Interdisciplinary course of

Design and Robotics

5° edition, 2017

Professors:
Andrea Bonarini, DEIB department, Politecnico di Milano
Maximiliano Romero , DPAC department, Università IUAV, Venezia

Tutors:
Andrea Brivio, Yael Bendit,

Students:
Simone Alborghetti, school of Product Design
Luca Morreale, school of CS and Engineering
Andrea Di Giosaffatte, school of CS and Engineering
Bianca Frigerio, school of Biomedical Engineering
Lucas Paiva Delgado, school of Electronic Engineering

Index

Abstract

In this document, aimed to report the work done to achieve the final prototype, it is possible to find the details of all this work and development regarding the robot of the group 01, CIN CIN. As first section of this report you will encounter a brief description of our robot with her interactions. Then, it is explained the state of the art which inspired us. Based on that we built our concept which is described in the relative section. Afterwards, the full development process is described, such as her interactions, shape, electronics, mechanics and informatics. The last section shows the conclusions about our work.

Description

For the 5th edition of the “Interdisciplinary course of Design and robotics”, the groups were asked to design and prototype a robot for the advertisement field, specifically for a restaurant.
The robot of this group, CIN CIN, is a mascot of a typical Chinese restaurant, her goal is to attract people, interact with them and try to convince them to enter. In order to do so, she likes to dance to Chinese music while watching people passing by on the sidewalk. When someone passes by within a certain range, she tries to get the person closer, saying “Hey Hey, come here!” first in italian, then in Chinese. If the person actually gets closer, she introduce herself saying “My name is CIN CIN, nice to meet you”. At this point, if the person is still there she will try to show the restaurant’s menu which is in her pouch. Afterwards, CIN CIN will try to convince the person to take a selfie with her and get in the restaurant to see the picture, saying that the customer will get a discount. At any point, If the human-robot interaction stops, CIN CIN will go back to dance looking for new potential customers.

Research

State of the Art

Informations were taken from the web, firstly in google and youtube, then on design and technology websites, such as fubiz.net, yankodesign.com and wired.com. In this way we have been able to get some inspirations by similar kind of robots. The keywords we searched for were: “Chinese design Alessi”, “Chinese culture”, “Chinese traditional food”, “Chinese traditional instrumental music”, “Chinese soybean”. The character of our robot is a Chinese caricature, so the front end is designed in order to follow the Chinese customs and semiotics. For this reason, we developed our research asking directly to Chinese people impressions and suggestions about the concept.

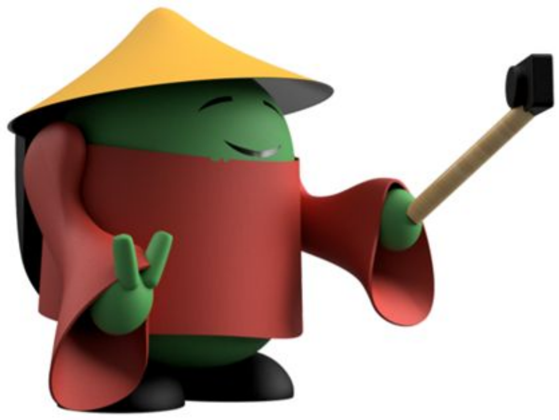
Concept

Inspiration

CIN CIN is inspired by a funny caricature of a Chinese soybean. The idea is to create a funny caricature to convince people to enter in the restaurant by establishing an interaction with her, for instance by offering them the restaurant menu and taking a selfie with her.

Representation

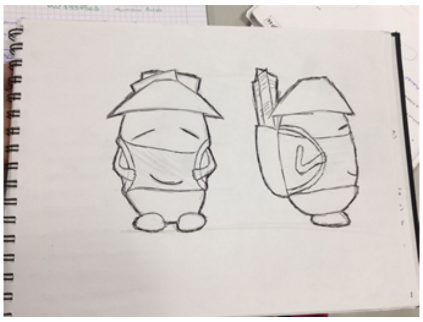
CIN CIN has a green skin color, in order to portray a soybean. The typical yellow chinese hat and the red-gold silk cloth, gives her the typical chinese look. Her behaviour is modeled to imitate chinese people. CIN CIN also likes to take a lot of pictures.

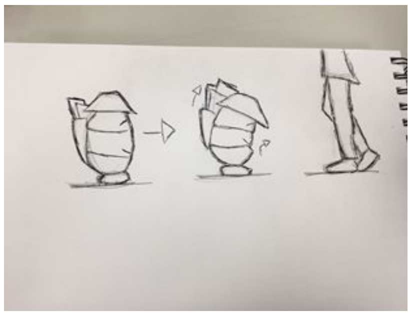


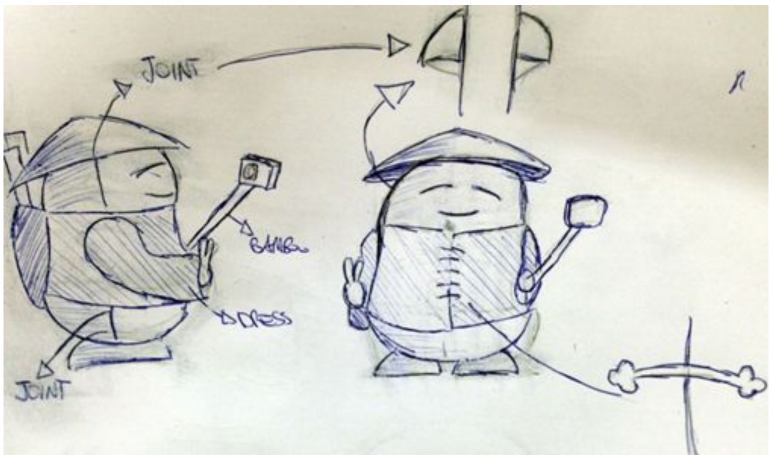
Development

Interaction

The robot will try to attract people passing by dancing to the tune of Chinese music. She will call the people saying out loud: “Hey Hey come here!”.
Once the person is in front of her, she introduces herself with a bow saying “My name is CIN CIN, nice to meet you”. Afterwards, she asks the person to have a look at the menu in the pouch saying: “Would you like to take a look at our menu? Take it from the pouch!”. When the client puts back the menu, CIN CIN says: “Have you seen how many good dishes we have?”. After a few seconds she asks: “Would you like to take a selfie with me? Just touch my hat and I will shoot the picture! Cheese!”. After the photo, she says: “What a wonderful picture! I’m going to send it to the cash register and they will give you a discount! It has been a pleasure to meet you.”. After making the photo, while CIN CIN is dancing, the Wi-Fi of the camera turns on and the cashier can download the photo through the phone application ‘iSmart DV’.



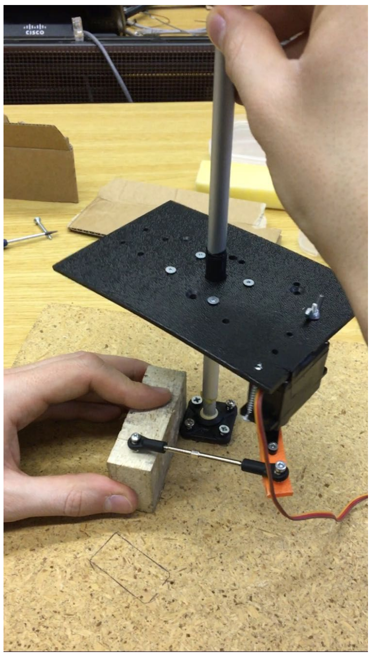
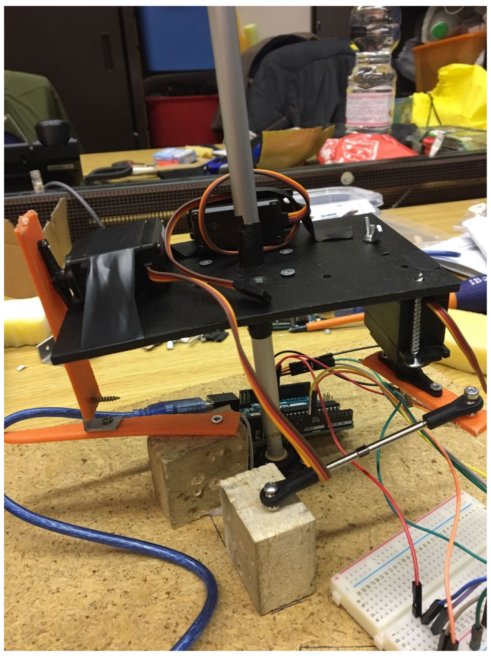
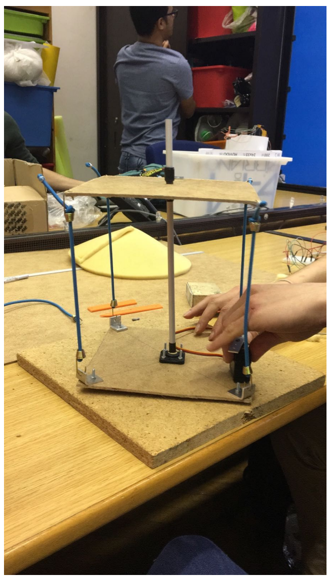
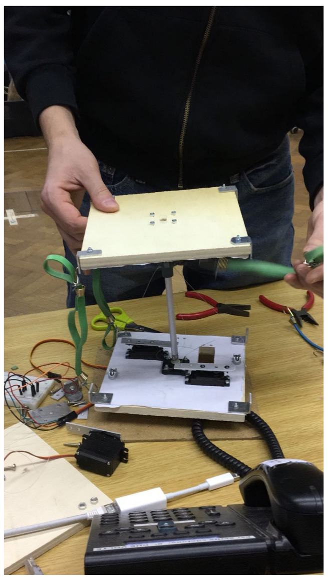
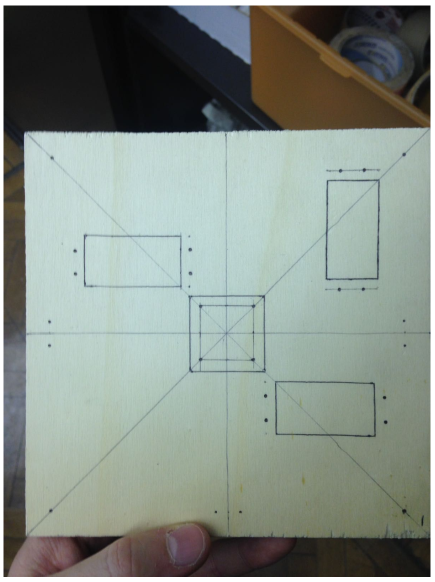
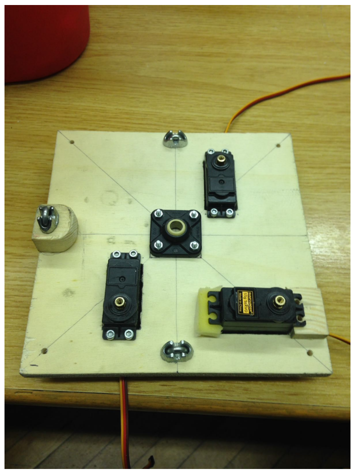


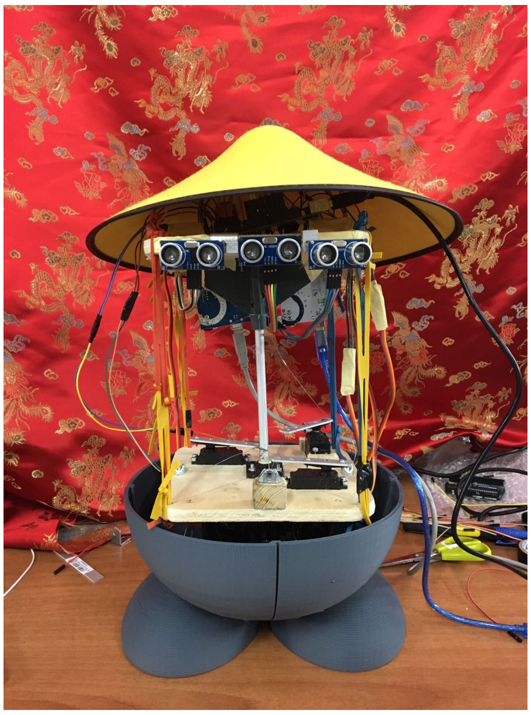


Shape

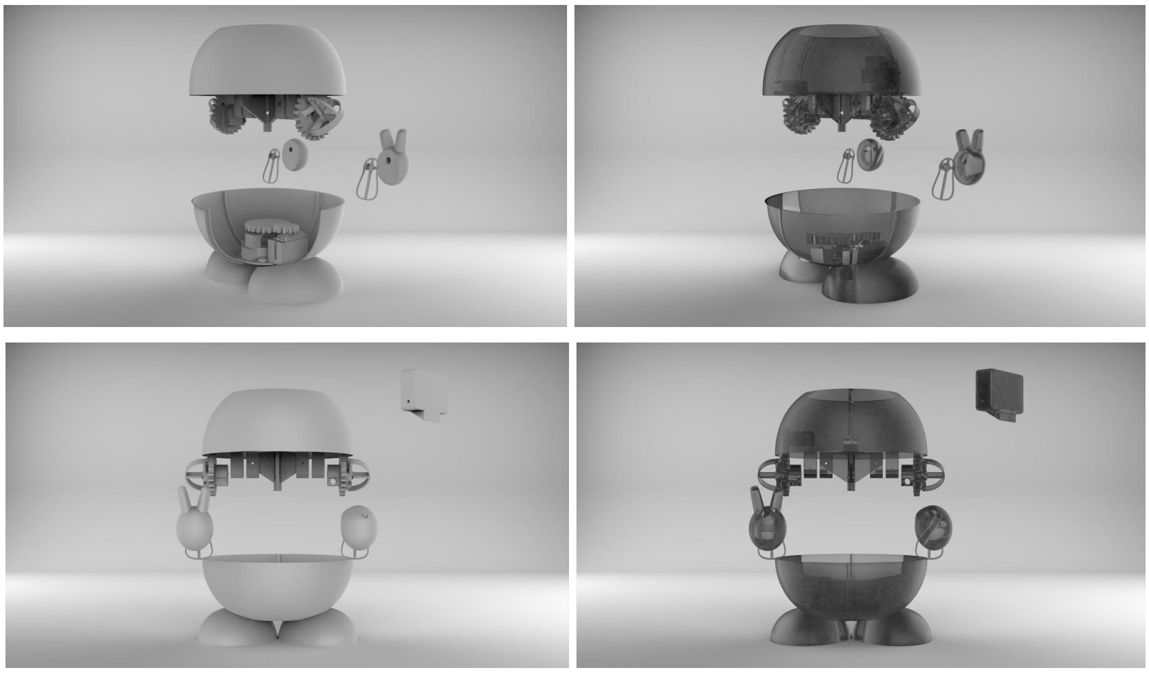
CINCIN has mainly a rigid body, while the center part is soft. It has also two little feet. Every part has been 3D printed with PLA. On the head she has a chinese hat. The shape itself comes from the soybean, a bean typically eaten in China with the name of “edamame”. Being a caricature, we wanted CIN CIN to be slightly chubby and funny.

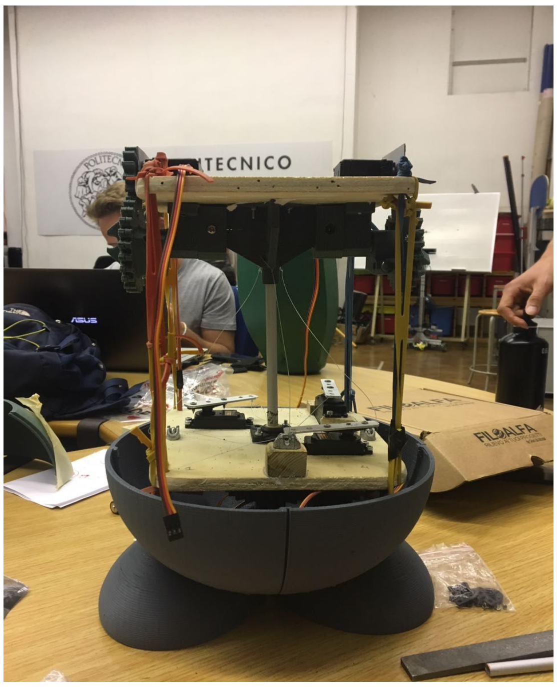
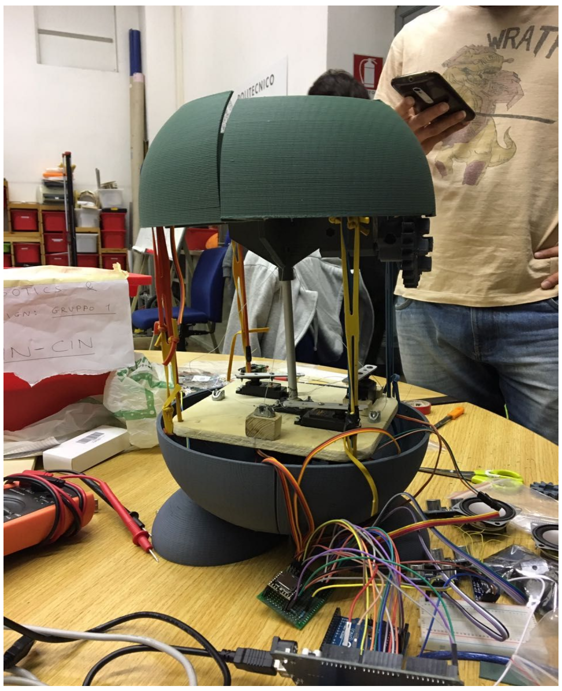
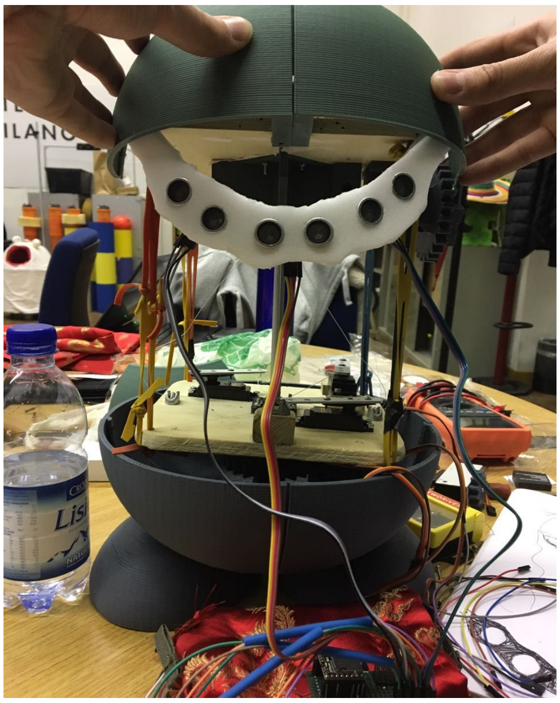
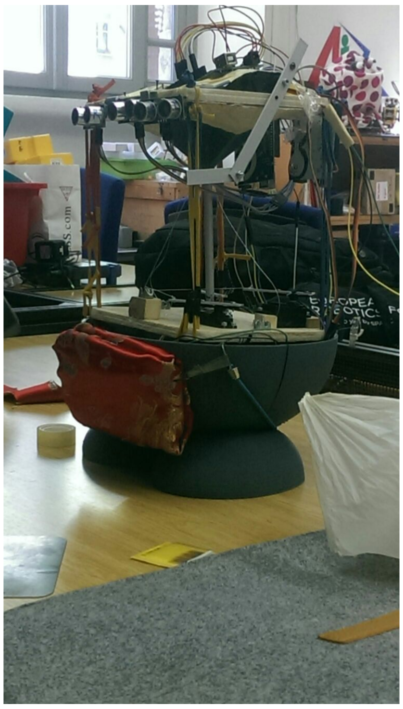
First deadline prototypes



Final deadline prototype

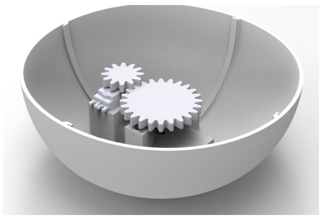
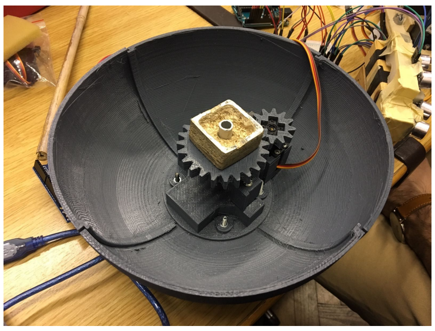
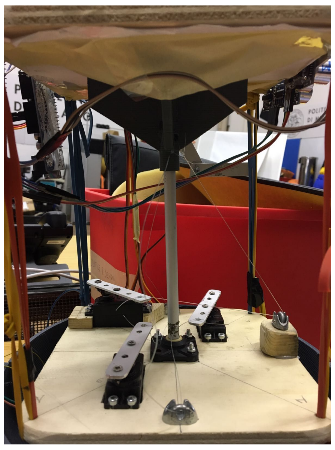
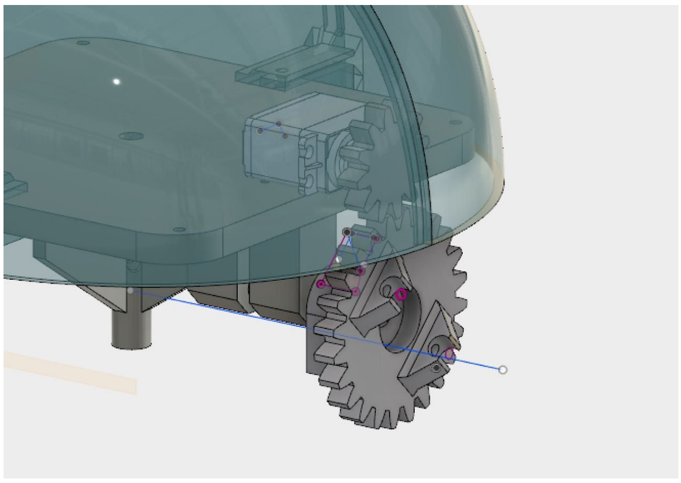
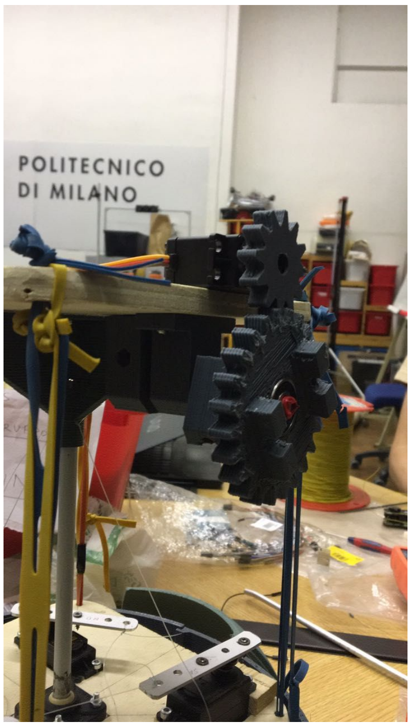



POLITECNICO
FILOALFA

POLITECNICO
GRUPPO 1



Mechanics

CIN CIN is a robot equipped with 6 servo motors. The first one manages the rotation of the basement around the z-axis. The second and the third one handle the left-right dance movement of the robot. The fourth motor manage to make CIN-CIN bow. The last two are used for the movements of the arms. After the first prototype shown on March 3th, we tried a different approach in order to simplify the mechanics. At the beginning we used some rigid staff to force the rotation of the body. Then we tried with elastic bands, it allows us to obtain more human-like movements. The mechanics is really simple and its components well

organized. For the rotation of the body and for the movement of the arms we used gear wheels in order to ease and make the movements stronger (as in the photo).
We exploited the same solution for the arms, moreover this allowed us to shift the shoulders' position below. Thus, our robot has a more human shape. (as shown in an image below) The dimension of the gears varies depending of the usage.
Due to the changes of the weight during the process the elastic bands have to be properly tuned to adjust the tilt.

Electronics

The electronics is relatively simple. It mainly involves components self-contained which do not need external parts. Our robot requires three distance (ultrasound) sensors to perceive clients. Two speakers and a module to reproduce music and the voice of our robot.

Everything is connected through a breadboard in order to simplify eventual changes and handle connections easily. From that breadboard all cables are connected with an Arduino Mega. That specific Arduino's version can handle more than 13 pins, thus a single board is needed.
For the first prototype two external boards were required to provide power to all components. For the final prototype only one power source is required (missing spec of power source); it will provide current to all components.
To control the presence of the menu in the pouch we used a photoresistor. To take the picture, we used a touch metal sensor.
In order to improve the internal organization of the cables, and thus of the robot, many boards have been created. In this way most the electronics will be centralized, so it will be easier to replace cables or, eventually, boards. For small sensors and components small boards have been created and spread in the robot.

Bill of material

- 6x SunFounder Metal Gear Digital RC Servo Motor High Torque 12 kg/cm(4.8V) or 15kg/cm(6V) - 70.80 €
- 1x DFPlayer Mini MP3 Module - 12,00 €
- 3x Sonar Sensor for Arduino - 9.99 €
- 2x Speakers 4ohm 3W - 7,27 €
- 1x Arduino Mega - 7.95 €
- 1x Y8 Action Camera - 39.99 €
- 1x Touch metal sensor 1.5 €
- 1x silk dress 20 €
- 2x PLA plastic for 3D print 40 €
- 3x Aluminium tube 6 €
- 1x Set jumper 5 €
- 2x Elastic band 10 €
- 1x Cardboard plates 4 €
- 1x Wood components (foot's rods and two layers) 5 €
- 1x Uniposca 4.5 €
- 1x menu print 3.5 €
- 3x bearing 6 €

Informatics

To efficiently perceive the clients passing by, CIN-CIN has to be coordinate movements and measurements. Thus, the movements cannot be performed directly, but they had to be broken down into smaller ones. During every partial movement, a measurements is carried out. This makes the code more complicated. Still we were able to put everything together. Until the first deadline in March most of our code concerned only the movements of the robot without the control of the audio part, the camera and the menu. Now we combined everything together as described below.

In order to achieve these results we used the following libraries:

```
#include <NewPing.h>                  // sonar library
#include <Servo.h>                    // servo library
#include <SoftwareSerial.h>           // mp3 player library
#include <DFPlayer_Mini_Mp3.h>        // mp3 player library
```

The following are the main functions we used:

void cin_cin_dance();   // CIN CIN dance the chinese music when nobody is front of her
As long as no one is at a distance higher than 130 cm, she dances exploiting 2 servo motors that allows the right and left tilt movements and moving her arms up and down.

void cin_cin_engagement();   // CIN CIN calls the people that stand in front of it
When someone is in front of the robot, she tries to engage him exploiting some mp3 audios. After the engagement, she introduces itself with a bow.

void cin_cin_menu();   // CIN CIN offers the menu to people standing in front of her
As long as the person is reading the menu, CIN CIN stays still. When the person has finished reading the menu and has put the menu back in the pouch she asks if the person liked the dishes.

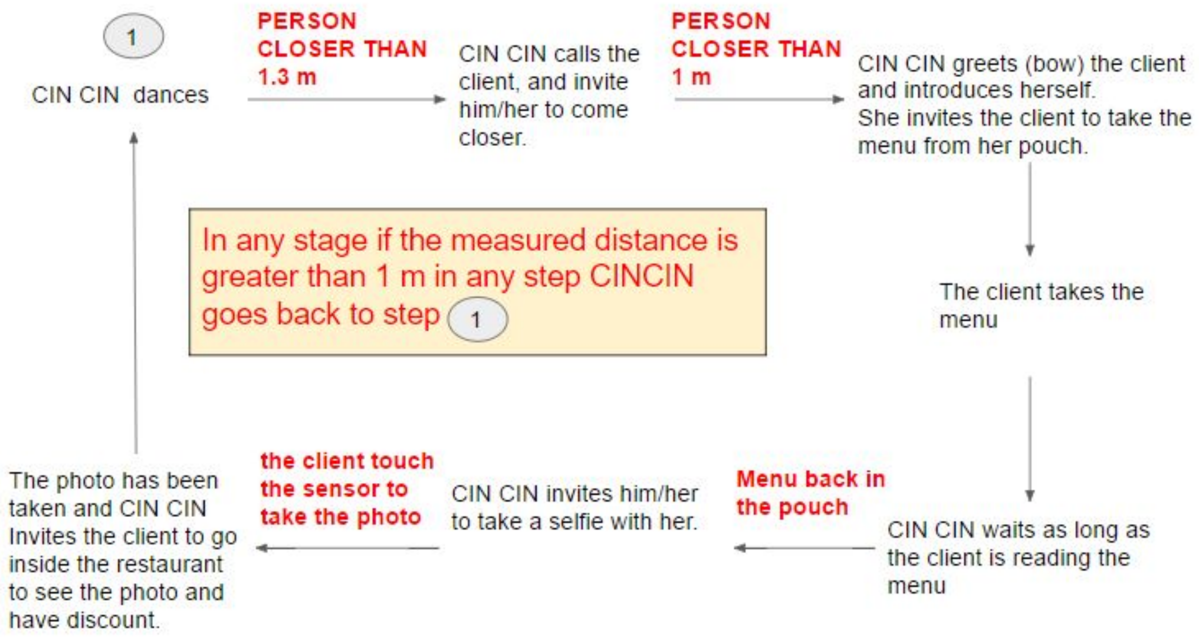
void cin_cin_selfie();   // CIN CIN turn horizontally and lifts the selfie stick
Cin Cin rotates and rises its arm to take a selfie with the client. When the client is ready can touch the metal touch sensor to take the photo. Then Cin Cin will rotate back to the rest position.

Conclusion

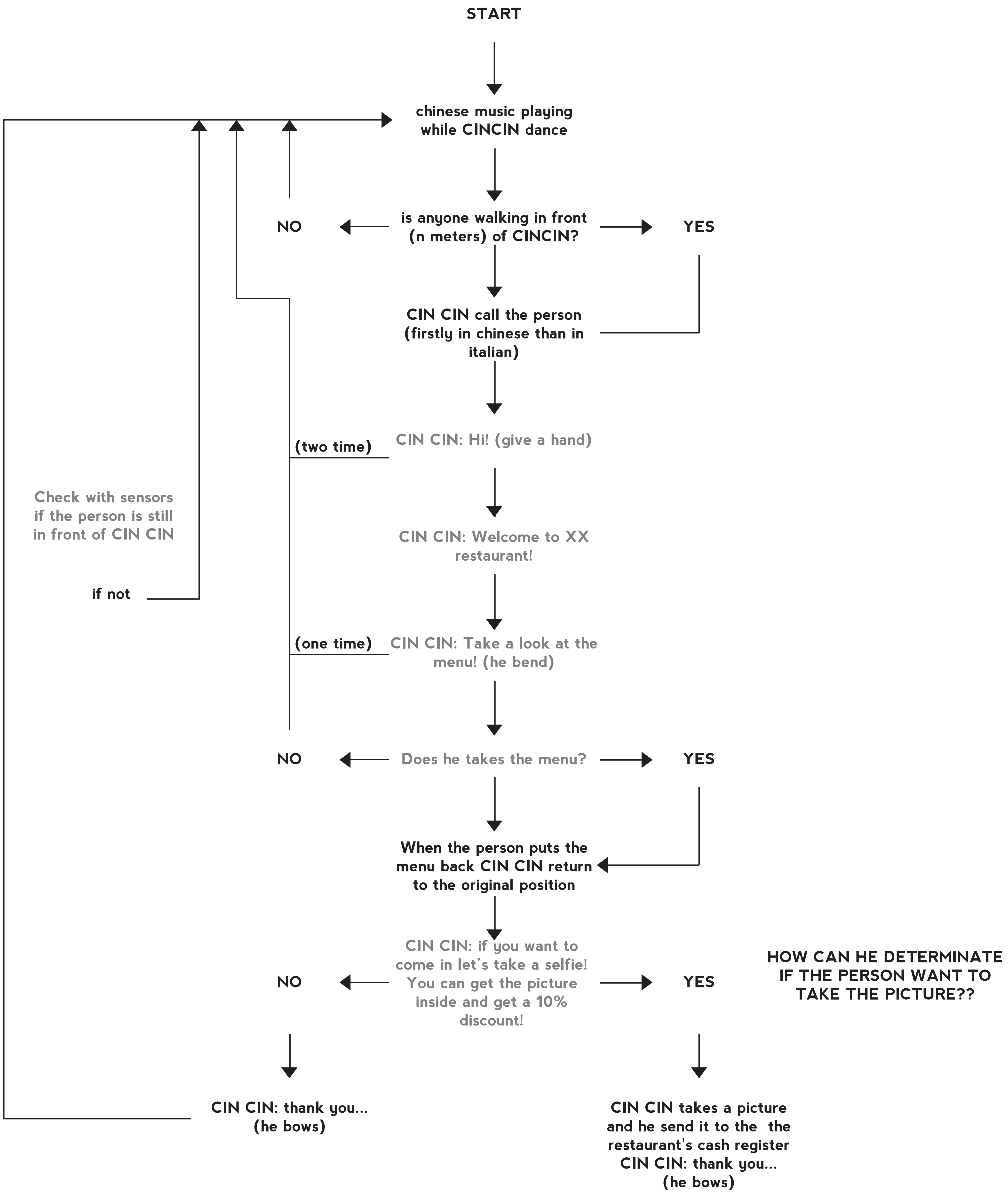
To achieve such result we worked all together. We value a lot team-work and think that is what allowed us to reach such point in such short time. The most difficult part was to produce a mechanics that works properly. This was only possible thanks to the combined effort of designers and engineers. The 3D model was fundamental to make everyone understand the plan, which was not so simple in early stages. During this two months we had many problem due to the alimentation, we broke 5 servo motors and 3 ultrasonic sensors but the determination and the contribution of each single person of our group has allowed us to produce the robot proposed on time.

Annex

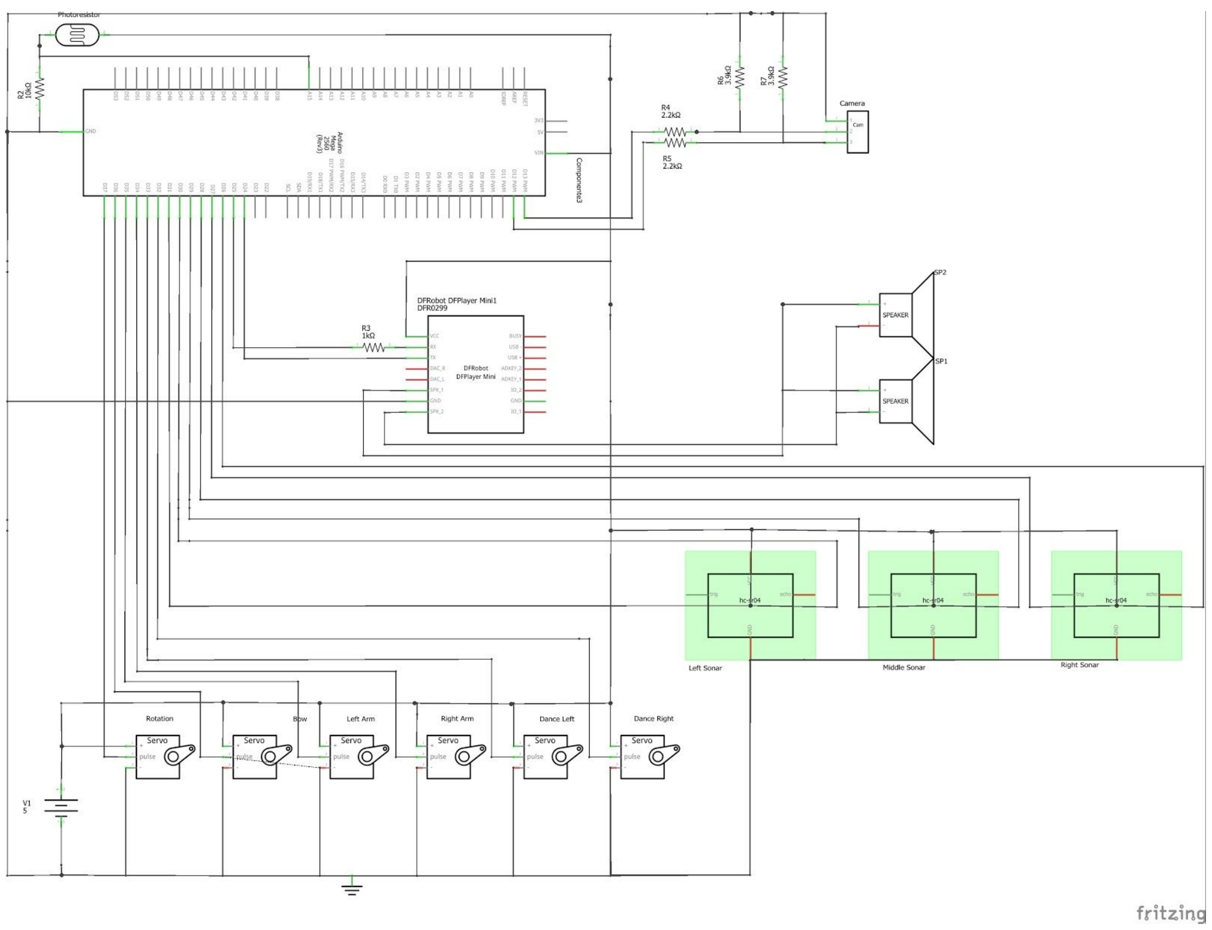
Interaction



1
CIN CIN dances
PERSON CLOSER THAN 1.3 m
CIN CIN calls the client, and invite him/her to come closer.
PERSON CLOSER THAN 1 m
CIN CIN greets (bow) the client and introduces herself. She invites the client to take the menu from her pouch.
The client takes the menu
CIN CIN waits as long as the client is reading the menu
Menu back in the pouch
CIN CIN invites him/her to take a selfie with her.
the client touch the sensor to take the photo
The photo has been taken and CIN CIN Invites the client to go inside the restaurant to see the photo and have discount.
In any stage if the measured distance is greater than 1 m in any step CINCIN goes back to step 1


START
chinese music playing while CINCIN dance
is anyone walking in front (n meters) of CINCIN?
NO
YES
CIN CIN call the person (firstly in chinese than in italian)
(two time)
CIN CIN: Hi! (give a hand)
Check with sensors if the person is still in front of CIN CIN
if not
CIN CIN: Welcome to XX restaurant!
(one time)
CIN CIN: Take a look at the menu! (he bend)
NO
Does he takes the menu?
YES
When the person puts the menu back CIN CIN return to the original position
NO
CIN CIN: if you want to come in let's take a selfie! You can get the picture inside and get a 10% discount!
YES
HOW CAN HE DETERMINATE IF THE PERSON WANT TO TAKE THE PICTURE??
CIN CIN: thank you... (he bows)
CIN CIN takes a picture and he send it to the the restaurant's cash register CIN CIN: thank you... (he bows)

Electronics



General schematics of the overall circuit

Informatics

The code has been uploaded on Beep as a pdf.

```
/* IMPORTANTISSIMO: I SONAR VANNO ALIMENTATI A PARTE (5 V)
 Trigger -> Grigio
 Echo -> Viola
 Gnd -> Verde
 Vcc -> Blu
*/

/*
 mp3 player: blu 13, verde 12
 motori: giallo 11, blu 10, rosso 9, beige 8, grigio 16, arancione 15
 sonar: arancione scuro 7, verde associato 6, arancione più chiaro 5, verde associato 4, beige 3, verde associato 2
 camera: blu 19, viola 20
 fotoresistenza: blu A0

 breadboard-breadboard:
   sonar: grigio GROUND, viola TENSIONE
   motori: beige GROUND, blu TENSIONE ...

*/

// ---- LIBRARIES ----
#include <NewPing.h>
#include <Servo.h>
#include <SoftwareSerial.h>
#include <DFPlayer_Mini_Mp3.h>

// ---- Constants ----
#define TRUE true
#define FALSE false
#define ON 1
#define OFF 0

// ---- SONAR DISTANCES DEBUGGING ----
#define DEBUG 1

// ---- MOTOR PINS ----
#define LEFT_PIN 9
#define RIGHT_PIN 10
#define BOW_PIN 15
```

```
#define RIGHT_ARM 11

#define LEFT_ARM 16

#define ROTATION_PIN 8


// ---- MOTOR INITIAL & FINAL POSITIONS ----

#define LEFT_STRETCHED_POSITION 30

#define LEFT_REST_POSITION 180

#define RIGHT_STRETCHED_POSITION 30

#define RIGHT_REST_POSITION 180

#define BOW_STRETCHED_POSITION 190

#define BOW_REST_POSITION 90

#define RARM_STRETCHED_POSITION 10

#define RARM_REST_POSITION 60

#define LARM_STRETCHED_POSITION 120

#define LARM_REST_POSITION 90

#define ROTATION_REST_POSITION 10

#define ROTATION_STRETCHED_POSITION 100


// ---- CAMERA's DEALYS ----

#define TURN_ON_DELAY 200

#define TURN_OFF_DELAY 4000

#define SHOOT_DELAY 500        // if increased will take a video!

#define SWITCH_STAGE_DELAY 500

#define WIFI_ON_DELAY 4000

#define WIFI_OFF_DELAY 4000


// ---- DELAYS ----

#define BOW_DOWN_DELAY 1500

#define BOW_UP_DELAY 800


// ---- SONAR PINS ----

#define TRIG_PIN_LEFT 2

#define ECHO_PIN_LEFT 3

#define TRIG_PIN_MIDDLE 4

#define ECHO_PIN_MIDDLE 5

#define TRIG_PIN_RIGHT 6

#define ECHO_PIN_RIGHT 7


#define MEASUREMENTS_NUMBER 30
```

```
// ---- MP3 PLAYER PINS ----

#define RXPIN 12

#define TXPIN 13


// ---- SONAR PINS ----

#define TRIG_PIN_LEFT 2

#define ECHO_PIN_LEFT 3

#define TRIG_PIN_MIDDLE 4

#define ECHO_PIN_MIDDLE 5

#define TRIG_PIN_RIGHT 6

#define ECHO_PIN_RIGHT 7


// ---- SONAR MINIMUM THRESHOLD (CM) ----

#define SONAR_MINIMUM_THRESHOLD 10


// ---- CAMERA's PINS ----

#define CAMERA_PIN 20

#define CAMERA_SHOT_PIN 19


// ---- TOUCH SENSOR PIN ----

#define TOUCH_SENSOR_PIN 14


// ---- MAXIMUM SONAR DISTANCE (CM) ----

#define MAX_DISTANCE 300


// ---- PHOTO RESISTOR THRESHOLD ----

#define PHOTO_RESISTOR_THRESHOLD 15


// ---- STOPPING DANCE DISTANCE (CM) ----

#define STOP_DISTANCE 80


// ---- SOMEONE DISTANCE (CM)

#define SOMEONE_DISTANCE 80


// ---- SONAR INITIALIZATION ----

long distanceLeft, distanceMiddle, distanceRight;

NewPing sonarLeft(TRIG_PIN_LEFT, ECHO_PIN_LEFT, MAX_DISTANCE);

NewPing sonarMiddle(TRIG_PIN_MIDDLE, ECHO_PIN_MIDDLE, MAX_DISTANCE);
NewPing sonarRight(TRIG_PIN_RIGHT, ECHO_PIN_RIGHT, MAX_DISTANCE);


// ---- MOTOR INITIALIZATION ----
Servo leftMotor, rightMotor, bowMotor, rarmMotor, larmMotor, rotationMotor;


// ---- MP3 PLAYER INITIALIZATION ----
SoftwareSerial mp3Serial(RXPIN, TXPIN);


// ---- STATE VARIABLES OF CAMERA ----
int camState = OFF;
int wifiState = OFF;


// ---- MARSUPIUM ANALOG PIN ----
int photoResistor = A0;


// ---- CONTROL VARIABLES ----
boolean someone = false; // booleano che indica la presenza o meno dell'utente
boolean engagement = false;
boolean touched = false;
boolean mp3_stopper = false;



// ---- FUNCTION PROTOTYPES ----
void attachServos();
void initServos();
void initSerial();
void initAnalog();
void initTouch();
void initPinCamera();


void cin_cin_dance();
void cin_cin_engagement();
void cin_cin_menu();
void cin_cin_selfie();


void sayHi();
void playDanceMusic();
void greetClient();
void offerMenu();
void forgottenMenu();
void sponsorMenu();
void proposeSelfie();
void explainSelfie();
void cheese();
void discount();
void sayBye();
void sayHiChinese();
void cmon();
void credits();

void goBackFromLeft();
void goBackFromRight();

void bow();
void standup();
void rotateForSelfie();
void rotateBackFromSelfie();
void doSelfie();
void pullUpLeftArm();
void pullDownLeftArm();

void updateDistances();
boolean isClose();
boolean thereIsSomeone();
boolean thereIsMenu();
boolean isTouched();
boolean clientEngaged();

void turnOnCamera();
void turnOffCamera();
void takePicture();
void switchOnWiFi();
void switchOffWiFi();

void debug(char description, long value);
void debug(char *description, long value);
void debug(char description, int value);
void debug(char *description, int value);
void debug(char *message);



void setup() {

  // Inizializzazione dei motori e dei vari componenti
  // Settaggio dei pin di OUTPUT ed INPUT

  attachServos();

  initServos();

  initSerial();

  initPinCamera();

  initAnalog();

  initTouch();

  distanceLeft = 0;
  distanceMiddle = 0;
  distanceRight = 0;

  camState = ON;
  turnOffCamera();
}

void attachServos()
{
  leftMotor.attach(LEFT_PIN);
  rightMotor.attach(RIGHT_PIN);
  bowMotor.attach(BOW_PIN);
  rarmMotor.attach(RIGHT_ARM);
  larmMotor.attach(LEFT_ARM);
  rotationMotor.attach(ROTATION_PIN);
}
```

```
}

void initServos()
{
  leftMotor.write(LEFT_REST_POSITION);
  rightMotor.write(RIGHT_REST_POSITION);
  bowMotor.write(BOW_REST_POSITION);
  rarmMotor.write(RARM_REST_POSITION);
  larmMotor.write(LARM_REST_POSITION);
  rotationMotor.write(ROTATION_REST_POSITION);
}

void initSerial()
{
  Serial.begin(9600);
  mp3Serial.begin(9600);
  mp3_set_serial (mp3Serial); //set Serial for DFPlayer-mini mp3 module
  mp3_set_volume (30); //max volume is 30
}

void initAnalog()
{
  pinMode(photoResistor, INPUT);// Set photoResistor - A0 pin as an input
}

void initTouch()
{
  pinMode (TOUCH_SENSOR_PIN, INPUT);
}

void initPinCamera()
{
  pinMode(CAMERA_PIN, OUTPUT);
  pinMode(CAMERA_SHOT_PIN, OUTPUT);

  digitalWrite(CAMERA_PIN, HIGH);
  digitalWrite(CAMERA_SHOT_PIN, LOW);
}
```

```
}


void loop() {


  cin_cin_dance();
  cin_cin_engagement();
  if (someone)
    cin_cin_menu();
  if (someone)
    cin_cin_selfie();
}


void cin_cin_dance() {


  // Funzione che fa ballare Cin Cin con musica cinese di sottofondo
  // Mentre balla rileva se passa qualcuno entro 2 metri c.a.
  // Quando rileva qualcuno si fermano i motori, la musica e la funzione ritorna dopo aver settato il flag someone a TRUE


  someone = false;


  if (thereIsSomeone()) {
    return;
  }


  while (true) {


    playDanceMusic();


    for (int i = LEFT_REST_POSITION; i >= LEFT_STRETCHED_POSITION; i--) {
      leftMotor.write(i);
      if ((LEFT_REST_POSITION - i) % 10 == 0) {
        if (thereIsSomeone()) {
          goBackFromLeft();
          return;
        }
      }
    }




 for (int i = LEFT_STRETCHED_POSITION; i < LEFT_REST_POSITION; i++) {
      leftMotor.write(i);
      if ((i - LEFT_STRETCHED_POSITION) % 10 == 0) {
        if (thereIsSomeone()) {
          goBackFromLeft();
          return;
        }
      }
    }

    for (int i = RIGHT_REST_POSITION; i >= RIGHT_STRETCHED_POSITION; i--) {
      rightMotor.write(i);
      if ((RIGHT_REST_POSITION - i) % 10 == 0) {
        if (thereIsSomeone()) {
          goBackFromRight();
          return;
        }
      }
    }

    for (int i = RIGHT_STRETCHED_POSITION; i < RIGHT_REST_POSITION; i++) {
      rightMotor.write(i);
      if ((i - RIGHT_STRETCHED_POSITION) % 10 == 0) {
        if (thereIsSomeone()) {
          goBackFromRight();
          return;
        }
      }
    }
  }

}

void cin_cin_engagement()
{
  engagement = false;
  sayHi();  // contains delay
  sayHiChinese();
  switchOffWiFi();

  int i = 0;
  while (i < 5 && !engagement) {
    if (thereIsSomeone()) {
      bow();
      standup();
      greetClient();
      engagement = true;
    }
    i++;
  }
}


void cin_cin_menu() {


  // Funzione che fa presentare il ristorante e offrire il menù all'utente
  // Cin Cin presenta il ristorante all'utente per poi invitarlo a dare un'occhiata al menÃ¹ facendo un inchino
  // Se l'utente non prende il menÃ¹ allora Cin Cin si rialza e verifica che l'utente sia ancora lÃ¬, ed eventualmente setta il flag someone a FALSE prima di ritornare
  // Se l'utente Ã¨ ancora lÃ¬ riprova ad offrirgli il menÃ¹, se non viene preso nemmeno questa volta la funzione ritorna
  // Se l'utente prende il menÃ¹ Cin Cin resta inchinato fino a che non viene rimesso a posto, per poi ritornare dopo aver eventualmente settato il flag someone


  engagement = false;


  int i = 0;
  while (i < 5 && !engagement) {
    if (thereIsSomeone()) {
      offerMenu();
      engagement = true;
    }
    i++;
  }


  if (engagement) {
    delay(3000);
    if (!thereIsMenu()) {
      for (int k = 0; k < 200; k++)
      {
        if (thereIsMenu())
          k = 200;
        else
          delay(100);
        if (k == 199) {
          forgottenMenu();
          delay(2000);
        }
      }
      sponsorMenu();
    }
    else {
      offerMenu();
      delay(3000);
      if (!thereIsMenu()) {
        for (int w = 0; w < 200; w++)
        {
          if (thereIsMenu())
            w = 200;
          else
            delay(100);
          if (w == 199) {
            forgottenMenu();
            delay(2000);
          }
        }
        sponsorMenu();
      }
    }
  }
}

void cin_cin_selfie()
{
  engagement = FALSE;
  turnOnCamera();
  int i = 0;
  //rotazione cin cin a 80°
  while (i < 5 && !clientEngaged()) {
    if (thereIsSomeone()) {
      proposeSelfie();
      rotateForSelfie();
      doSelfie();
      rotateBackFromSelfie();
      if (touched)
        discount();
      engagement = TRUE;
      switchOnWiFi();
    }
    i++;
  }
  if (engagement) {
    sayBye();
    for (int z = 0; z < 40; z++)
    {
      if (isTouched()) {
        z = 40;
        credits();
      }
      else
        delay(100);
    }
    pullDownLeftArm();
  }
}

// -------------- HELPER FUNCTIONS --------------

void playDanceMusic()
{
  if (!mp3_stopper) {
    mp3_play(1); // play 0001.mp3
    mp3_single_loop(true);
    mp3_stopper = true;
    delay(3000);
  }
}
```

```
void sayHi()
{
  mp3_play(2); //"EHI EHI VIENI QUI!!!
  delay(3000);
}

void greetClient()
{
  mp3_play(3); //"(inchino) PIACERE DI CONOSCERTI, IO SONO CIN CIN"
  delay(4000);
}

void offerMenu()
{
  mp3_play(4); //"VUOI GUARDARE IL MENU' DEL NOSTRO RISTORANTE? PRENDILO DAL MARSUPIO!"
  delay(6000);
}

void sponsorMenu()
{
  mp3_play(5); //"HAI VISTO CHE PIATTI BUONI CHE ABBIAMO?"
  delay(3000);
}

void proposeSelfie()
{
  mp3_play(6); //"TI VA DI FARE UN SELFIE CON ME?"
  delay(2000);
}

void explainSelfie()
{
  mp3_play(7); //"BASTA CHE SCHIACCI IL CAPPELLO!"
  delay(3000);
}

void cheese()
{
  mp3_play(8); //"FAI CHEESEEEE!"
  delay(2000);

}


void discount()

{

  mp3_play(9); //"SIAMO VENUTI BENISSIMO!"

  delay(6000);

}


void sayBye()

{

  mp3_play(10); //"E' STATO UN PIACERE CONOSCERTI!"

  delay(4000);

}


void forgottenMenu()

{

  mp3_play(11); //"EHI NON SCAPPARE CON IL MENU'"

  delay(2000);

}


void cmon()

{

  mp3_play(12); //"DAI DAI SBRIGATI!"

  delay(3000);

}


void sayHiChinese()

{

  mp3_play(13); //"EHI EHI QUE GOLA EH?"

  delay(3000);

}


void credits()

{

  mp3_play(14); //"TI SCATTERO' UNA FOTO..."

  delay(32000);

}
```

```
void goBackFromLeft()
{
  leftMotor.write(LEFT_REST_POSITION);
  someone = true;
  delay(400);
  mp3_stop();
  mp3_stopper = false;
}

void goBackFromRight()
{
  rightMotor.write(RIGHT_REST_POSITION);
  someone = true;
  delay(400);
  mp3_stop ();
  mp3_stopper = false;
}

void bow()
{
  for (int i = BOW_REST_POSITION; i < BOW_STRETCHED_POSITION; i++) {
    bowMotor.write(i);
    delay(7);
  }
  delay(1000);    // delay to let the motor reach the position
}

void standup() {
  for (int i = BOW_STRETCHED_POSITION; i > BOW_REST_POSITION; i--) {
    bowMotor.write(i);
    delay(7);
  }
  delay(1000);    // delay to let the motor reach the position
}

void rotateForSelfie()
{
  for (int i = ROTATION_REST_POSITION; i < ROTATION_STRETCHED_POSITION; i++) {
    rotationMotor.write(i);
    delay(5);
  }
}

void pullUpLeftArm()
{
  for (int i = LARM_REST_POSITION; i < LARM_STRETCHED_POSITION; i++) {
    larmMotor.write(i);
    delay(5);
  }
}

void pullDownLeftArm()
{
  for (int i = LARM_STRETCHED_POSITION; i > LARM_REST_POSITION; i--) {
    larmMotor.write(i);
    delay(5);
  }
}

void rotateBackFromSelfie()
{
  for (int i = ROTATION_STRETCHED_POSITION; i > ROTATION_REST_POSITION; i--) {
    rotationMotor.write(i);
    delay(5);
  }
}

void doSelfie()
{
  rarmMotor.write(RARM_STRETCHED_POSITION);
  delay(500);
  pullUpLeftArm();
  explainSelfie();
  //la persona tocca il sensore touch
  for (int k = 0; k < 80; k++)
  {
    if (isTouched()) {
      k = 80;
      cheese();
      takePicture();
      touched = true;
    }
    else
      delay(100);
    if (k == 79) {
      cmon();
      for (int j = 0; j < 60; j++)
      {
        if (isTouched()) {
          j = 60;
          cheese();
          takePicture();
          touched = true;
        }
        else
          delay(100);
      }
    }
  }
  rarmMotor.write(RARM_REST_POSITION);
}



void updateDistances()
{
  distanceRight = sonarRight.ping_cm();
  distanceRight = (distanceRight == 0 || distanceRight <= SONAR_MINIMUM_THRESHOLD) ? MAX_DISTANCE : distanceRight;

  distanceMiddle = sonarMiddle.ping_cm();
  distanceMiddle = (distanceMiddle == 0 || distanceMiddle <= SONAR_MINIMUM_THRESHOLD) ? MAX_DISTANCE : distanceMiddle;

  distanceLeft = sonarLeft.ping_cm();
  distanceLeft = (distanceLeft == 0 || distanceLeft <= SONAR_MINIMUM_THRESHOLD) ? MAX_DISTANCE : distanceLeft;

  debug('R', distanceRight);
  debug('M', distanceMiddle);
  debug('L', distanceLeft);
}

boolean isClose()
{
  return distanceRight < STOP_DISTANCE || distanceMiddle < STOP_DISTANCE || distanceLeft < STOP_DISTANCE;
}

boolean thereIsMenu()
{
  int analog_value = 0;
  analog_value = analogRead(photoResistor);
  if (analog_value > PHOTO_RESISTOR_THRESHOLD)
    return false;
  return true;
}

boolean isTouched()
{
  int touch_value = 0;
  touch_value = digitalRead(TOUCH_SENSOR_PIN);
  if (touch_value == HIGH)
    return true;
  return false;
}

boolean thereIsSomeone()
{
  updateDistances();

  if (isClose()) {
    someone = TRUE;
  } else {
    someone = FALSE;
  }
  return someone;
}
```

```
boolean clientEngaged()
{
  return engagement;
}

void turnOnCamera()
{
  if (camState == OFF) {
    digitalWrite(CAMERA_PIN, LOW);
    delay(TURN_ON_DELAY);
    digitalWrite(CAMERA_PIN, HIGH);
    delay(5000);    // wait it turns on
    camState = ON;
  }
}

void turnOffCamera()
{
  if (camState == ON) {
    digitalWrite(CAMERA_PIN, LOW);
    delay(TURN_OFF_DELAY);
    digitalWrite(CAMERA_PIN, HIGH);

    camState = OFF;
  }
}


void takePicture()
{
  if (camState == ON) {
    switchToPictureStage();
    digitalWrite(CAMERA_SHOT_PIN, HIGH);
    delay(SHOOT_DELAY);
    digitalWrite(CAMERA_SHOT_PIN, LOW);
    delay(SHOOT_DELAY);
  }
}
```

```
void switchToPictureStage()
{
  for (int i = 0; i < 2; i++) {
    digitalWrite(CAMERA_PIN, LOW);
    delay(SWITCH_STAGE_DELAY);
    digitalWrite(CAMERA_PIN, HIGH);
    delay(SWITCH_STAGE_DELAY);
  }
}

void switchOnWiFi()
{
  if (camState == ON && wifiState == OFF) {
    digitalWrite(CAMERA_SHOT_PIN, HIGH);
    delay(WIFI_ON_DELAY);
    digitalWrite(CAMERA_SHOT_PIN, LOW);

    wifiState = ON;
  }
}

void switchOffWiFi()
{
  if (camState == ON && wifiState == ON) {
    digitalWrite(CAMERA_SHOT_PIN, HIGH);
    delay(WIFI_OFF_DELAY);
    digitalWrite(CAMERA_SHOT_PIN, LOW);

    wifiState = OFF;
  }
}


void debug(char description, long value)
{
#if DEBUG
  char buff[10];
  sprintf(buff, "%c%ld", description, value);
  debug(buff);
#endif
}

void debug(char *description, long value)
{
#if DEBUG
  char buff[50];
  sprintf(buff, "%s%ld", description, value);
  debug(buff);
#endif
}

void debug(char description, int value)
{
#if DEBUG
  char buff[20];
  sprintf(buff, "%c%d", description, value);
  debug(buff);
#endif
}

void debug(char *description, int value)
{
#if DEBUG
  char buff[50];
  sprintf(buff, "%s%d", description, value);
  debug(buff);
#endif
}

void debug(char *message)
{
#if DEBUG
  Serial.println(message);
#endif
}
```