y\_default, KEEP);

set\_site(2, x\_default, y\_default, KEEP);

set\_site(3, x\_default, y\_default, KEEP);

wait\_all\_reach();

//stand();

set\_site(0, x\_default, y\_default, z\_default - 20);

set\_site(1, x\_default, y\_default, z\_default - 20);

set\_site(2, x\_default, y\_default, z\_default - 20);

set\_site(3, x\_default, y\_default, z\_default - 20);

wait\_all\_reach();

move\_speed = body\_dance\_speed;

head\_up(30);

for (int j = 0; j < i; j++)

{

if (j > i / 4)

move\_speed = body\_dance\_speed \* 2;

if (j > i / 2)

move\_speed = body\_dance\_speed \* 3;

set\_site(0, KEEP, y\_default - 20, KEEP);

set\_site(1, KEEP, y\_default + 20, KEEP);

set\_site(2, KEEP, y\_default - 20, KEEP);

set\_site(3, KEEP, y\_default + 20, KEEP);

wait\_all\_reach();

set\_site(0, KEEP, y\_default + 20, KEEP);

set\_site(1, KEEP, y\_default - 20, KEEP);

set\_site(2, KEEP, y\_default + 20, KEEP);

set\_site(3, KEEP, y\_default - 20, KEEP);

wait\_all\_reach();

}

move\_speed = body\_dance\_speed;

head\_down(30);

}

/\*

- microservos service /timer interrupt function/50Hz

- when set site expected,this function move the end point to it in a straight line

- temp\_speed[4][3] should be set before set expect site,it make sure the end point

move in a straight line,and decide move speed.

---------------------------------------------------------------------------\*/

void servo\_service(void)

{

sei();

static float alpha, beta, gamma;

for (int i = 0; i < 4; i++)

{

for (int j = 0; j < 3; j++)

{

if (abs(site\_now[i][j] - site\_expect[i][j]) >= abs(temp\_speed[i][j]))

site\_now[i][j] += temp\_speed[i][j];

else

site\_now[i][j] = site\_expect[i][j];

}

cartesian\_to\_polar(alpha, beta, gamma, site\_now[i][0], site\_now[i][1], site\_now[i][2]);

polar\_to\_servo(i, alpha, beta, gamma);

}

rest\_counter++;

}

/\*

- set one of end points' expect site

- this founction will set temp\_speed[4][3] at same time

- non - blocking function

---------------------------------------------------------------------------\*/

void set\_site(int leg, float x, float y, float z)

{

float length\_x = 0, length\_y = 0, length\_z = 0;

if (x != KEEP)

length\_x = x - site\_now[leg][0];

if (y != KEEP)

length\_y = y - site\_now[leg][1];

if (z != KEEP)

length\_z = z - site\_now[leg][2];

float length = sqrt(pow(length\_x, 2) + pow(length\_y, 2) + pow(length\_z, 2));

temp\_speed[leg][0] = length\_x / length \* move\_speed \* speed\_multiple;

temp\_speed[leg][1] = length\_y / length \* move\_speed \* speed\_multiple;

temp\_speed[leg][2] = length\_z / length \* move\_speed \* speed\_multiple;

if (x != KEEP)

site\_expect[leg][0] = x;

if (y != KEEP)

site\_expect[leg][1] = y;

if (z != KEEP)

site\_expect[leg][2] = z;

}

/\*

- wait one of end points move to expect site

- blocking function

---------------------------------------------------------------------------\*/

void wait\_reach(int leg)

{

while (1)

if (site\_now[leg][0] == site\_expect[leg][0])

if (site\_now[leg][1] == site\_expect[leg][1])

if (site\_now[leg][2] == site\_expect[leg][2])

break;

}

/\*

- wait all of end points move to expect site

- blocking function

---------------------------------------------------------------------------\*/

void wait\_all\_reach(void)

{

for (int i = 0; i < 4; i++)

wait\_reach(i);

}

/\*

- trans site from cartesian to polar

- mathematical model 2/2

---------------------------------------------------------------------------\*/

void cartesian\_to\_polar(volatile float &alpha, volatile float &beta, volatile float &gamma, volatile float x, volatile float y, volatile float z)

{

//calculate w-z degree

float v, w;

w = (x >= 0 ? 1 : -1) \* (sqrt(pow(x, 2) + pow(y, 2)));

v = w - length\_c;

alpha = atan2(z, v) + acos((pow(length\_a, 2) - pow(length\_b, 2) + pow(v, 2) + pow(z, 2)) / 2 / length\_a / sqrt(pow(v, 2) + pow(z, 2)));

beta = acos((pow(length\_a, 2) + pow(length\_b, 2) - pow(v, 2) - pow(z, 2)) / 2 / length\_a / length\_b);

//calculate x-y-z degree

gamma = (w >= 0) ? atan2(y, x) : atan2(-y, -x);

//trans degree pi->180

alpha = alpha / pi \* 180;

beta = beta / pi \* 180;

gamma = gamma / pi \* 180;

}

/\*

- trans site from polar to microservos

- mathematical model map to fact

- the errors saved in eeprom will be add

---------------------------------------------------------------------------\*/

void polar\_to\_servo(int leg, float alpha, float beta, float gamma)

{

if (leg == 0)

{

alpha = 90 - alpha;

beta = beta;

gamma += 90;

}

else if (leg == 1)

{

alpha += 90;

beta = 180 - beta;

gamma = 90 - gamma;

}

else if (leg == 2)

{

alpha += 90;

beta = 180 - beta;

gamma = 90 - gamma;

}

else if (leg == 3)

{

alpha = 90 - alpha;

beta = beta;

gamma += 90;

}

servo[leg][0].write(alpha);

servo[leg][1].write(beta);

servo[leg][2].write(gamma);

}

***THANK YOU***