THE ICARUS BOT

Tangible Programming

Tristan Weis
tristan.weis@uni-weimar.de
MTR902210
WS2014/2015
Bauhaus-Universität Weimar

„Let me warn you, Icarus, to take the middle way, in case the moisture weighs down your wings, if you fly too low, or if you go too high, the sun scorches them.“

—Daedalus in Metamorphoses, Book VIII

Publius Ovidius Naso

Introducing Icarus

While learning about the fundamentals of electric circuits in Darsha Hewitt's class We make machines not art photoresistors and motors caught my special attention. I was fascinated by both quantity and complexity of functions these two simple elements offer in combination.

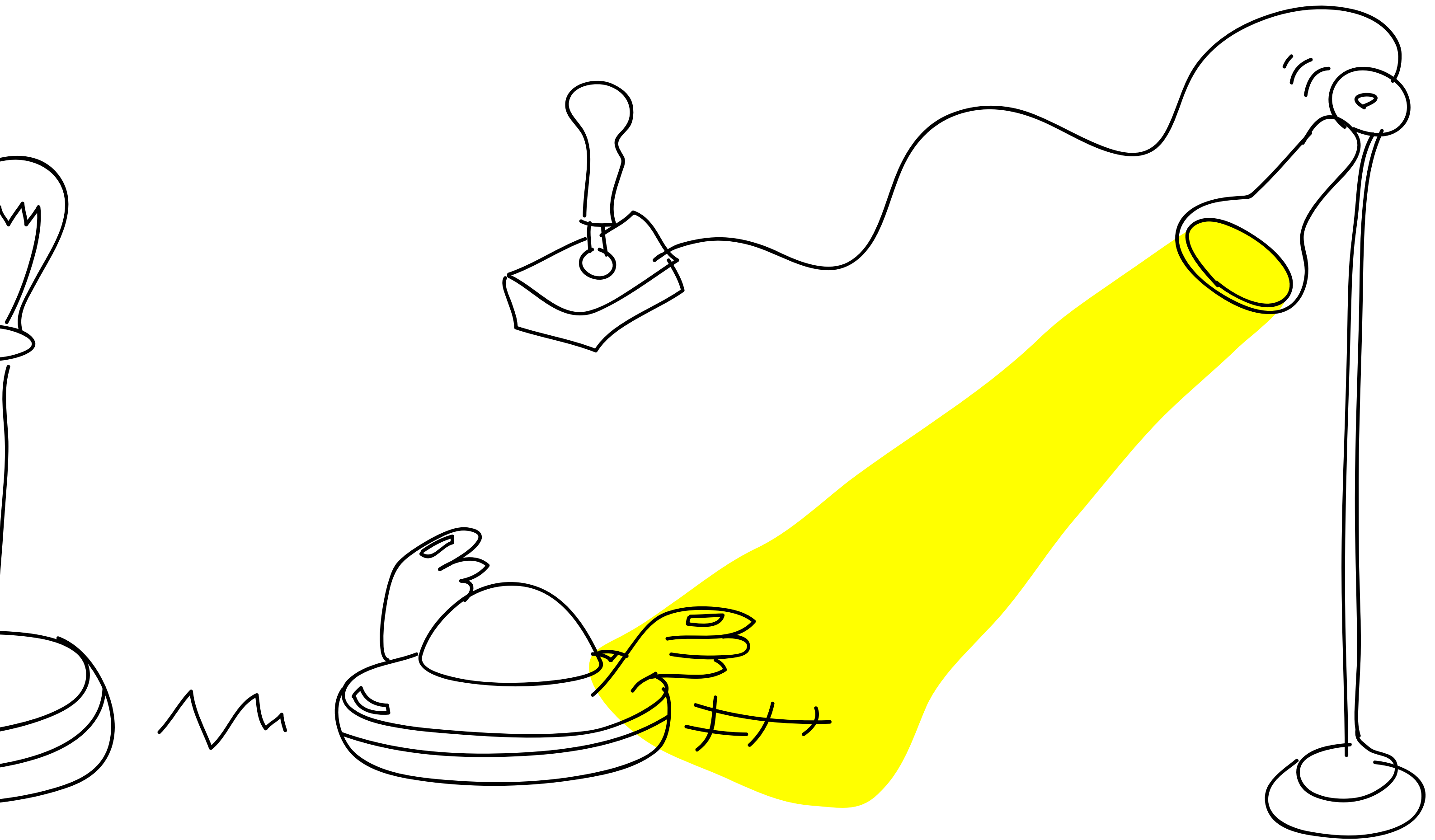
As a human being, moving through space for the biggest part means collecting and analyzing a constant stream of visual data while sending commands to the muscles according to the calculated results.
Armed with basic knowledge of how to create simple electronic devices supported by integrated circuits, I wanted to tackle my idea of creating a machine which follows that principle.

My plan was to reduce both input and output of the machine to a manageable level, which lead to the concept of a little robot moving on a plane, utilising two orthogonal wheeled motor axes.

The first version of the robot utilized four photocells to measure brightness and moved according to those levels – constantly fleeing the light. The circuit I built was purely analogue, which offered me a lot of insight into electronics basics, but naturally suffered from drawbacks. For the next step I decided to rely on an Arduino board to controll the robot's behaviour digitally based on what I learned in Johannes Deichs's class Tanglible programming – An introduction.

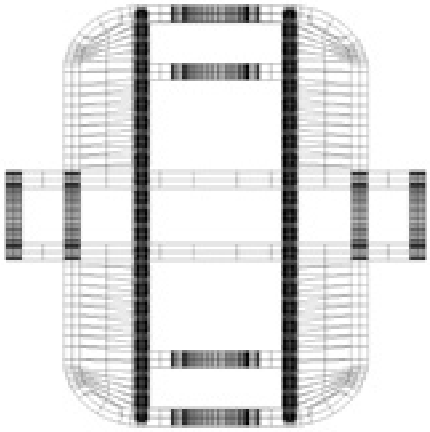
In addition to redesigning the circuit for the advanced arduino controls, I also optimized the hardware design. The previous version was not able to move by itself which led me to further optimize the wheels.

Altogether I was able to finally bring my little creature to a state, where it truly flees the light and – although still bound to power supply – is a lot of fun to play with.

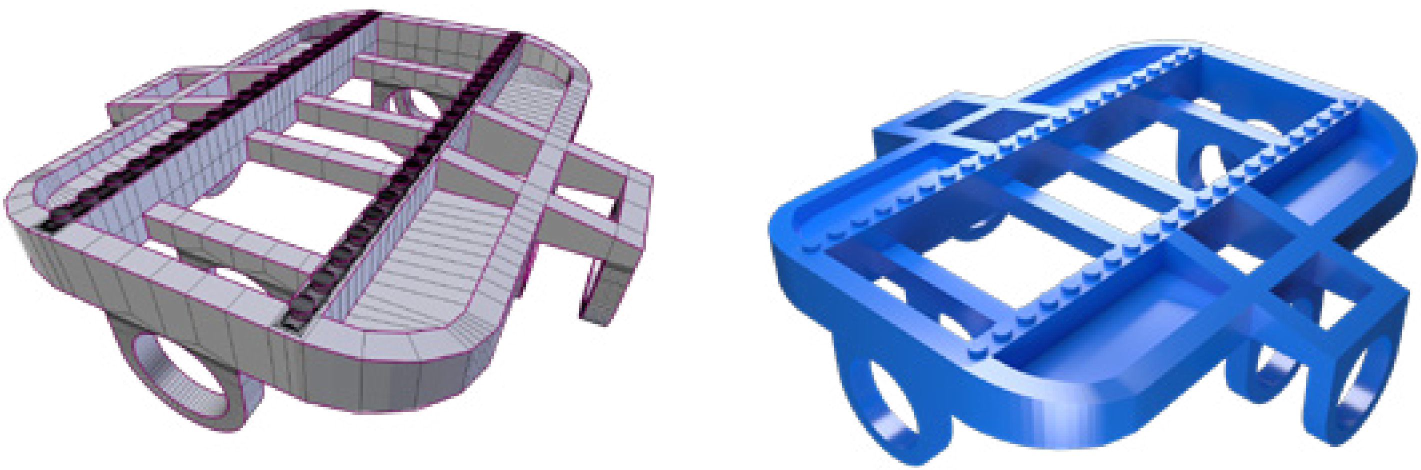


Building the body

In order to have a reliable base to work on, I chose to model the body in a 3D environment and materialize it with the help of a 3D printer. It was my first fairly complex model to be printed and other than getting used to the printing process I designed it to be as versatile as possible, keeping possible future robot projects in mind.



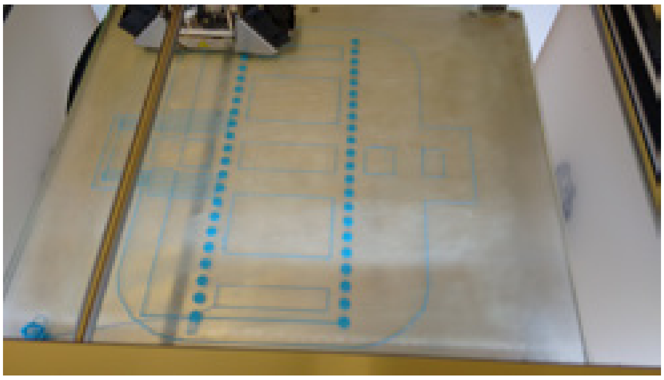
Modelling the base mesh of the body – orthographic side, front and top views

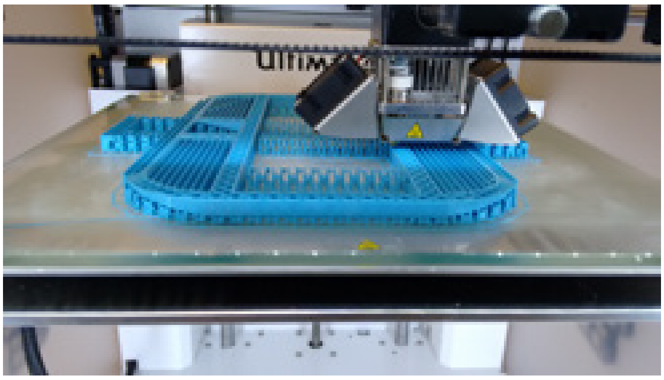


Perspective view of the finished model: solid grid and shadered

The model should provide a base for any bi-axial machine I want to build in the future, with enough space to hold a standard breadboard as well as some batteries. It's holding the motors I obtained in place and allows them direct access to the ground. The body is easy on printing material while still being robust enough for possible experiments. I chose to also add Lego compability to the model, in case I need to upgrade it with another board or other structures.

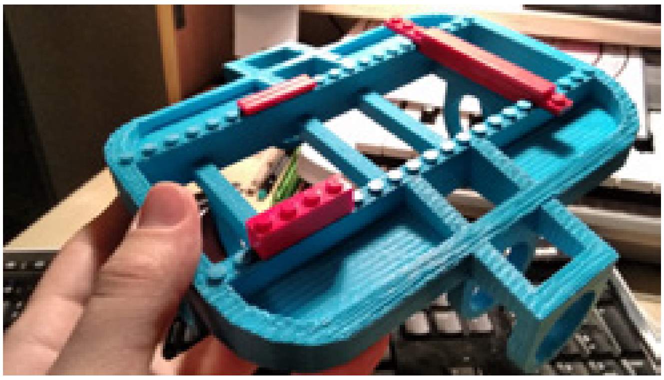
Printing the body





Printing in process, starting upside down and building on support structure

Finished print, after 16 hours of non-stop overnight printing

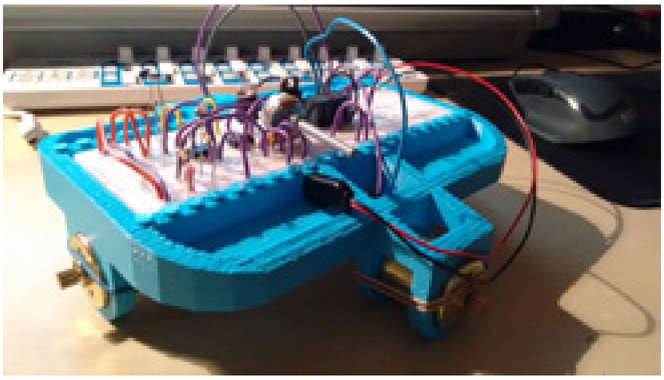


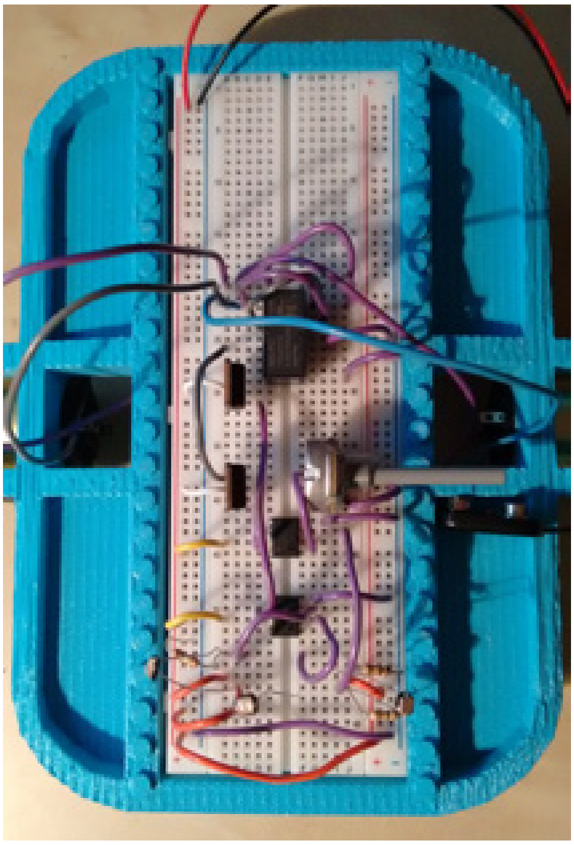
The support structures had to be carefully removed

Proof that the Lego support buttons work with actual bricks

Stacking parts

Planting my first test circuit into the model was an easy task. The motors fit into their mounts and rubber bands are holding them in place and keep them from turning. Two holes provide access to the breadboard, which sits safely on top.





Designing wheels

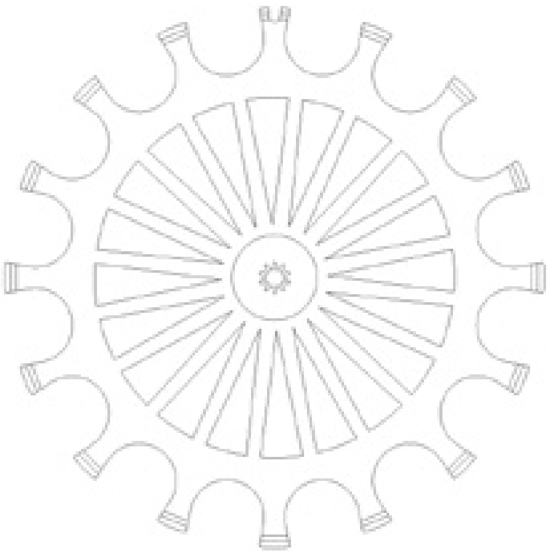
Omni wheels used in robotics

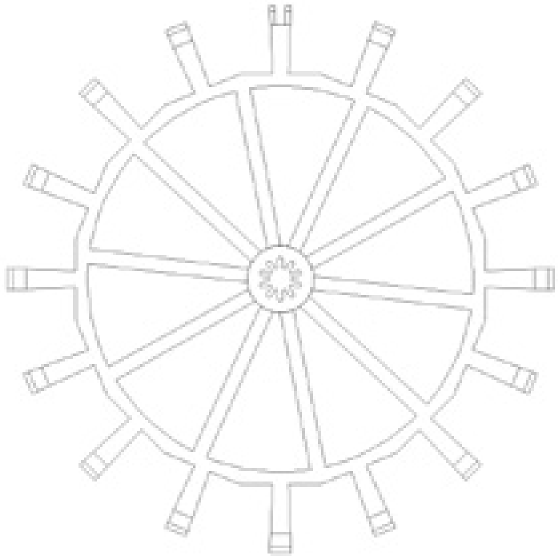
One of the biggest obstacles is the design of wheels, that allows bi-directional movement over orthogonal axes. A movement apparatus containing steering would have been to complicated to control electronically, which is why I had to seek some inspiration looking at *classic* robotic examples.

What I found was the principle of the omni wheel. Omni wheels have small discs around their circumference, perpendicular to their turning direction. They can be driven in their natural direction of turning, while still being able to slide laterally.

Omni wheels seemed perfect for my need to create a *holonomic* machine within a two-dimensional space. As it turned out, effective omni wheels are not easy to craft and seem to require cleverly adjusted structures and material properties.

Constructing wheels



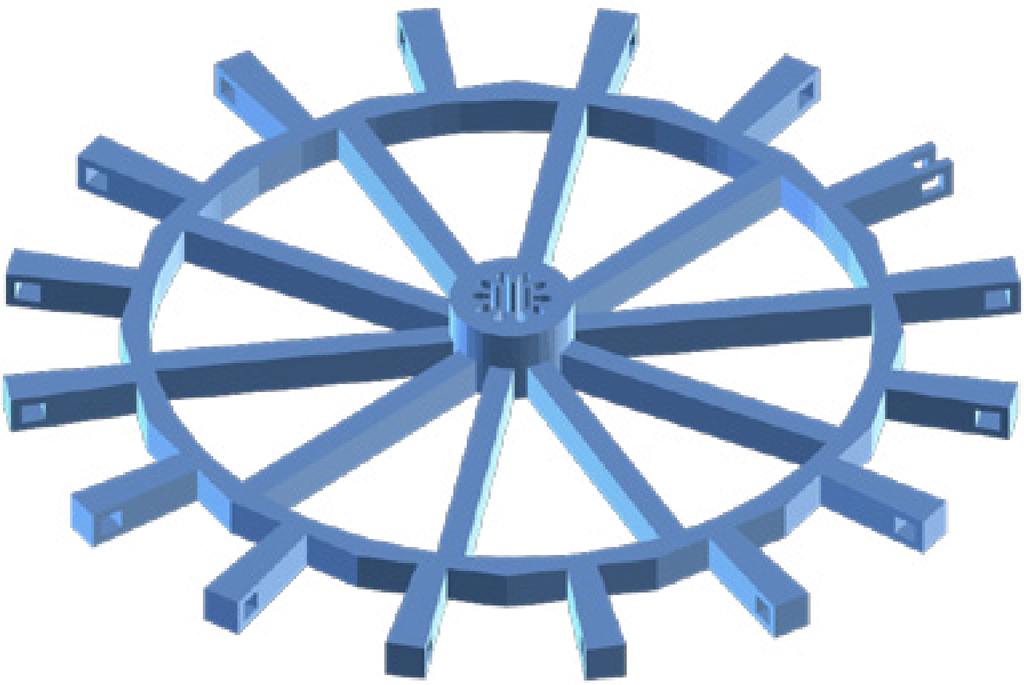


My first design was based on a simple wheel with spokes, holding 16 wooden balls in place with a wire.

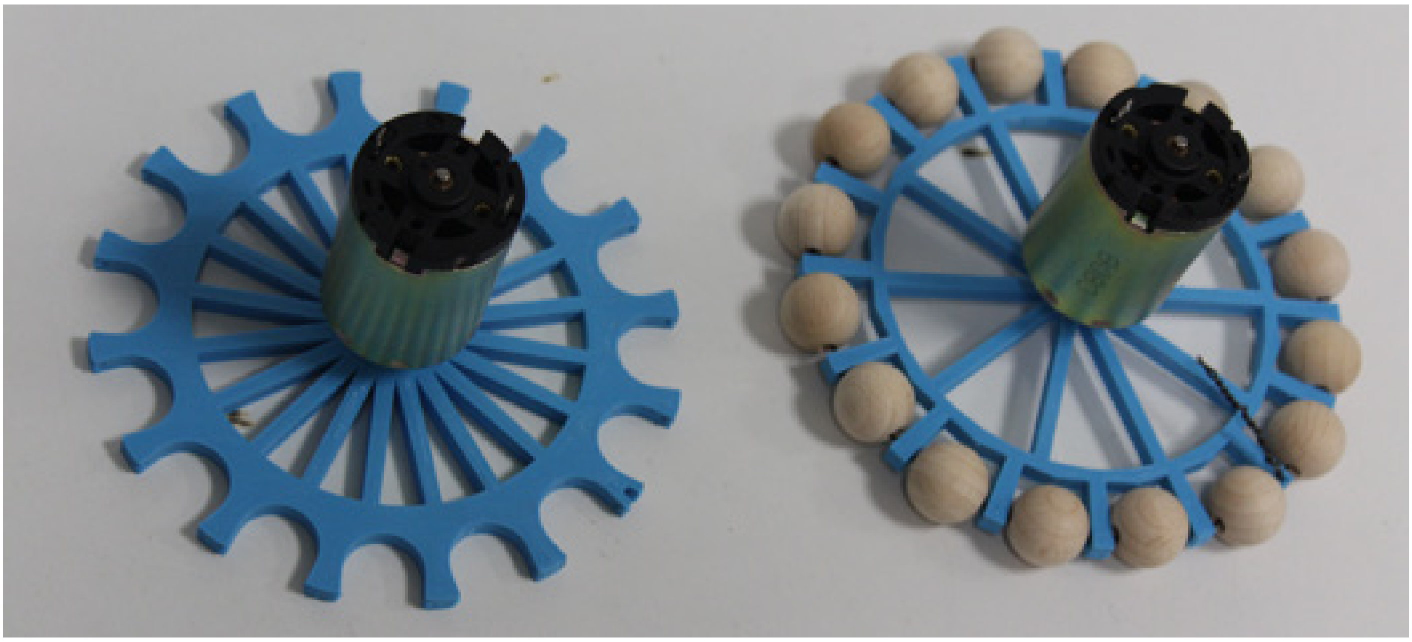
Sadly – although my measurements seemed exact – the printer got most of the holes and connections slightly wrong. There was no way to get the wire through all of the loops and the connection to the motor was too small. One way to avoid this would have been to tweak the overall quality of the prints, which results in extremely high printing times.

I decided to rather improve the design and use the first prototype to see where I can save material.

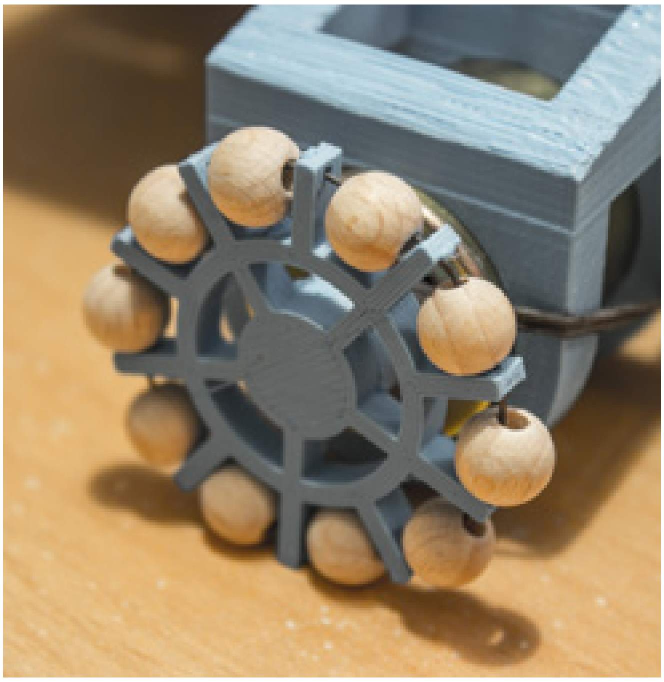
The new design should still be stable enough to support the whole robot in movement, but use less resources and provide a solid way to attach to the motors.



Shaded polygon mesh model of the second omni wheel design



Both first and second design side by side



Final wheel

Sadly, those big wheels wouldn't allow the motors to gain enough momentum, so I had to go for one final omni wheel design. The wheels of the third version are barely big enough to keep the machine's body a few millimeters above the floor. The wooden balls' diameter is reduced from 14mm to 7mm and their number shrinked to ten per wheel.

The new design pays off. Those little spinners allow for anough momentum and where the robot before wouldn't be able to move at all without help, it now speeds off with "squealing tires".

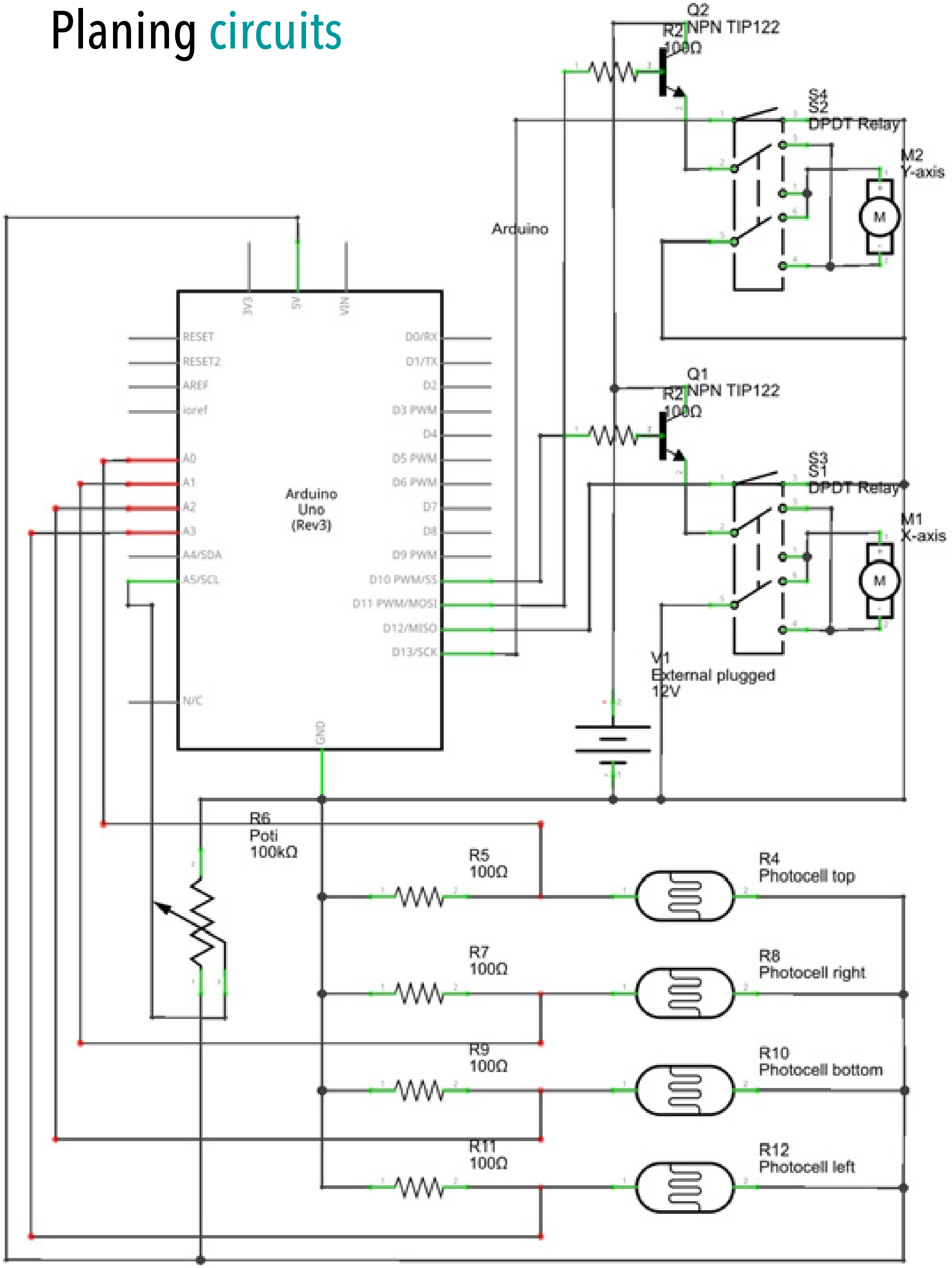
Happy with the result I could finally move on, approaching the core of the machine.

Writing code


```
//  Define pins on Arduino board
int motorXupPin = 10;
int motorYupPin = 11;
int motorXdirPin = 12;
int motorYdirPin = 13;

int photoYtopPin = 0;
int photoXrightPin = 1;
int photoYbottomPin = 2;
int photoXleftPin = 3;

int potiPin = 5;

//  Variablos to store pure input
int photoYtop;
int photoXright;
int photoYbottom;
int photoXleft;

int lightX;
int lightY;
int poti;

//  Setting up variables to store speed
int motorXspeed;
int motorYspeed;

//  Define minimum speed and difference multiplicator
int speedBase = 255;
int speedMod = 1;

//  Setup
void setup(){
  Serial.begin(9600);
  pinMode(motorXupPin, OUTPUT);
  pinMode(motorYupPin, OUTPUT);
  pinMode(motorXdirPin, OUTPUT);
  pinMode(motorYdirPin, OUTPUT);
  pinMode(photoXrightPin, INPUT);
  pinMode(photoYbottomPin, INPUT);
  pinMode(photoXleftPin, INPUT);
  pinMode(potiPin, INPUT);
}
```

```
//  Loop
void loop(){

//  Write poti factor
  poti = analogRead(potiPin);

//  Measure and store light
  photoYtop = analogRead(photoYtopPin);
  photoXright = analogRead(photoXrightPin);
  photoYbottom = analogRead(photoYbottomPin);
  photoXleft = analogRead(photoXleftPin);
  lightX = max(photoXright, photoXleft);
  lightY = max(photoYtop, photoYbottom);

//  Y: See if there is enough light to trigger movement
  if (lightY <= poti){
    analogWrite(motorYupPin, LOW);
    }
  if (lightY > poti){

//  Y: Change motor direction by triggering relays
    if (photoYtop >= photoYbottom){
      digitalWrite(motorYdirPin, HIGH);
      }
    if (photoYtop < photoYbottom){
      digitalWrite(motorYdirPin, LOW);
      }

// Y: Power up motors with PWM based on light comparison
    motorYspeed = min(speedBase + (abs(photoYbottom - photoYtop) / speedMod), 255);
    analogWrite(motorYupPin, motorYspeed);
  }

//  X: See if there is enough light to trigger movement
  if (lightX <= poti){
    analogWrite(motorXupPin, LOW);
    }
  if (lightX > poti){

//  X: Change motor direction by triggering relays
    if (photoXleft >= photoXright){
      digitalWrite(motorXdirPin, HIGH);
      }
    if (photoXleft < photoXright){
      digitalWrite(motorXdirPin, LOW);
      }

// X: Power up motors with PWM based on light comparison
    motorXspeed = min(speedBase + (abs(photoXright - photoXleft) / speedMod), 255);
    analogWrite(motorXupPin, motorXspeed);
  }
}
```

Planing circuits



Assembling elements

So, what do I need to build this thing? I split the functions into basic modules: I need four photocells measuring light in each direction – therefore four inputs monitoring their resistance. I need two PWM driven outputs to control motor speed; one for each axis. The PWMs should each feed a transistor, enabling me to use an external power supply for the motors (they're hungry for 12V). The motor's direction will be switched by one relay per axis, changing their poles via a DPDT switch – giving me two more arduino outputs needed. Last but not least: A simple potentiometer, to manually set a resistance which can be compared to the photo resistors. This way, I can adjust my robot to the current light situation of the room.

Altogether, I have nine connections to the arduino.

Five inputs:

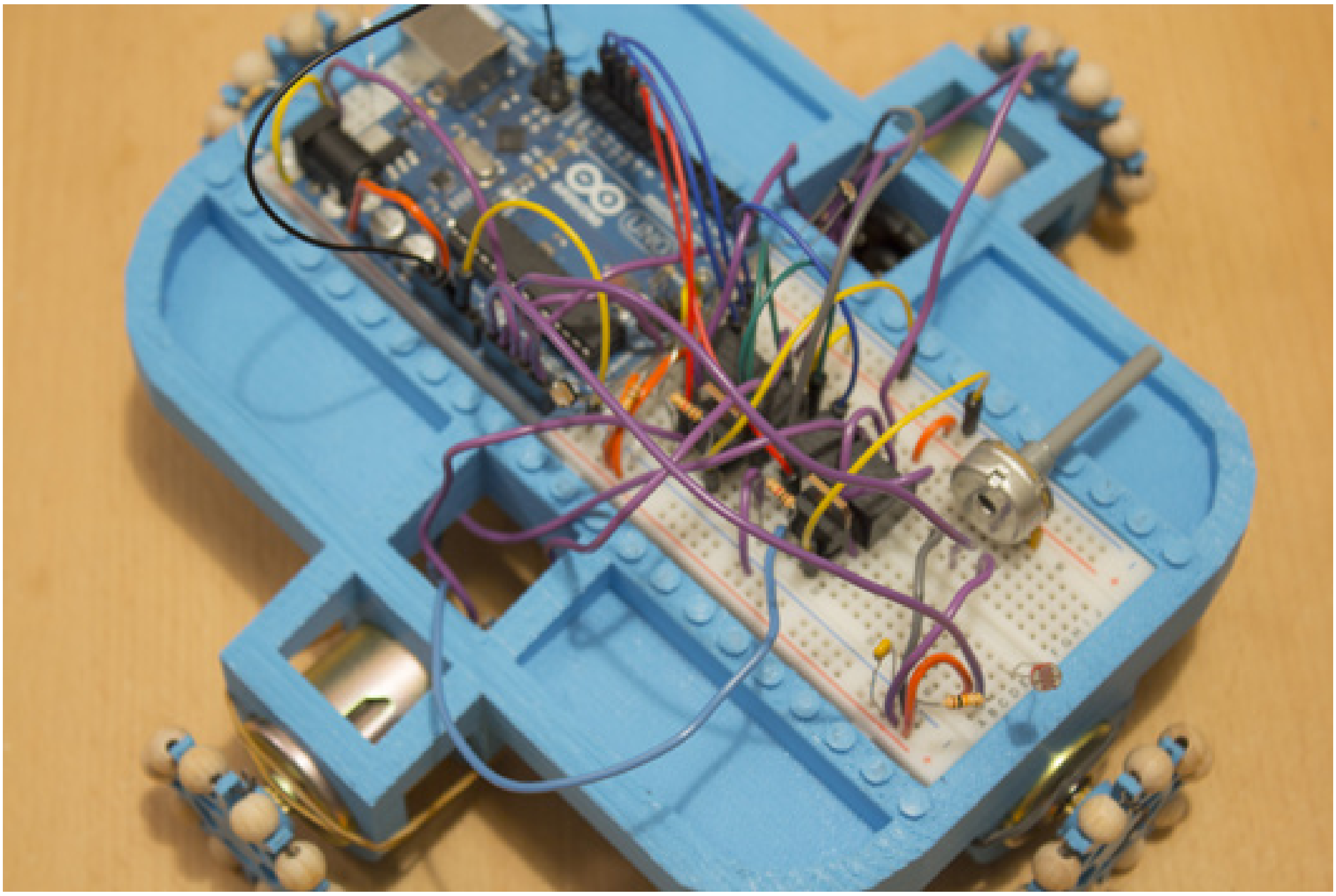
- Photocell top
- Photocell right
- Photocell bottom
- Photocell left
- Variable resistance (poti)

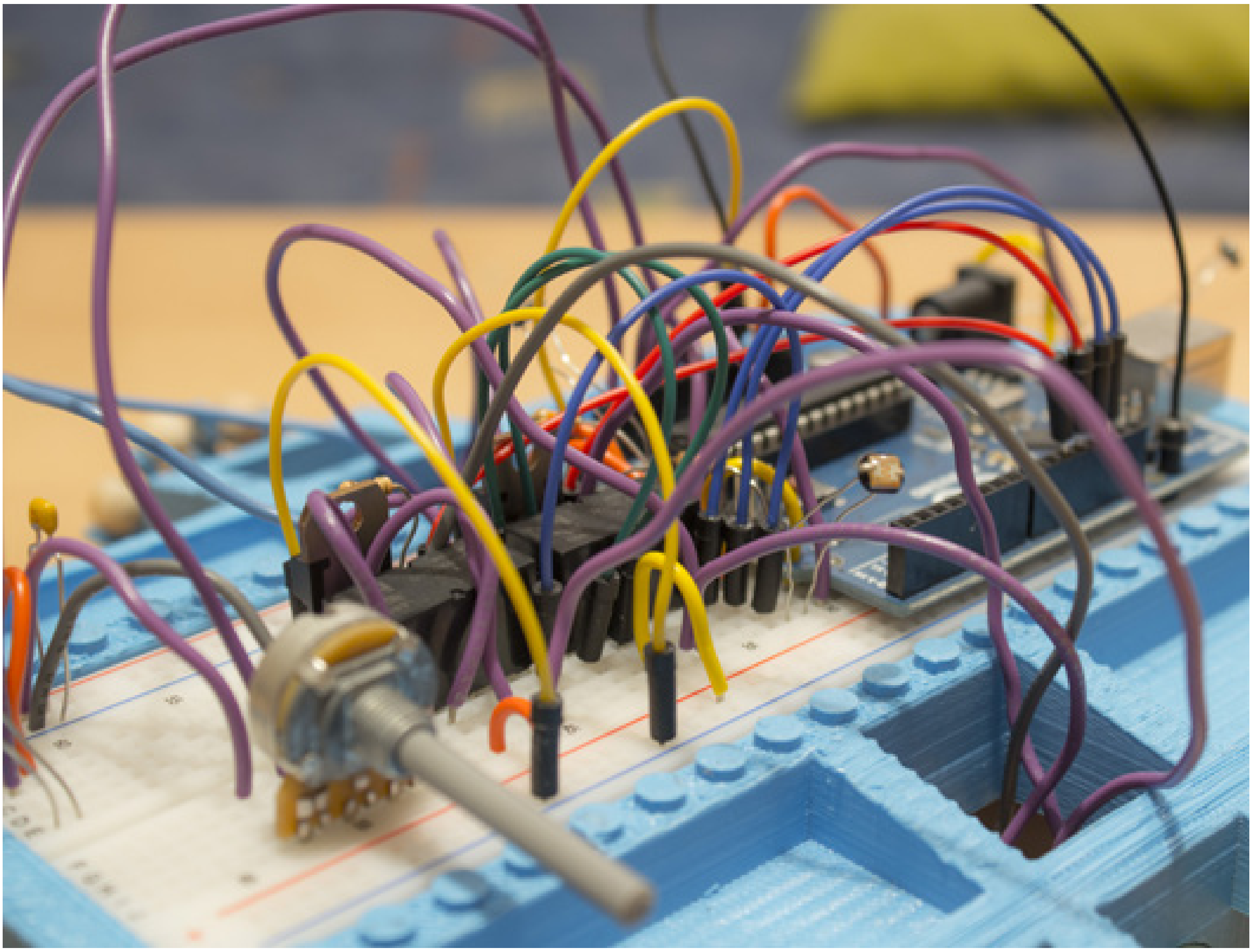
And four outputs:

- X-axis motor speed (PWM)
- X-axis motor direction switch
- Y-axis motor speed (PWM)
- Y-axis motor direction switch

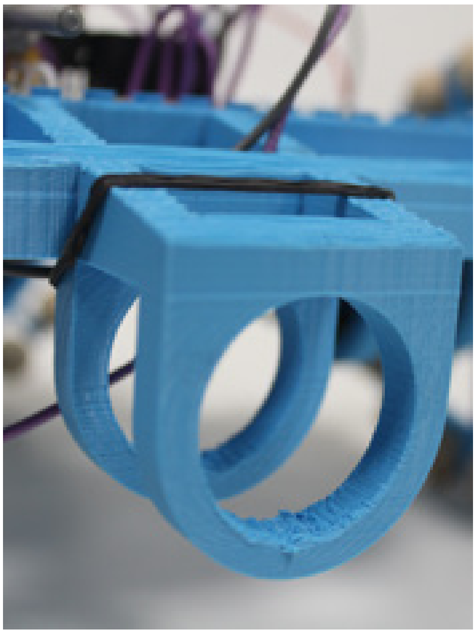
After putting down the concept, the process of coding and assembling was very quick; however being followed by a painful (hardware-) bug hunt.

Top view of the beautiful wire chaos

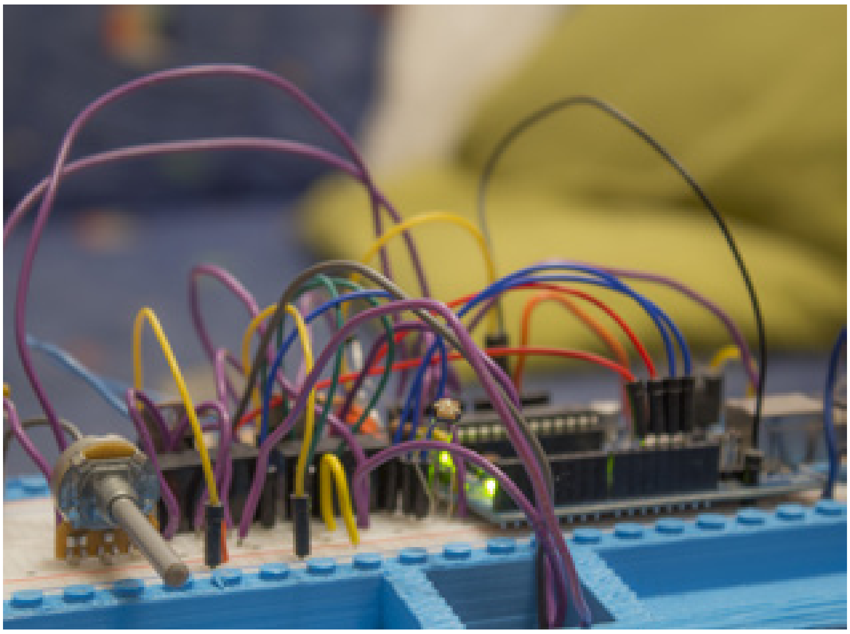




Close-up of the circuit; pairs of relays and transistors in the middle and arduino in the background

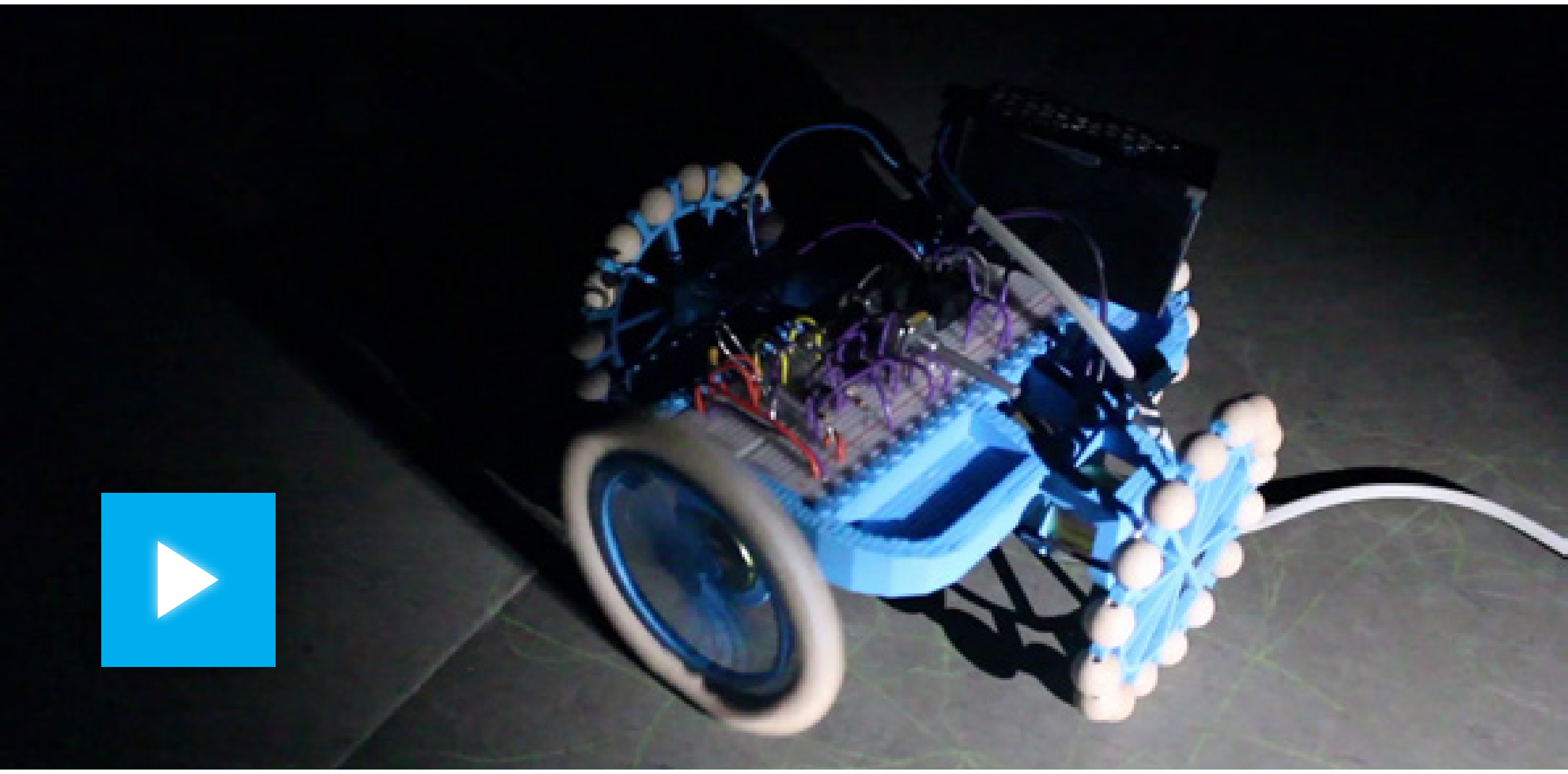


Structure holding the motors

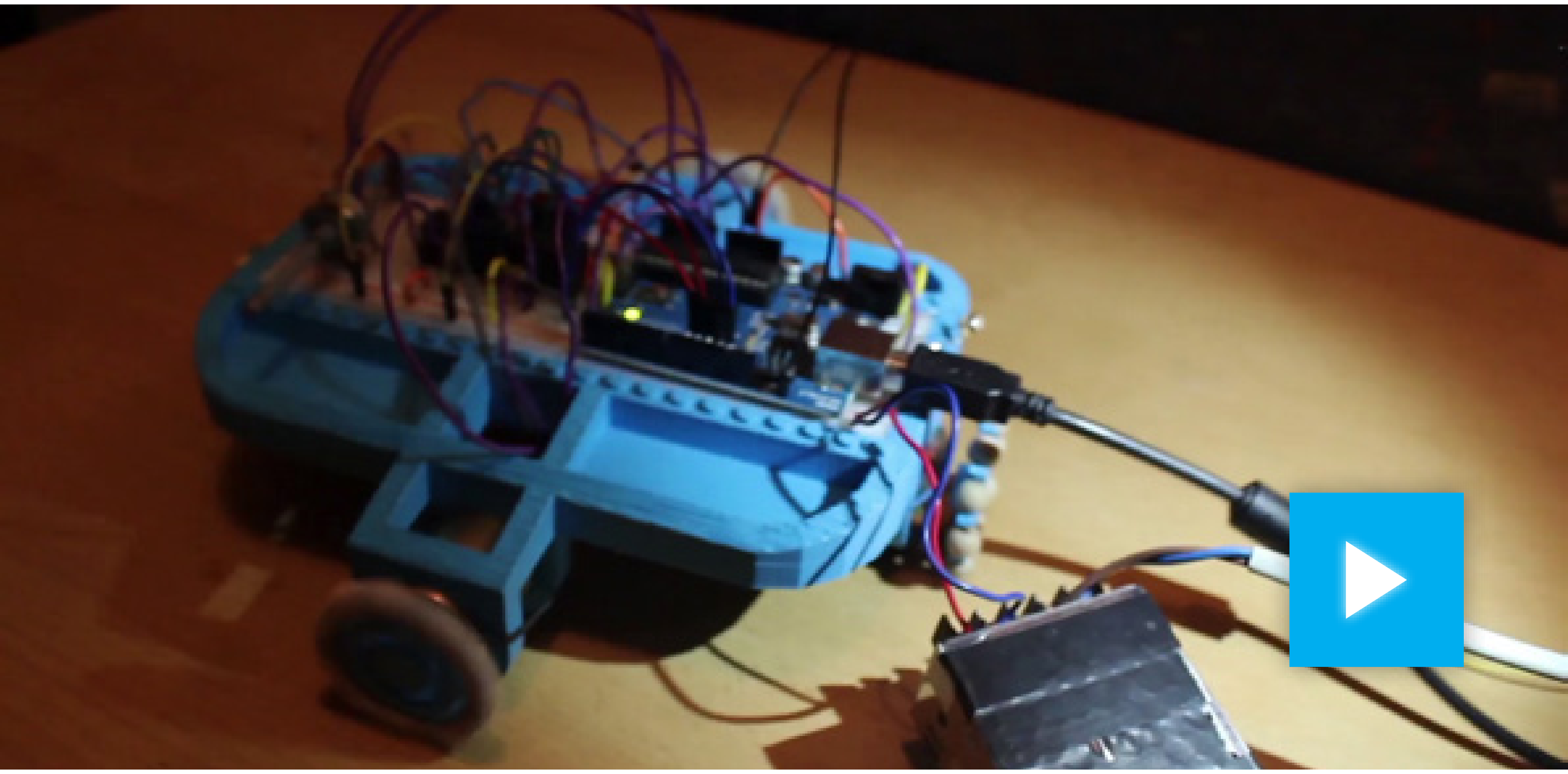


Circuit from the side

Recording action

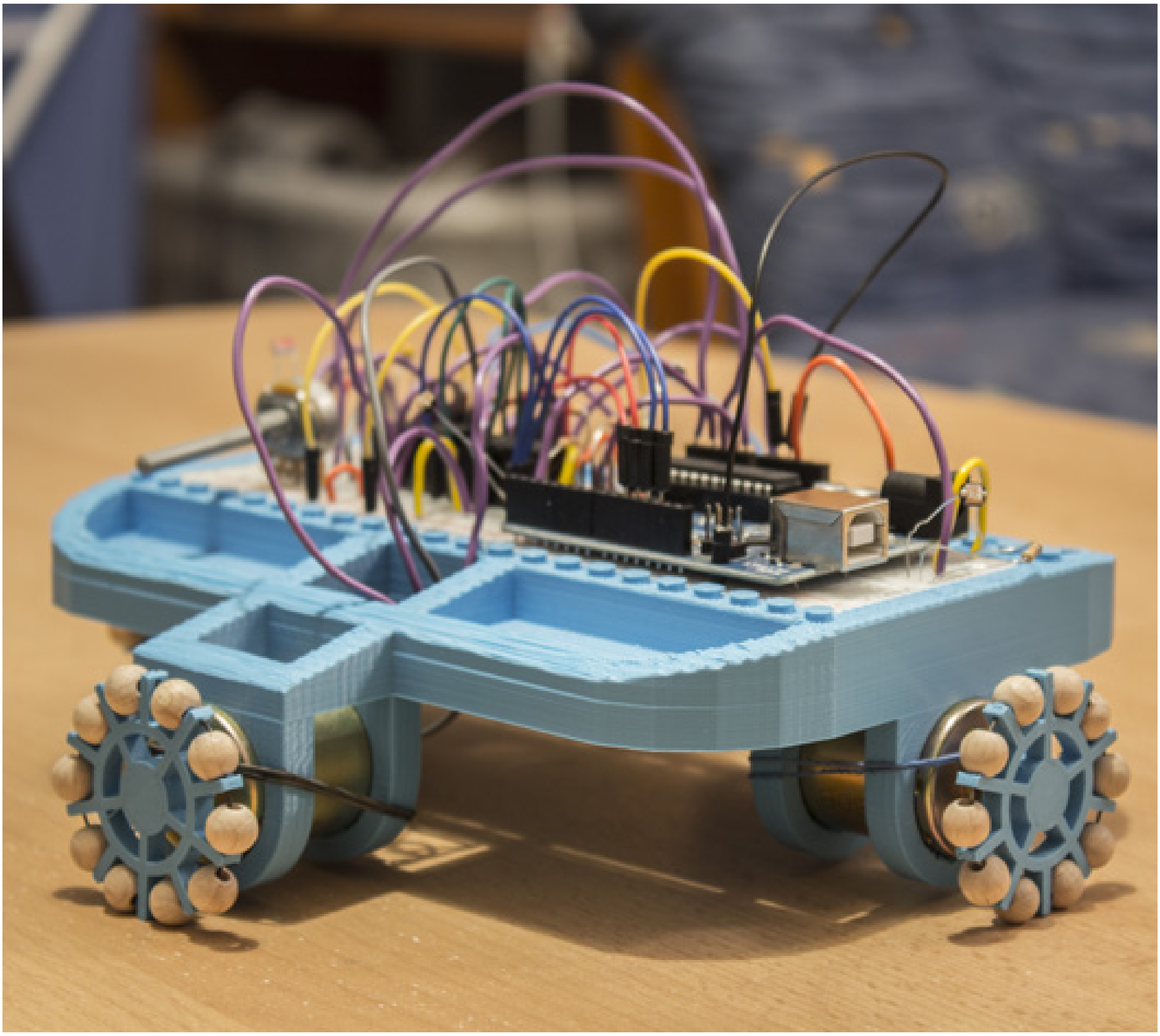


First, arduino-less demonstration of the bot (Youtube)



Final demonstration with arduino and new wheels (Youtube)

Discussing further development



After working on the wheels over and over again, I am happy I was finally able to create a working solution. For some time I thought I'd never see this thing move – witnessing how it now even drags its power supply for some distance gives me almost fatherly feelings. The addition of the arduino provided my machine with the much needed flexibility and control to make it fully functional. I am very happy with the result and I most definitely learned a lot through this project – be it electronics, working with the arduino or 3D printing. The next (and probably last) step is to find a portable power supply that allows fully unplugged movement. Once the robot is wireless and mobile I can start experimenting with infinite inputs to control it instead of light!

Thanks to Johannes Deich for his awesome class!