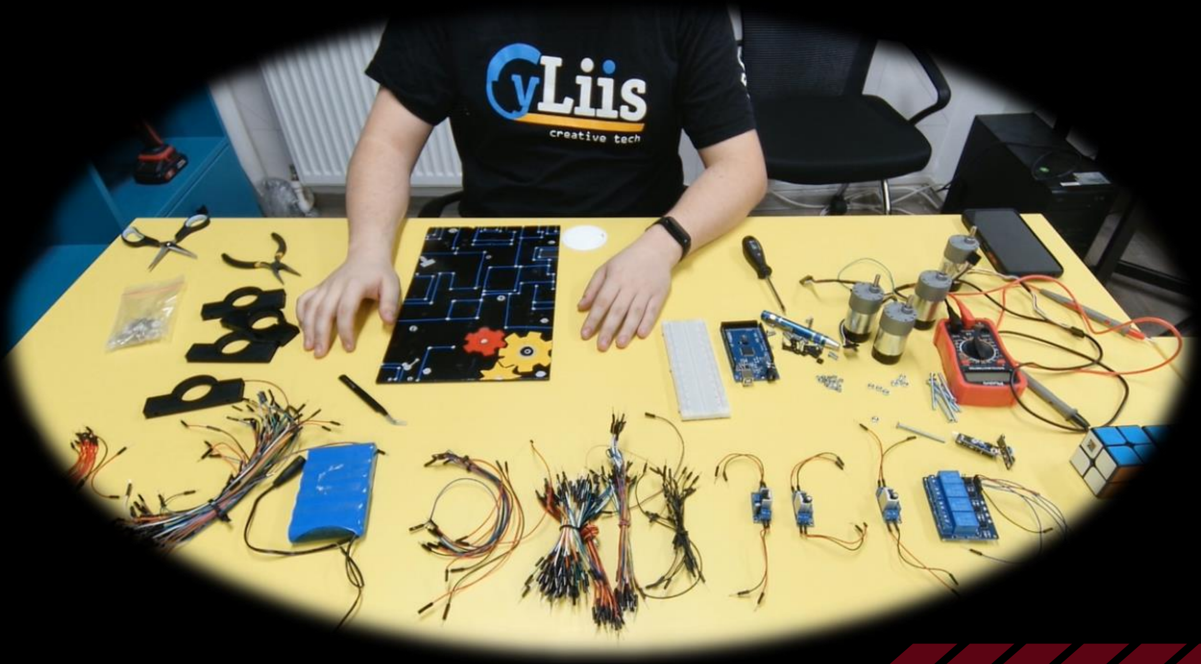
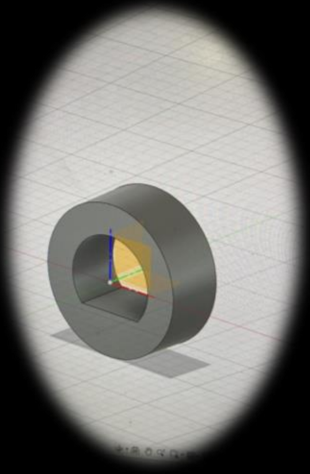
LINE FOLLOWING DEVICE

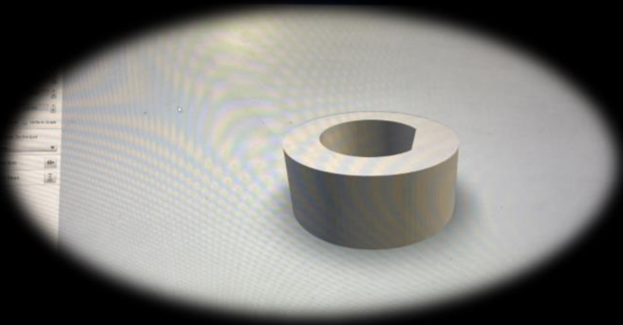
Required components



ENGINES AND THEIR POSITION

For the propulsion of the robot we decided to use four 12 volt motors. They are very powerful, and move the robot very fast. To keep them attached to the robot, we created a motor system in FUSION360, which was 3D printed.



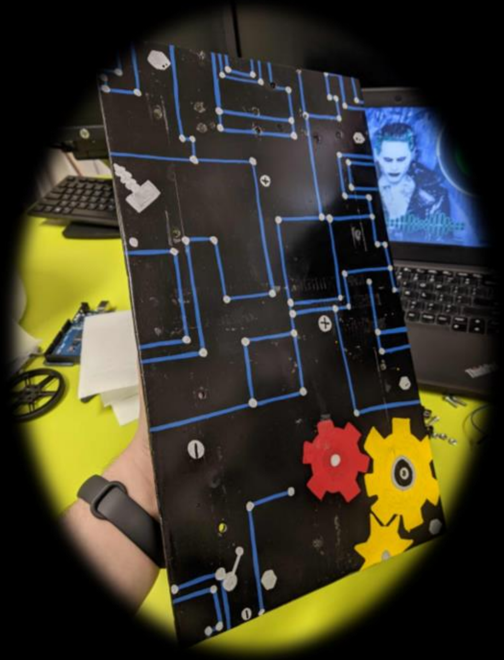


To transmit the power of the motors to the ground, we used some wheels together with some 3D printed adapters that matched our motors. The adapters, as well as the motor mounts, were made in FUSION360.

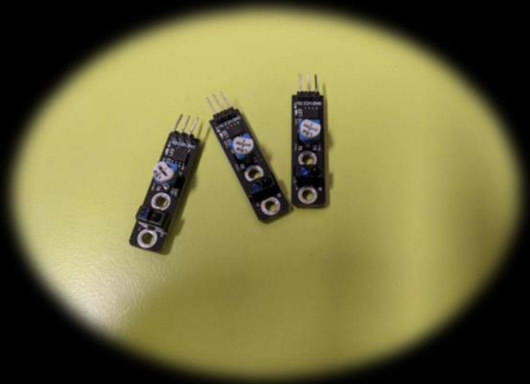
WHEELS AND ADAPTERS

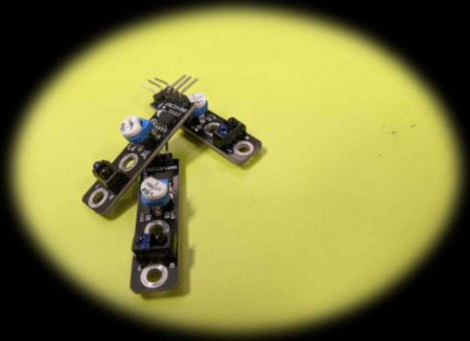
CHASSIS AND ITS DESIGN

We decided to build the robot on a PlexiGlass base, because it is a material that is quite easy to obtain and strong enough to have a solid robot. The initial idea was to leave it transparent, but after the tests there were many scratches and imperfections on the surface, so we decided to decorate it. Thanks to the design team, the robot chassis now looks like this:



We decided to use three sensors to detect which line the robot should follow, placed next to each other on the robot. Thus, the central sensor checks if the robot is still on the line, the one on the right if it can take a curve to the right, and the one on the left if it can take a curve to the left. The code uses the information received from the sensors to determine the robot's actions.

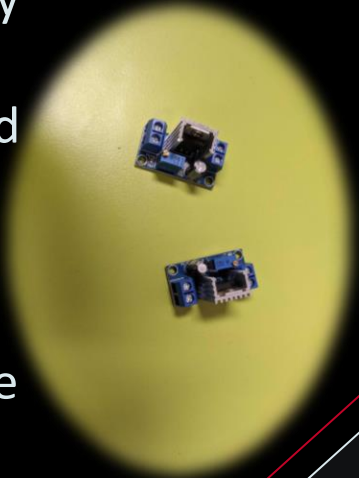




SENSORS

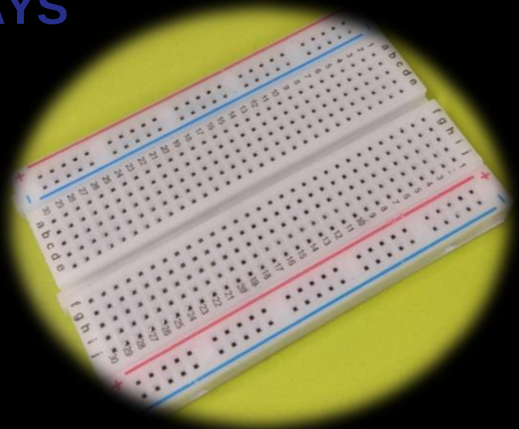
BATTERY AND VOLTAGE REGULATORS

To power the robot, we decided to use a 12V battery. It only powers the Arduino board directly, with the rest of the components taking the necessary current from the 5V pins on the Arduino. The motors are supplied separately from the battery by means of voltage regulators, which reduce the voltage from 12V to 1.5, in order to reduce the speed of the motors.



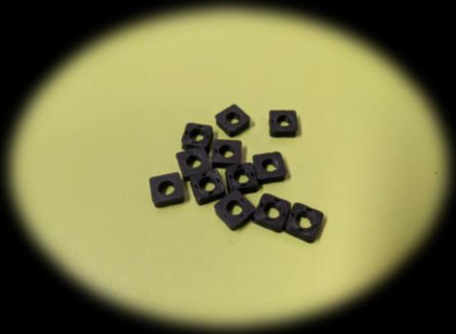
ARDUINO BOARD, BREADBOARD AND THE RELAYS

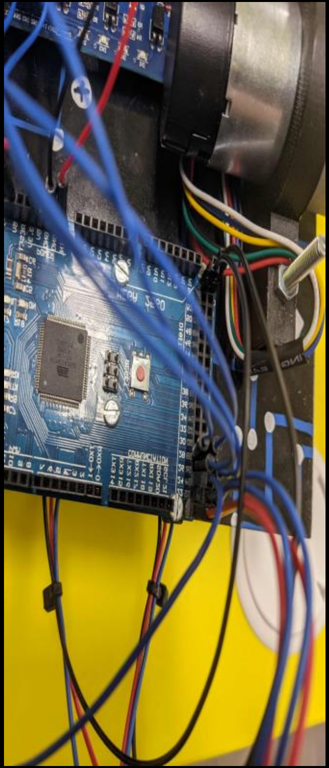
To control the robot, we decided to use an Arduino Mega 2560 board, because it gives us a lot of flexibility. To connect and distribute all signals and power sources, we decided to use a breadboard. To start and stop the engines, we use a board with 4 relays, one for each engine.



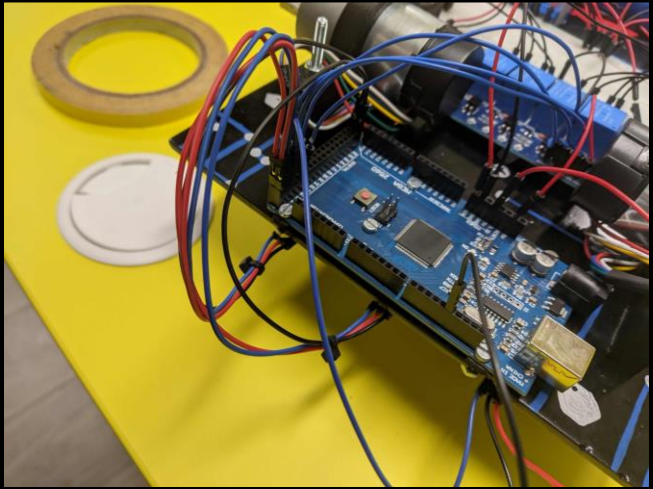
OTHER NECESSARY THINGS

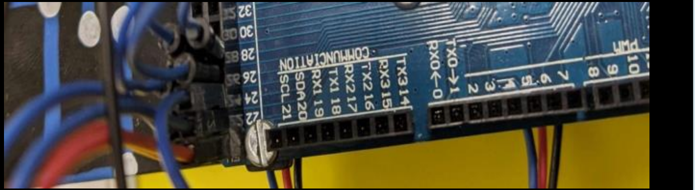
In addition to the parts mentioned above, screws and nuts, 3D-printed spacers are needed to distance the electronic plexiglass boards, cables, double-sided tape and Threadlocker to lock the nuts on the screws.





Below is the Arduino Mega 2560. It was used by us to execute the source code and make the robot come to life. Careful cable management is needed so that everything goes smoothly, as many cables have been used. At the top we declare the pins we will use. We need to make sure that they have the same number written on the Arduino. We can see that all the pins on the Arduino are numbered.





This is the code behind the robot. We have many features that we use such as:

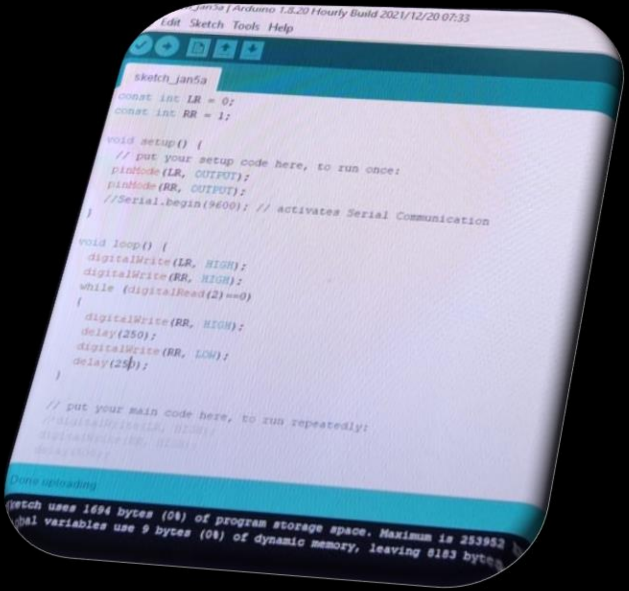
-digitalWrite (to turn on / off the current from a pin)

-delay (to wait a few seconds)

-pinMode (to initialize a pin as output / input)

-while (to repeat a set of instructions in May many times)

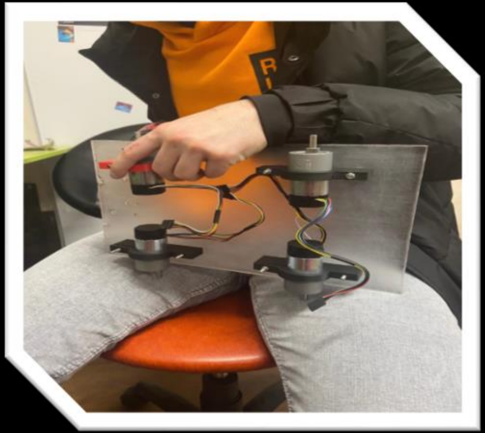
-for (to repeat a set of instructions by a number specific times)



```
const int LR = 0;
const int RR = 1;

void setup() {
  // put your setup code here, to run once:
  pinMode(LR, OUTPUT);
  pinMode(RR, OUTPUT);
  //Serial.begin(9600); // activates Serial Communication
}

void loop() {
  digitalWrite(LR, HIGH);
  digitalWrite(RR, HIGH);
  while (digitalRead(2)==0)
  {
   digitalWrite(RR, HIGH);
   delay(250);
   digitalWrite(RR, LOW);
   delay(250);
  }

  // put your main code here, to run repeatedly:
  //digitalWrite(LR, HIGH);
  digitalWrite(RR, HIGH);
  [illegible]
```

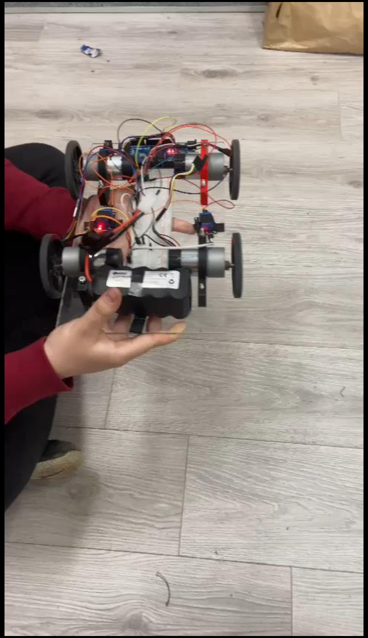
Done uploading

```
etch uses 1694 bytes (0%) of program storage space. Maximum is 253952
bal variables use 9 bytes (0%) of dynamic memory, leaving 8183 byte
```

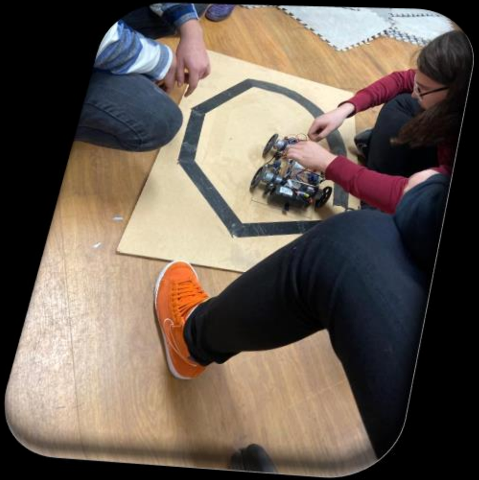


The coding team almost finished writing the robot code per piece, so our assembly team started assembling the robot in the alpha phase. So we started with ecstasy in joining the components on the plexiglass backplate.

Unfortunately, immediately after the first assembly, the robot did not work properly. The wheels on the right side of the robot moved in the opposite direction to those on the left, so that the robot rotated in place.



In these photos we completed each function of the robot in the alpha phase, it works and we made a route template to test the capabilities of the robot and see if the code works in normal parameters, and if the sensors react properly to an unexpected change . Unfortunately we ran into problems, the code being written a bit inefficient, so our robot had lag in a change of direction so the coders team had to rewrite some of it.



Thus, we reached the final assembly. The assembly process lasted one hour, being performed by one person. Thanks to the design tested and redone several times, no difficulties were encountered.