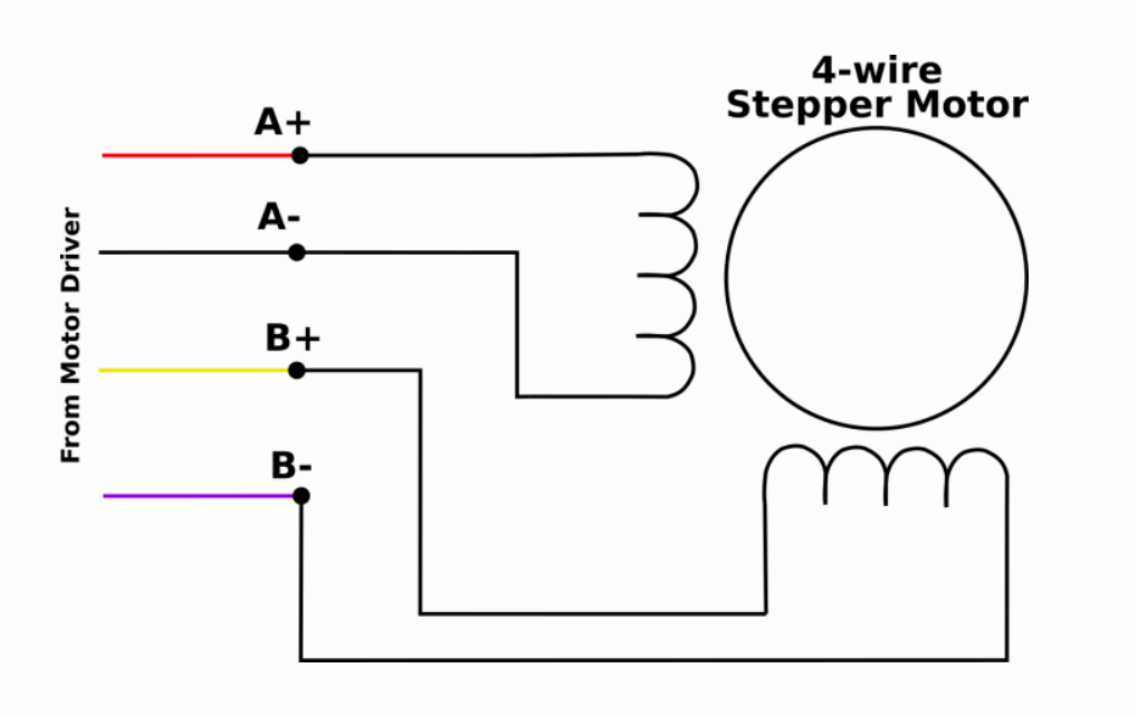
Arduino Uno Stepper motor

An arduino is a very small microcontroller, having several in/out lines and different connectivity means. The Arduino that we have used is the Arduino UNO. The Stepper motor is the Sanyo Denki series 103h7823-1740. This motor has a dual bipolar connection, i.e. with two different coils. The first coils has two ends (A+, A-) and the second one (B+, B-).



Besides that, the stepper motor has to be supplied with an intensity of at least 2amp in order to move the motor. The Arduino cannot do this, so that we used an H-bridge (L298N Deriver) that interfaces the Arduino with the Stepper motor.

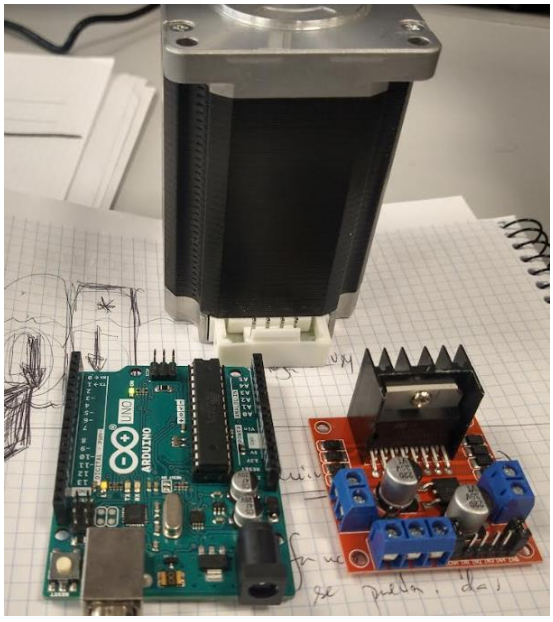


Figure 1. Arduino UNO, 103h7823-1740 Stepper motor, L298N Deriver

The programming of the Arduino is very straightforward. We download the Arduino IDE (https://docs.arduino.cc/software/ide-v1/tutorials/Windows). The Arduino UNO connects to a PC though the USB, both to programming and as a serial line using the USB. The installation of the IDE makes all this in an automatic way.

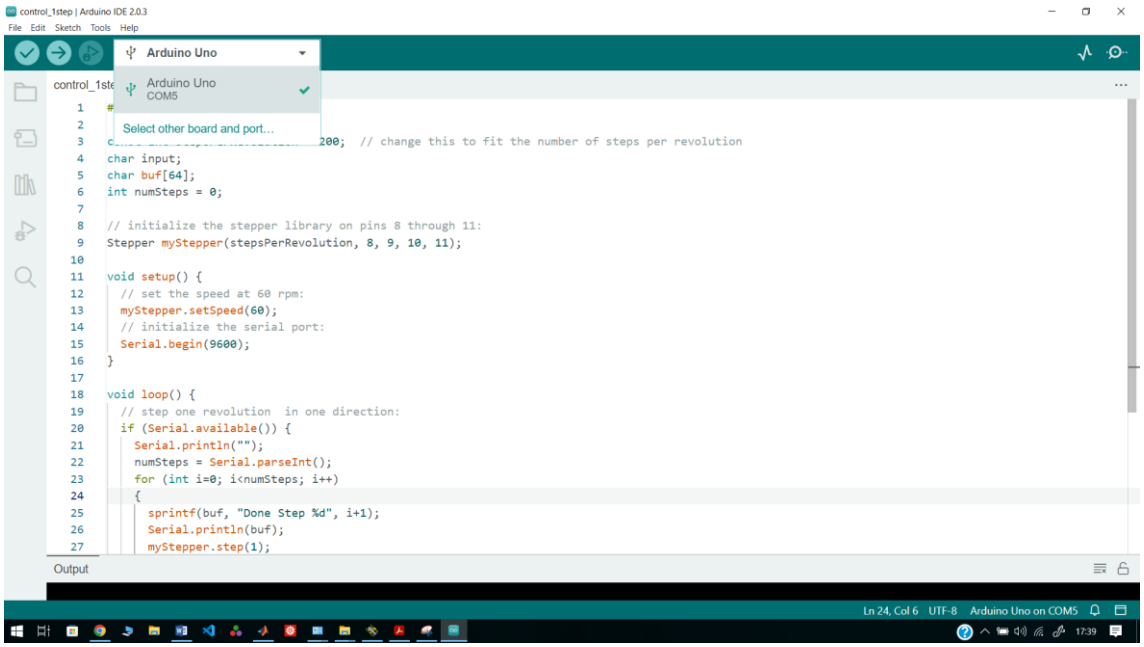


Figure 2 – Arduino connects to the serial COM5 via the USB

The connections of the Arduino to the L298N and the 4 inputs A+ A- B+ B- are the following:

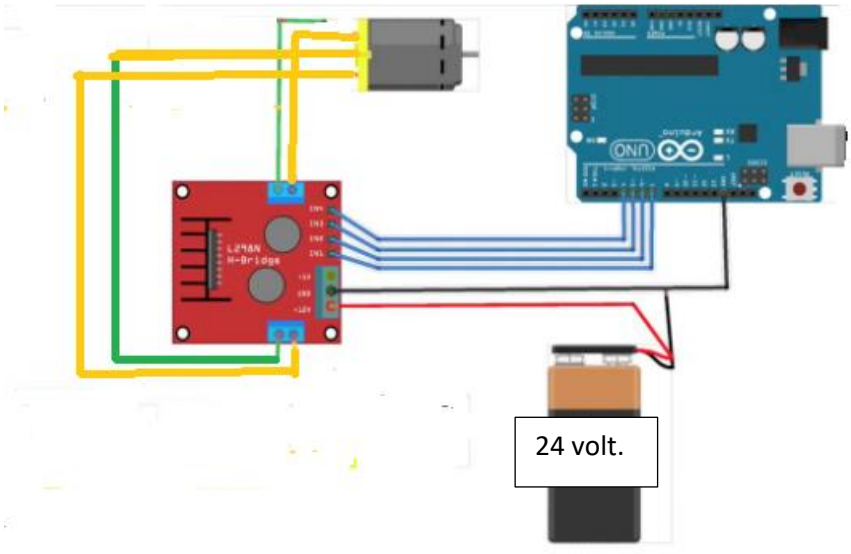


Figure 3 - Schematics of the connections of the Arduino to the L298N

The Arduino program is very simple. It opens the Serial Line COM5, and waits a command with the number of steps to execute, which are executed one step at the time. The stepper has 200 steps per revolution. After executing one step, the motor inputs are set to zero to avoid feeding the motors.

```
#include <Stepper.h>

const int stepsPerRevolution = 200;  // change this to fit the number of
steps per revolution
char input;
char buf[64];
int numSteps = 0;

// initialize the stepper library on pins 8 through 11:
Stepper myStepper(stepsPerRevolution, 8, 9, 10, 11);

void setup() {
  // set the speed at 60 rpm:
  myStepper.setSpeed(60);
  // initialize the serial port:
  Serial.begin(9600);
}

void loop() {
  // step one revolution  in one direction:
  if (Serial.available()) {
    Serial.println("");
    numSteps = Serial.parseInt();
    for (int i=0; i<numSteps; i++)
    {
      sprintf(buf, "Done Step %d", i+1);
      Serial.println(buf);
      myStepper.step(1);
      delay(500);
    }
    digitalWrite(8, LOW);
    digitalWrite(9, LOW);
    digitalWrite(10, LOW);
    digitalWrite(11, LOW);

  }else
  {
    delay(100);
  }
}
```

Figure 4 – Control program of the Arduino

Both Gige Cameras and the serial line can be controlled with Matlab. The program of laser interpolation with camera, consists of a CCD and a laser whose laser plane goes through the axis of rotation of a plate. We rotate the plate taking 200 images per revolution, allowing us to reconstruct the object. The connection of the USB camera in Matlab is straightforward. First, we check the number of cameras:

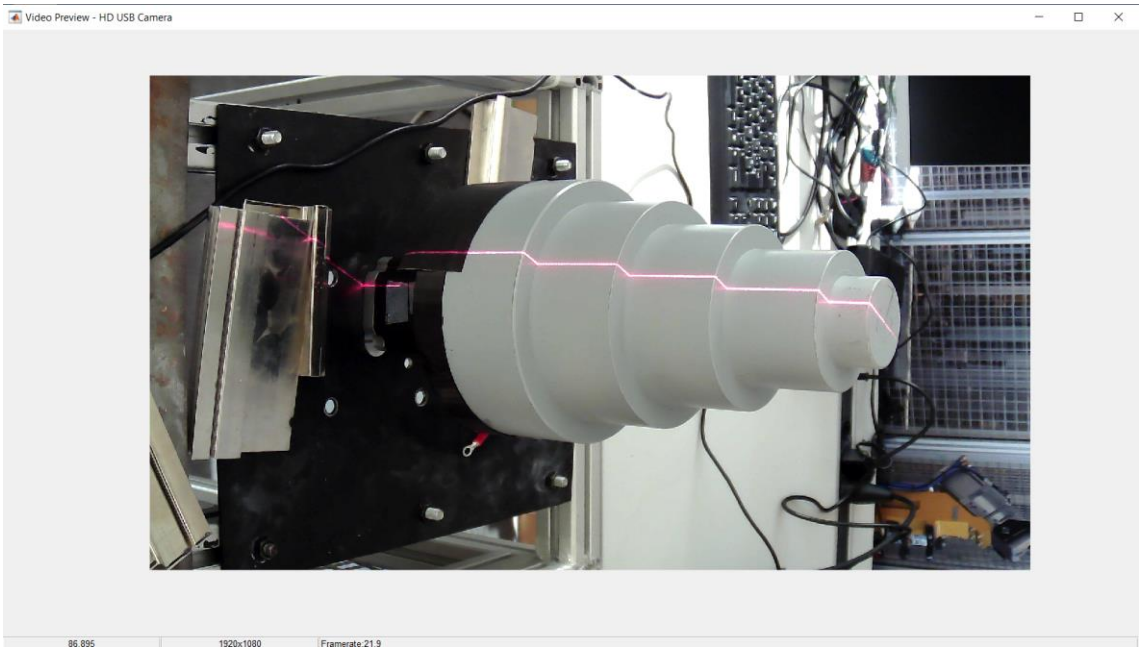
```
>> webcamlist
ans =

  2×1 cell array

    {'HP HD Camera' }
    {'HD USB Camera'}
```

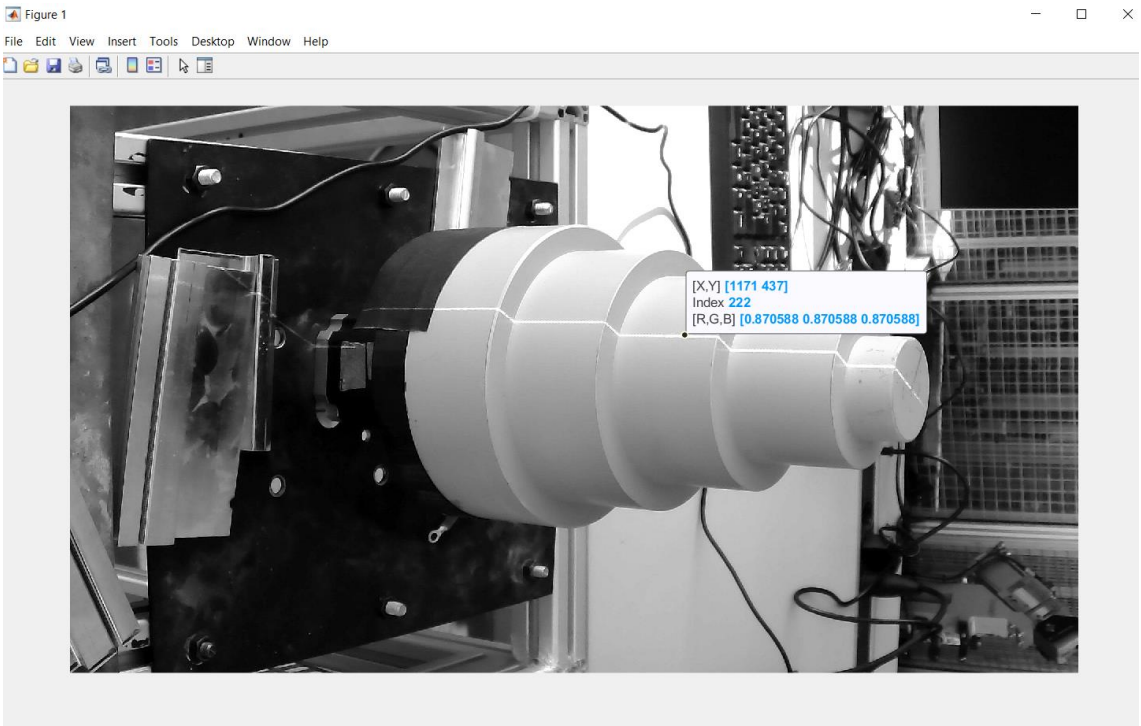
The first camera corresponds to the Laptop, and the second to our camera. We connect it:

```
>> cam = webcam(2);
>> preview(cam);
```



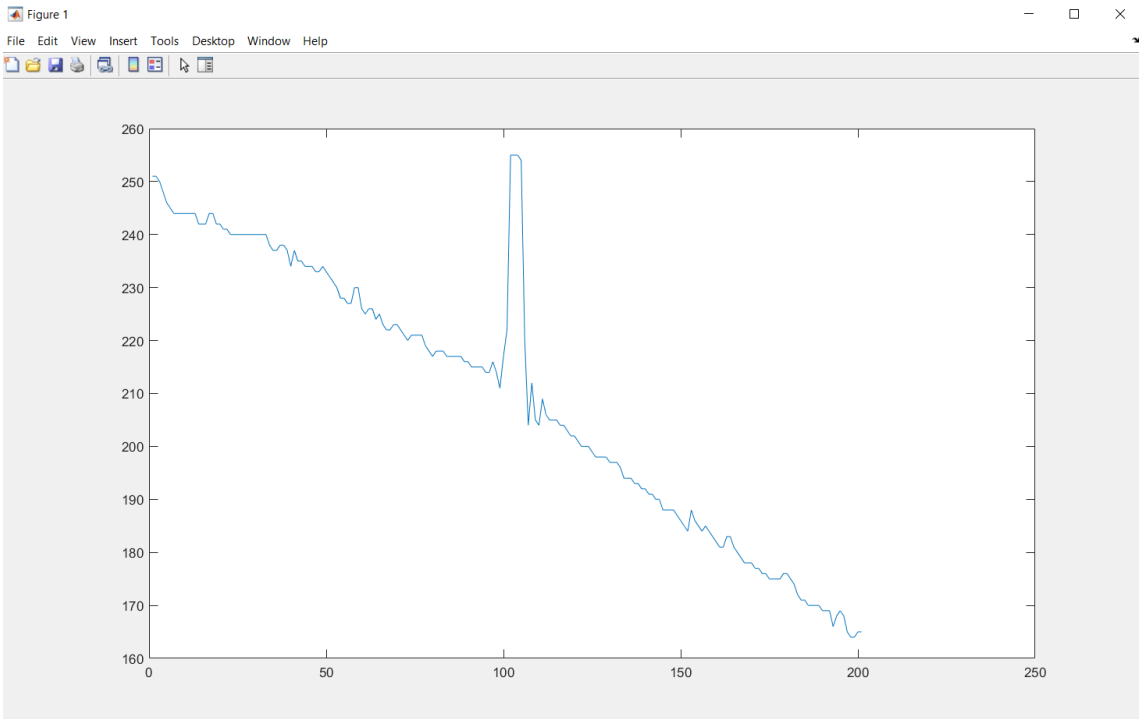
To take one image:

```
>> img = snapshot(cam);
>> imgGray = rgb2gray(img);
>> imshow(imgGray)
```



We can plot the line 1171, and see the laser profile, around the point 437:4

```
>> line = imgGray(:, 1171);
>> plot(line(437-100:437+100))
```



The laser line is saturated and the background should be darker. In order to do a fitting to obtain a sub-pixel precision, we should perform either fittings to parabols, exponential curves, or calculate the center of gravity.

There are cameras that calculate in real time the subpixel of the laser line. For example, the camera (C4-4090-gigE) delivers lines with the maximal of the laser peaks instead of the images.

https://www.automationtechnology.de/cms/wp-content/uploads/2015/07/C4-4090-gigE_web.pdf

Finally, we can also control the stepper motor with Matlab. We can open the serial line:

```
>> s = serialport('COM5', 9600);
```

We send 10 as string to the serial line, which moves 20 steps in the stepper motor.

```
>> write(s, "10", "string")
```

In order to close the handle, we write:

```
>> s = [];
```

Bibliography

https://www.instructables.com/Control-DC-and-stepper-motors-with-L298N-Dual-Moto/

https://create.arduino.cc/editor/albertoizaguirre/a63c16f8-e6a5-4348-aa01-4e8fcf1a15c4

https://www.norwegiancreations.com/2017/12/arduino-tutorial-serial-inputs/

https://docs.arduino.cc/learn/electronics/stepper-motors

https://datasheetspdf.com/pdf/1317040/SanyoDenki/103H7823-5740/1

https://programarfacil.com/blog/arduino-blog/motor-paso-a-paso-uln2003-l298n/

https://forum.arduino.cc/t/control-4-amp-nema-23-114mm-stepper-motor-with-a4988/680114/11

https://www.pololu.com/product/3730

https://sedilec.com/producto/fuente-de-alimentacion-conmutada-12v-2-amp-sin-caja-ref-c-701-smps-s-c