Interdisciplinary course of

Design and Robotics

XI° edition, 2023

Project:
CamaLele

Professors:
Andrea Bonarini, DEIB department, Politecnico di Milano
Maximiliano Romero , Design department, IUAV, Venice University

Tutors:
Federico Espositi

Group number 3

Students:
DE LUCA VALERIO, school of ENGINEERING
MASSACCI PAOLO, school of DESIGN
MASTROMAURO MATTEO, school of ENGINEERING
WANG SIYI , school of DESIGN

Index

Abstract

Perfetti Sonosciuti (Perfect Strangers) is a 2016 film directed by Paolo Genovese where a group of friends, during a dinner, decide to put every phone on the table and let anyone see all the incoming messages and phone calls, which will progressively reveal the secret lives of each one of the characters.
The scope of the course is to design a robot which represents a character based on the ones in the movie, and to react to events that happen on a possibly simulated set of mobile phones. The robots are implemented as heads with different mechanical architectures, shapes, and behaviours, which may move and produce sentences according to the situation set by the phone events.
In particular, each robot should be able to read out loud a pre-recorded message arriving at it, turn towards who reads the message, and expressively react to them. The environment for the simulation should represent the same of the movie, so the robots will be placed around a circular table.

The following report describes the realisation process of CamaLele, the robot developed by Group 3 that represents the character of the movie called Lele, which is interpreted by Valerio Mastandrea. In the report there will be described the various phases of this process, including the project management, the character analysis, the researches carried out, the first definition of the robot, the development phase and the delivery.
The report will also provide a user manual to help some users to correctly play with the robot and a maintenance manual to help them with the maintenance of the product.

Phase 1: Discover

In this phase we start to organise ourselves, the team management, the tools we will use to communicate and we start to develop a common language.

Team Organization

In the group the team leader is Paolo, and we split the responsibilities as follow:

- Rapporteur: Matteo
- Schedule Manager: Paolo
- Art Director: Siyi
- Tech Manager: Valerio

Project Management

Our team plans to solve the tasks according to the following GANTT:

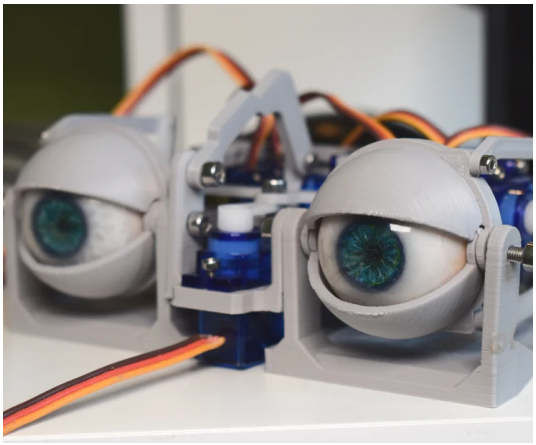
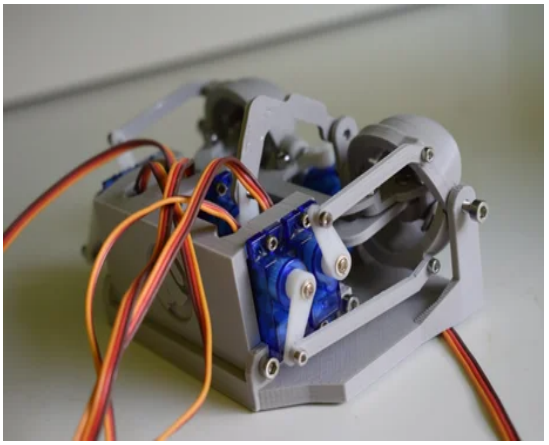
WBS NO.	TASK TITLE	TASK OWNER	START DATE	DUE DATE	DURATION	PCT OF TASK COMPLETE
1	**Phase 1: Discover**					
1,1	Create a Gantt chart	Valerio	7-mar-2023	7-mar-2023	0	100%
1,2	Decide the animal to associate	All	7-mar-2023	10-mar-2023	3	100%
1,3	Assign roles	All	10-mar-2023	10-mar-2023	0	100%
1,4	Make Moodboard	Siyi	11-mar-2023	14-mar-2023	3	100%
1,5	Make Storyboard	Siyi	14-mar-2023	17-mar-2023	3	100%
1,6	Decide on robot's characteristics	All	17-mar-2023	24-mar-2023	7	100%
1,7	Think of possible electronics	Matteo	18-mar-2023	24-mar-2023	6	100%
1,8	Bill of materials	Matteo	24-mar-2023	24-mar-2023	0	100%
1,9	Supegroup meeting	Valerio	24-mar-2023	24-mar-2023	0	100%
1,10	Make prototypes	All	24-mar-2023	27-mar-2023	3	100%
1,11	Creat a presentaion template	Siyi	27-mar-2023	27-mar-2023	0	100%
2	**Phase 2: Define**					
2,1	Reshape eyes	Paolo	28-mar-2023	8-apr-2023	10	100%
2,2	Sketch a draft of structure	Paolo & Matteo	1-apr-2023	6-apr-2023	5	100%
2,3	Initial code setup	Valerio	2-apr-2023	9-apr-2023	7	100%
2,4	Design a draft of the circuit	Matteo	3-apr-2023	10-apr-2023	7	100%
2,5	Code for basic servo movements	Valerio & Matteo	6-apr-2023	13-apr-2023	7	100%

2,6	Design of the shape	Paolo	7-apr-2023	17-apr-2023	10	100%
2,7	Order electronic components	Matteo	8-apr-2023	15-apr-2023	7	100%
2,8	First model of the tail	Siyi	10-apr-2023	17-apr-2023	7	100%
2,9	State diagram of the emotions	Valerio	14-apr-2023	14-apr-2023	0	100%
2,10	Supegroup meeting	Valerio	18-apr-2023	18-apr-2023	0	100%
3	**Phase 3: Develop**					
3,1	Model neck	Paolo	18-apr-2023	25-apr-2023	7	100%
3,2	Model chest	Paolo	18-apr-2023	25-apr-2023	7	100%
3,3	Unwrap eyes and eyelids	All	18-apr-2023	1-mag-2023	13	100%
3,4	Manual controller	Valerio	20-apr-2023	30-apr-2023	10	100%
3,5	3D print skeleton	Paolo & Siyi	24-apr-2023	3-mag-2023	9	100%
3,6	Printing of the components	Paolo	25-apr-2023	1-mag-2023	6	100%
3,7	First tail prototype	Siyi	25-apr-2023	1-mag-2023	6	100%
3,8	Circuit implementation on	Matteo	1-mag-2023	6-mag-2023	5	100%
3,9	Tuning of the motors	Valerio & Matteo	1-mag-2023	8-mag-2023	7	100%
3,10	Tuning of basic movements	Valerio & Matteo	1-mag-2023	8-mag-2023	7	100%
3,11	Prototype for chest	Paolo & Matteo	2-mag-2023	5-mag-2023	3	100%
3,12	Prototype for neck	Paolo & Matteo	2-mag-2023	5-mag-2023	3	100%
3,13	Supergroup meeting	Valerio	4-mag-2023	4-mag-2023	0	100%
3,14	First components assembly	Matteo	5-mag-2023	5-mag-2023	0	100%
3,15	Fix power supply problems	Valerio & Matteo	6-mag-2023	13-mag-2023	7	100%
3,16	Implementation of sad emotion	Valerio	6-mag-2023	8-mag-2023	2	100%
3,17	Implementation of angry emotion	Valerio	6-mag-2023	8-mag-2023	2	100%
3,18	Implementation of shocked	Valerio	6-mag-2023	8-mag-2023	2	100%
3,19	Assemble the robot	All	8-mag-2023	8-mag-2023	1	100%
3,20	Dual core task managing	Valerio	8-mag-2023	8-mag-2023	1	100%
4	**Phase 4: Deliver**					
4,1	Printing of reduced eyes and eyelids	Paolo	9-mag-2023	10-mag-2023	1	100%
4,2	Printing of the base	Paolo	10-mag-2023	10-mag-2023	0	100%
4,3	Polipropilene tail creation	Siyi	10-mag-2023	18-mag-2023	8	100%
4,4	Supergroup discussion	Valerio	11-mag-2023	13-mag-2023	2	100%
4,5	Model of the tail connection	Paolo & Matteo	11-mag-2023	14-mag-2023	3	100%
4,6	Reprint wrong components	Paolo	11-mag-2023	20-mag-2023	9	100%
4,7	Mp3 connection	Matteo	14-mag-2023	17-mag-2023	3	100%
4,8	Supergroup meeting	Valerio	15-mag-2023	15-mag-2023	0	100%
4,9	Painting components	All	15-mag-2023	18-mag-2023	3	100%
4,10	Message buffer with priority	Valerio	15-mag-2023	22-mag-2023	7	100%
4,11	Code debugging	Valerio	15-mag-2023	22-mag-2023	7	100%
4,12	Mp3 code setup	Valerio	17-mag-2023	17-mag-2023	0	100%
4,13	Text-to-speech generation	Valerio	18-mag-2023	18-mag-2023	0	100%
4,14	Select audio for emotions	Valerio	18-mag-2023	21-mag-2023	3	100%
4,15	Make final tail move correctly	Matteo	18-mag-2023	21-mag-2023	3	100%
4,16	Implementation of cautious	Valerio & Siyi	18-mag-2023	18-mag-2023	0	100%
4,17	Implementation of happy emotion	Valerio & Siyi	19-mag-2023	19-mag-2023	0	100%
4,18	Implementation of annoyed	Valerio & Siyi	20-mag-2023	20-mag-2023	0	100%
4,19	Implementation of idle state	Valerio	21-mag-2023	21-mag-2023	0	100%
4,20	Solder the circut	Matteo	19-mag-2023	19-mag-2023	0	100%
4,21	Slides for final presentation	Paolo	20-mag-2023	22-mag-2023	2	100%
4,22	Fix audio problems	Valerio	21-mag-2023	22-mag-2023	1	100%
4,23	Robot assembly	Matteo	21-mag-2023	22-mag-2023	1	100%

Research

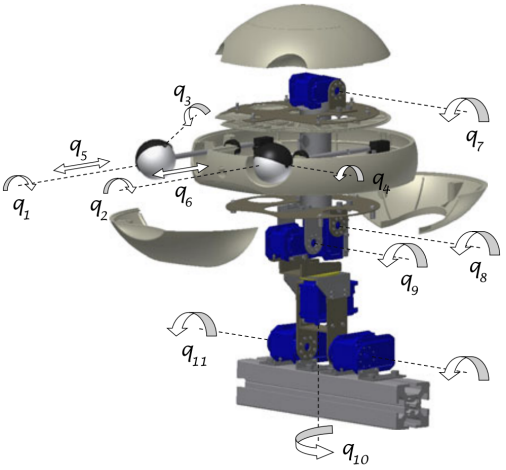
We decided to rely on online solutions as a starting point for the kinematics and then adapt them to our needs. These are the following 3D models from which we took inspiration:

- **eyes**: for the eyes we found on Instructables the following project:

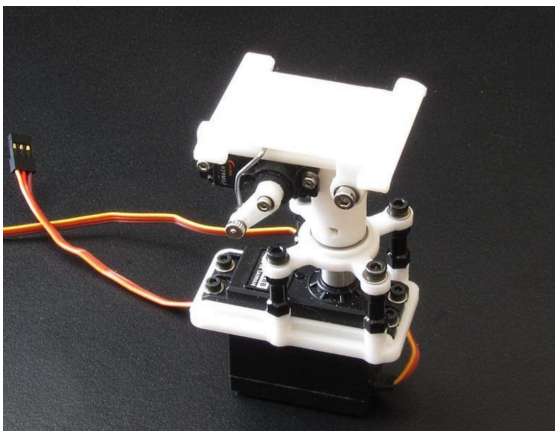



We resized and changed it in some parts, as described in the following sections, in order to meet our needs

- **chest:** for the chest we found online the following projects:
  a) first project:

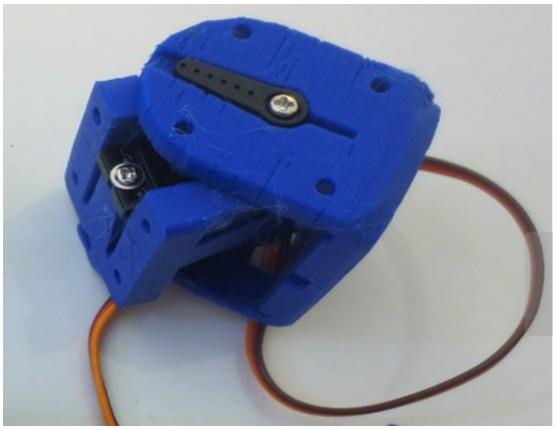


  b) second project (camera mount for drones):



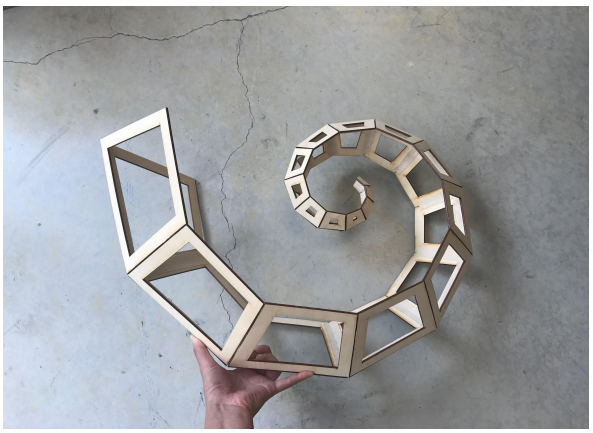
The idea was to combine this to project taking from the a) project the bottom part, related to the connection between the chest and the base by means of the two motors in position q11 in the picture a)). Then insert project b) for the vertical rotational motion.

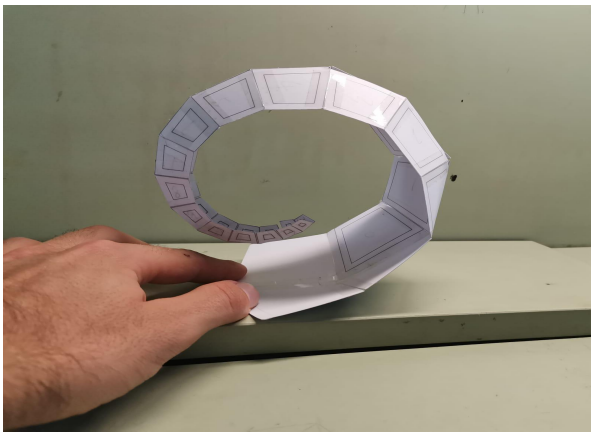
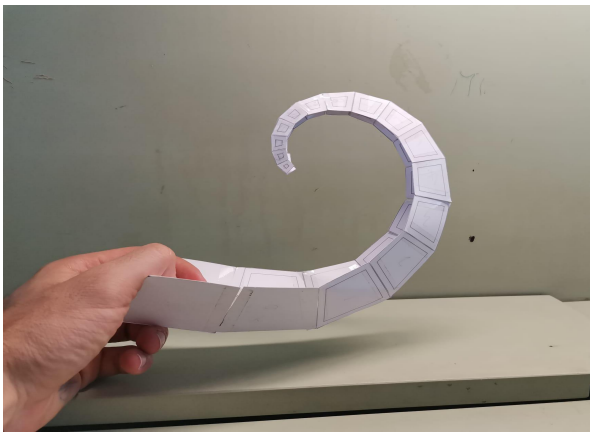
- **neck**: for the neck we found online the following project (It was used for a wrist):



Also in this part, starting from this project, we customised the attachments so that the part can link the head to the chest.

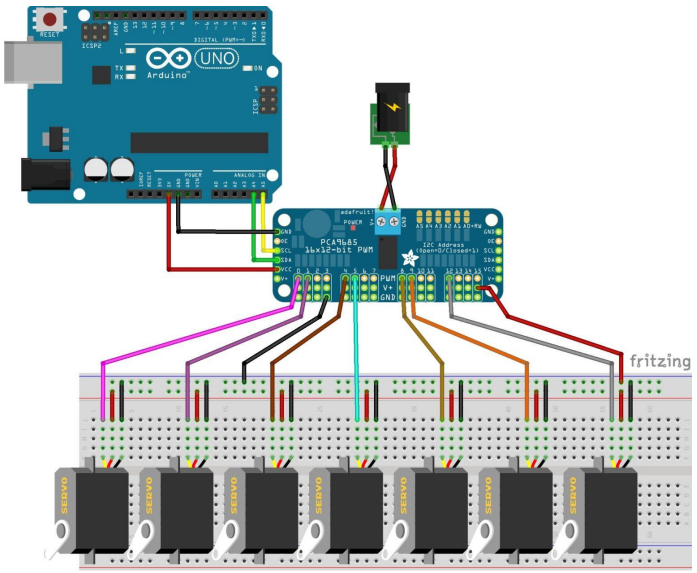
- **tail**: for the tail we found on Instructable the following project:



This particular tail can be moved by acting on the two final pieces of it, indeed opening and closing the two final parts it is possible to open and close it.

- **motor connection:** our project required 11 motors and so we use the PCA9685 board to drive all them. We learnt how to use it from a online site, in particular we saw:
    a) how to connect the motors to the PCA9685



So the idea is to connect the several motors to the PCA9685 with an additional power supply for the motors, and then use the IC2 connection protocol to communicate with the board from the esp32.

b) how to drive the motors through the PCA9685: in the site there is a example code of how to use it and in particular the fundamental lines are:

```
#include <Wire.h>
#include <Adafruit_PWMServoDriver.h>

Adafruit_PWMServoDriver pwm0 = Adafruit_PWMServoDriver();
```

```
void setup() {
  Serial.begin(9600);
  Serial.println("8 channel Servo test!");

  pwm.begin();
  pwm.setOscillatorFrequency(27000000);
  pwm.setPWMFreq(SERVO_FREQ);  // Analog servos run at ~50 Hz updates

  delay(10);
}
```

```
pwm.setPWM(n, 0, pulse);
```

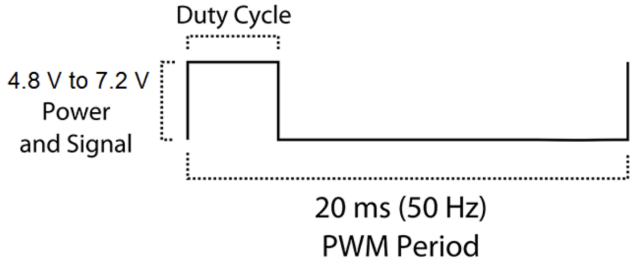
- we use the library *Adafruit_PWMServoDriver.h* to use the PCA9685
- *setOscillatorFrequency(27000000)* -> it is used to set frequency oscillation
- *setPWMFreq(SERVO_FREQ)* -> it is used to set the frequency of the servo motor and usually it is set at 50Hz
- *setPWM(n, 0,pulse)* -> it is used to directly drive the servo motor to a specific position, in particular how it works:

the *pwm.setPWM* function needs 3 parameter:

pwm.setPWM(channel, on, off)

channel is used to select the one of the 16 servo motors the single board can handle. The other two parameters are used to specify the starting(*on*) and the ending(*off*) of the pulse inside a period (period = 1/SERVO_FREQ). These two parameters can only assume values in the range [0;4095]. (the interval is discretized into 4095 parts). So:

channel: the channel must be updated with new values in the range [0;15]
on: the value [0;4095] when the PWM signal changes from low to high in a period
off: the value [0;4095] when the PWM signal changes from high to low in a period



Character analysis

In the first meeting we decided, according to the other groups, that the character for our robot would been Lele:

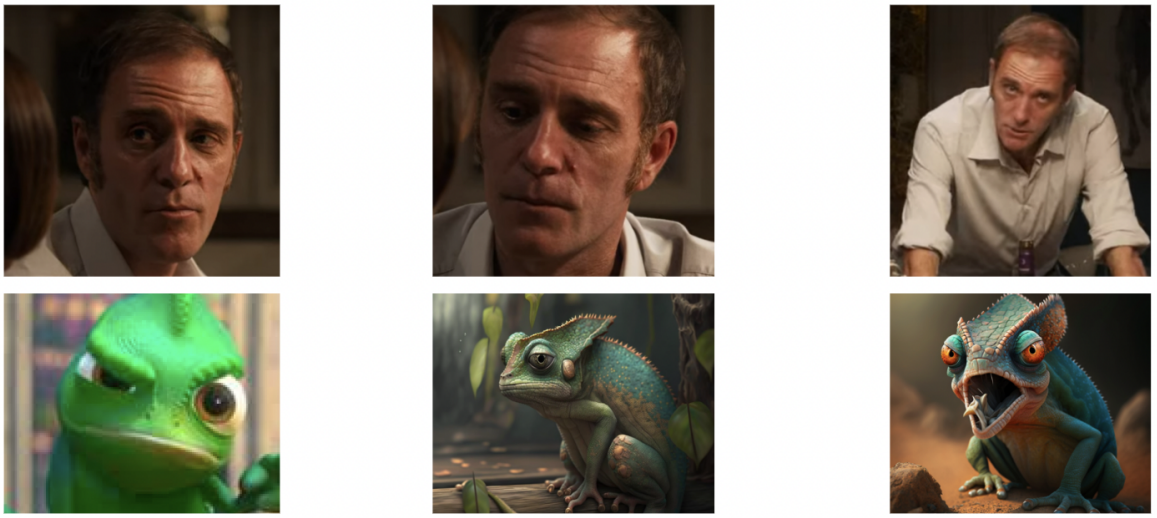
Lele is one of the main characters and friends in the group that gathers for a dinner party.He is married to Carlotta, and they have two children together. Lele is devoted to his family, and his interactions with his wife and children showcase his affectionate and nurturing side.However, Lele's character is not without flaws. He is engaged in a secret extramarital affair with a younger woman, which he tries to keep hidden from his friends and wife. This aspect of his life reveals a more complex side to Lele, highlighting the internal conflicts he experiences. On one hand, he appears to be a loving family man, but on the other hand, he is unable to resist the temptation of pursuing a clandestine relationship.

Animals strategies

In order to identify Lele's behaviour and emotions, usually actors link to their character an animal that fits as much as possible with him. In our case, we associated to Lele the chameleon that's because we see some similarities that are listed as follow:

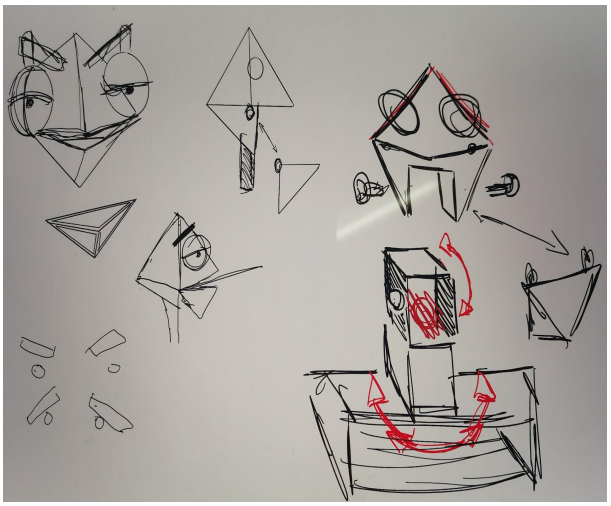
- Lele's intention is to hide the truth for his safety, but at the end the target turns out in defending his friend (generous). In both cases the way he chooses is cheating.
- He is shy and calm when he is left alone, but when he is under attack he speaks loudly, he can defend himself very well.
- He does not regret it. He is always cheating. He is able to adapt himself to the situation, indeed after the changing phones he mimics perfectly the role of the gay person. Always watching down when he resigns. hide his emotion. poker face

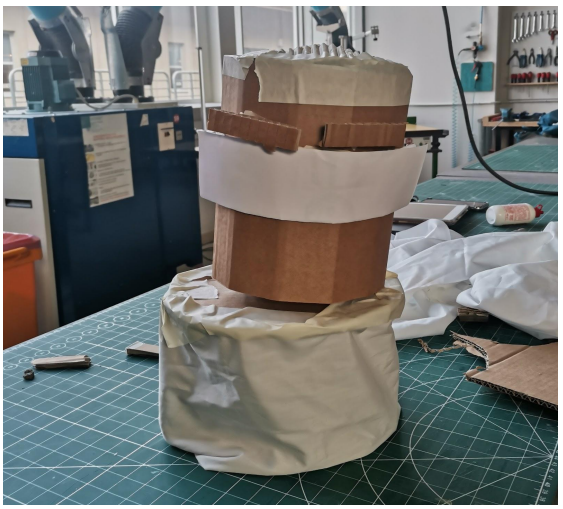
And, based on the previous considerations, we created the following moodboard:



Robot strategies

In the second meeting we have been asked to realise a first prototype of the robot so that we can start thinking about the possible movements and functionalities that it has to have in order to express the emotions. First, we did some drawings because it is the easiest way to develop very different shapes in a short time:





The idea was to have a rotational axis around the z-axis (vertical axis) to orient the face toward the other character around the table, other two rotational axes around the y-axis (later axis) to move backwards and forwards the main body and to make the robot looks up and down (useful to express happiness and sadness, for example). As shown in the physical model, in the first attempt the idea was to use a lcd matrix to animate the eyes, but, after a suggestion by the prof, we decided to replace the matrix with a physical one because we felt that a 3D shape fo the eyes could be more suitable for a chameleon than a digital one.

Functionalities

The purpose of this project is to deliver a robot able to interact with others by means of a centralised communication structure. In this way robots communicate and react autonomously, plus the server computer (called “god”) from which is possible to communicate with them.

Phase 2: Define

Strategy

As suggested by the professors, we decided to rely on online solutions for the kinematics and then modify them for our specific application. Instead for the design of the robot we did all by ourselves printing all the parts by means of a 3D resin printer (Elegoo Saturn S)

Functionalities

The functionalities for our robot were:

- eyes motion: up and down, left and right
- eyelids motions: 4 independent motion of the eyelids, one for each one of them
- neck motion: around y-axis
- chest motions: around z-axis and y-axis
- mouth motion: open and close
- audio: we decided to use recorded sounds found online to animate the emotions and use pre-recorded audio to reproduce the sentences.

Electronics

The microcontroller used to drive the robot is the ESP32, we select it because: it can easily connect to a Wi-Fi network to connect to the internet (station mode), or create its own Wi-Fi wireless network (access point mode) so other devices can connect to it. It is Dual-core and it comes with 2 Xtensa 32-bit LX6 microprocessors(core 0 and core 1), in this way we can manage at the same time the motor handling and client-server connection.

The driver motor is the PCA9685 that allows us to connect up to 16 motors and it can be commanded through the IC2 that is a multi-master/multi-slave (controller/target) serial communication bus.

The motors used are:
SG90

Specifications	
Weight	9 g
Dimension	26mm*13mm*24mm
Stall torque	1.2kg/cm (4.8V)
Operating speed	0.12sec/60º(4.8V)
Operating voltage	4.8 V
Stall Current	750 mA (6V)
Dead band width	7 µs

MG996R

Specifications	
Weight	55 g
Dimension	40.7 x 19.7 x 42.9 mm
Stall torque	9.4 kgf·cm (4.8 V)
Operating speed	0.17 s/60º (4.8 V)
Operating voltage	4.8 V
Running Current	500 mA
Stall Current	2.5 A (6V)
Dead band width	5 µs

In order to select the correct power supply, first we identify the power requirement to move the motors; using 8 SG-90 at 5V 500mA and 4 MG996R at 5V 2.5A we need at least 50.2W, so we select the TASVICOO 50 Watt Power Supply , 5V 10A (the computation of the power is done in the worst case, considering the stall current and not the running current).

For the audio, we select a 3W speaker drived by a MP3player board.

Coding

In the first phase of the robot creation process, after having in mind a possible structure of the project folder, the main tasks to do at code level were to think of a solid structure for handling the communication between the robots and to design a mechanism for the reaction to the received messages.

Project structure

The decision about how to organise the project code has been to divide it in several libraries, and import each of them in one main file to manage an aspect of the whole robot behaviour. We decided to use Arduino IDE to program the ESP32 and to keep in the *.ino* file the main loop with the implementation of the logic of the reception and reaction to the messages, while more technical aspects have been moved in C++ files and the respective header files have been imported in the Arduino code.
To manage all the coding process, we decided to use GitHub to ease the team work and to have a safe cloud to constantly update and save our progress.

Robot communication

To decide how to make the robot communicate with each other, a supergroup has been created by selecting a member for each group, who has the responsibility to be the spokesperson for the group. In our case, we decided to select our tech manager as a member of the supergroup for his computer science background.

The main issue to discuss was about implementing a client-server communication or a peer-to-peer one, and after some meeting the supergroup decided to choose for the first one, to guarantee a centralised structure and let it take care of forwarding all the messages. So, it has been decided that each robot would send a message to a *god server* when performing an emotion, and this server would have broadcasted all the incoming messages to all the robots and sent the external messages simulating the phone messages whenever requested by input. Also, it would have been able to halt the communication exchange when

requested by input to stop the chain of emotions. On the other side, instead, the client has been requested to manage the queue of incoming messages and properly choose which of them will react to, and also to decide when and how.

After the choice of the structure, the next problem was about thinking about a protocol to exchange the messages. The final choice was to send a two characters long string for a god message and a four characters long string for an emotion sent by a robot. The god message always starts with letter G and then is followed by another character: from 1 to E (in hexadecimal order) to symbolise a phone message, F to send the halt message, G to send the end of the reading of a message by a robot. Phone messages are two for each group and are ordered in ascending group order, so messages to be received by our group are G5 and G6. The emotions sent by a robot are instead defined as follows:

[1-7] from	[B-L] emotion	[0-7] toward	[1-3] intensity
6	C	7	1

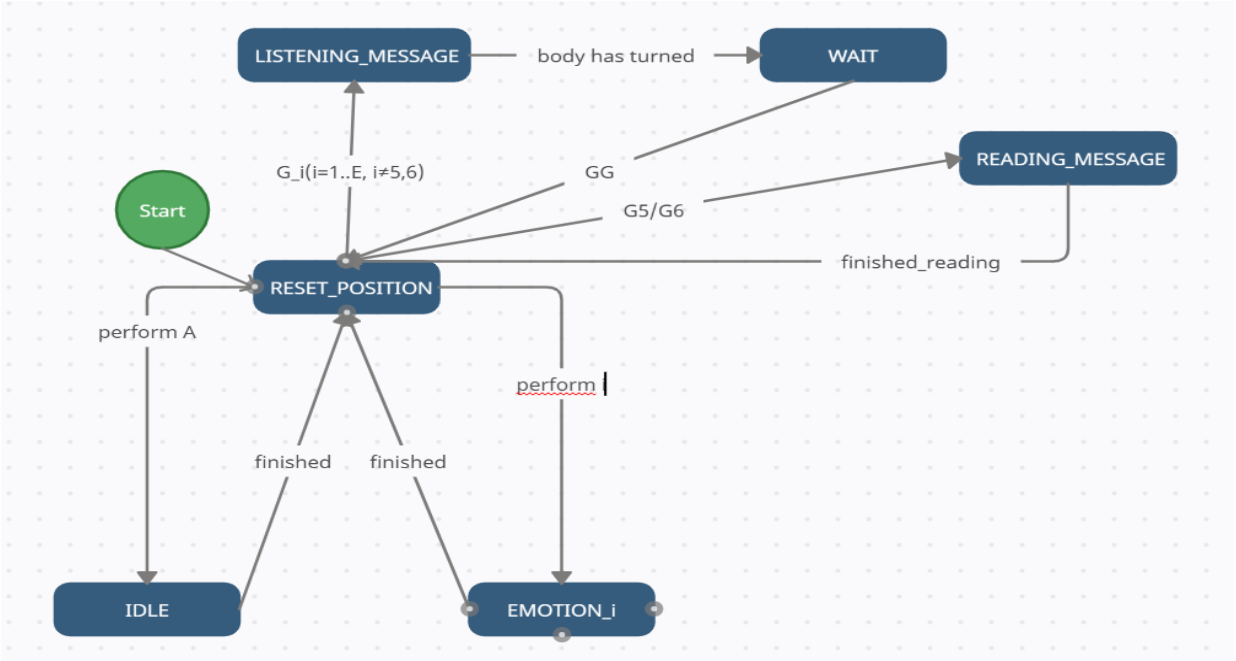
First and third character are a group number representing sender and receiver of message, respectively (0 if message has no specific receiver); second character is a letter encoding an emotion and last character is a number symbolising the intensity level of the emotion.

Last topic was to decide which emotion a robot should perform. After some discussion the supergroup selected a list of available emotions and each group implemented some of them. Our group decided to implement the following emotions:

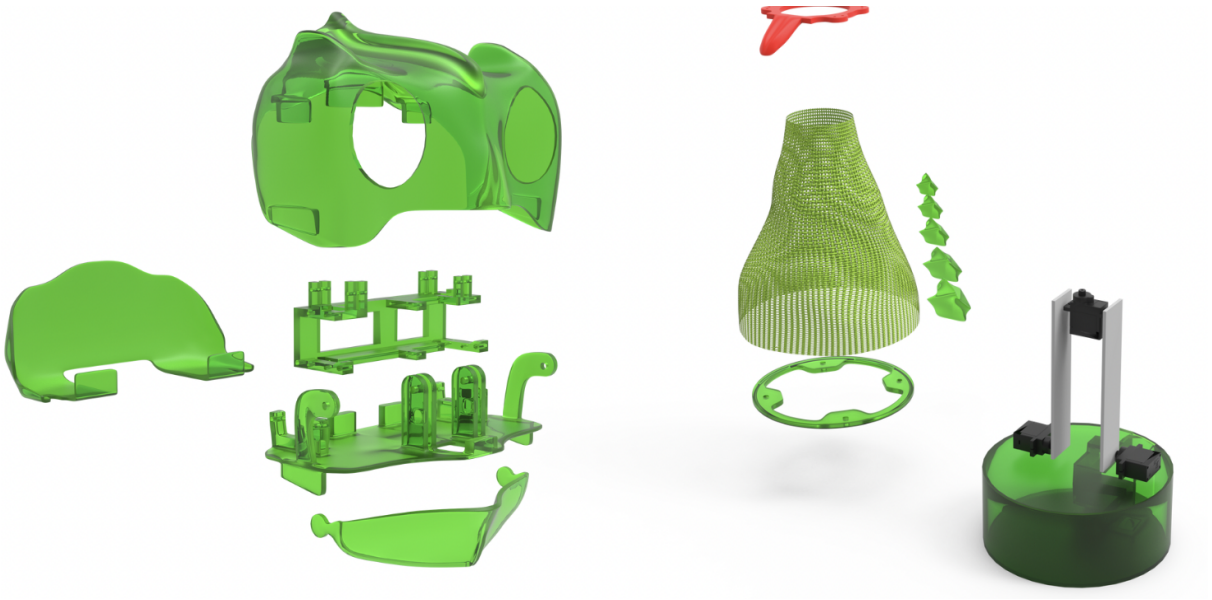
A: idle
B: happy
C: angry
D: shocked
E: sad
H: cautious
J: annoyed

State diagram for the emotions

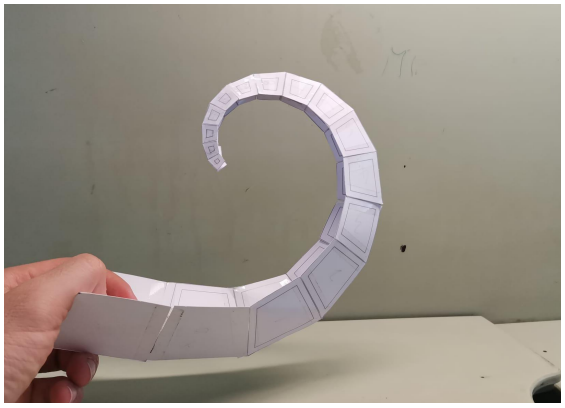
Once that the emotions to implement and the mechanism with which they would have been sent were clear, we could follow in the design of the reaction process of our robot. In particular, we decided to structure the code life cycle of the robot in states: we thought about having a state for each emotion, in which to make the robot perform the proper reaction, plus a state for resetting robot position, one for reading the phone messages, one for turning to the other robots when they were reading phone messages, and the last one to wait until the other robot finished reading. After performing any action, our robot is intending to return to the reset position state, where all motors come back to the starting position and it is checked whether to send an emotion message, change state again or stay in idle. The behaviour of our robot is synthesised in the following state diagram:


LISTENING_MESSAGE
body has turned
WAIT
READING_MESSAGE
G_i(i=1..E, i≠5,6)
GG
G5/G6
Start
finished_reading
RESET_POSITION
perform A
perform
finished
finished
IDLE
EMOTION_i

Structure

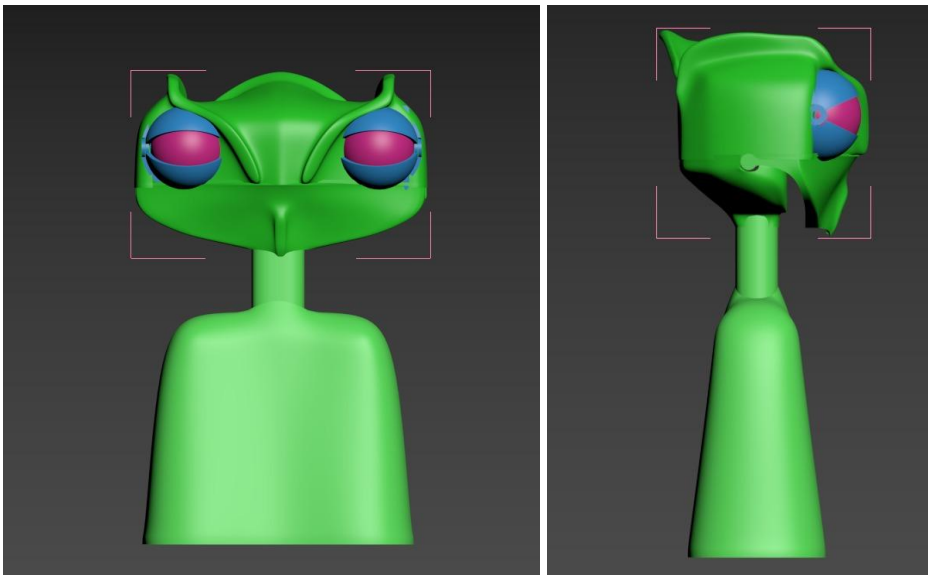


About the tail, after we had found online a project for the tail, we resized it, because it was too big, and doing so we deleted the internal holes to make it more resistant as we did not need to save weight with such a small and light shape. Then we created it using paper connecting the two parts with paper tape, after several attempts, we got the following result:

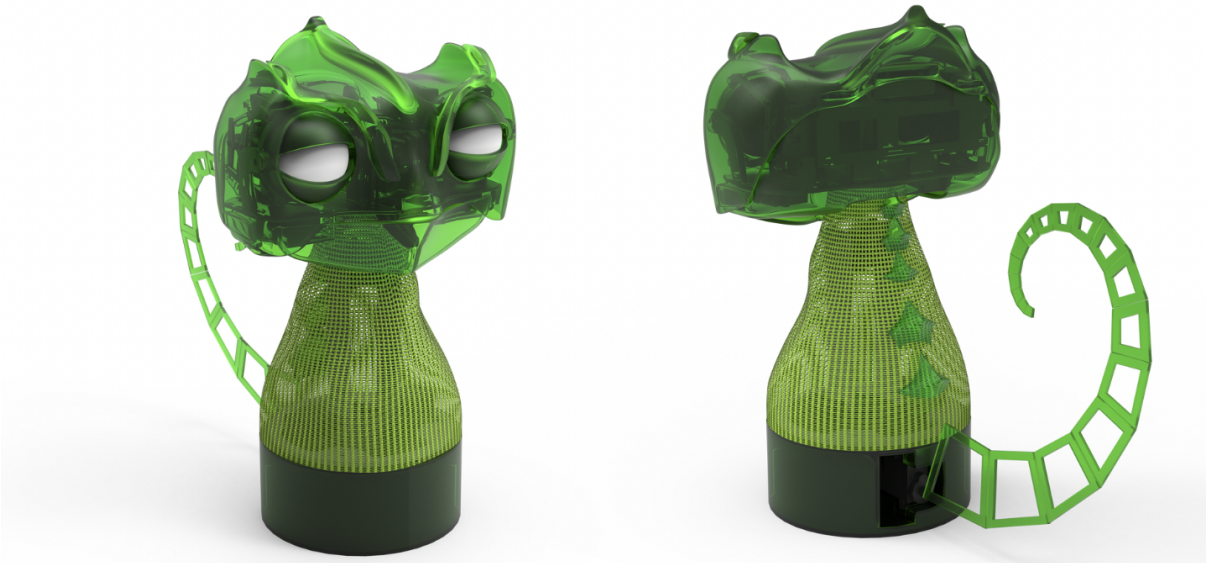


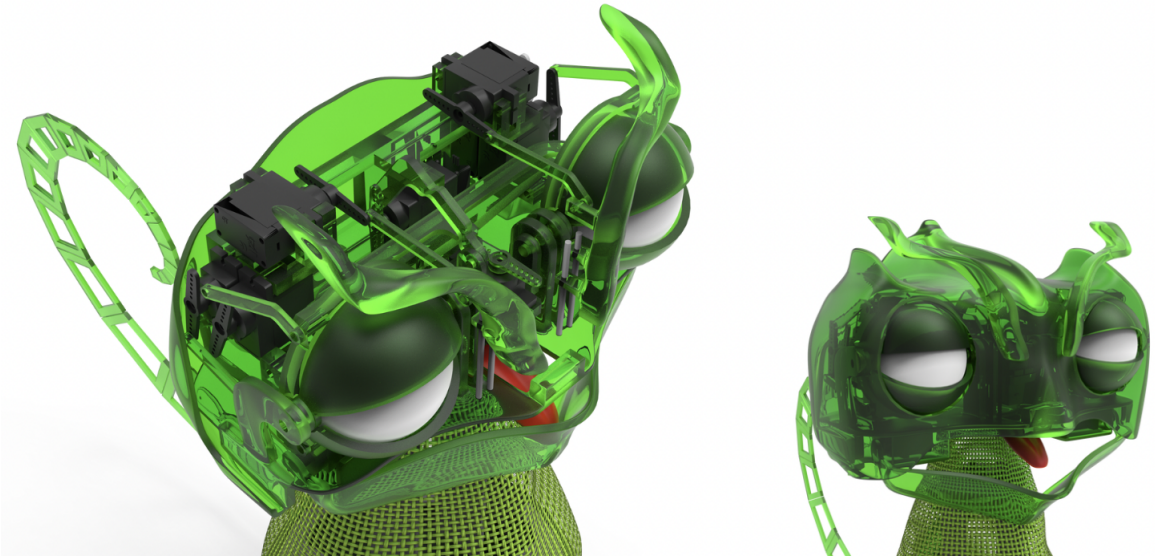
Shape:

- fist shape:



- second shape:





Phase 3: Develop

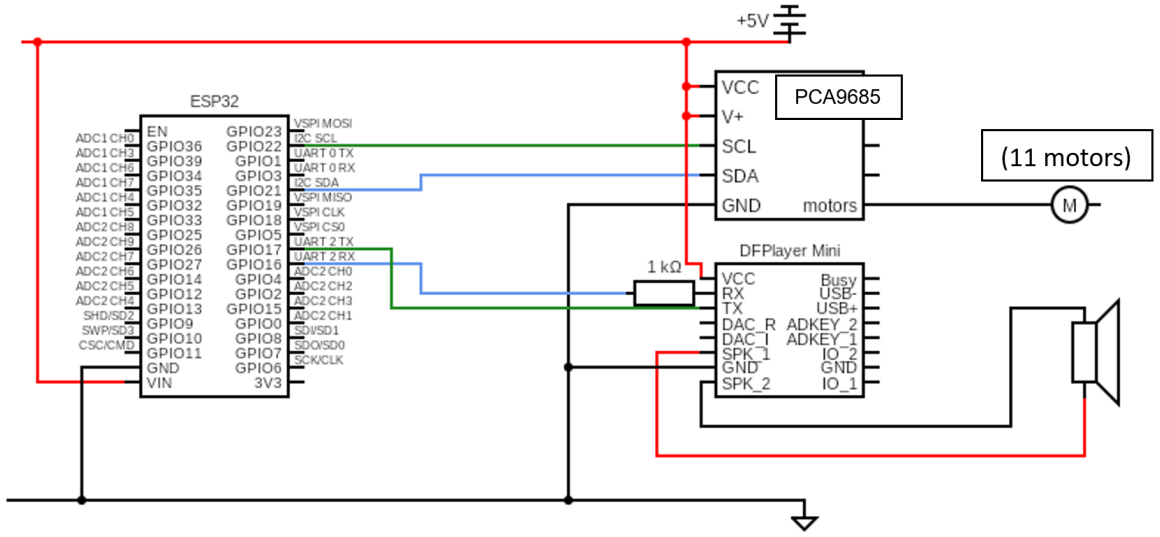
In this phase we describe the development process, departing from the first prototype to the final improvements regarding:

Strategy

For the developing part, we first focused on the 3D model of the eyes because it was the most critical part from the point of view of the kinematics, indeed it required a farther re-shaping to adapt the shape and size to our project and the usage of 6 motors at the same time, then a part of group starts working on the motor handling while the other part continue to shape the neck, chest and the basis. In this way as a new component was printed, it was possible to assemble the corresponding motors and program it.

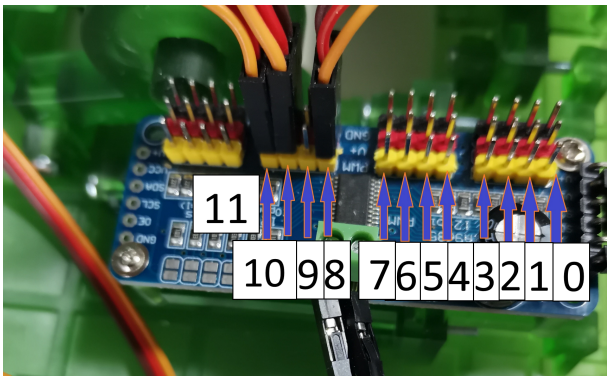
Electronics

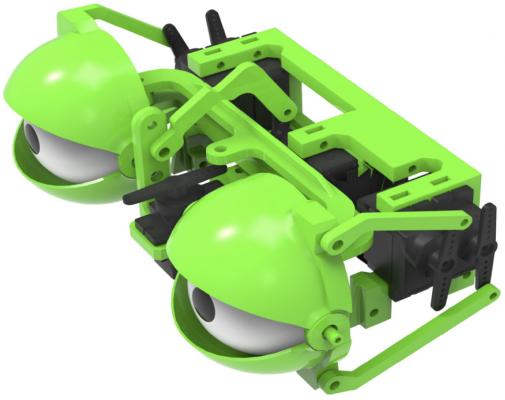
After several attempts, we finally decided to implement the following circuit scheme, realised with an online tool “Circuit-Diagram”:

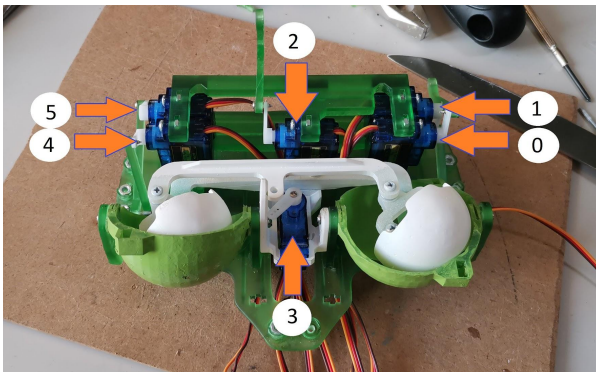


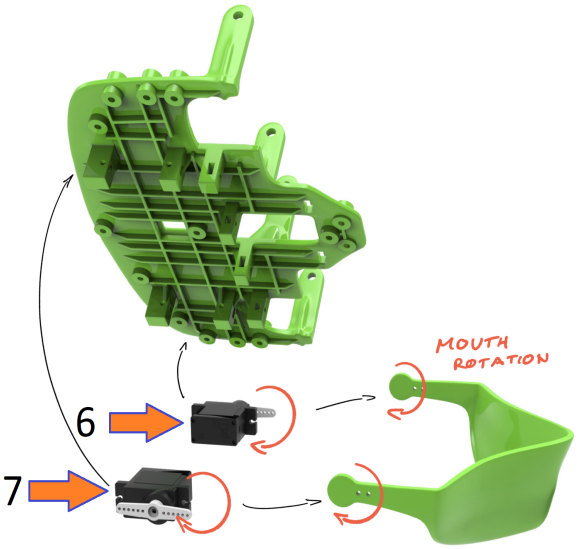
As it is possible to see in the scheme, we connected

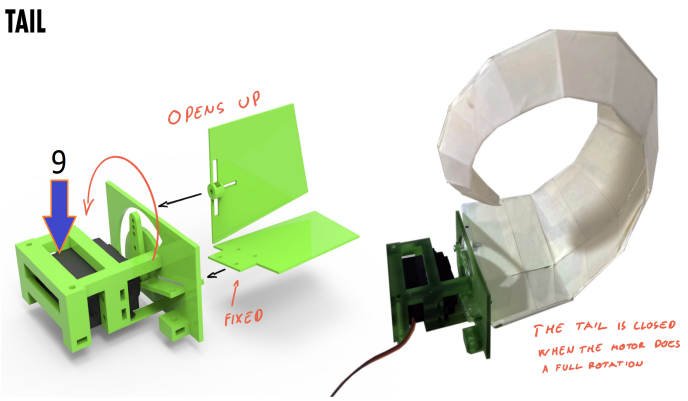
- 16 17 pins of ESP32 to the RX TX pins of the MP3player (for the serial connection), using a 1Khom resistance in series between the 17 and RX pins
- 21 22 pins of ESP32 to the SDA SCL pins of the PCA9685 module (for the IC2 connection)
- 5V and GND pins of ESP32 to the V+ and V- of the power supply
- 5V and GND pins of MP3player to the V+ and V- of the power supply
- SPK1 and SPK2 pins of MP3player to black and red wires of the speaker
- V+ VCC pins of the PCA9685 module to the V+ pin of the power supply (GND pin to V- pin of the power supply)
- the motors pin to the PCA9685 module in the correct sequence as shown in the following picture:

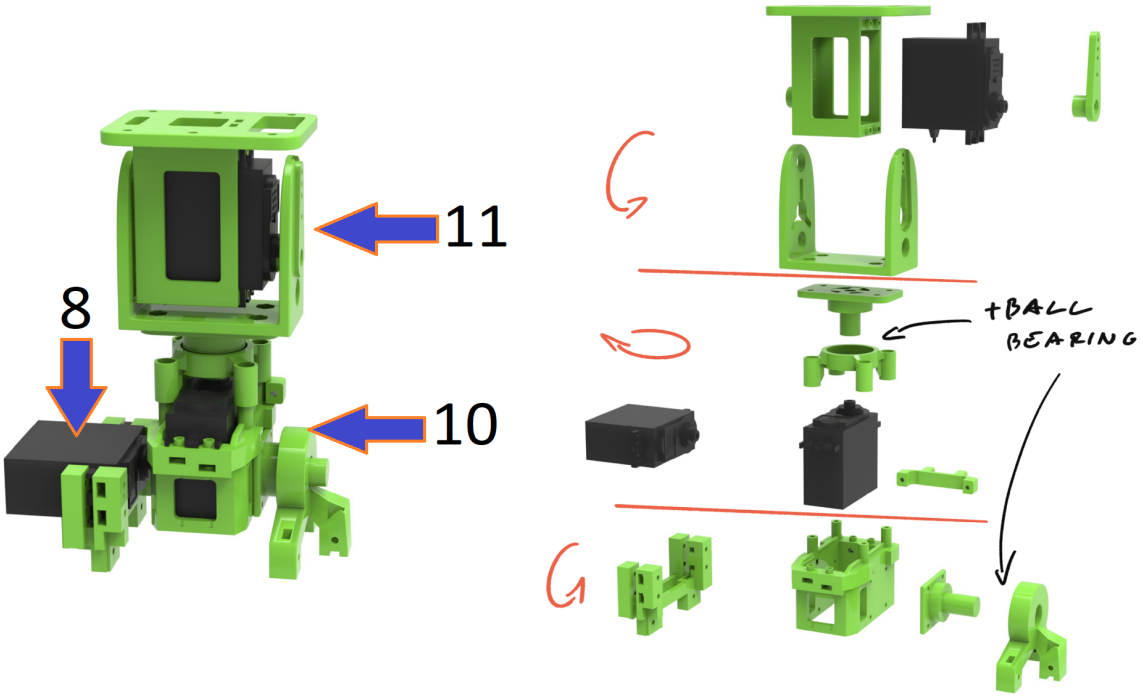

11
10 9 8 7 6 5 4 3 2 1 0



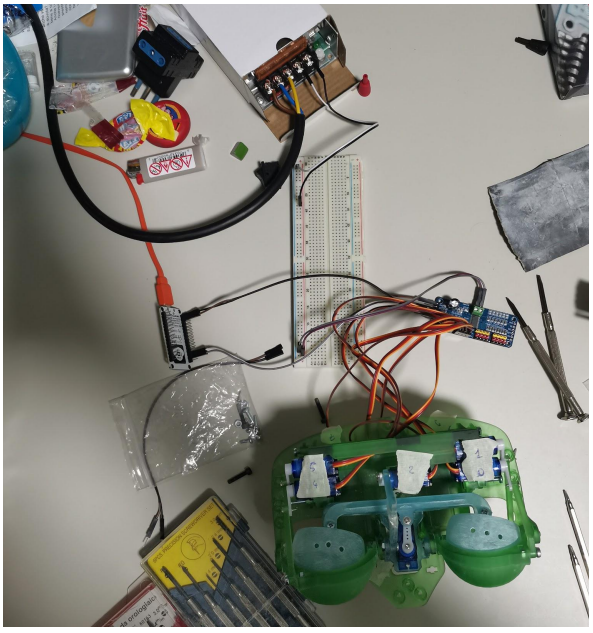

2
5
4
1
0
3


MOUTH ROTATION
6
7


TAIL
OPENS UP
9
FIXED
THE TAIL IS CLOSED WHEN THE MOTOR DOES A FULL ROTATION



In the developing step, we used the breadboard to test the circuit, adding step by step new elements: first we started connecting only the motors related to the eyes and eyelids, then adding the ones related to the chest and neck, finally the one related to the tail. the last element added was the Mp3Player.



During this stage, we met several problems related to the power supply, indeed the motors didn't work very well all the time and we had to restart the robot many times. Also the Mp3Player didn’t work well from the beginning because the wires weren’t well connected to

the breadboard (they continuously disconnected). For these technical issues related to the connections, the implementation of the Mp3Player became a very critical part and we had to use an external power supply to use it during all the development stage. (once the circuits were realised, we used the correct power supply and all worked fine)

Coding

During the development phase, the whole architecture previously defined has been finally realised. So, as planned, the reasoning and emotion-reaction behaviour of the robot has been kept in the arduino file while the other tasks have been deployed in *.cpp* files and their headers have been imported by the main file. This helped us to have a robust and scalar architecture, and we could try several options for the logic of the robot reaction just replacing the main file without having to copy tons of code lines each time. We have created three main libraries: one to manage the movements of the robot and to implement the sequence of actions that define an emotion, one for the wifi and client setup and one for the mp3 setup and audio management.

Wifi_setup

The wifi_setup library provides some APIs to set up the connection to the wifi and to the god server. Here all the variables regarding the connection are defined, in particular the id and password of the router to use to connect the robots to the internet and the ip and port of the god server. So, if any of these addresses changes for any reason (e.g. change of router or change of dynamic ip of the god server) these variables should be promptly modified.
In addition to this, in the wifi_setup library there are defined two functions: one that sets up the connection to the wifi and subsequently connects the client (robot) to the god server, and another function that just handles the connection between client and server. In this way, it is impossible to connect the client to the wifi but not to the server: this is obviously done on purpose because, given the project specifications, the robot has no reason to work if not connected to the god server. The API that just connects robot with god, instead, is made to have a connection that is resilient to disconnections or server crashes, so if the Wifi connection is stable but the server disconnects and then returns available, our client can immediately reconnect to it without having to reboot the ESP32.
Finally, a simple API to send a test message is provided for debug purposes.

External libraries used for setting up the connection and web server are the Arduino libraries *Wifi.h* and *WebServer.h* .

Mp3Player_setup

The mp3_setup library uses the Arduino libraries *DFRobotDFPlayerMini.h* and *HardwareSerial.h* to implement the software setup of the mp3 player and offers a function that extends the function that plays a song included in the external libraries by returning a boolean parameter that is set to true when the song finishes to play: this feature will be helpful to detect when the robot has finished reading a phone message. The mp3 setup is done using pins 17 and 16 of the ESP32 for TX and RX respectively.

In order to correctly read the audio files stored in the SD card in the mp3 player, they have to be in mp3 format, with Mono audio channel and a frequency of 44 kHz. The mp3 files in the SD card must also follow a specific naming convention with a precise folder structure: all the files must be stored in a folder called mp3 and named “000X.mp3”, where X is the order numbering of the song. In this way, it is possible to call the function play_song(X) in the mp3_setup library to play song X.

Movement

The movement library is the most complex library implemented in our code, that we used to implement the motion of the level providing low level APIs to move just a single motor to a certain position, medium level APIs to move many motors in parallel representing the movement of a body part (e.g. the movement of the eyelids is represented by four motors moving at the same time), and high level ones to implement a sequence of low and medium-level motions that realise the full movement related to a specific emotion. The main external libraries used to realise these functions are *Adafruit_PWMServoDriver.h* and *Wire.h*.

The first part of the library is used to initialise constants. Each of the eleven motors have been tuned and the value of the PWM for each relevant position of the motors have been stored in several constants. Basically, each motor has at least a constant for when the respective body part is fully open/close and for when it is at rest, plus some constants for specific positions and rotations if needed to implement a specific configuration for an emotion.
Another set of constants is provided for the relative rotation of the robot body with respect to any other robot position. This is needed because the robot does not have sensors and cannot perceive the position of the other robots, so it has to be known a priori. So, if a robot has to look at another one and then perform an emotion, it has to turn by an angle that is stored in these constants, and then, if the emotion requires a rotation of the body, the robot must be able to turn left and right with respect to that angle. Such constants provide the PWM for each angle between our initial robot position and the other robots and the PWM for a relative shift of 30 degrees left and right.
The problem of how to position the robots for the simulation have been discussed in the supergroup and with the professors, and the decision changed during the development phase, so we opted for having a MatLab algorithm to calculate by interpolation the transformation between the given angles and their respective PWM. The way we calculated the proper angles for the rotation given the initial environment configuration will be explained in the appendix, while the choice for the final environment setup will be described in the delivery section.
Last group of constants is the one for the enumeration of emotions and characters, that maps each character’s name and each emotion to the corresponding letters and numbers we chose for the embedding in the define phase.

After the initialization of the constants, a movement class representing a servo motor in an object-oriented style is defined. A movement for a servo has some private parameters like the servo number, the servo initial and final positions, the velocity of the transition, a long to store the current time and a char to store the delay between each transition step, a char to set the acceptable error between the desired final position and the actual one, and two boolean flags that tell if the servo is performing a movement and if it has finished the

transition. Getters and setters for the boolean values and for the servo position are provided, while all the other parameters must stay private.
Then, a constructor for the class is provided: the constructor initialises all the internal variables, with special care to the initial position of the servo that changes depending on which servo is being initialised. At this point, two functions are provided: one that sets the position of the specific servo to a new one with a certain velocity and one that lets the servo in stall for some amount of time. The first one is realised with a control law that makes the transition smooth during time: the servo does not instantly move to the desired position, but makes a small transition each couple of milliseconds (given by the private parameter delay). This control algorithm is ensured to converge to the final desired position, as explained in the appendix. The second function is used instead to make the robot wait without blocking the execution of other functions (as it happens with the delay function in Arduino).

At this point, the APIs for the movement of the body parts are provided. Each one of them takes as input a number identifying the degree of openness or closure of the respective motors, and the velocity of the transition. The functions realised are for moving the eyelids, the mouth, the chest (front or back movement), the neck, the eyes left/right and up/down, the tail and the body rotation. For the last function, also a character could be provided as input to indicate that the rotation should be performed with respect to that character. All the functions return a boolean that is true only when the motion has finished to perform.

Now is the turn of the high level APIs to create the sequence of movements that form a full emotion. The functions are created in the same way as the previous one, but an iterator manages the sequencing of the various actions and then is reset when the motion has finished. Each function also have a character as additional optional parameter so that the emotion can be performed looking at a specific robot
The reset position function just sets all the body parts to their initial configurations, making the robot return to the position it starts when first powering it.
For the idle state, a random number is picked and based on it the robot will perform an up/down movement of the tail, a body rotation or a blink, and then will wait for a second and start again.
For the happy emotion, the robot opens the eyelids, sets the neck backwards, rolls the eyes up and then starts opening and closing the mouth, moving the tail up and down and the chest back and forth, meanwhile a laughing audio is played. This gives the sensation that CamaLele is happily laughing.
For the sad emotion, the robot moves the neck and the chest forward, completely closes the tail, starts rolling eyes and lower eyelids up and down and then rotates to the right, meanwhile playing a crying audio. This gives the sensation that CamaLele is sadly crying.
For the angry emotion, the robot closes the eyes and starts opening and closing rapidly the mouth while rotating left and right, meanwhile playing a lion roar audio. This gives the sensation that CamaLele is angrily shouting.
For the shocked emotion, the robot first raises the tail a bit, then keeps eyes wide open, makes the mouth fall until completely open and completely closes the tail, meanwhile playing a ghost audio. Then, it blinks and stares again at everyone with eyes and mouth wide open. This gives the sensation that CamaLele is shocked by something.
For the doubtful emotion, the robot starts turning around with the chest forward while playing some perplexed sound, then the chest returns to normal position, the eyelids close and the eyes move left and right. Finally, the chest goes forward again and one eye opens wide

looking at someone. This gives the sensation that CamaLele is really suspicious about something or about a specific character.
For the annoyed emotion, the robot closes the upper eyelids, looks down then up to someone, then moves the eyes to the right and then rotates the body to the right. This gives the sensation that CamaLele does not care about a situation or a specific character.
Last function is the one for reading the phone messages, in which the audio of the messages is played and the mouth moves a bit up and down to simulate talking.

ESP32.ino

In the esp32_*version*.ino files the logic of the robot and its interactions in the environment are implemented. In the Arduino setup function, the setup functions from the previously defined libraries are called to setup the servos, the wifi connection and the mp3 player. During the set up, a web server is also initialised and turned on in order to make the ESP32 accessible for testing via web browser. The manual controller used to test the robot will be explained later on in this document.
Other setups are the task handlers for the two main loops and the random seed for the probabilistic behaviour.

The main code is organised in two main loops, each one of them is handled by one of the two cores of the ESP32. The first loop manages the switching of the states described according to the state machine, the second loop manages the network connection and takes care of receiving the messages, storing and computing how to react to them and gives this information back to the other loop that changes the state of the robot.

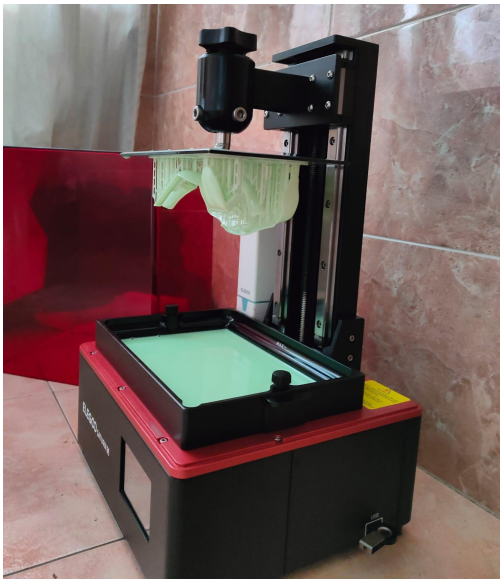
To make the connection to the Wifi and the god server resilient, it is periodically checked and eventually reestablished. To make it insensitive to messages with wrong format, instead, a check on the correctness of the messages is performed in order to detect if they have the proper length and sequence of characters.

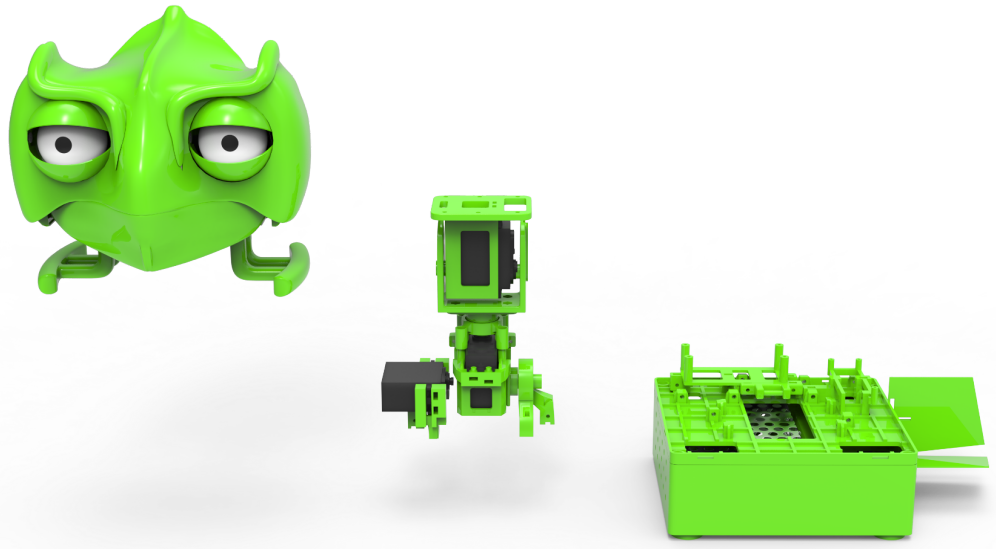
Manual controller

The manual controller is a website from which we send commands to the robot for making him perform arbitrary emotions with just a click. The controller has been realised mainly for debug purposes, to test the connection, understand what was happening during a specific emotion, and tune each of them. This is also the method we used to present the robot, while in the simulation phase the web server is inactive and useless. The website has been developed with node.js for server side and a simple HTML page for client side. The client has a really basic interface where the user can decide to set a servo to a specific position or to perform an emotion.
If the “Control” button is pressed, an HTTP request is sent to the ESP32 web server to move the selected servo to the selected position, while if the “Send emotion” button is pressed it will be sent the request to perform the selected emotion. If the request goes wrong a 404 page is prompted.
When testing the robot with the manual controller, there is no care about the messaging mechanism, so the robot will send wrong strings or no emotion, but this is outside the debugging scope of this controller and left to the Serial input and output.

Structure

We entirely 3D printed the plastic structure using 3 different types of resin for a total of 2.8 kg, only 1.3 kg were used in the final model as many pieces had to be tested and redesigned. The resins used are 0.3kg Elegoo water washable clear blue, 1kg Anycubic ABS like +, 1 kg Elegoo plant based though transparent green, 0.5 kg Elegoo plant based though white. The parts were first printed, washed in 99.9% IPA and later put under a UV lamp for the final cleaning and hardening.



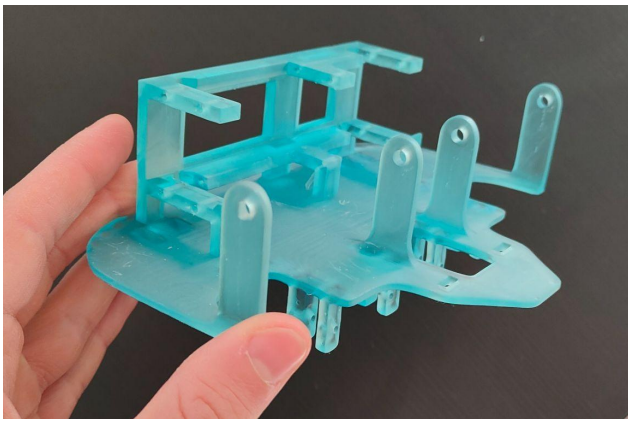
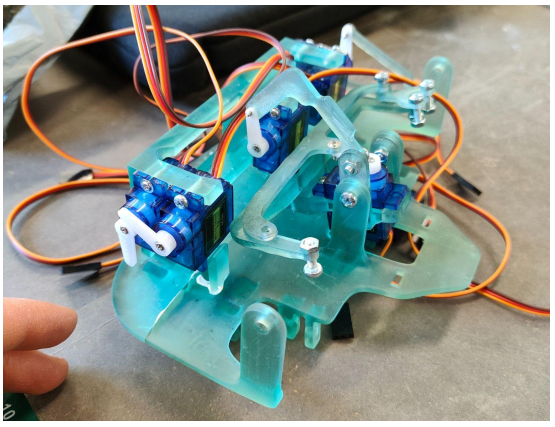


We planned the body in a way that is composed of three main pieces: the head with shoulders, neck and base with electronics. They are easily detachable as 4 screws connect

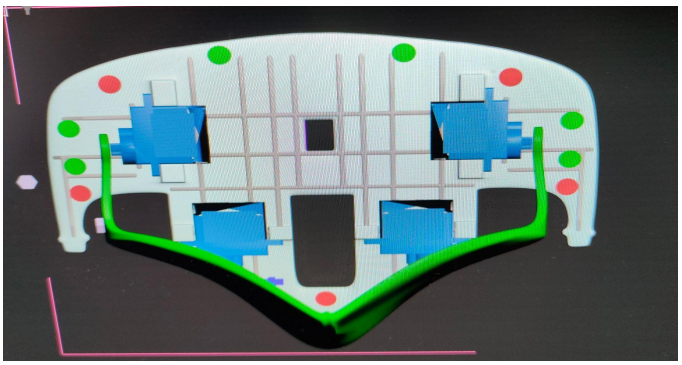
the neck and head and another four connect body and base. just lifting the shirt will reveal all screw placement. The neck and head have a hole to allow cables to pass through in an orderly fashion without interfering with movement.

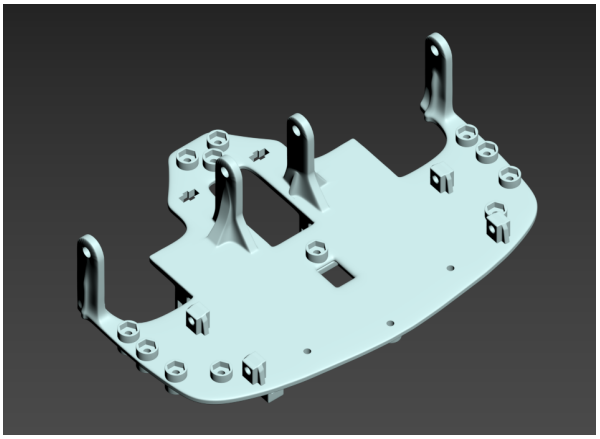
Head structure

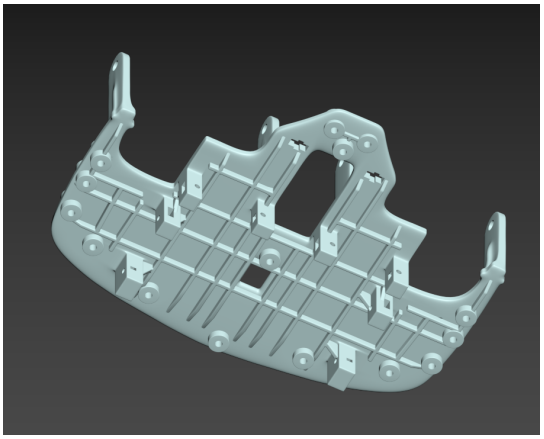
We started by using a clear blue water-washable resin and printed the first eyes and head support. The plate was completely flat and too brittle so it broke and had to be glued together. The motor supports were first attached to the base so we decided to split it in two pieces for further models in order to allow more flexibility.

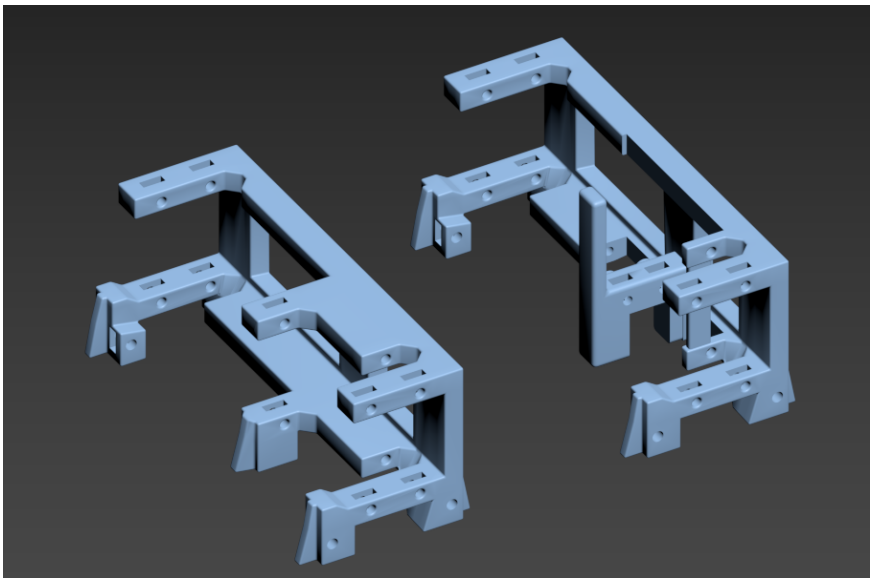
After further testing we decided to change the resin used to allow more flexibility and chose an Anycubic ABS like + resin, which allows up to 30% distortion before breaking.
A new base was redesigned adding plastic ribs to support the vertical eye supports, other improvements are the divided structure for the motors, sockets to place the screws and a hole in the middle to group the cables and bring them towards the base with the boards. Screw placement was decided as green for the back cover and face, red for the neck connection, later it was modified as shown in the following pictures. Rectangular structures are placed under it to fix the motors for the mouth and eyebrows, they have holes to insert 2.5 mm bolts and screws. The rest of the holes for the covers are for 3 mm bolts and screws. A cross shaped hole is placed in the nose to allow the eyebrows to be connected to the motors underneath it as there wasn't enough place in the top part to put another two motors without badly affecting the eye movement and aesthetics of Camalele.





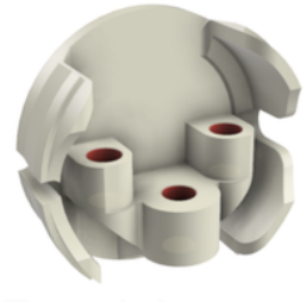


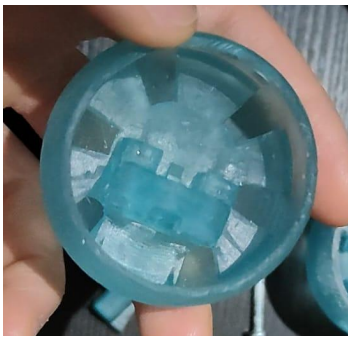
The new motor supports were printed two more times as separate pieces: the model on the right is adapted to hold 5 small servo motors, while the one on the right was intended for 4 servos and a bigger motor as the eyes seemed to need more strength to be lift up and down due to their big size. After printing and testing with both structures we decided to use the first one and simply use a longer lever for the connection. The structure was hollowed and redesigned to use less material than the one found online.

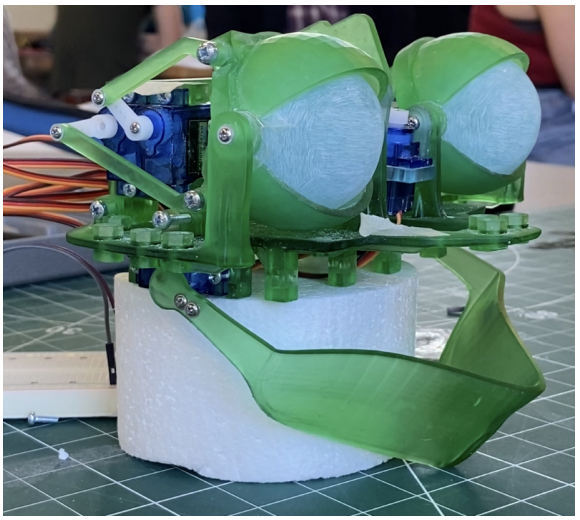


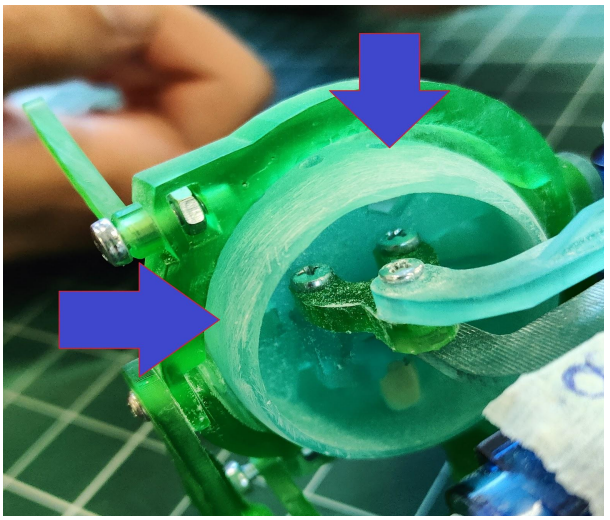
Eyes

We started by modifying the original eyes as they were flat and needed to be assembled with a second eye cover, compared to the original design (shown on the left it was hollowed and a structure with circular ribs was added.

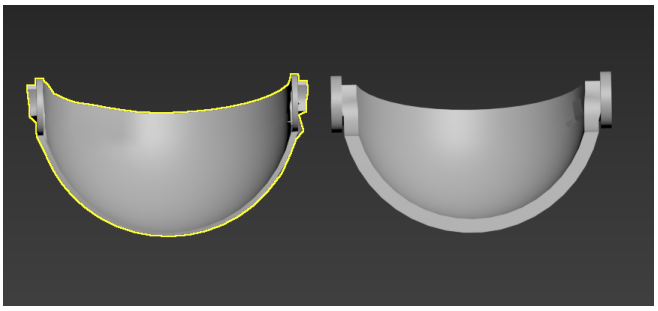


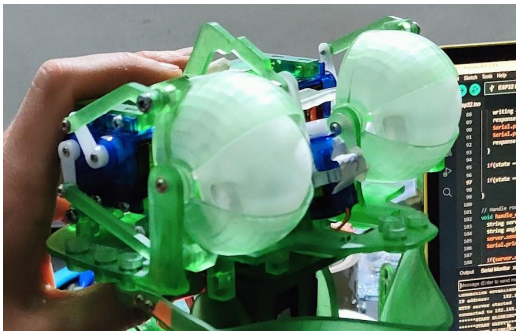




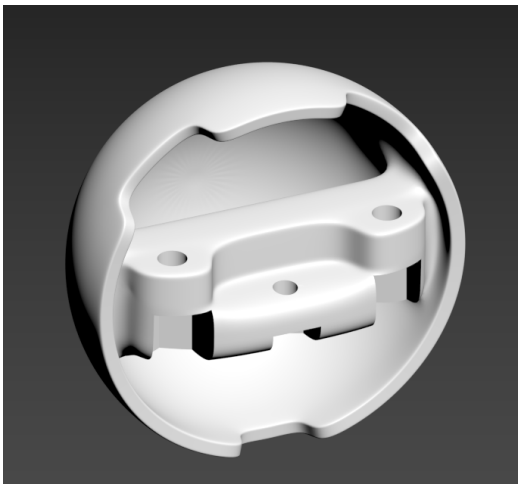


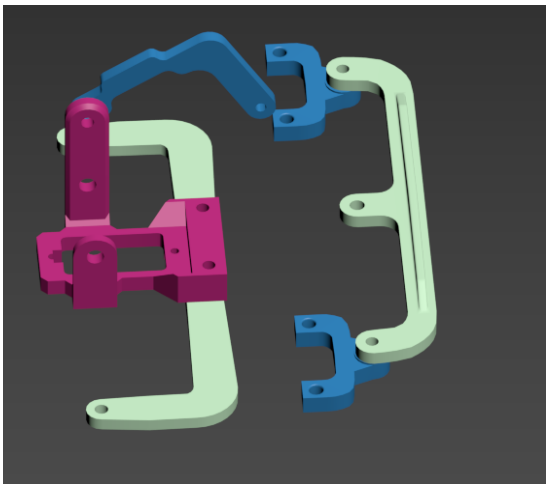
In the first prototype, it worked enough well, being our first attempt, from the point of view of the kinematics: using the eyes and the eyelids, in a separated way, both moved well, but we met problems when we put the components together because the eyes got in touch with the internal surface of the eyelids, as shown in the figure, and as result the motors weren't able to move smoothly the parts due to the presence of high friction. As a first attempt to solve this problem, we tried to unwrap the external surface of the eyes and the internal one of the eyelids, but the situation didn't improve enough. So we re-printed these parts making the eyes smaller and reducing the thickness of the eyelids, in this way we made enough room between the parts. In this way we solved the problem about friction, obtaining smooth motions.





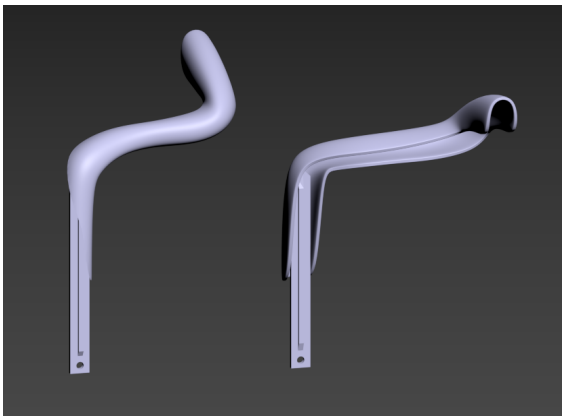
As it is possible to see from the figure, we solved the friction problem, but we got too much transparent eyelids, indeed also when they were closed, it was possible to see clearly the eyes. So this was one of the main reasons why we decided to paint the robot (another reason was that the final product was too ruined, full of indentations).
A new version of the eye was designed to help placing screws and bolt and make it lighter. In the first model we had to drill small holes on the top of the eye to allow a screwdriver to enter perpendicularly.





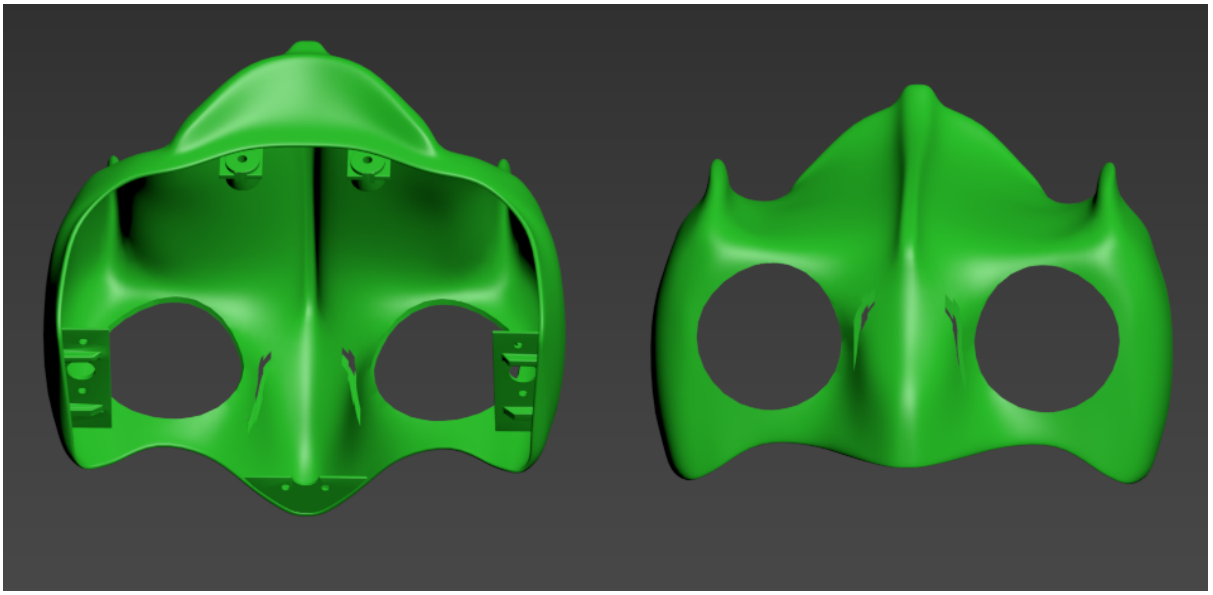
A new version of the structures to connect it was designed to help accommodate the larger eyes (more apart from each other and twice the size compared to the online model). A rib is added on the top part to make it less flexible and more resistant, the connecting levers were made longer so that the eyes could move further apart and allow a bigger rotation.

Eyebrows

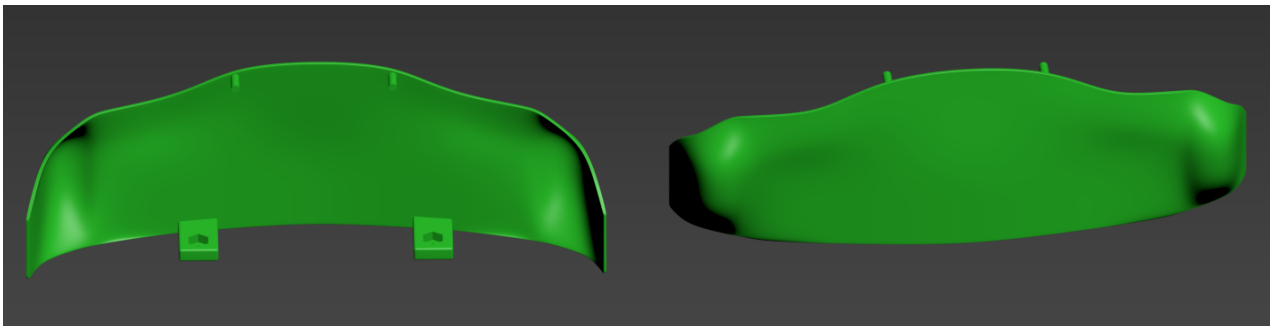


The eyebrows are designed with a hollow structure and a cross shaped pole that connects it to the motors underneath it.

Head

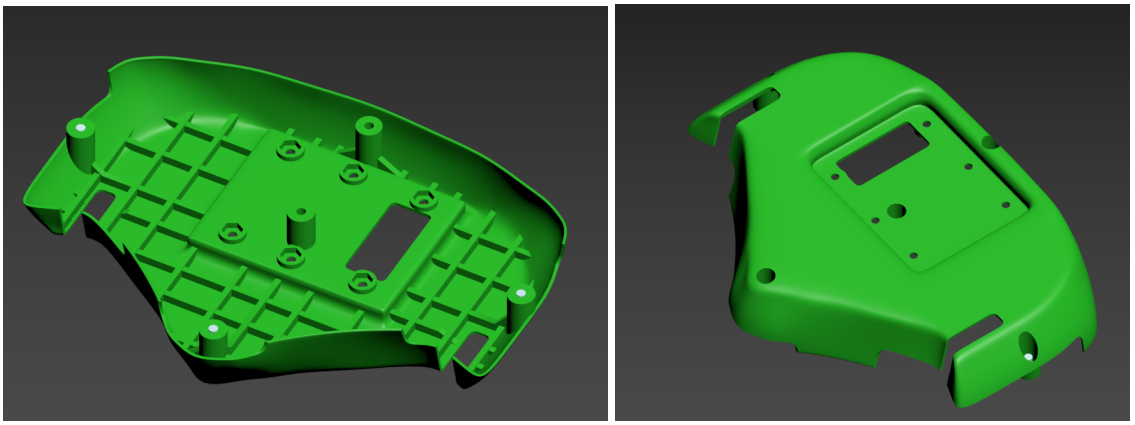


Two pieces are used as covers: the front of the face and its back. While the back only has holes for screwing it and connecting it to the face on the top part and the head base on the bottom, the face also has two extra holes for the eyes and two for the eyebrows.



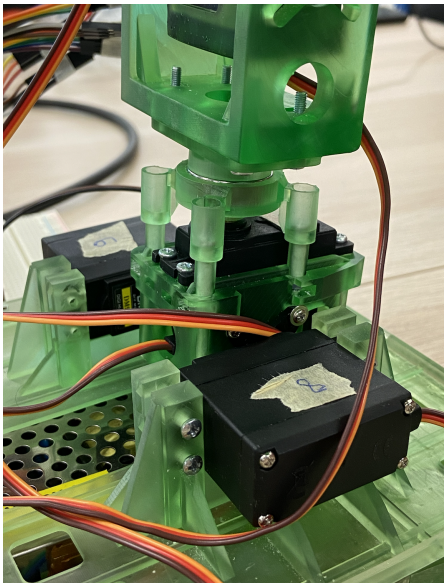
The two cylinders on the top work to align the back of the head, at first we decided to screw it together but we realised they would be visible so only the lower part was screwed to obtain a seamless surface.

a dark line is visible in the picture, it is not an empty space but the base for the head, we decided not to colour it to give a more interesting texture.



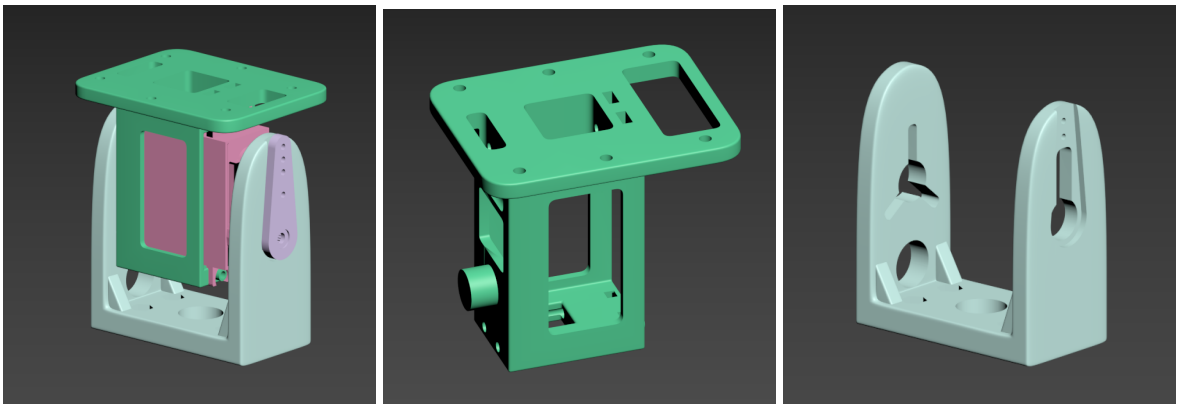
The lower part of the head was designed with an allotment for the top part of the neck, 6 m2.5 screws and their slots were planned to connect the two parts but we decided to use only 4 of them. 5 holes 3mm wide are placed on the sides and centre to connect it to the rest of the head. A hole in the middle is used to make the cables connect to the rest of the body. The cuts on the sides allow the mouth to move freely.

Neck



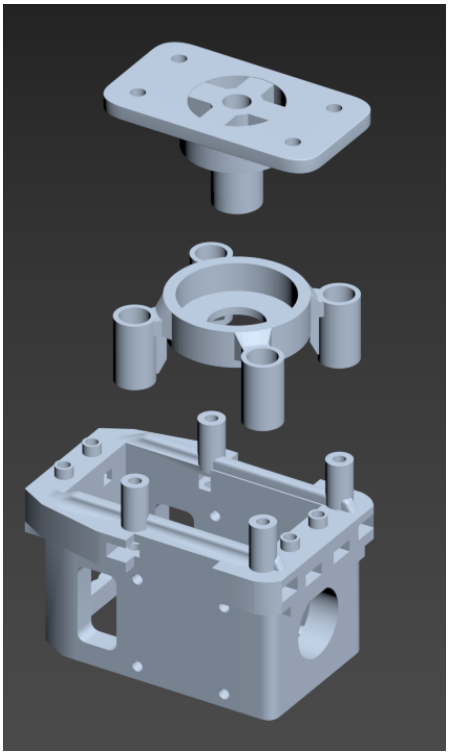

In the first prototype it was realised using two motors as shown in the figure; the main issue of this configuration was that they had to move perfectly synchronously, we added this in the code driving the motors with the same commands in order to move them together. However, even if we correctly programmed them, we met several problems related to a resistive torque that one motor acted on the other and as result both of them absorbed a stall current of 500mA, each one, from the power supply. This situation brought to a problem related to the power supply and a continuous restart of ESP32, making the usage of the robot impossible. So we modified the chest replacing a servo motor with a ball-bearing as shown in the figure. With this solution all the problems were solved and still the robot was able to rotate with only a motor.

The neck is made of three main pieces with different joints: the top part for head inclination, the middle for rotation around the z axis and the last one to tilt all the body.

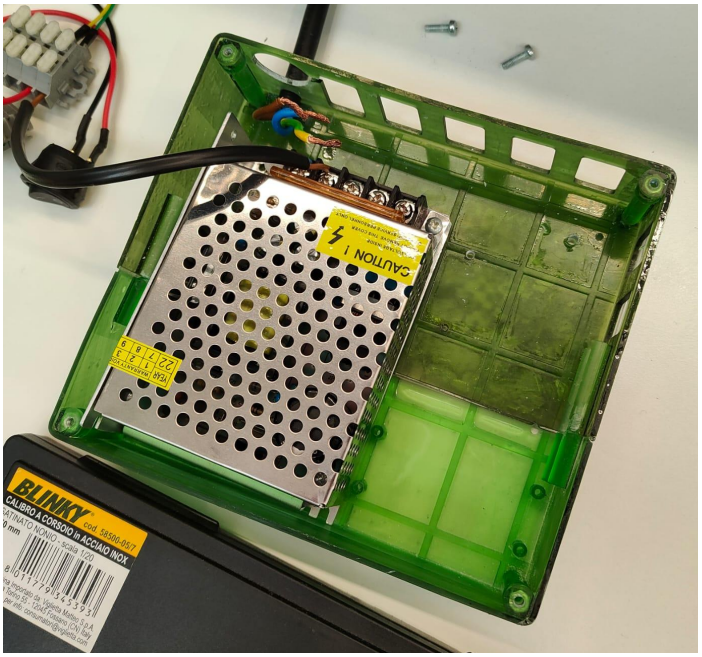


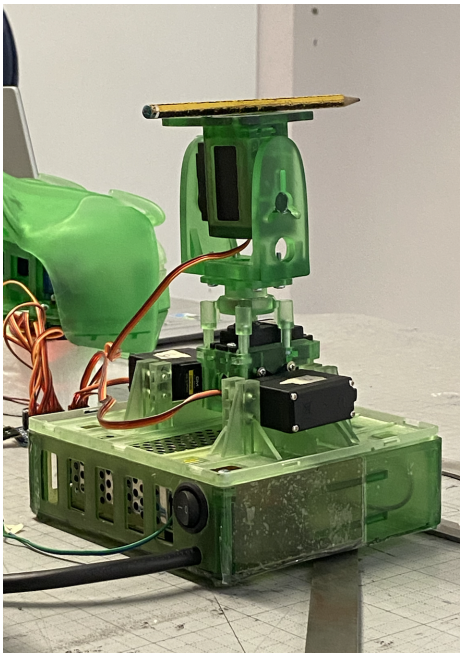
The first piece is composed of a container for the motor and a part that is fixed to the rest of the neck. a hole is placed in both pieces to allow cables to pass through. it works by placing the cylinder in the left hole and the motor gear in the other hole, it is later fixed in place with the horns of the motor. We developed it based on a design found online and adapted it to our project, in the online file there was no way to fix it as we wanted and it was planned for use with a different motor.

The second piece is made based on a camera tilter for drones we found online. We reduced the number of pieces and fixed the top part directly to the motor, while the project we found used a metal shaft in the middle. A ball bearing is placed between the two pieces to allow a smooth rotation. The bottom piece was entirely designed by us, the 4 holes on each side allow it to be fixed to the two motors (that later became one motor and a shaft) on the base.



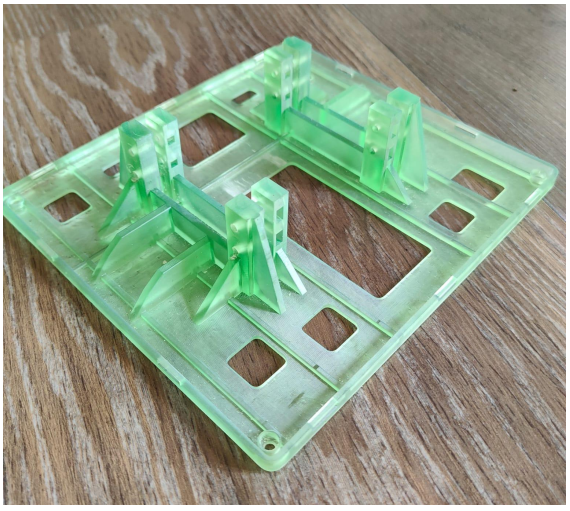
Base

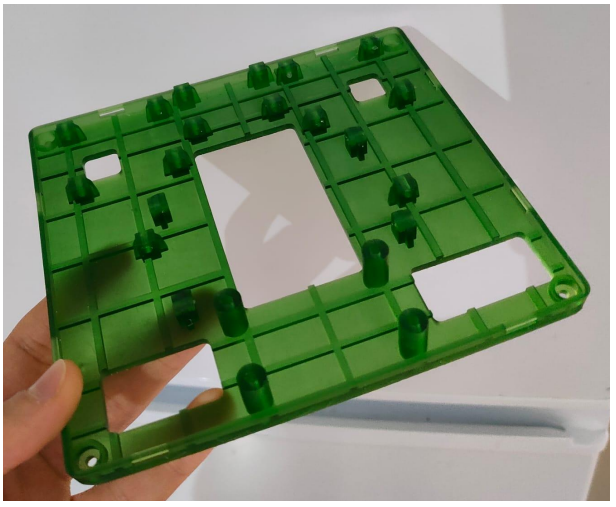




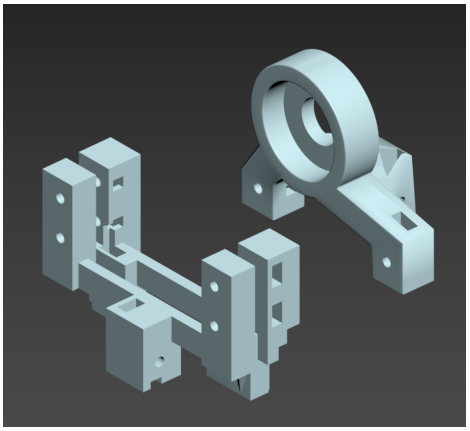
The first base design had big rectangular holes on the sides to allow air flow and was divided in two pieces, we used different resins to print the halves but we planned to paint it. The cable and switch are put on the side. The top part only had motor allotments.

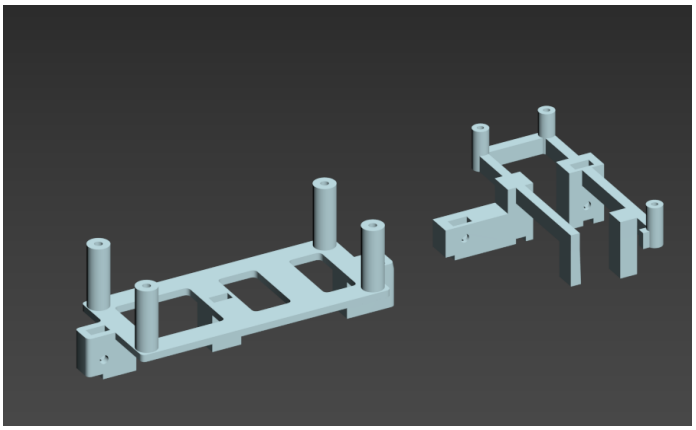
We realized this was not a good design for the motor alignment so we decided to change the lid and developed a new one with the tail on top of it and slots to fix some motors, we didn't directly create the motor slots on it in order to be more flexible. Four slots for m3 screws are placed on the sides.





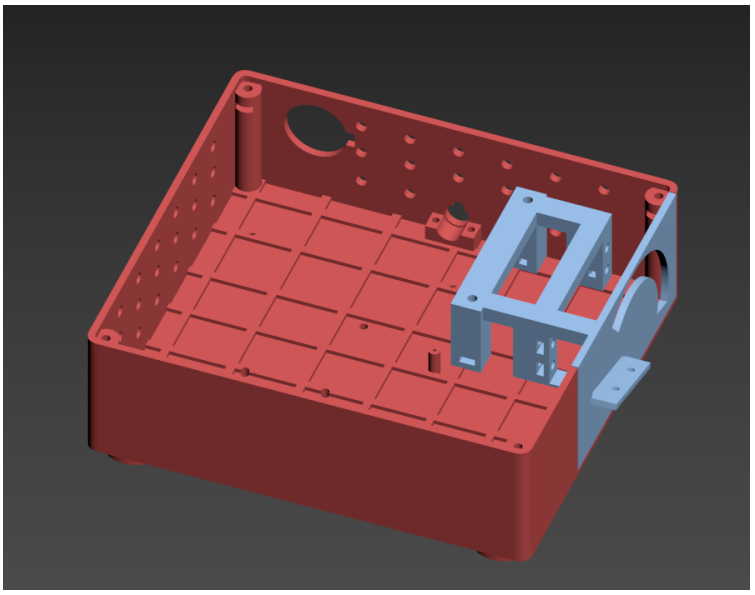
Later we decided only to use it for one motor, speaker, esp and ball gear so we designed some pieces to put on it.





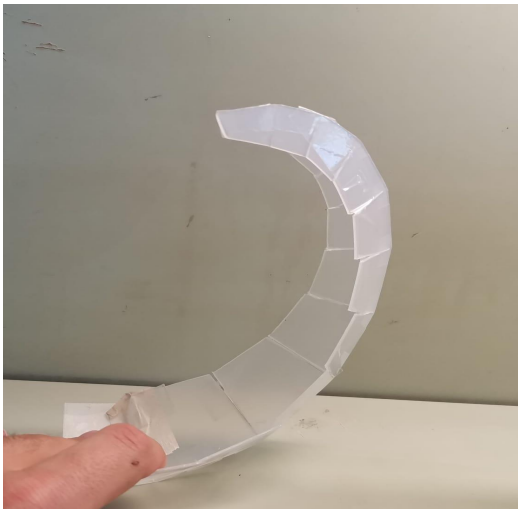
On the left picture there are the ball gear slot and motor slot while in the right one the slot for the speaker and the esp, since they were designed to avoid printing again the base and wasting material some shapes seem to be weird yet effective.

The last version of the base was printed as a single piece, the cable had to be moved since the power supply was rotated by 90 degrees and the cable could not fit anymore, the switch remained on the side. The power supply is contained in the left part, meanwhile in the right part a board is placed and a different piece to move the tail.

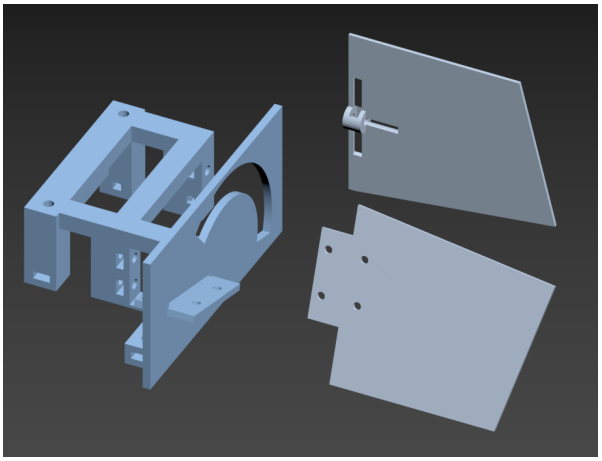


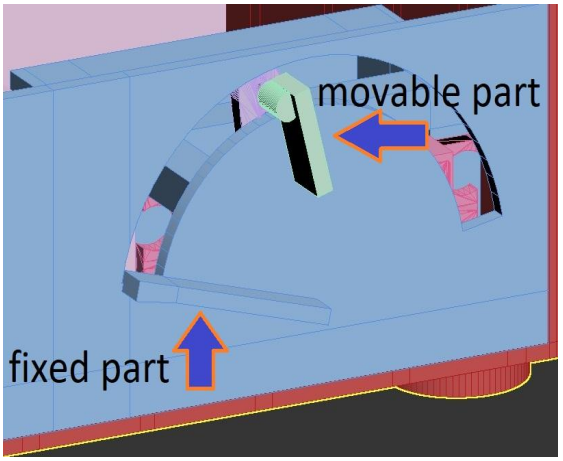
Tail

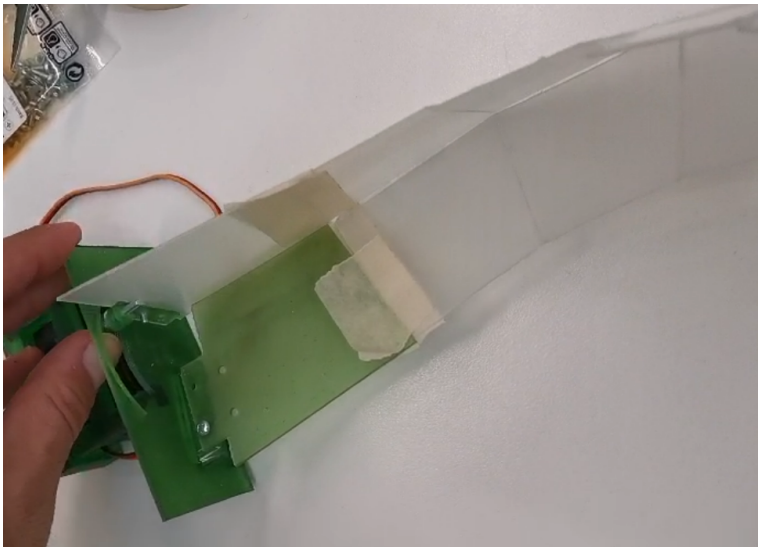
About the tail: once the dimension and shape were selected (done in the define section), we realised it using a sheet of polypropylene, in particular we use a scissors to cut it and normal scotch to connect the parts, and we got:



The tail was not cut all the way through so it was cut again from the leftover polypropylene sheet, separating every joint one from another.
Once we realised the final tail, we found a way to move it using servo motors. Unlike the other kinematics, this one should be entirely designed by us because in the original project the idea was to move it by hand. So we solved the problem fixing a part of the tail to the basic of the robot and the other one to the servo motor:

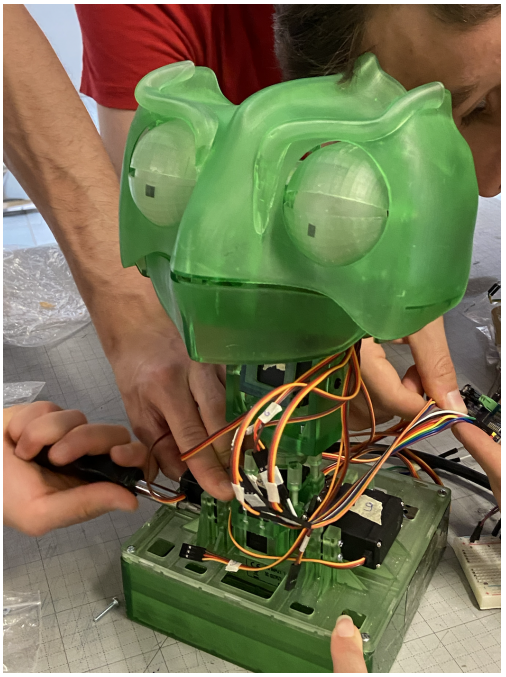


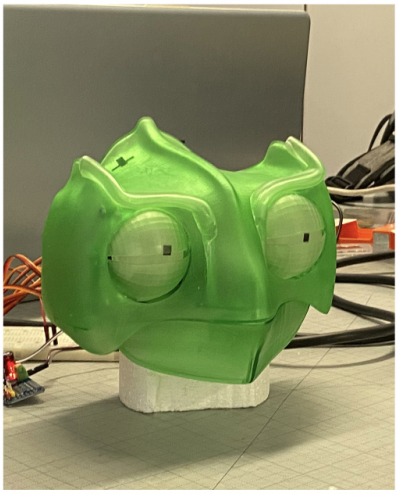


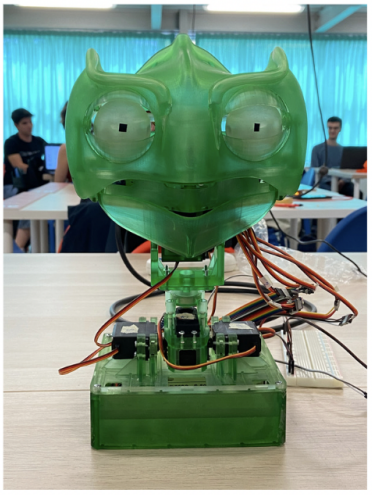


The first two pieces were printed so they could be fixed and screwed on the base. The motor slot is screwed on the base from underneath it.

Shape:

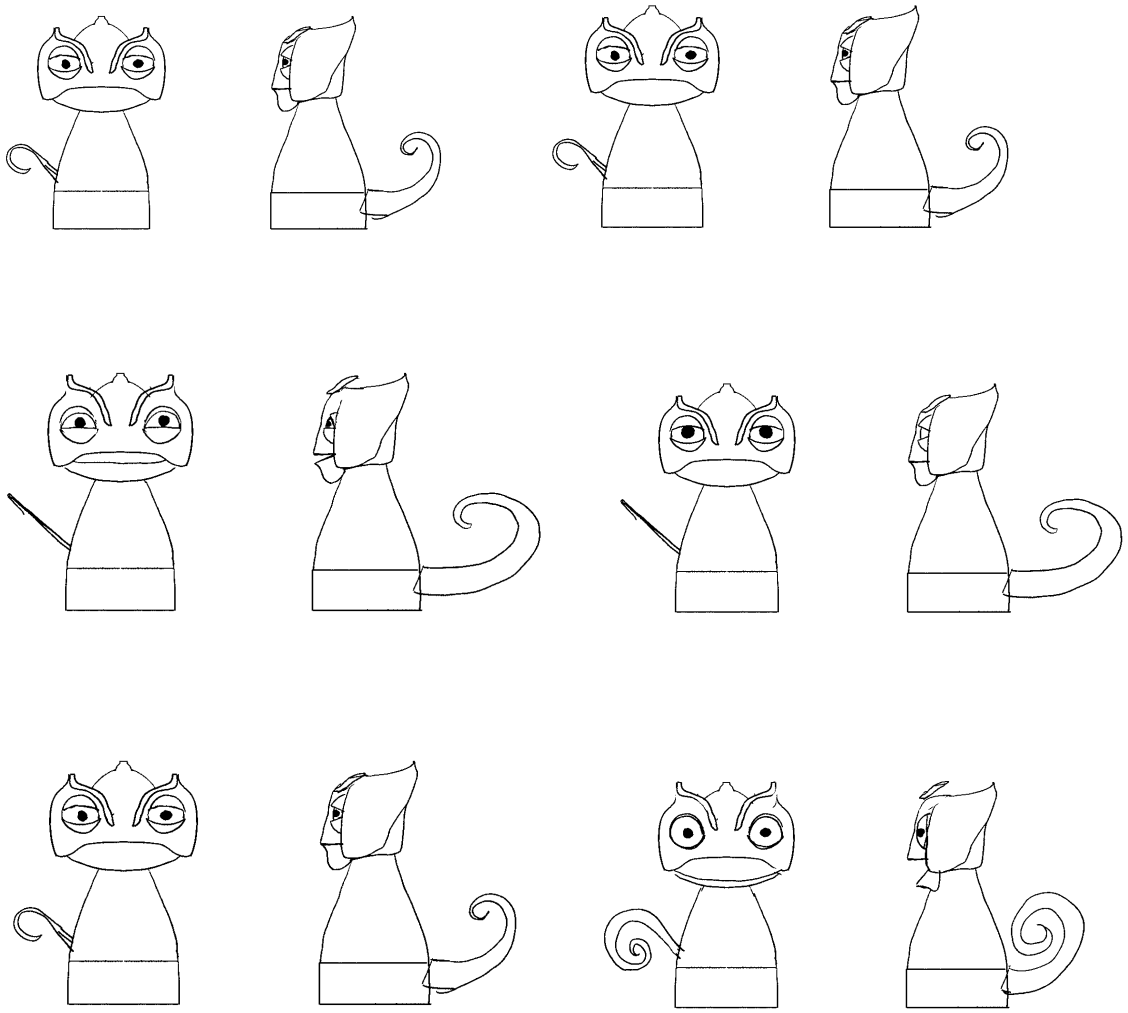






This is the point we got at the final revision, after it the eyebrows were reprinted and the base and board supports were designed again in an improved way. It was still missing the back of the head and wires had to be cut.

Emotions:



Emotions were first designed with some animated drawings which we later discarded as they didn’t consider neck movement and we had to remove the eyebrows movement.

Phase 4: Deliver

In this phase we describe the final robot.

Final Robot description

Strategy

At the presentation, our robot was able to:

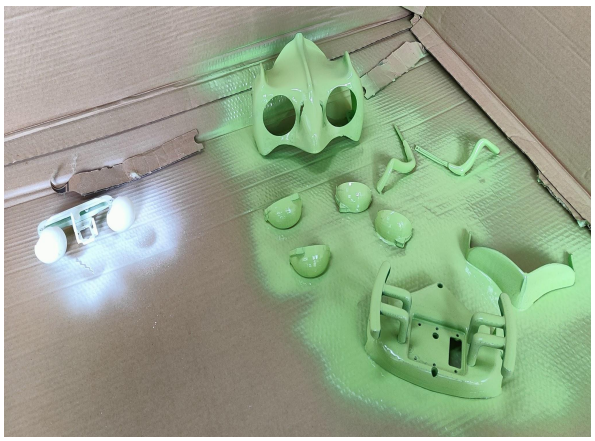
- connect to the local network (also able to reconnect in case of restart of the god server).
- receive a message from other robots, then reproduce the corresponding emotion.
- If another robot was playing its sentence, wait until the “GG” message, then reproduce the corresponding emotion.
- receive a message from god server, then, based on the case, reproduce prerecorded sentences, notify the end of the message (by means of the “GG” message) and reproduce the corresponding emotion.

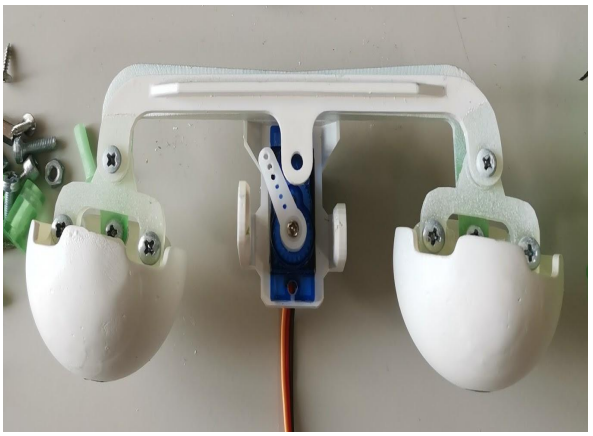
Shape

The final 3d model can be downloaded at this link:

https://drive.google.com/file/d/1jA7dnD3a8s3VRmKyglTxgAuXqDSeuEC2/view?usp=drive_link

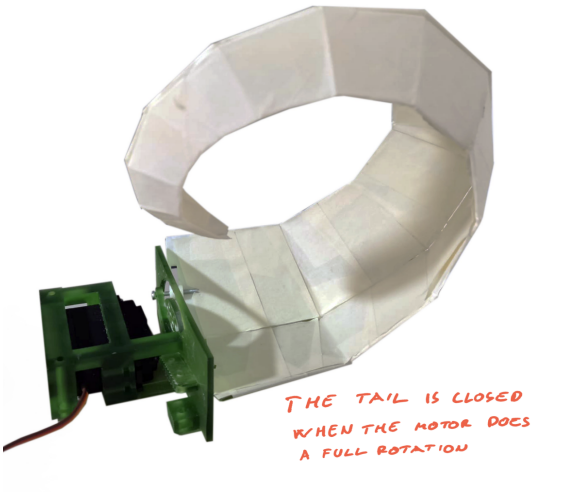
To give the robot a better appearance for the presentation, we decided to paint all the external parts of it and buy a shirt. We selected a green paint recalling the colour of the chameleon and after a search in different shops, we found the suitable green. We painted the robot many times in order to give him a good colour. Unfortunately it rained when we painted it so it proved to be a hard task and we ended up using a hair dryer to dry it, this resulted in some cracks in the paint.





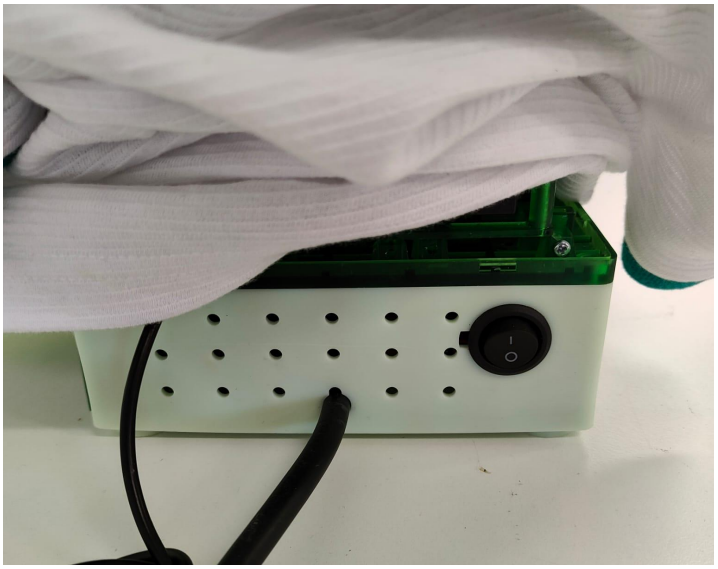
We put a shirt on and painted the inside of the mouth in red so it could be seen as actually open in a more realistic yet fun way.

About the tail: having the plastic one realised before, we had just to paint it. To do so, we first cover it with paper tape and use glue to fix it, getting:

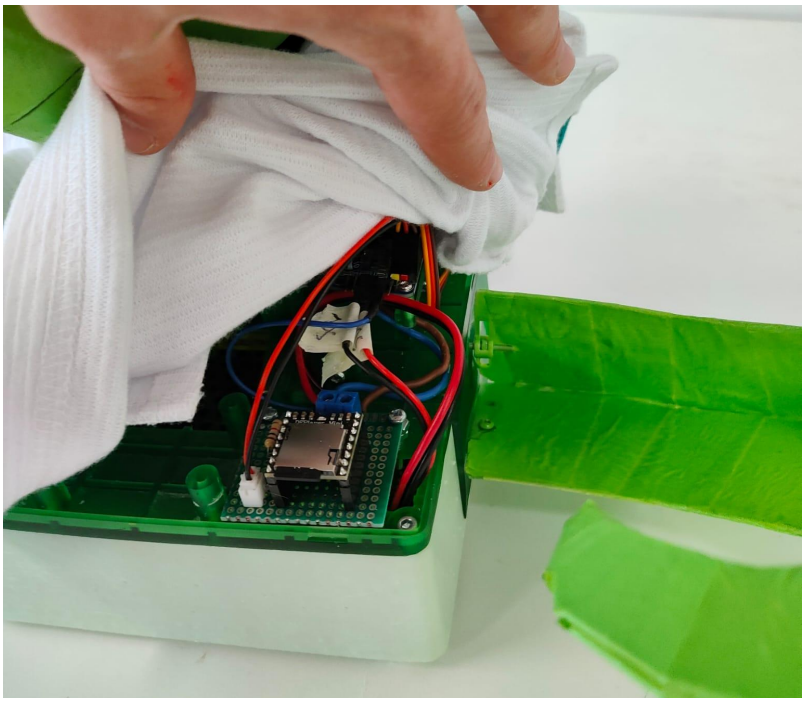


Then it was ready to be painted, for this task we used a water-based paint, and the result was:

About the base:

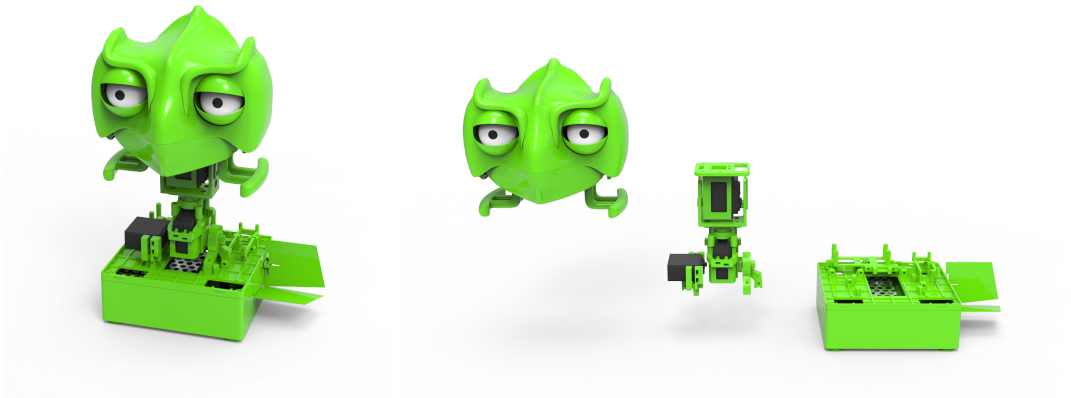


In the new base the air holes are round and the cable exits from the centre, on the side there is the switch for on and off. As it is covered by the shirt we decided not to paint it and keep the different colours.

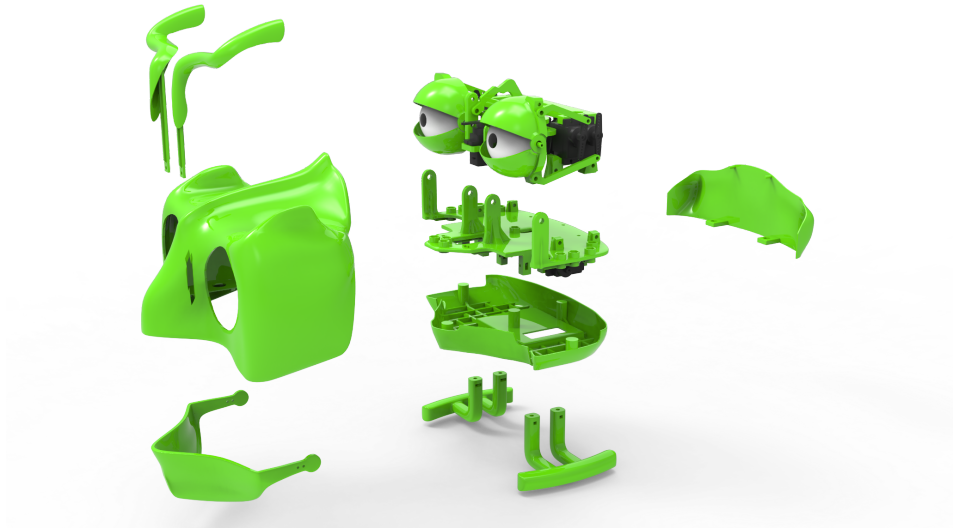


The slot for the mp3 card is screwed on the front, by lifting the shirt it is very easy to access and change the sounds if needed.

CamaLele has his own resting position and the neck needs to be tilted first towards the back and then the head towards the front. When turned on he will automatically straighten his neck and wake up.
We planned to put a spring to make him stay “awake” also when not in use and printed the pieces needed to connect it but didn’t manage to set and test the motors with the spring attached before the presentation so we had to leave him in the sleeping position.

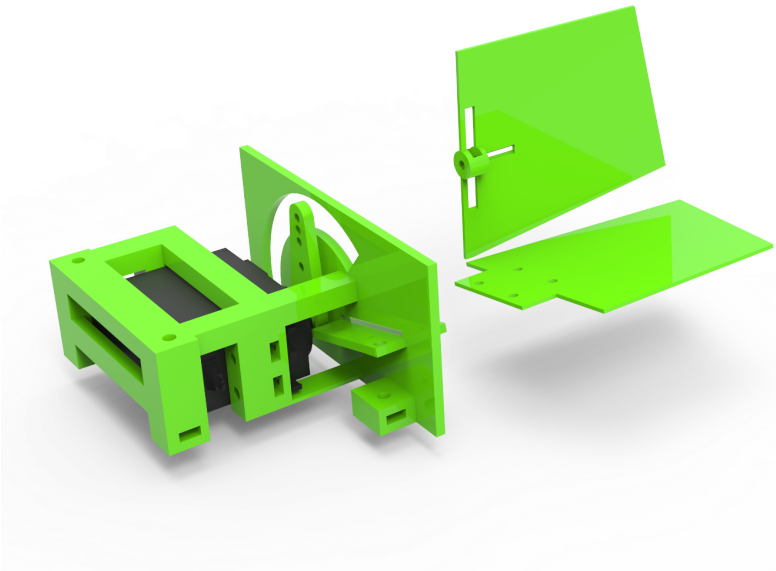


Final model with all the pieces and assembly

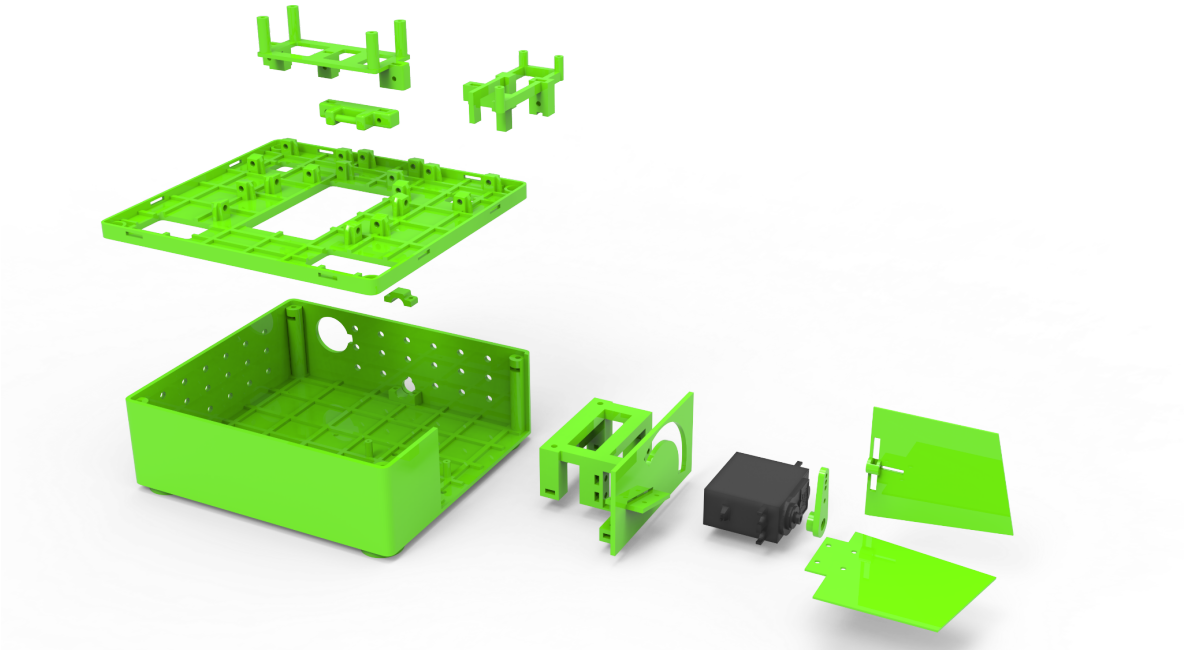


Head assembly.

Mouth assembly.

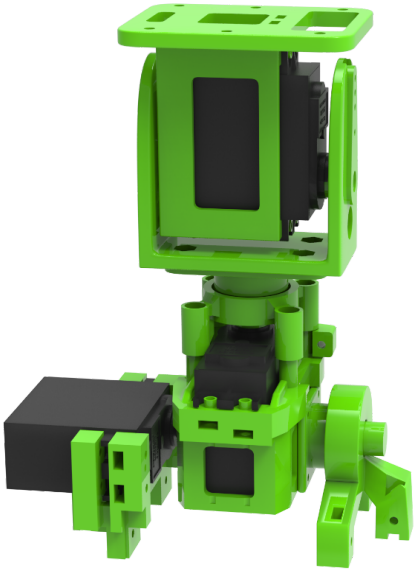


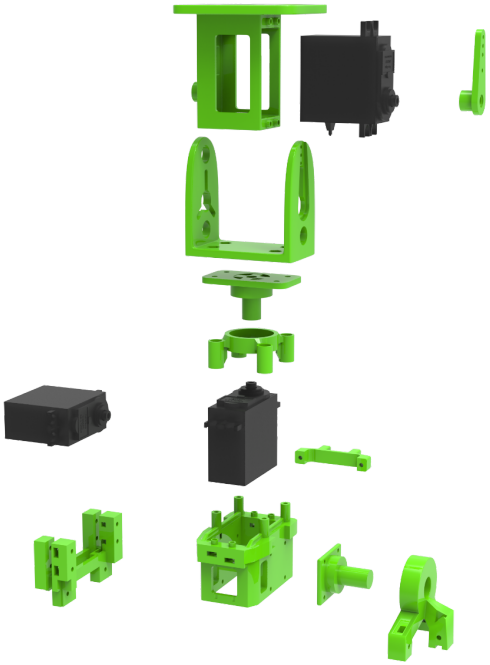
Tail connection.



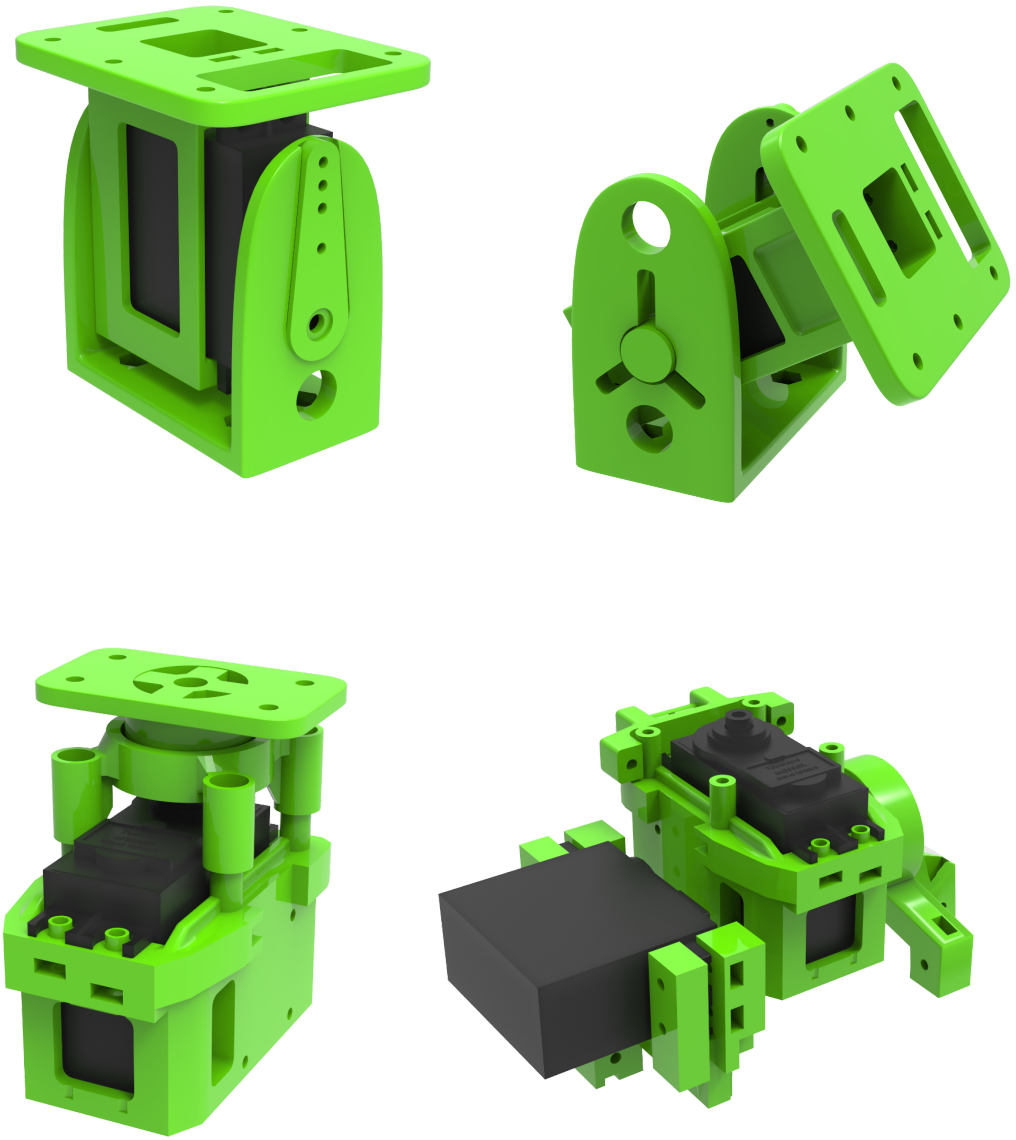
Body assembly.

Eye mechanism.





Neck assembly.



Neck pieces.

Emotions:

Happy emotion

CamaLele starts from idle and tilts the neck and eyes upwards, he starts laughing and moving the tail. While laughing he moves the mouth revealing its inside and a laugh sound is played. The mouth opens up but not excessively.

Sad emotion

CamaLele starts looking down first with only the eyes and then with all the body. He looks left and right while a crying sound is played. Tail movement is minimal

Angry emotion

CamaLele lifts his tail to the maximum position and squints the eyes, he looks left and right while opening the mouth very wide and fast. A roaring dinosaur sound is played.ù

Shocked emotion

CamaLele lowers opens the eyes and mouth really wide and later he tilts his head upwards and moves the tail down. He does a few disbelieving blinks while an ethereal sound is played.

Annoyed emotion

CamaLele lowers the top eyelids while he opens the lower ones, he looks downwards with the head and eyeballs and then he brings up the eyeballs again to make them half visible. He ends up looking a little pissed right and left.

Cautious emotion

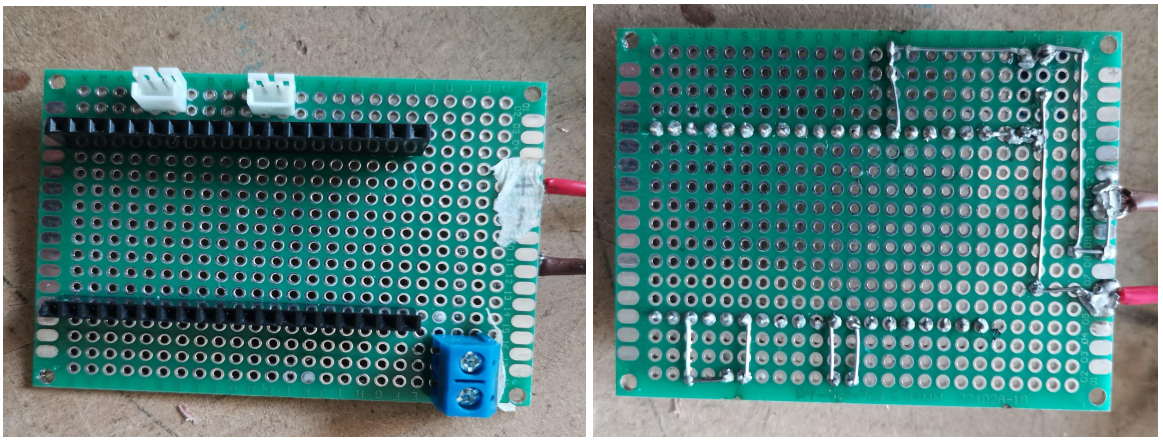
CamaLele looks left and right with the eyes open by rotating his neck and tilting forward both with the neck and body. He gets straight again and this time he closes the eyes and moves only the eyeballs in a suspicious way. He tilts forward and rotates his body with and eye squinted and one fully open.

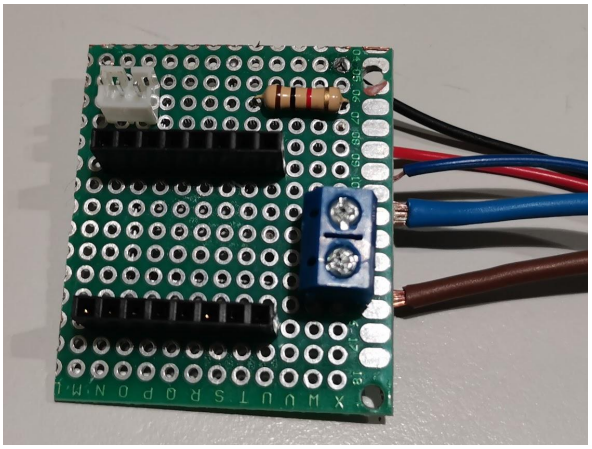
Idle position

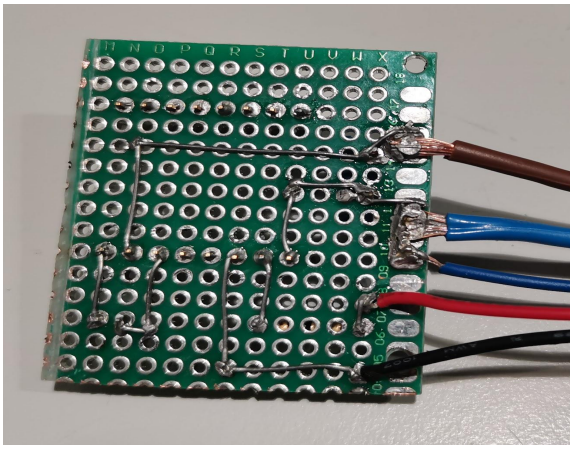
Idle position includes blinking, moving the head left or right and small tail movement, he can do one of the 4 things with a 25% probability and will wait a bit before doing another movement.

Electronics

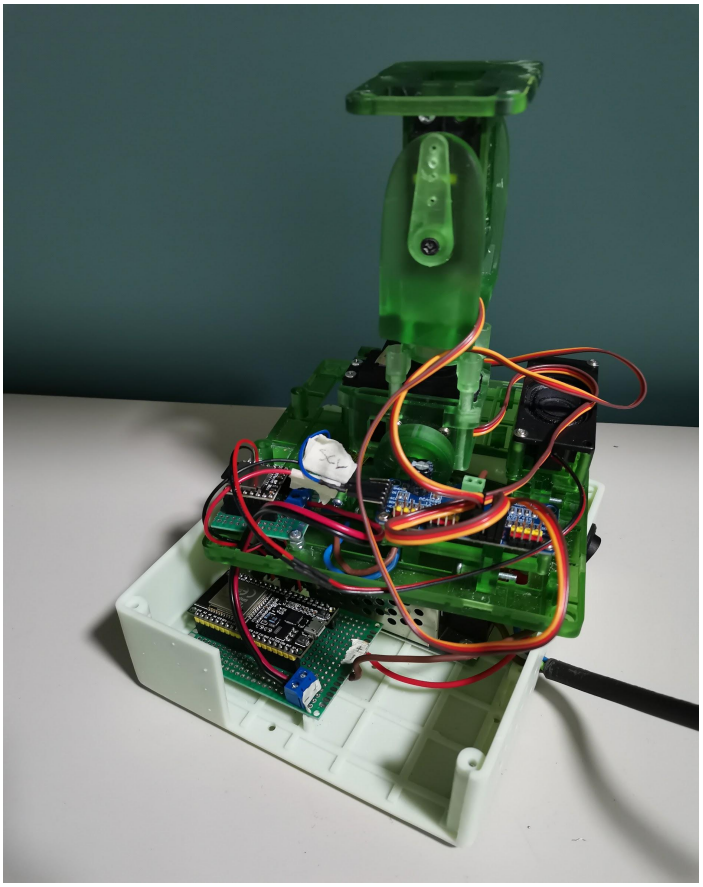
Schematics:
Once we finished to print all the parts of the robot and make the final tests, we realised the circuit:







The most challenging part was to displace the electronic parts on the board in the right position taking into account were the board would be assemble, indeed many pins must be accessible also after the assembly of the board on the robot, the orientation of the ESP32 should be so that it is possible to connect the it to the laptop by means of the cable and for the Mp3Player it had to be possible to remove the SD card to update it.
So taking into account where the boards would be placed inside the robot, we also had to organise the space on the board in order to create all the links between the different pins avoiding the overlapping of wire (they are made of tin, so they can't get in touch).
Final result with all the circuit connected:



Bill of material:

component	model/typo	cost	quantity	total cost
servo motors 10pcs	SG90	€ 21,98	1	€ 21,98
servo motors 5pcs	SG90	€ 7,99	1	€ 7,99
Cavi per Arduino 5M		€ 10,99	1	€ 10,99
Servomotore 5pcs	Servomotore Digitale MG996R	€ 37,26	1	€ 37,26
Resina per Stampante 3D	Resina ABS-Like Plus	€ 34,99	1	€ 34,99
Alcool Isopropilico PURO	Cleanser Ipa art. 102 isopropanolo	€ 12,50	1	€ 12,50
Modulo PCA9685 16 Canali	AZDelivery	€ 13,49	1	€ 13,49
Alimentatore di rete	5V 10A, TASVICOO 50 Watt	€ 16,00	1	€ 16,00
connettori	Pin JST Kit Connettore	€ 15,99	1	€ 15,99
speaker		€ 11,99	1	€ 11,99
power on/off		€ 3,99	1	€ 3,99
				€ 183,18

Informatics

To recap what we did at code level from the beginning of the robot creation process until the delivery phase, the decision about how to organise the project code has been to divide it in several libraries, and import each of them in one main file to manage an aspect of the whole robot behaviour.
For the communication, it has been decided that each robot would send a message to a *god server* when performing an emotion, and this server would have broadcasted all the incoming messages to all the robots and sent the external messages simulating the phone messages whenever requested by input.
The final choice for the protocol for exchanging messages was to send a two characters long string for a god message and a four characters long string for an emotion sent by a robot.
Our group then decided to implement the following emotions: idle, happy, angry, shocked, sad, cautious, annoyed.
We decided to structure the code life cycle of the robot in states: we thought about having a state for each emotion, in which to make the robot perform the proper reaction, plus a state for resetting robot position, one for reading the phone messages, one for turning to the other robots when they were reading phone messages, and the last one to wait until the other robot finished reading.

During the development phase, the whole architecture previously defined has been finally realised. We have created three main libraries: one to manage the movements of the robot and to implement the sequence of actions that define an emotion, one for the wifi and client setup and one for the mp3 setup and audio management.
External libraries used for setting up the connection and web server are the Arduino libraries *Wifi.h* and *WebServer.h* .

The mp3_setup library uses the Arduino libraries *DFRobotDFPlayerMini.h* and *HardwareSerial.h* to implement the software setup of the mp3 player.
The main external libraries used to realise the functions for robot movement are *Adafruit_PWMServoDriver.h* and *Wire.h*.

At the end of the development phase, we had a ready-to-work prototype to test and fine tune. The last strategy to discuss before proceeding with the testing phase has been to decide how to react to the phone messages and to the emotion sent by other agents. To deal with the phone messages, we decided to react to each of them with a different deterministic emotion, as shown in the following table:

code	to	sms	our reaction
G1	[1]Rocco	The house in Salina is confirmed	[C]happy
G2	[1]Rocco	You embarrassed me today when you gave me the condoms	[D]shocked
G3	[2]Eva	Good evening Eva. I'm the surgeon who operated on the enlargement of your breast. Are you doing okay with the new implants?	[D]shocked
G4	[2]Eva	Hi mom, tonight I'm sleeping at a friend's house. See you tomorrow. Dad has already said it's ok.	[H]cautious
G5	[3]Lele	I miss your kisses so much.	[C]angry
G6	[3]Lele	What are you doing? You're such an idiot! You know what the truth is? I don't think you even know if you like men or women!	[D]shocked
G7	[4]Carlotta	You got a new message from +39346237173. Dear Mrs De Rosa. We wanted to inform you that we have availability this month. Thank you for choosing us. Kind regards, Residence La Quiete	[E]sad
G8	[4]Carlotta	Did you… wear the panties tonight?	[C]angry
G9	[5]Peppe	It's time for your workout!	[C]happy
GA	[5]Peppe	Nude	[C]happy
GB	[6]Bianca	"... I want to f**k"	[D]shocked
GC	[6]Bianca	I'll make you lasagne when you come with Cosimo this weekend	[J]annoyed

GD	[7]Cosimo	Did your wife like the earrings?	[H]cautious
GE	[7]Cosimo	Hello love, I took the test and two lines appeared. I think I'm pregnant.	[D]shocked

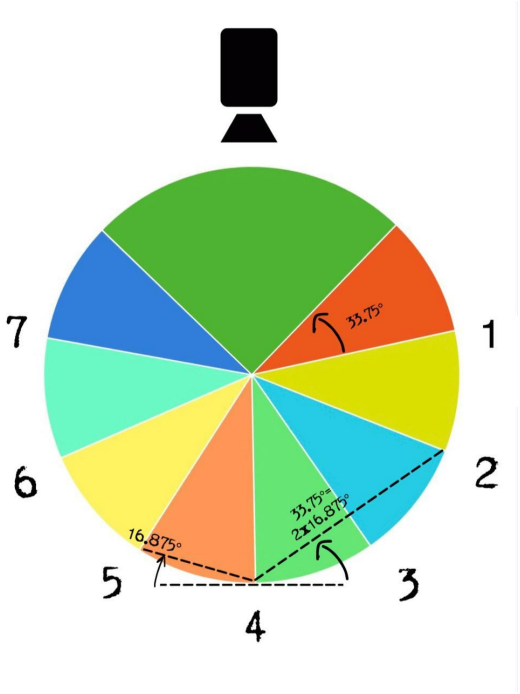
To deal with the emotions sent by the other robots, instead, we decided to create a buffer of three elements in which we store all the messages that were relevant for us, if present. We decided to store only a few messages because otherwise there could be the risk of reacting to messages that were too distant in time and this could impact on the realism of the performance. So we decided to check the integrity of each incoming message and store it if the buffer is not full and if the message is interesting according to our character behaviour (e.g. an emotion sent between two characters not relevant to us is simply discarded). If the buffer is already full, instead, a priority mechanism has been implemented to check which message to remove from the queue: this mechanism is mainly based on the sentimental closeness of the sender to our character, the kind of emotion, the receiver and the difference between past actions (in order not to react always to same character in same way and create loops).

When the robot is ready to perform a new action, it checks the availability of a message in the buffer, removes the one with highest priority from the queue and executes the relative action. The response to a message depends on a behaviour implemented to us that rìtries to respect the actual feelings of Lele, the character imprisoned by CamaLele in the simulation. The algorithm involves some deterministic behaviour for some kind of messages and some stochastic behaviour for other ones.

After the end of the development phase, we had to fix some problems we faced from the informatics side. First problem we found was about how to record the phone messages to read and how to raise the volume of the audio, because the speakers were already at maximum volume but the sound was still quite low. For the phone messages, we decided to generate the audio files with some text-to-speech generator tool found on the web, that created some audio files with the deepfake voice of the actor Liam Neeson (that has an English voice similar to Lele’s one) reading our phone messages. For the audio volume, we found an online tool to raise the decibels of the original files with good results.

Another problem we found was that the idle state was too noisy and eventful, so we decided to slow down some movements and to delay the time between each of them.

Last problem we faced was how to correctly place our robot in the environment to make it turn correctly and simulate a more realistic talk with the other robots. As stated in the previous sections, we have developed an algorithm to calculate the PWM given the angle between CamaLele and the other robots and then store the value in a constant. Initially, the environment should have been formalised with a regular octagon inscribed in a circumference, so the angle between two robots was easily calculated. Then, due to the exigence to have a wider angle for the camera, the decision moved to another more irregular configuration that we can see below:



The geometry to calculate the angles between CamaLele default angle and the other robots is shown in the appendix.

Here the links to the final version of the code, available in the google drive folder of the course and in the github repository of our group:

- GitHub repository of the project: puch66/Robotics-Design (github.com)
- Drive folder for the software: SW - Google Drive

Conclusion

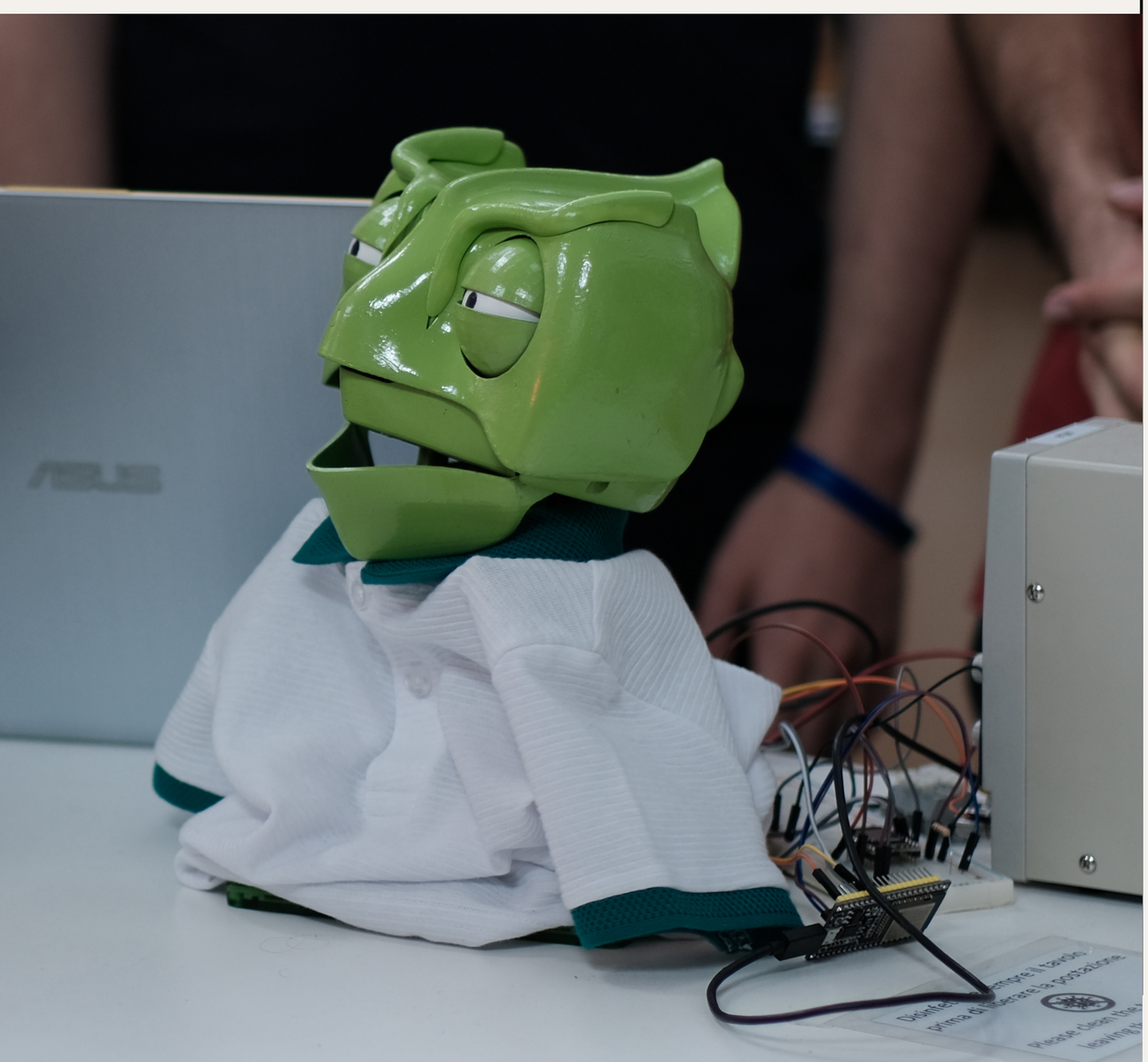
The simulation provided us with some interesting insights: the robot was able to perform all the emotions correctly when asked and in an expressive way. The state machine worked fine, apparently with no bugs, and the robot could each time stay in the idle state, react whenever an interesting message came, turn to the other robots while they were speaking, and read out loud messages. The informatic structure demonstrated to be resilient to the several server crashes, as the robot always reconnected automatically and in a short time. The message protocol was correctly implemented from our client side, as CamaLele always turned towards who was speaking and waited for the “GG” signal despite some other robots were yet performing reactions, and correctly reacted to the god messages, reading out loud the message and sending “GG” after finishing, and halting when “GF” was sent.
Unfortunately, the priority mechanism could not be tested that much because the robots which were closer in our hierarchy almost sent no message during the whole simulation. Nevertheless, the (few) reactions to the other robots demonstrated to be correctly implemented, even if some reactions seemed to be inappropriate to the context, maybe because they were delayed in time or because the less realistic chain of emotion has been followed by our algorithm.
A possible solution for the whole presentation to be more realistic and understandable could have been to let the god server manage the priority of the messages and identify which character should react at any time, letting the client side choose only which emotion to perform given a more simplified and understandable context.

The teamwork chain worked really well, the group had the ability to split the job in equal parts and work together most of the time, creating an optimal atmosphere for working happily and enjoying all the aspects of the projects, even the ones that were less relatable with the personal previous studies.
During the development phase, we noticed that we missed some components or that the one that we bought were insufficient in number for the realisation of the prototype, most of the types because of our errors during the development process. This made us understand that it is always a good choice to buy extra components in case of faults.
The power supply has been the main issue during our development process: in the first presentation, our robot suddenly stopped working and after many meetings we found out that the problem was not about the stability of the motors but instead it was a matter of correctly supplying power to the circuit. Also in the case of setting up the mp3, it took us some time to discover that there was initially an incorrect setup of the power supply circuit that made us lose some time in identifying the problems. This experience helped us a lot to understand the delicacy of the power supply and that, in general, when developing a robot the power supply is a crucial aspect that must be designed and implemented with extreme care.
The movements were the funniest part to develop because we could see the robot acting and expressing as we desired, and the testing phase took no exaggerated amount of time. The timing was well managed between the design phase, the printing of the components, the setup of the motors and the code and the implementation of the emotions, and a correct timing between all of us let us finish the work just in time for the presentation.

Overall, this experience has been really exciting for all of us, bringing a result that is proportional to our great efforts and expresses all the involvement, dedication and passion we put for developing our CamaLele.

APPENDIX

MASTER DEGREE IN ARTIFICIAL INTELLIGENCE AND ROBOTICS



Design and Robotics

CAMA LELE

AIRLAB
ARTIFICIAL INTELLIGENCE AND ROBOTICS LAB

XI° edition, 2023
Group number 3

Professors:
Andrea Bonarini, DEIB department, Politecnico di Milano
Maximiliano Romero , Design department, IUAV, Venice University

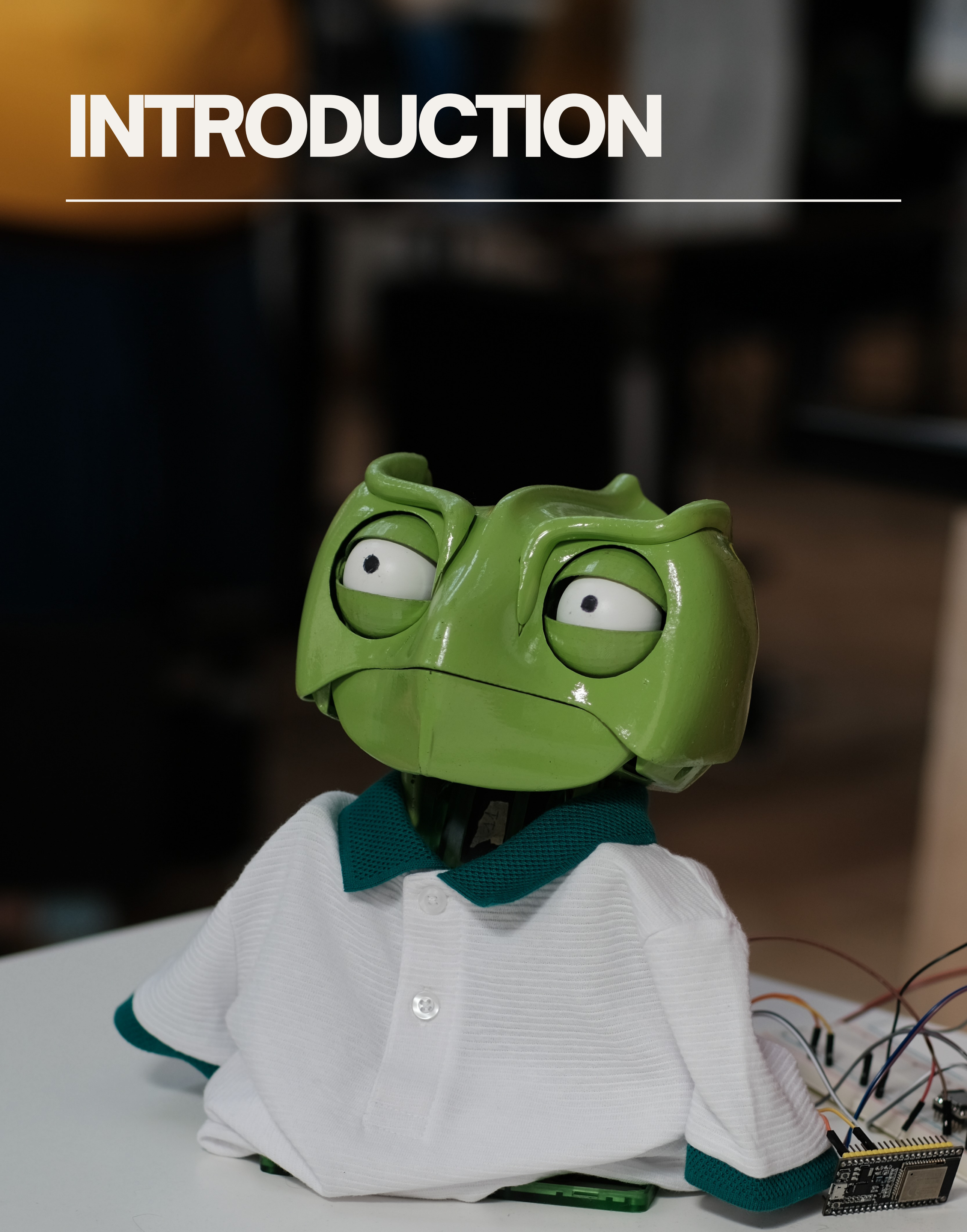
Tutors:
Federico Espositi

Group number 3

Students:
DE LUCA VALERIO, school of ENGINEERING
MASSACCI PAOLO, school of DESIGN
MASTROMAURO MATTEO, school of ENGINEERING
WANG SIYI , school of DESIGN

INDEX

INTRODUCTION



INTRODUCTION

The Perfect Strangers chameleon is here!

CamaLele is a robot that represents the character Lele in Perfect Strangers movie.

As the character referred to, CamaLele is a chameleon that hides the truth to defend himself and his friends, he is shy and calm, but when he is under attack he speaks loudly, he can defend himself very well. He fastly adapts to the situation and mimic the other people, and he even cheats!

Put him on a table with the other Perfect Stranger robots and let the magic happen! CamaLele will behave as Lele and will react to the phone message he will receive and express emotions towards the other robots.

Learn everything about CamaLele at
Group03 - Google Drive

3+

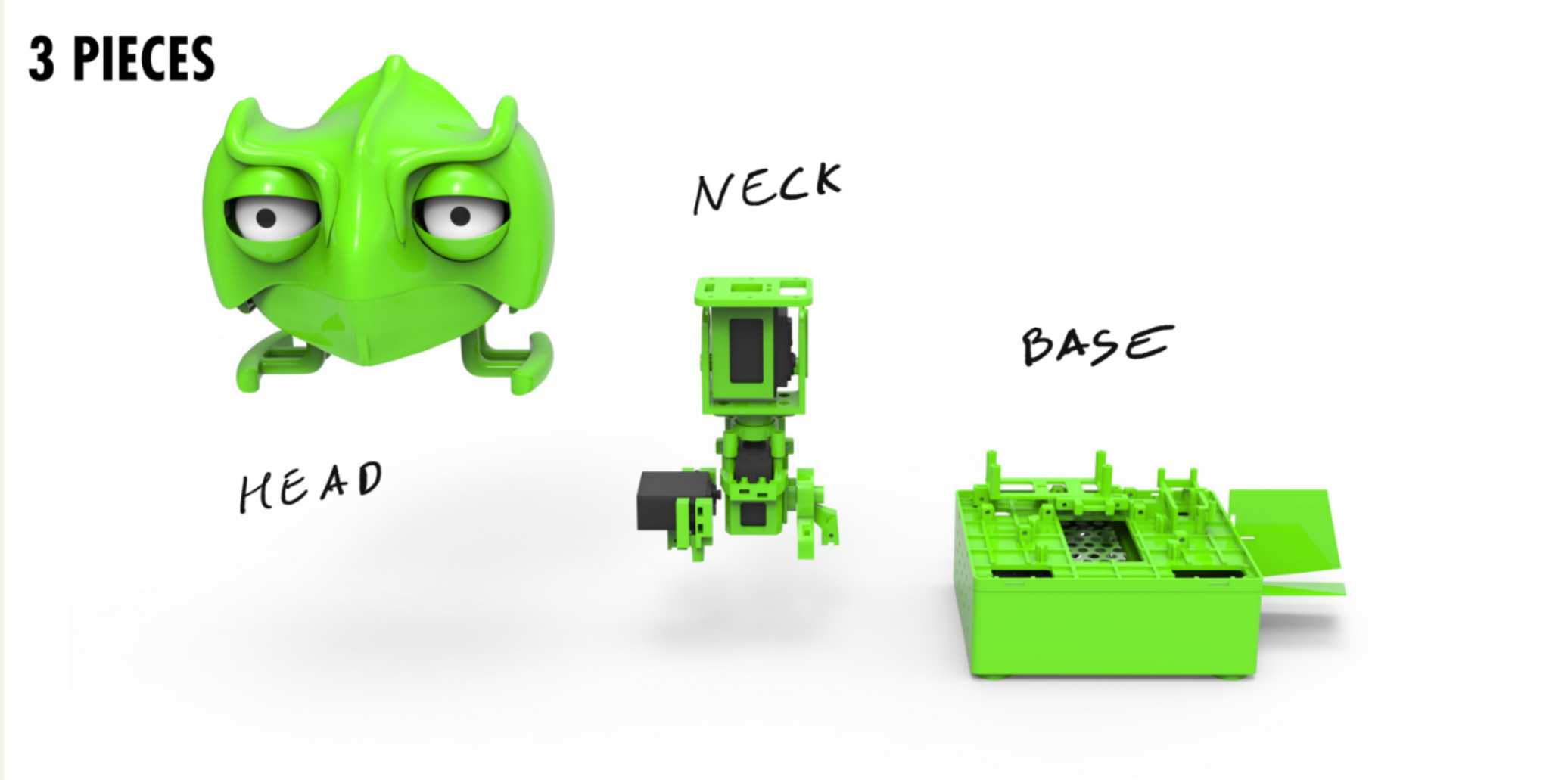
CamaLele is suitable for children from 3 years of age

11 MOTORS

Servo motors control the movements of each body part

HOW TO PLAY

FIRST HARDWARE CONFIGURATION



Take CamaLele out of the box and mount the three parts between them using the screwdriver. Connect the neck to the head with three M3 screws and connect the neck to the base with 4 M3 screws.
Then put the t-shirt on and he will be ready to act!

To power supply CamaLele, insert the plug of the base into a socket and then turn on the button behind the base. CamaLele will turn on and wait for some action to perform

HOW TO PLAY

SECOND SOFTWARE CONFIGURATION

SERVER CONFIGURATION

Step 1: Open on your Laptop "server.py".
Step 2: Modify the fields "HOST" and "PORT" with the corresponding informations about the IP address ot the laptop on which the server runs and the Port used.

```python
import socket
import threading
import time
import sys
import os

HOST = '192.168.206.177'  # server host
PORT = 8090  # server port

debug_mode = False

CHARACTERS = ["All","Rocco","Eva","Lele","Carlotta","Peppe","Bianca","Cosimo"]
IPs = [HOST,"","","","","","",""]
EMOTIONS = {"A": "idle", "B":"happy","C": "angry","D": "shocked","E": "sad","F": "relaxed", "G": "afraid","H":
# create a TCP/IP socket
sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

```

Step 3: Server ready for the execution, run the python script "server.py".

HOW TO PLAY

SECOND SOFTWARE CONFIGURATION

CLIENT CONFIGURATION

Step 1: connect the ESP32 to your Laptop
Step 2: Open on your Laptop "wifi_setup.h" on Arduino IDE and change the fields "ssid" and "password" with SSID (name) of the Wi-Fi network you want to connect to and its password:

```
//WiFi credentials
//CHANGE WIFI CREDENTIAL WITH YOURS!
//const char* ssid = "AndroidAPFD5D";    // The SSID (name) of the Wi-Fi network you want to connect to
//const char* password = "valerioo";     // The password of the Wi-Fi network
const char* ssid = "Triskarone";
const char* password = "triskarone";
```

Step 3: client ready to be executed. Open on Arduino IDE "Esp32_final.ino" in the same folder with all the other header files. Load on the ESP32 "Esp32_final.ino"

HOW TO PLAY

START TO PLAY

In order to work correctly, CamaLele needs to interact with his friends. Connect them all to the server and send them some messages!

If message G5 or G6 is sent, CamaLele will read out loud the content of the message to all the other robots and then he will become angry or shocked, respectively.

If any other message GX is sent (with X from 0 to 9 or from A to E) CamaLele will turn towards the receiver of the message and, when she finishes reading, he will perform an emotion.

HOW TO PLAY

START TO PLAY

If somebody makes fool of his friend Pepp.e, CamaLele will become sad.

If his beloved wife Robotta behaves badly with him, CamaLele will turn out angry.

If Eva bothers him, CamaLele becomes really annoyed

HOW TO PLAY

START TO PLAY

If someone tells a joke or an hilarious phone message arrives, CamaLele will laugh out loud.

If he discovers that the situation is getting suspicious or that there is some lier in the group, CamaLele becomes cautious.

If some strange message arrives or some unpredictable event happens, CamaLele will be terribly shocked.

EMOTIONS

A: idle

B: happy

C: angry

D: shocked

E: sad

H: cautious

J: annoyed

EMOTIONS

IDLE

Head : stays neutral

Eyes : blink

Neck : stays up

Body : rotates left and right

Tail : moves up and down

Sound : no sound

EMOTIONS

HAPPY

Head : laughs with mouth

Eyes : roll up

Neck : goes backwards

Body : moves back and forth

Tail : moves rapidly up and down

Sound : happy laugh

EMOTIONS

ANGRY

Head : rotates while biting with mouth

Eyes : almost close

Neck : goes a bit forward

Body : slightly rotates

Tail : completely up

Sound : tiger roar

EMOTIONS

SHOCKED

Head : stands still with mouth wide open

Eyes : wide open

Neck : goes a bit backwards

Body : stands still

Tail : up and then completely down

Sound : ghost sound

EMOTIONS

SAD

Head : looks down

Eyes : look down with superior eyelids down

Neck : goes forward

Body : rotates right

Tail : completely down

Sound : crying puppy

EMOTIONS

CAUTIONS

Head : turns towards someone

Eyes : one closed and the other wide open

Neck : goes a bit forward

Body : huge rotation left and right

Tail : stays normal

Sound : "mmh"

EMOTIONS

ANNOYED

Head : looks down from above its target

Eyes : superior eyelids totally down

Neck : stays up

Body: turns right

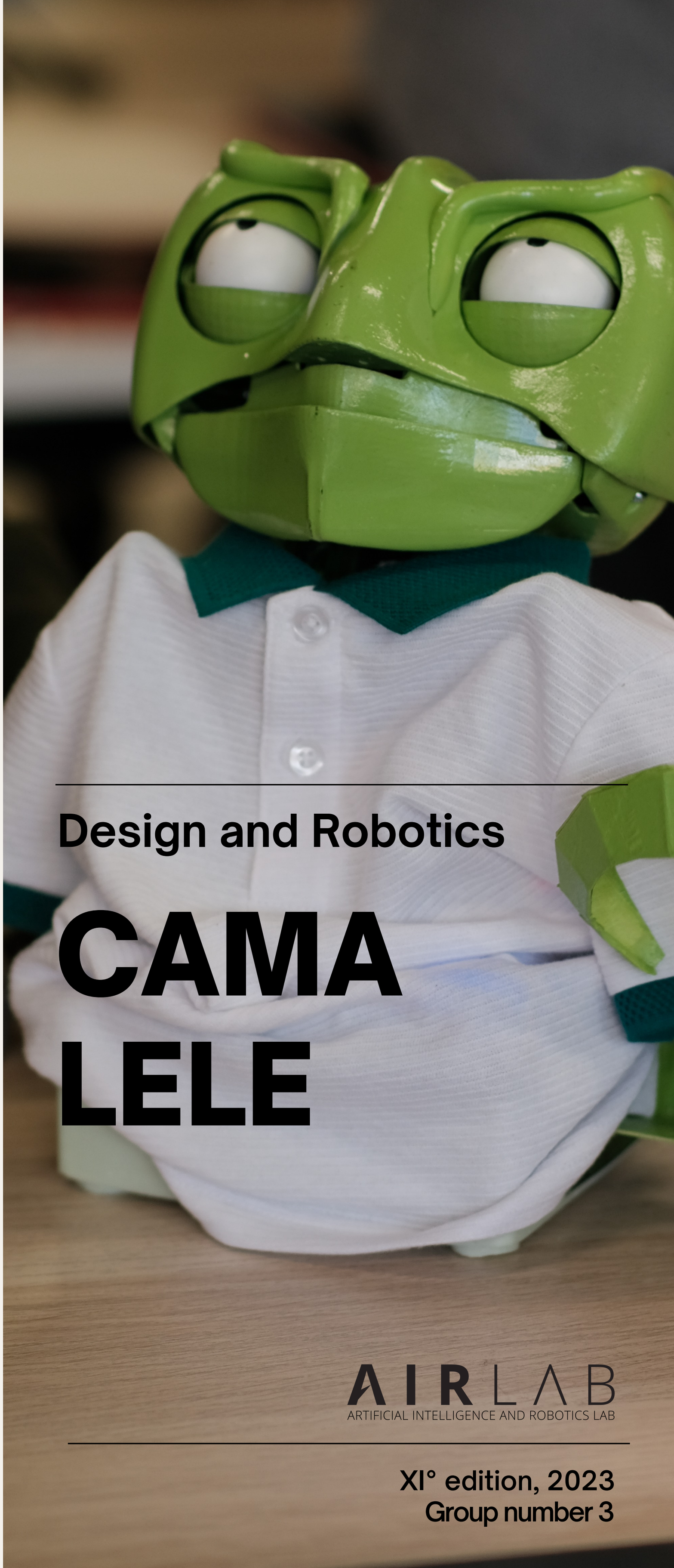
Tail : stays normal

Sound : no sound

POLITECNICO
MILANO 1863

AIRLAB
ARTIFICIAL INTELLIGENCE AND ROBOTICS LAB

MANUAL DE CAMPO



Design and Robotics

CAMA LELE

AIRLAB
ARTIFICIAL INTELLIGENCE AND ROBOTICS LAB

XI° edition, 2023
Group number 3

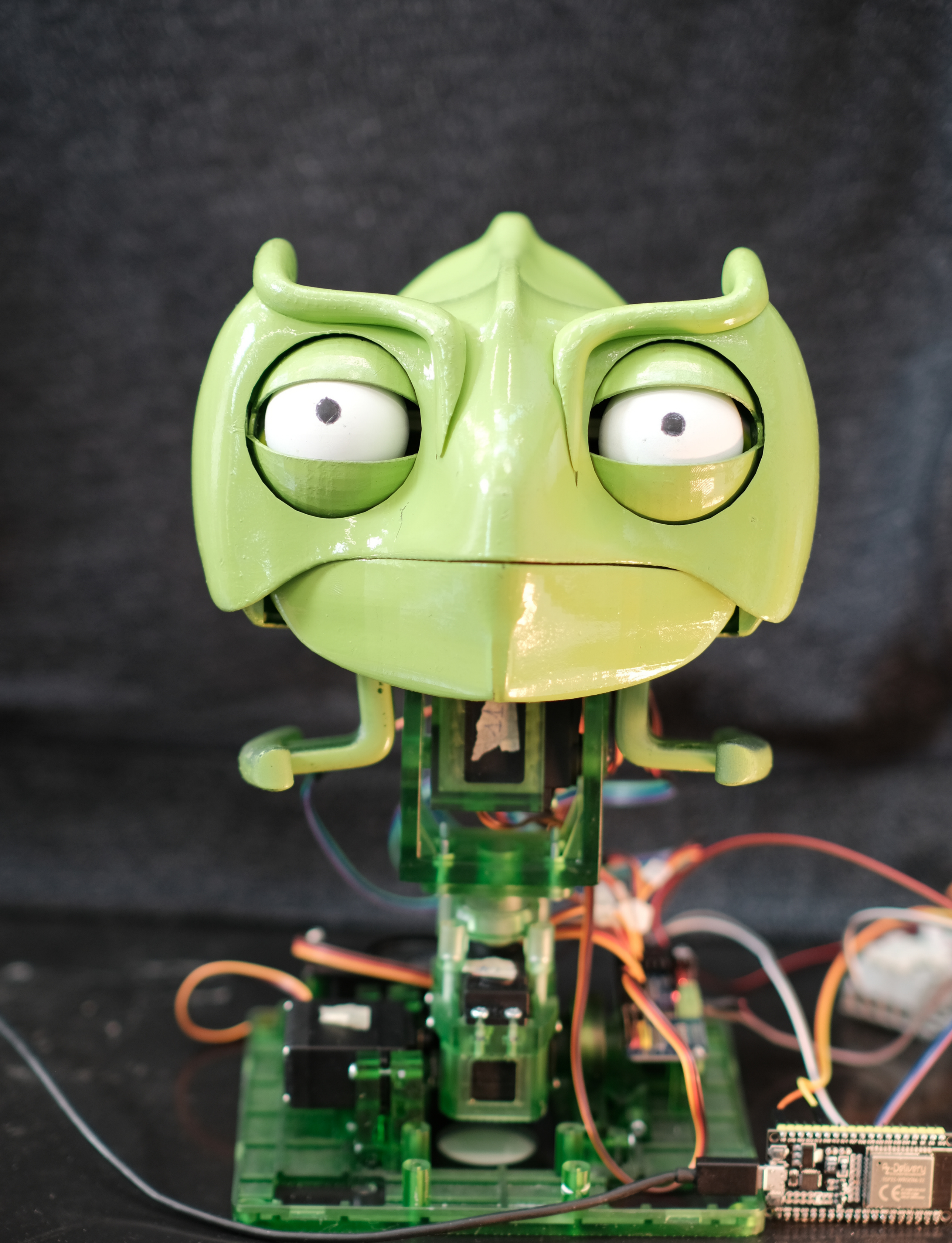
Professors:
Andrea Bonarini, DEIB department, Politecnico di Milano
Maximiliano Romero , Design department, IUAV, Venice University

Tutors:
Federico Espositi

Group number 3

Students:
DE LUCA VALERIO, school of ENGINEERING
MASSACCI PAOLO, school of DESIGN
MASTROMAURO MATTEO, school of ENGINEERING
WANG SIYI , school of DESIGN

INDEX



INTRODUCTION

The Perfect Strangers chameleon is here!

CamaLele is a robot that represents the character Lele in Perfect Strangers movie.

As the character referred to, CamaLele is a chameleon that hides the truth to defend himself and his friends, he is shy and calm, but when he is under attack he speaks loudly, he can defend himself very well. He fastly adapts to the situation and mimic the other people, and he even cheats!

Put him on a table with the other Perfect Stranger robots and let the magic happen! CamaLele will behave as Lele and will react to the phone message he will receive and express emotions towards the other robots.

Learn everything about CamaLele at
Group03 - Google Drive

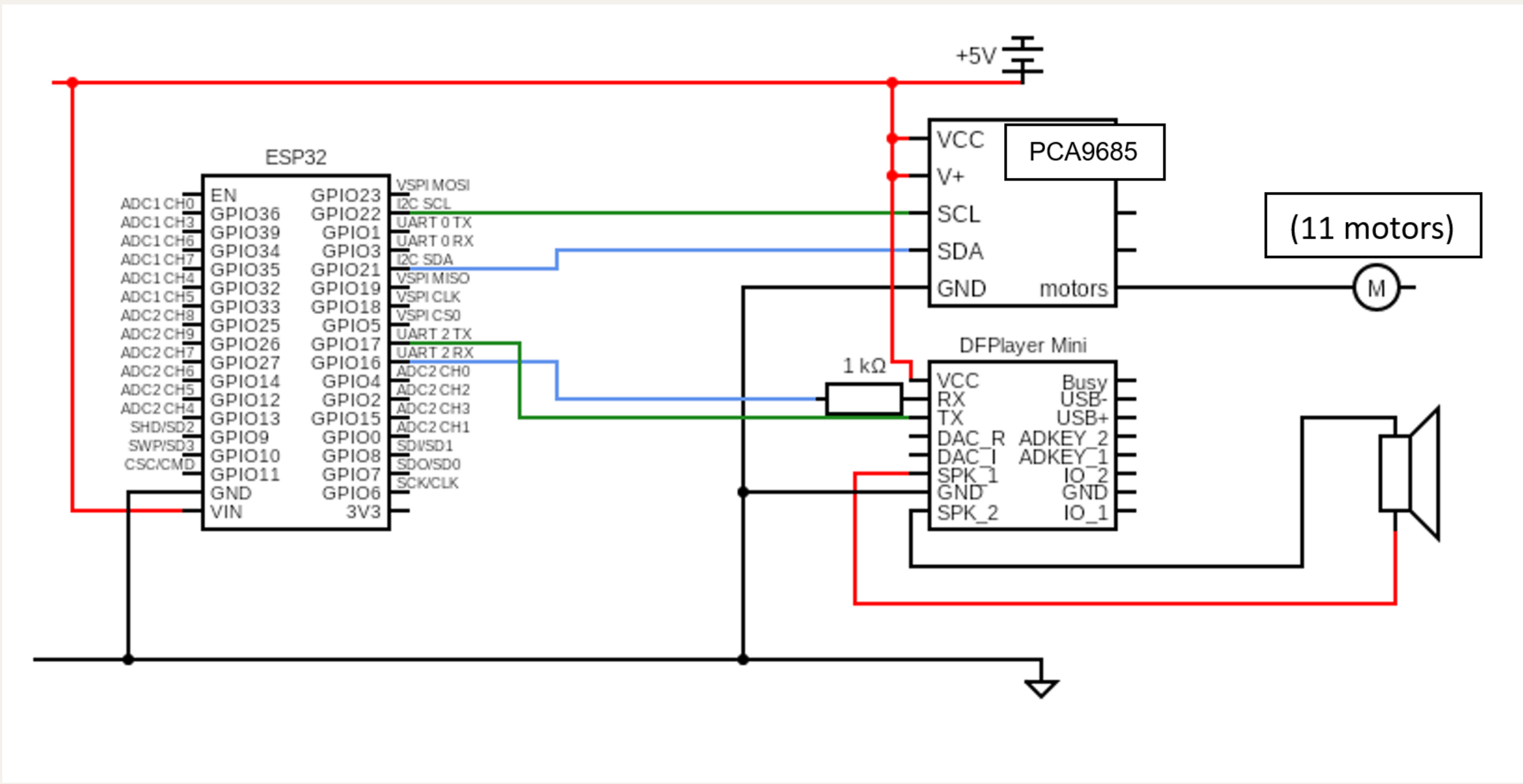
3+

CamaLele is suitable for children from 3 years of age

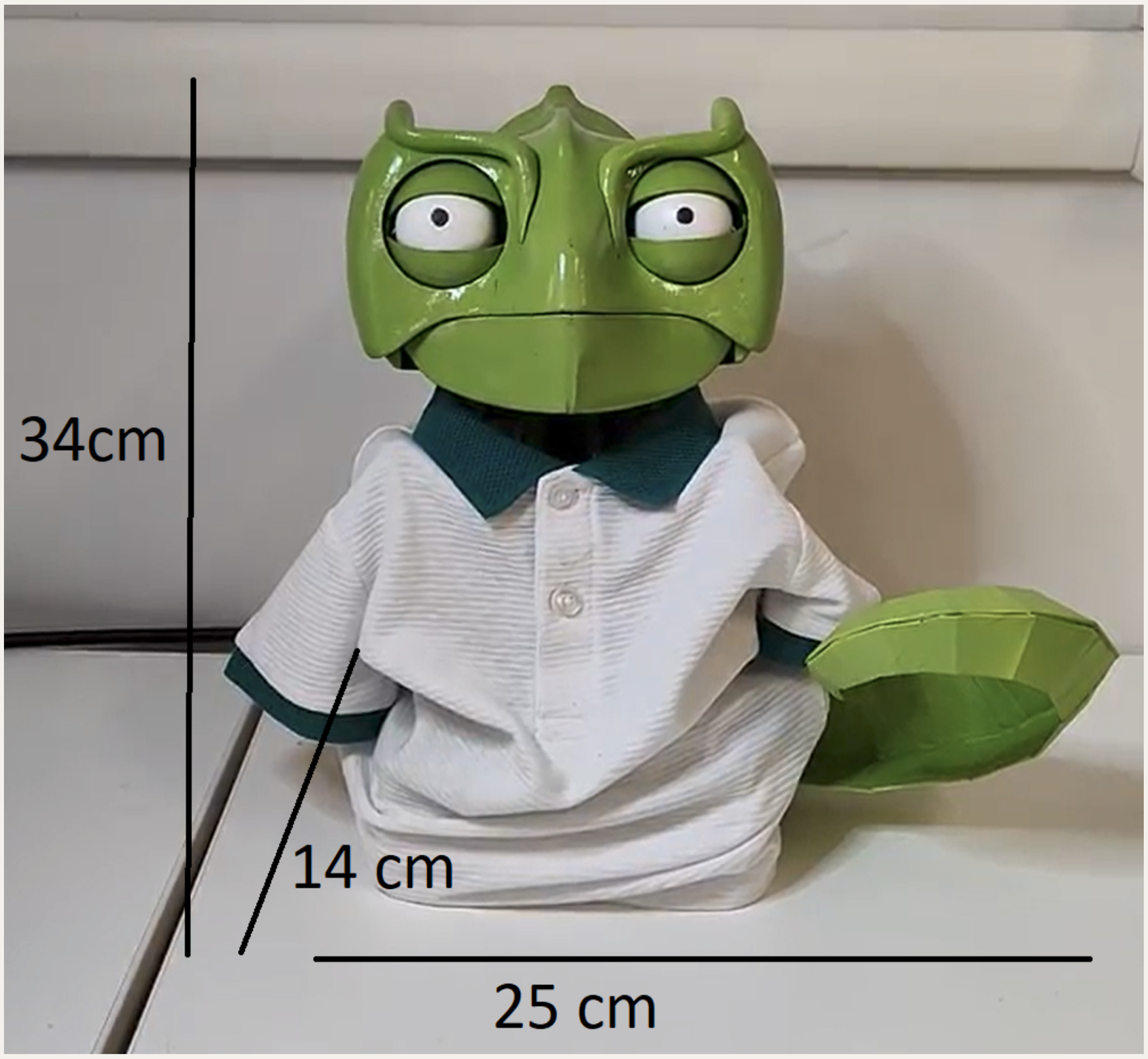
11 MOTORS

Servo motors control the movements of each body part

WIRECONFIGURATION SCHEME

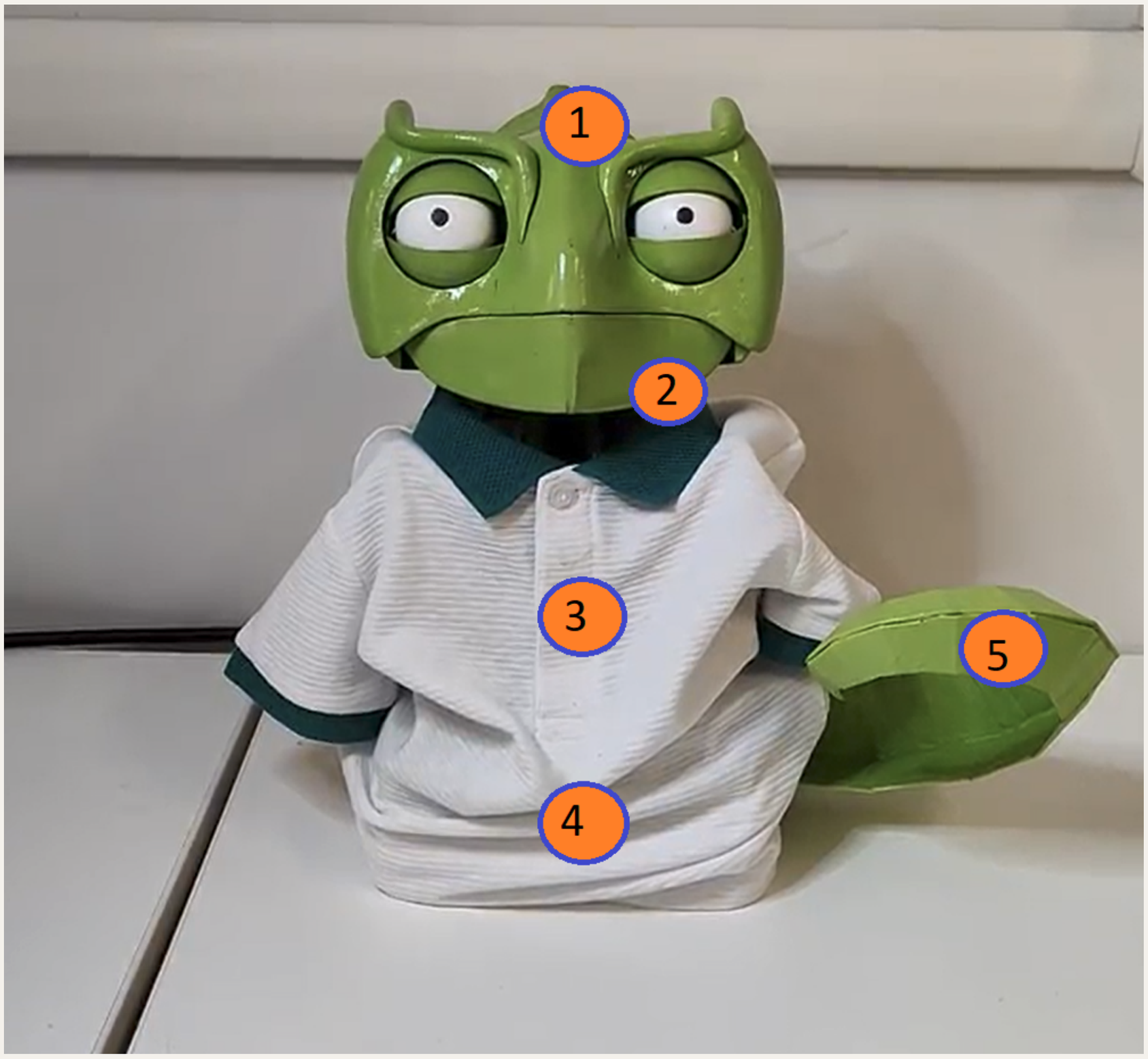


GENERAL DIMENSIONS



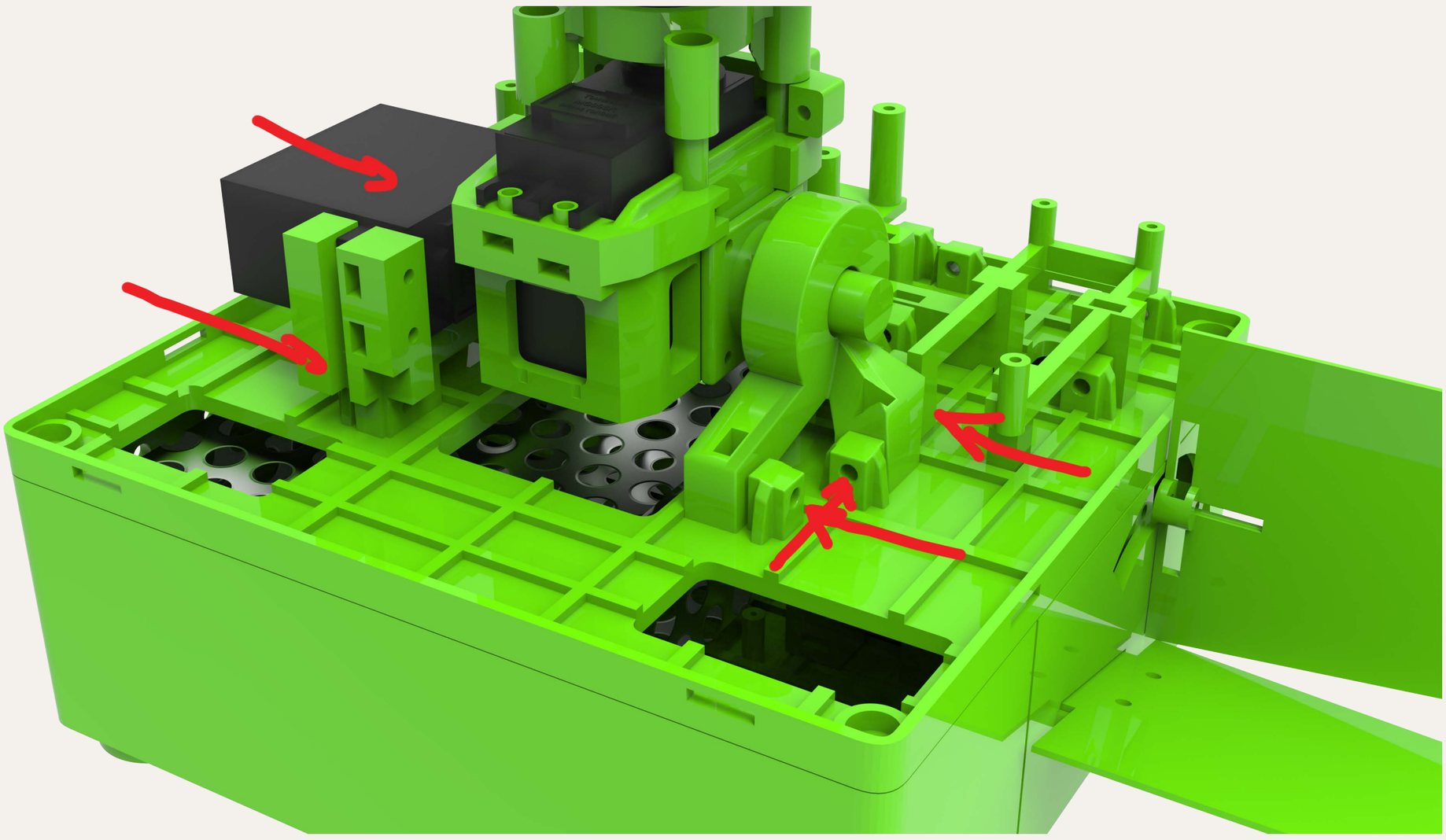
Dimensions of CamaLele: 25 x 34 x 14 cm

ARCHITECTURE

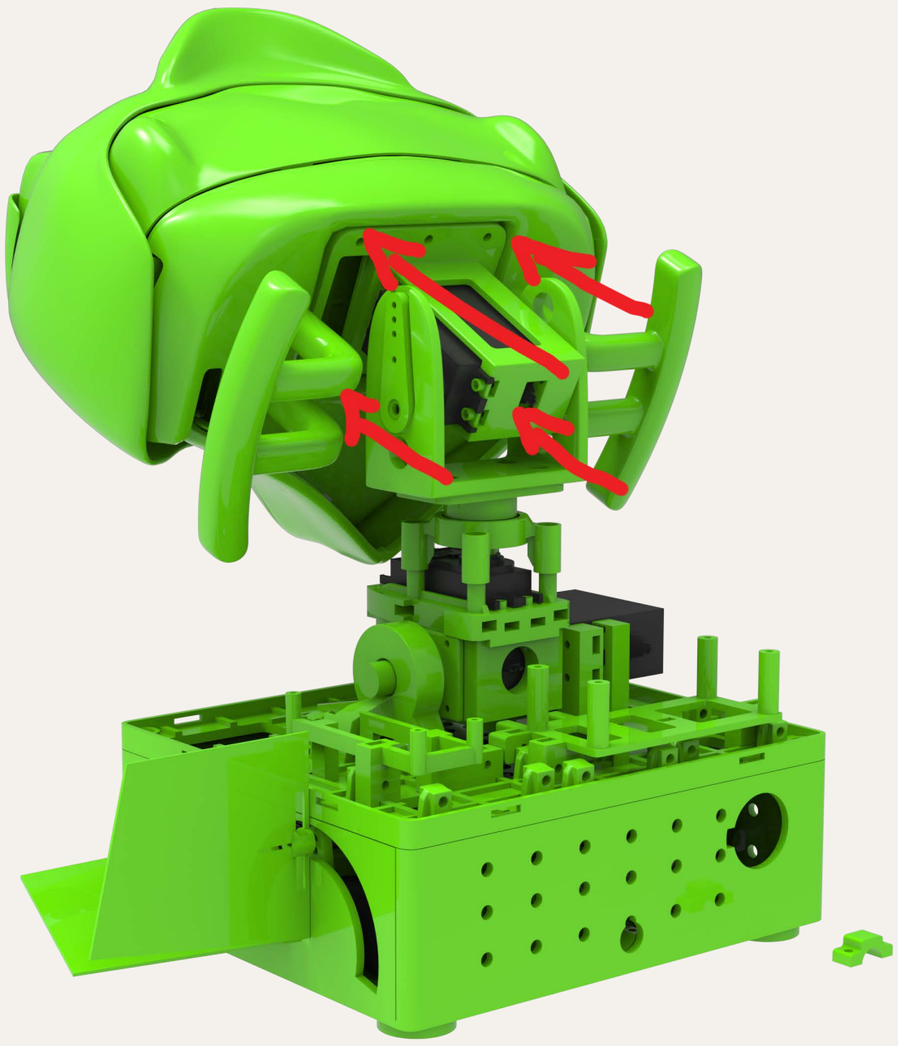


The robot is made of:
1) head
2) mount
3) chest and neck
4) base
5) tail

MAINTENANCE

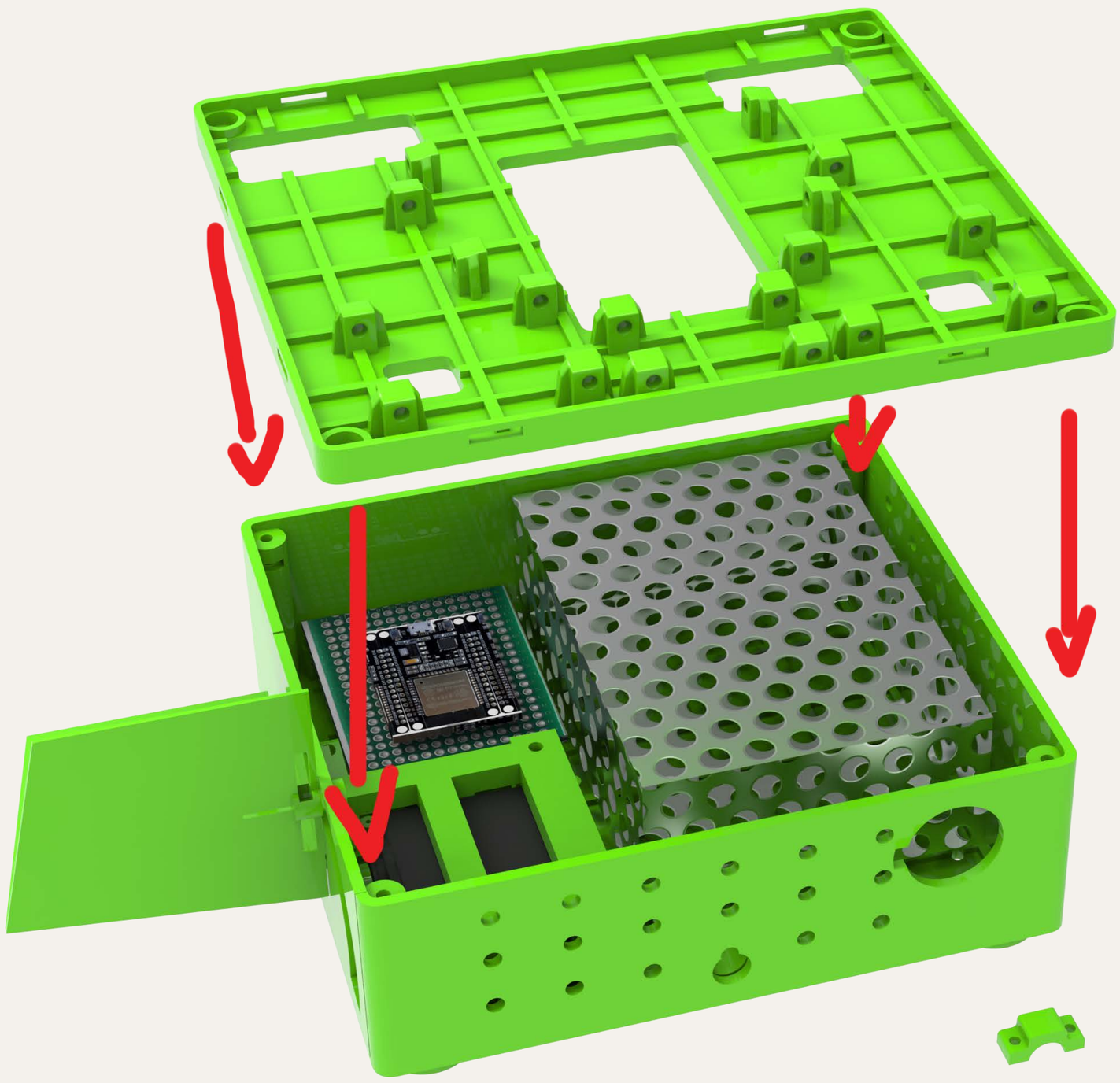


To remove the neck from the base unscrew the 6 m2.5 screws indicated by the red arrows.



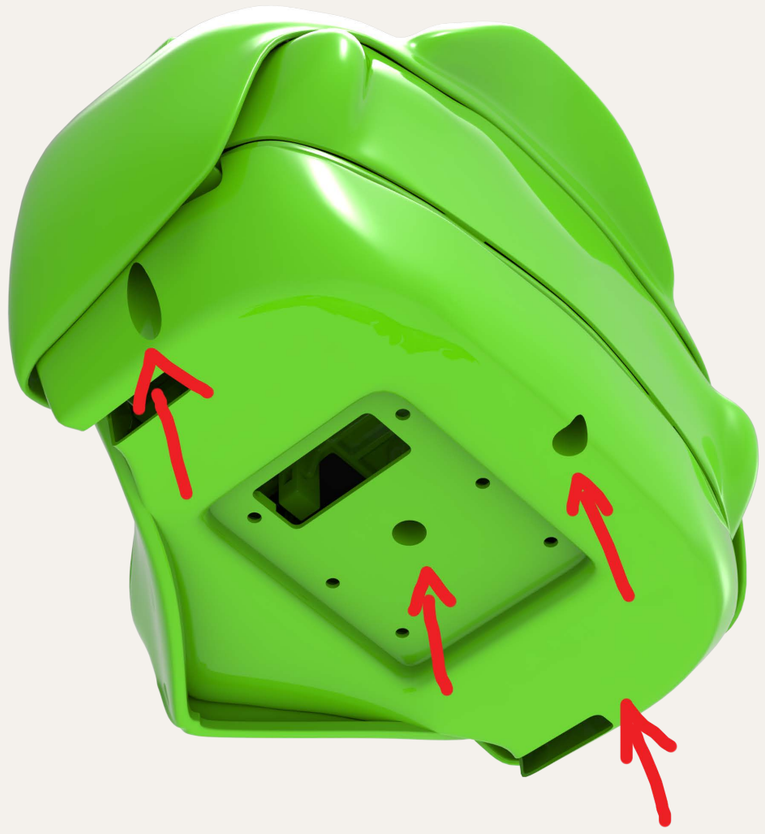
To remove the head from the neck, tilt the top part and remove the 4 m3 screws connecting it, be careful to hold the head to prevent it from falling.

MAINTENANCE



The boards and power supply inside the base is accessible by removing the 4 m3 screws on the lid.

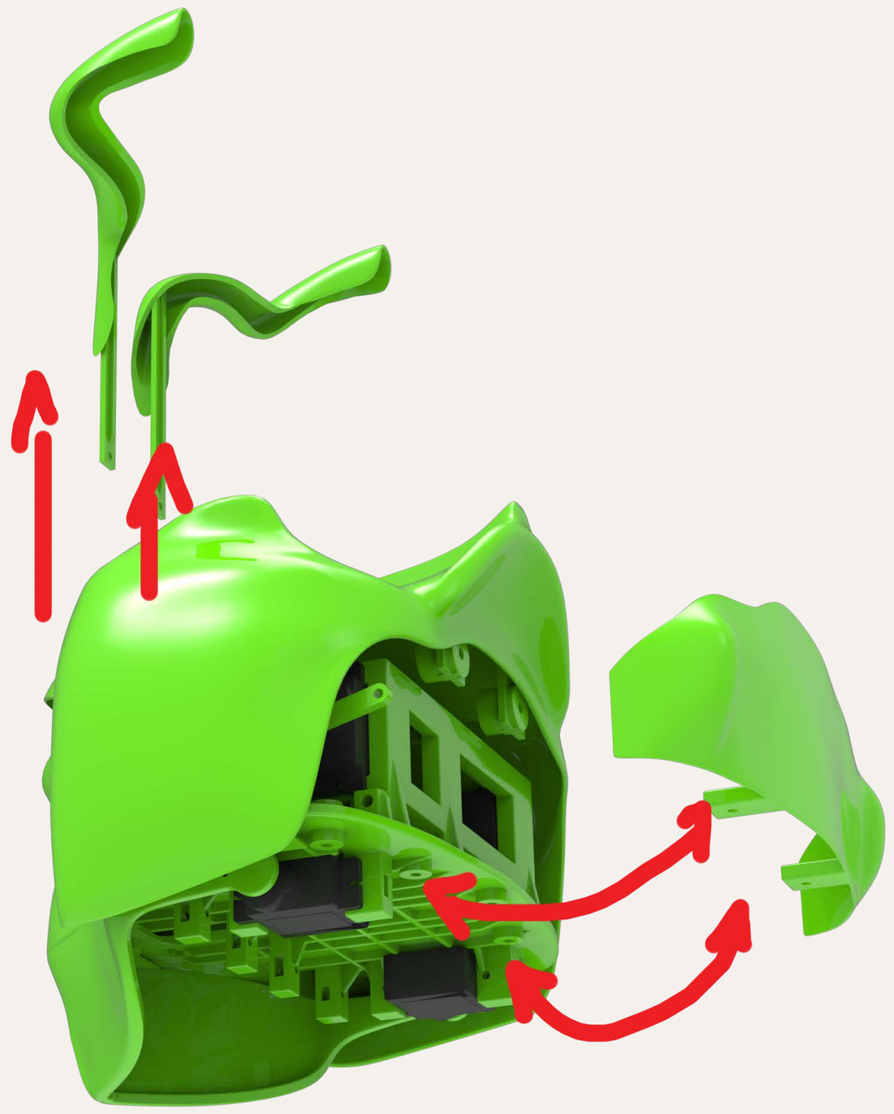
MAINTENANCE

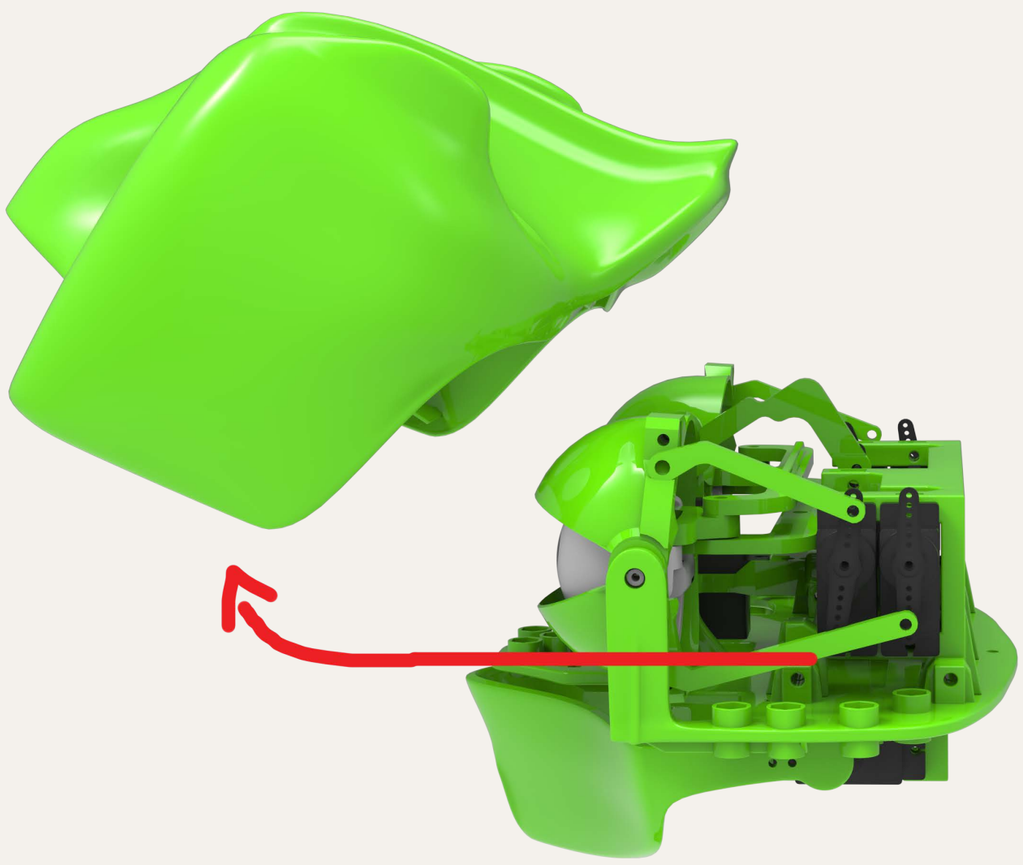


To disassemble the head, start from the bottom and remove the 4 m3 screws that connect its lower cover to the plate.

Later remove the two m3 screws for the back cover and pull out the eyebrows.

Finally remove the last 4 m3 screws under it and slide the face forward, when it stops sliding tilt it upwards and pull it out.





POLITECNICO
MILANO 1863

AIRLAB
ARTIFICIAL INTELLIGENCE AND ROBOTICS LAB

Minutes of the Meetings

Meeting 10/03/2023

Time: 15:00 - 18:00
Venue: Bovisa
Attendees: All

- During the whole week each one had thought about a possible animal to associate with Lele, then during the meeting each one explained his idea and then we voted to select the best association. At the end the chameleon was selected.
- things done:
    - animal selection (each one explained his idea)
    - discussion about the possible features shared between the animal and the character
    - moodboard of the character

Meeting 24/03/2023

Time: 14:00 - 17:00
Venue: Bovisa
Attendees: All

- We continued the prototype we had started in class. A further development of the kinematics of the robot.
- things done:
    - Decision about which movements are important to express emotions
    - The shape of the robot
    - Further development of the shape of the robot in order to understand which parts make moves
    - Online search about possible kinematics of the robot
    - Bill of material
- split the responsibilities among us, and planning the tasks for the next meetings

Meeting 31/03/2023

Time: 9.00 - 16-00
Venue: Bovisa
Attendees: Paolo and Siyi

- First meeting about the design. We started to reshape the eyes and eyelids, enlarge them and adapt them to the base of the head.

Meeting 06/04/2023

Time: 17:00 - 20:00
Venue: Bovisa
Attendees: Matteo and Valerio

- First meeting about the code. First, we decided to use an oriented approach to write the code, then we started to deal with the motor class, how to make motors move without using for cycles and delay() functions.

Meeting 14/04/2023

Time: 15:00 - 19:00
Venue: Bovisa
Attendees: All

- Valerio continued to implement the servo motor class, adding new functions as get() set() functions. Define the function to make smooth motions with the servo motors.
- Matteo designed a first sketch for the electric circuit, in particular only for the ESP32 board.
- Paolo and Siyi assembled the printed eyes and eyelids
- Valerio and Matteo started to test the code on the first prototype of the eyes (with bad news because there was too much friction and so it wasn't possible to assess the correctness of the code).

Meeting 28/04/2023

Time: 14:00 - 17:00
Venue: Bovisa
Attendees: All

- Paolo started to model the neck and the chest.
- Matteo and Valerio continued to test the eyes trying to unwrap the external surface of the eyes and the internal one of the eyelids.

Meeting 30/04/2023

Time: 14:00 - 17:00
Venue: Bovisa
Attendees: Paolo

- Paolo finished to shape the chest and the neck, so he printed all the components in the following days.

Meeting 05/05/2020

Time: 13:00 - 19:00
Venue: Bovisa
Attendees: All

- Paolo assembled the remaining part of the printed chest and neck.
- Matteo and Valerio start tuning the new motors and then implementing the first movements of the entire robot. So far we have used the breadboard for the electric circuit.
- Same problems for the eyes, friction was too much. The motors of the chest didn't work well and the ESP32 sometimes restarted.

Meeting 08/05/2023

Time: 14:00 - 18:00
Venue: Bovisa
Attendees: Matteo and Valerio

- Finished to implement some emotions for the presentation.

Meeting 12/05/2023

Time: 14:00 - 20:00
Venue: Bovisa
Attendees: All

- Paolo modified the eyes and the eyelids, reducing the size for the first one and the thickness for the second ones.
- Paolo designed the tail connection and the base.
- Matteo and Valerio continued to debug the code and started the part related to the wifi connection.
- Siyi did the first prototype of the tail using paper.

Meeting 15/05/2023

Time: 10:00 - 20:00
Venue: AirLab
Attendees:

- Paolo substituted the eyes and eyelids and connected the tail to its motor.
- Valerio continued to debug the code related to the motor handling, then first connection of the Mp3Player and first its usage.
- Matteo finished soldering the ESP32 board and starting the Mp3Player one.

Meeting 16/05/2023

Time: 10:00 - 18:00
Venue: AirLab
Attendees: All

- Paolo reprinted in the past days, a new base different from the previous one to make room for the tail's motor. So he assembled the motor, the ESP32 board and the power supply to the base.
- Valerio finished implementing the functions related to the Mp3Player and debugging the rest of the code.
- Matteo remake the board of the Mp3Player in order to orient correctly the components on it taking in account the robot's component and constraint around the board

Meeting 19/05/2023

Time: 10:00 - 19:00
Venue: AirLab

Attendees: All

- Paolo and Matteo disassembled the entire robot, then painted it.(painting it many times)
- Valerio found on internet the sounds and use the text to speech tool to record the sentences related to our character
- Matteo connected the motors to the PCA9685, then assembled all the boards to the robot.

Meeting 20/05/2023

Time: 11:00 - 18:00
Venue: Home
Attendees: All

- Paolo and Matteo continued painting the robot.
- Valerio continued debugging the code.
- Matteo covered the tail with the paper and glue, tape and then he painted it.

Meeting 21/05/2023

Time: 11:00 - 21:00
Venue: Home
Attendees: All

- Paolo and Matteo assembled the robot
- Matteo assembled the tail to the base
- Valerio started to program the emotions

Meeting 22/05/2023

Time: 11:00 - 17:00
Venue: AriLab
Attendees: All

- Valerio programmed the emotions, the others gave advices about how to make the robot move

Bibliography

Eyes:

- Simplified 3D Printed Animatronic Dual Eye Mechanism : 4 Steps (with Pictures) - Instructables

Chest

- (PDF) EMYS-emotive head of a social robot (researchgate.net)
- Heavy Duty Pan and Tilt - ItsQv

Neck:

- Pivoting Servo Joint Module *Tiny_CNC_Collection by MechEngineerMike - Thingiverse

Tail:

- Curling Spiral Kinetic Sculpture : 6 Steps (with Pictures) - Instructables

Servo motors control:

- PCA9685 Controllo I2C a 16 canali PWM per LED e Servomotori - Makerslab Fablab

Circuit diagram:

- Circuit Diagram - A Circuit Diagram Maker (circuit-diagram.org)

State diagram:

- Creately

Text-to-speech generator:

- FakeYou. Deep Fake Text to Speech.

Mp3 converter

- Convertio — Convertitore file

ENROLLMENT FORM

DESIGN AND ROBOTICS V11 2022-23

CLOSING FORM

DESIGN AND ROBOTICS V11 2022-23

Minutes of the Meetings

Meeting 10/03/2023

Time: 15:00 - 18:00
Venue: Bovisa
Attendees: All

- During the whole week each one had thought about a possible animal to associate with Lele, then during the meeting each one explained his idea and then we voted to select the best association. At the end the chameleon was selected.
- things done:
    - animal selection (each one explained his idea)
    - discussion about the possible features shared between the animal and the character
    - moodboard of the character

Meeting 24/03/2023

Time: 14:00 - 17:00
Venue: Bovisa
Attendees: All

- We continued the prototype we had started in class. A further development of the kinematics of the robot.
- things done:
    - Decision about which movements are important to express emotions
    - The shape of the robot
    - Further development of the shape of the robot in order to understand which parts make moves
    - Online search about possible kinematics of the robot
    - Bill of material
- split the responsibilities among us, and planning the tasks for the next meetings

Meeting 31/03/2023

Time: 9.00 - 16-00
Venue: Bovisa
Attendees: Paolo and Siyi

- First meeting about the design. We started to reshape the eyes and eyelids, enlarge them and adapt them to the base of the head.

Meeting 06/04/2023

Time: 17:00 - 20:00
Venue: Bovisa
Attendees: Matteo and Valerio

- First meeting about the code. First, we decided to use an oriented approach to write the code, then we started to deal with the motor class, how to make motors move without using for cycles and delay() functions.

Meeting 14/04/2023

Time: 15:00 - 19:00
Venue: Bovisa
Attendees: All

- Valerio continued to implement the servo motor class, adding new functions as get() set() functions. Define the function to make smooth motions with the servo motors.
- Matteo designed a first sketch for the electric circuit, in particular only for the ESP32 board.
- Paolo and Siyi assembled the printed eyes and eyelids
- Valerio and Matteo started to test the code on the first prototype of the eyes (with bad news because there was too much friction and so it wasn't possible to assess the correctness of the code).

Meeting 28/04/2023

Time: 14:00 - 17:00
Venue: Bovisa
Attendees: All

- Paolo started to model the neck and the chest.
- Matteo and Valerio continued to test the eyes trying to unwrap the external surface of the eyes and the internal one of the eyelids.

Meeting 30/04/2023

Time: 14:00 - 17:00
Venue: Bovisa
Attendees: Paolo

- Paolo finished to shape the chest and the neck, so he printed all the components in the following days.

Meeting 05/05/2020

Time: 13:00 - 19:00
Venue: Bovisa
Attendees: All

- Paolo assembled the remaining part of the printed chest and neck.
- Matteo and Valerio start tuning the new motors and then implementing the first movements of the entire robot. So far we have used the breadboard for the electric circuit.
- Same problems for the eyes, friction was too much. The motors of the chest didn't work well and the ESP32 sometimes restarted.

Meeting 08/05/2023

Time: 14:00 - 18:00
Venue: Bovisa
Attendees: Matteo and Valerio

- Finished to implement some emotions for the presentation.

Meeting 12/05/2023

Time: 14:00 - 20:00
Venue: Bovisa
Attendees: All

- Paolo modified the eyes and the eyelids, reducing the size for the first one and the thickness for the second ones.
- Paolo designed the tail connection and the base.
- Matteo and Valerio continued to debug the code and started the part related to the wifi connection.
- Siyi did the first prototype of the tail using paper.

Meeting 15/05/2023

Time: 10:00 - 20:00
Venue: AirLab
Attendees:

- Paolo substituted the eyes and eyelids and connected the tail to its motor.
- Valerio continued to debug the code related to the motor handling, then first connection of the Mp3Player and first its usage.
- Matteo finished soldering the ESP32 board and starting the Mp3Player one.

Meeting 16/05/2023

Time: 10:00 - 18:00
Venue: AirLab
Attendees: All

- Paolo reprinted in the past days, a new base different from the previous one to make room for the tail's motor. So he assembled the motor, the ESP32 board and the power supply to the base.
- Valerio finished implementing the functions related to the Mp3Player and debugging the rest of the code.
- Matteo remake the board of the Mp3Player in order to orient correctly the components on it taking in account the robot's component and constraint around the board

Meeting 19/05/2023

Time: 10:00 - 19:00
Venue: AirLab

Attendees: All

- Paolo and Matteo disassembled the entire robot, then painted it.(painting it many times)
- Valerio found on internet the sounds and use the text to speech tool to record the sentences related to our character
- Matteo connected the motors to the PCA9685, then assembled all the boards to the robot.

Meeting 20/05/2023

Time: 11:00 - 18:00
Venue: Home
Attendees: All

- Paolo and Matteo continued painting the robot.
- Valerio continued debugging the code.
- Matteo covered the tail with the paper and glue, tape and then he painted it.

Meeting 21/05/2023

Time: 11:00 - 21:00
Venue: Home
Attendees: All

- Paolo and Matteo assembled the robot
- Matteo assembled the tail to the base
- Valerio started to program the emotions

Meeting 22/05/2023

Time: 11:00 - 17:00
Venue: AriLab
Attendees: All

- Valerio programmed the emotions, the others gave advices about how to make the robot move

Bibliography

Eyes:

- Simplified 3D Printed Animatronic Dual Eye Mechanism : 4 Steps (with Pictures) - Instructables

Chest

- (PDF) EMYS-emotive head of a social robot (researchgate.net)
- Heavy Duty Pan and Tilt - ItsQv

Neck:

- Pivoting Servo Joint Module *Tiny_CNC_Collection by MechEngineerMike - Thingiverse

Tail:

- Curling Spiral Kinetic Sculpture : 6 Steps (with Pictures) - Instructables

Servo motors control:

- PCA9685 Controllo I2C a 16 canali PWM per LED e Servomotori - Makerslab Fablab

Circuit diagram:

- Circuit Diagram - A Circuit Diagram Maker (circuit-diagram.org)

State diagram:

- Creately

Text-to-speech generator:

- FakeYou. Deep Fake Text to Speech.

Mp3 converter

- Convertio — Convertitore file

ENROLLMENT FORM

DESIGN AND ROBOTICS V11 2022-23

CLOSING FORM

DESIGN AND ROBOTICS V11 2022-23