Póvoa de Varzim

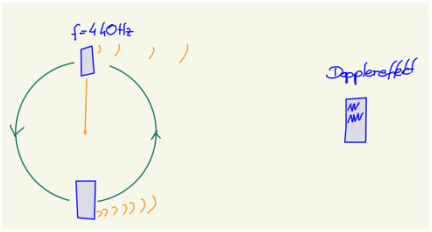
Escola Secundária de Rocha Peixoto

W A V E S

17.05. – 20.05.2022

Topic	Age	Country	Date
Dopplereffect with Smartphones (Phyphox)	>14	Germany	Mai 2022

Dopplereffect with a Smartphone



Experimental setup materials:

2 Smartphones

1 case

1 rope

Experimental setup and procedure:

Start the **Phyphox App** on your first smartphone and setup the experiment „Tone generator“. Mount the smartphone in the case and fix the rope to your case. Play the frequency 440Hz with the phyphox App. Do movements of circles with the rope and smartphone over your head. Try to circle in a constant angular velocity.

Pay attention, nobody should stand next to you!

The other students have to stand 3 m away from you, while they are listening to the tone !

Experiment evaluation:

1) Describe, what you hear in a distance of 3 m!

2) Explain what you hear, at the part of the circle, where the smartphone moves in your direction. Explain the connection of the wavelength you hear λ_E and the velocity of the smartphone v_{Smart}!

3) Measure the frequency, you hear f_E with a second smartphone and the program „frequency“ in the Phyphoxapp . The frequency of the smartphone is called f_S .

   Calculate the speed of the Smartphones v_{Smart}, with the help of Doppler-formula:

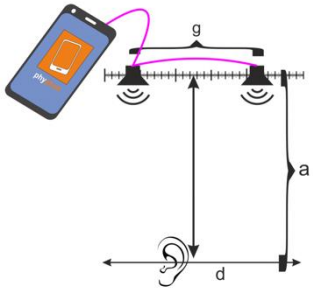
$$f_E = f_S \cdot \frac{1}{1 \pm \frac{v_{Smart}}{v_{Schall}}}$$

   (v_{schall}= 342m/s (Speed of Sound))

4) Open the program „Dopplereffect“ on your second Smartphone. Write *440Hz* in the settings and click to *results*. Now you can measure the shift in wavelength directly; you can also compare the speed you have calculated with the speed which shown in the phyphox App.

Co-funded by the Erasmus+ Programme of the European Union

Topic	Age	Country	Date
Interference with two smartphones (Phyphox)	>14	Germany	Mai 2022



Experimental materials:

2 smartphones/ tablets

1 tape measure

1 duct tape

Experimental setup and procedure:

Start the **Phyphox App** on both smartphones and open the experiment „Tongenerator“. Fix them in a horizontal position, so that the speaker points towards you. The distance between both speakers has to be 0,80 m.

Start the frequency of 440Hz on both smartphones. Close one of your ears with your hand and go slowly to a distance of a=3,00 m along line d.

Experiment evaluation:

1) Describe, what you hear while you are walking line d.

2) How do the positions of the accoustic maxima change, if you use different frequencies?

3) Find out a connection for the wavelength λ and the distances of the positions of the maxima d_k.

4) The frequency is defined as:

$$f = \frac{c_L}{\lambda} = \frac{343\frac{m}{s}}{\lambda}$$

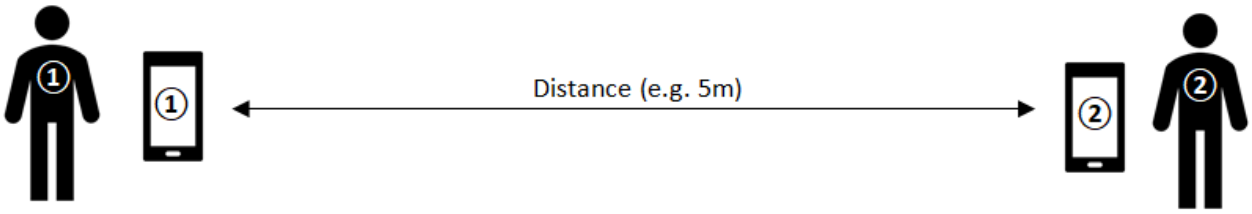
5) Find a connection between the distance of the speakers b and the positions of the accustic maxima d_k.

6) Experiment with the distances of the speakers b, what do you observe especially for short distances?

7) Can you find a set-up with positions of total silence?

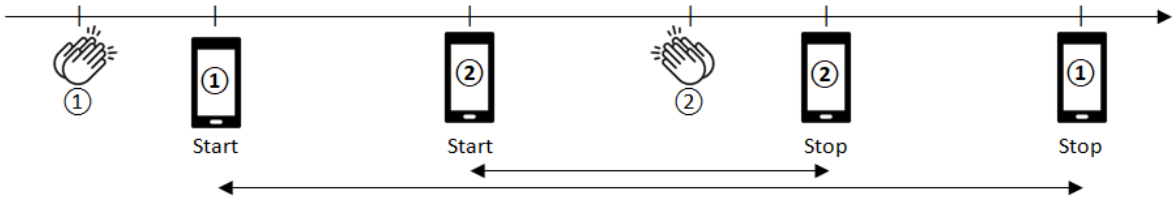
Topic	Age	Country	Date
Speed of sound (Phyphox)	>14	Germany	May 2022

- 2 Smartphones
- 2 persons
- Tape measure

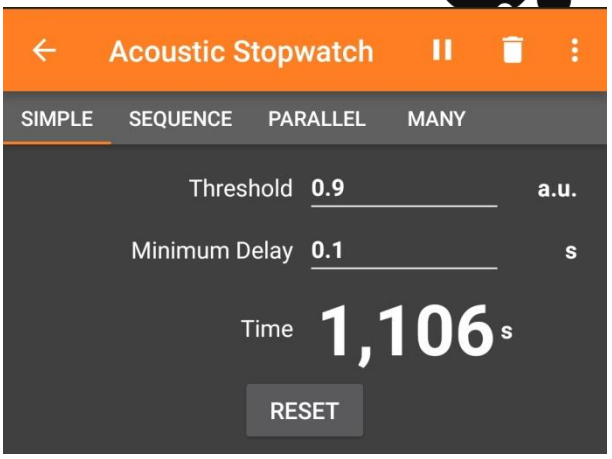


In this experiment you can approximate the speed of sound. 2 persons have a certain distance (tape measure!) from each other (e.g. 5m).

For this experiment you need the acoustic stopwatch of "Phyphox" to get the time between two acoustic events. The first signal causes from the clapping of the first person. The second signal by clapping of the second person stops the clock running.



At the end you have to subtract the time of smartphone 2 from the time of smartphone 1. So you get the time for the double distance of the sound.



It is very important to put the threshold to such a level, that the noise from the area doesn't disturb your measure.

Now we can calculate the speed of sound by using the distance d and the time difference Δt :

Calculated value: c =

Theoretical value: c =

Topic: Waves	Age	Country	Date
Math and Music	>14	Poland	May 2022

Function, realisation

Some sound sources generate very pure tones (e.g. sound fork). The sounds generated by musical instruments are composed of many tones of different frequencies. The task of the experiment is to find the peak frequency (using Fourier transform FFT). To also get the timbre of the instrument would require extraction of the overtones as well.
The aim of the experiment is to show that the frequencies of the octave sound form a geometric sequence with the q=2^(1/12).

Hardware required

- Vernier Analogue Microphone sensor MCA-BTA
- Labquest mini interface (microcontroller).
- PC or Mac computer
- sound sources: sound fork, guitar, flute, piano
- calculator
- tone generator mobile app
-

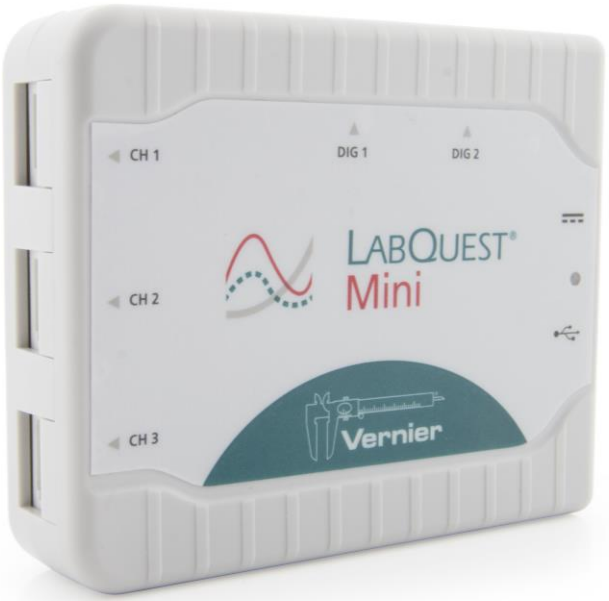
Materials required

- paper for note

Software required

-Logger Pro software from Vernier

Setup Hardware

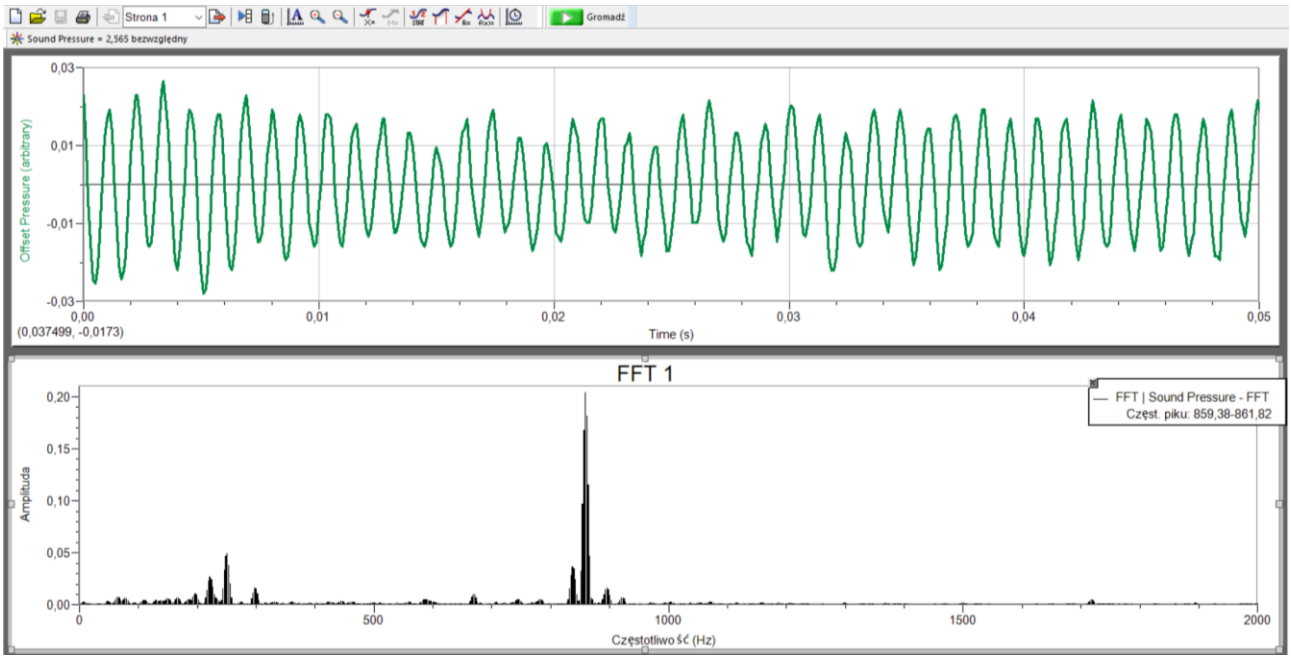


We must connect microphone plug to selected 1 of 3 available analog sockets in Labquest mini microcontroller. Then we must connect Labquest mini wit USB cable to PC or Mac.

Setup software

For starting experiment we must find and open experiment file Mathemattics and Musics (from the set Physics with Vernier)

For start measure frequency of the sound we must click green button (see illustration below)



Tasks- testing steps

- measure tone frequency generated by sound fork
- finding find the peak frequency of the sound generated by different musical instruments: guitar, flute, piano (the same note)
- testing frequency of the sound generated by mobile app.

After testing the operation of the system on various instruments and sound sources, we move on to finding the relationship between frequency and note pitch.

Experiment Evaluation

System testing:
-measure tone frequency generated by sound fork

Write down Your results

Nr	key	Frequency Hz	ratio to previous note	ratio to A4
1	A4	440	----------	1
2	A4#			
3	H4			
4	C4			
5	C4#			
6	D4			
7	D4#			
8	E4			
9	F4			
10	F4#			
11	G4			
12	G4#			
13	A5			

Go further

We can do the same using Python script (example given https://pythonnumericalmethods.berkeley.edu/notebooks/chapter24.04-FFT-in-Python.html), but the code is a bit long, and and not all is easy to understand by all students. This code use Numpy, Pandas and Matplotlib libraries.

Topi: Waves Principle of operation of the RFID system	Age >14	Country Poland	Date May 2022

Function, realisation

Build experimental system which provide testing tags, cards and reading information stored inside RFID tags, which use radio waves

Hardware required

- Microcontroller Raspberry Pi Pico with Micropython 1.18 or above
- Cytron Maker Pico docking station for Pico (or different docking station for Pico: Waveshare, Seeedstudio)
- RFID-RC522 card reader
- Mifare 13,56kHZ cards, tags
- PC or Mac computer

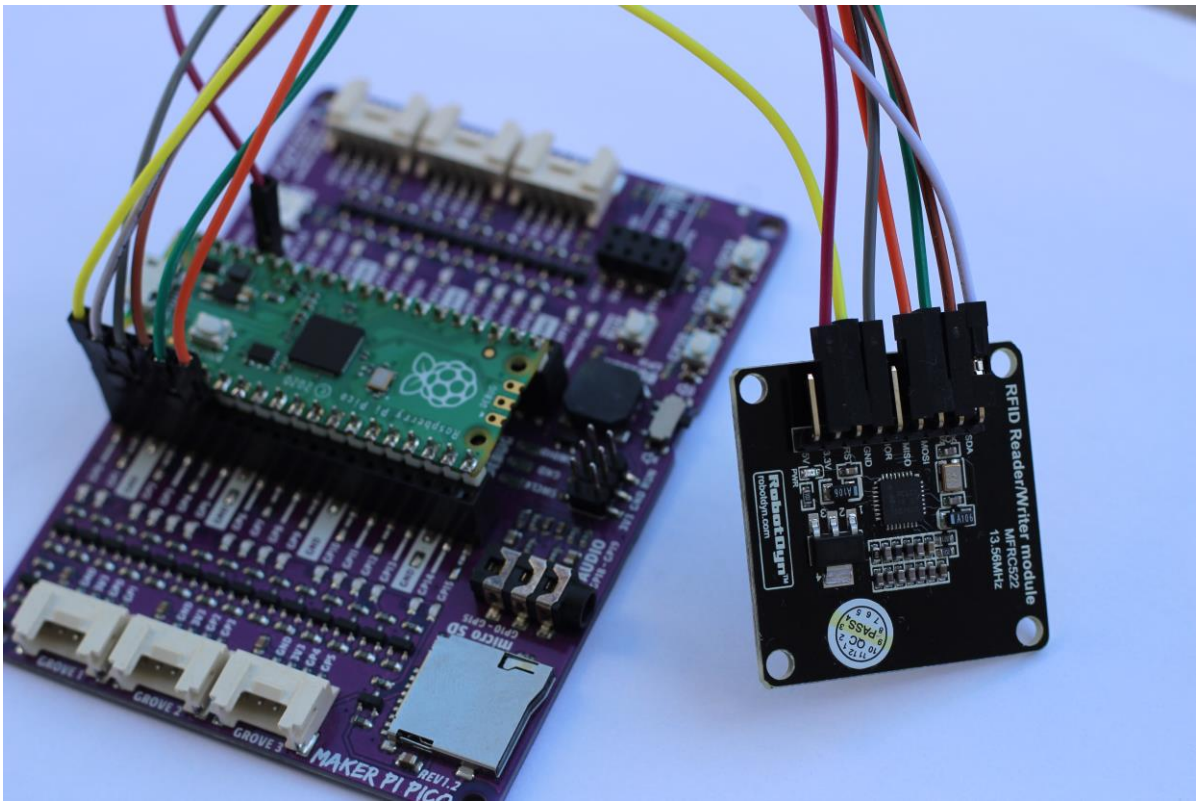
Materials required

- 7 wires in different colours (Male-Female DuPont wire)
- micro usb 2.0 high speed (15cm or 30 cm)
- paper for note

Software required

-Thonny (thonny.org)
-library for RFID https://github.com/danjperron/micropython-mfrc522/blob/master/mfrc522.py
- script https://github.com/danjperron/micropython-mfrc522/blob/master/Pico_example/Pico_read.py

Setup Hardware



RFID reader use serial communication in **SPI standard**, so we need a bit more wires then in I2C case.

Reader Pin	Raspberry Pi Pico	GPIO ESP32	GPIO ESP8266
sck	**GP2**	18	0
mosi	**GP3**	23	2
miso	**GP4**	19	4
rst	**GP0**	22	5
cs (SDA)	**GP1**	21	14

The table also contains information on how to connect to other popular microcontrollers (ESP32, Esp8266)
The reader use 3.3V power (not 5!!!). Usually we connect red wire to pin 3V3 and black wire to the ground (GND)

Setup software

The file mfrc522.py which contain library for the reader we must put to lib folder on our Pico microcontroller. We can do this with Thonny software.

The script for testing

```
from mfrc522 import MFRC522
import utime
def uidToString(uid):
    mystring = ""
    for i in uid:
        mystring = "%02X" % i + mystring
    return mystring
reader = MFRC522(spi_id=0,sck=2,miso=4,mosi=3,cs=1,rst=0)
print("")
print("Please place card on reader")
print("")
PreviousCard = [0]
try:
    while True:

        reader.init()
        (stat, tag_type) = reader.request(reader.REQIDL)
        #print('request stat:',stat,' tag_type:',tag_type)
        if stat == reader.OK:
            (stat, uid) = reader.SelectTagSN()
            if uid == PreviousCard:
                continue
            if stat == reader.OK:
                print("Card detected {}  uid={}".format(hex(int.from_bytes(bytes(uid),"little",False)).upper(),reader.tohexstring(uid)))
                defaultKey = [255,255,255,255,255,255]
                reader.MFRC522_DumpClassic1K(uid, Start=0, End=64, keyA=defaultKey)
                print("Done")
                PreviousCard = uid
            else:
                pass
        else:
            PreviousCard=[0]
        utime.sleep_ms(50)
except KeyboardInterrupt:
    pass
```

```
MicroPython v1.18 on 2022-01-17; Raspberry Pi Pico with RP2040
Type "help()" for more information.
>>> %Run -c $EDITOR_CONTENT

  Please place card on reader

  Card detected 0XFB862396  uid=[0x96, 0x23, 0x86, 0xFB]
  00 S00 B0: Authentication error
  Done
  Card detected 0XFB862396  uid=[0x96, 0x23, 0x86, 0xFB]
  00 S00 B0: Authentication error
  Done
  Card detected 0X59C01E0D  uid=[0x0D, 0x1E, 0xC0, 0x59]
  00 S00 B0: 0D 1E C0 59 8A 08 04 00 62 63 64 65 66 67 68 69   ...Y....bcdefg
  hi
  01 S00 B1: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
  ..
  02 S00 B2: 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00   ................
```

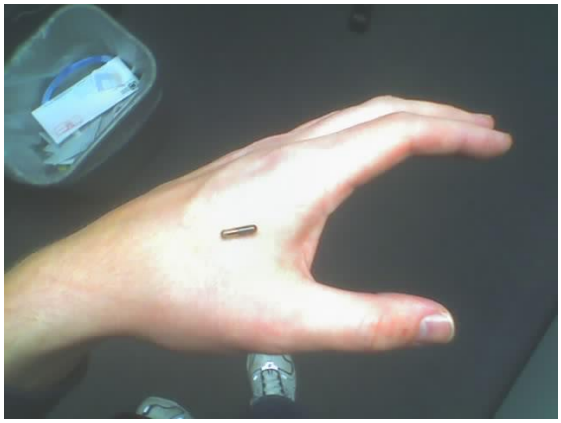
Terminal window with results of reading

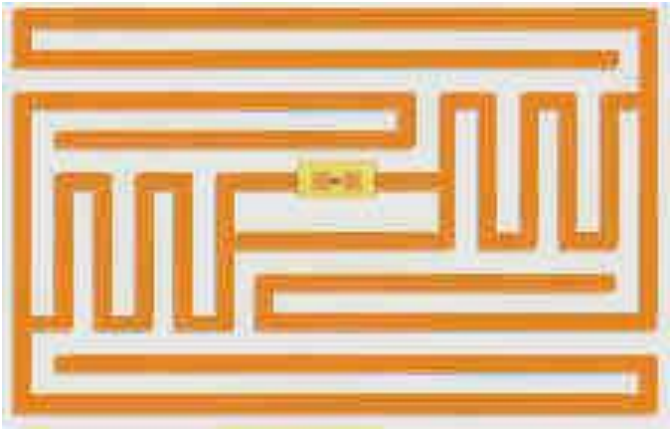
The principle of operation of RFID

RFID (Radio-frequency identification)- introduction

A technology that uses radio waves to transmit data and power the electronic circuit (RFID tag) of an object tag by a reader to identify the object. The technique makes it possible to read and sometimes also write an RFID chip. Depending on the design, it allows you to read labels from a distance of up to several dozen centimetres or several meters from the reader antenna.

The system operation is as follows:
the reader uses the transmitter antenna to generate an electromagnetic wave, the same or the second antenna receives electromagnetic waves, which are then filtered and decoded so as to read the responses of the tags.





Passive tags do not have their own power supply, when they are in the electromagnetic field with the resonant frequency of the receiving system, they collect the received energy in a capacitor contained in the tag's structure.
Upon receipt of sufficient energy, a reply containing the marker code is sent.
Popular standards (in our case 13.56 kHz compatible with the Mifare standard)
125kHz, 13.56kHz and others

image source: wikipedia.org

(Mifare also supports ATM cards, tickets)

Experiment Evaluation

System testing:
-trying to read different receivers: identification cards, tags;
-practical check of the range of tags and reader
-learning about the information sent to the reader
-checking the effectiveness of security measures against card cloning

Write down Your results

card/ tag	data sentt from card	card/tag	effective distance

Topic: Waves	Age	Country	Date
Sound level meter	>14	Poland	May 2022

Function, realisation

Construction of a circuit based on microcontroller and Python that measures the sound level

Hardware required

- Microcontroller Raspberry Pi Pico with Micropython 1.18 or above
- Cytron Maker Pico docking station for Pico
- Dfrobot Level meter module (analog 3.3V for Pico, 5V for Arduino)
- PC or Mac computer

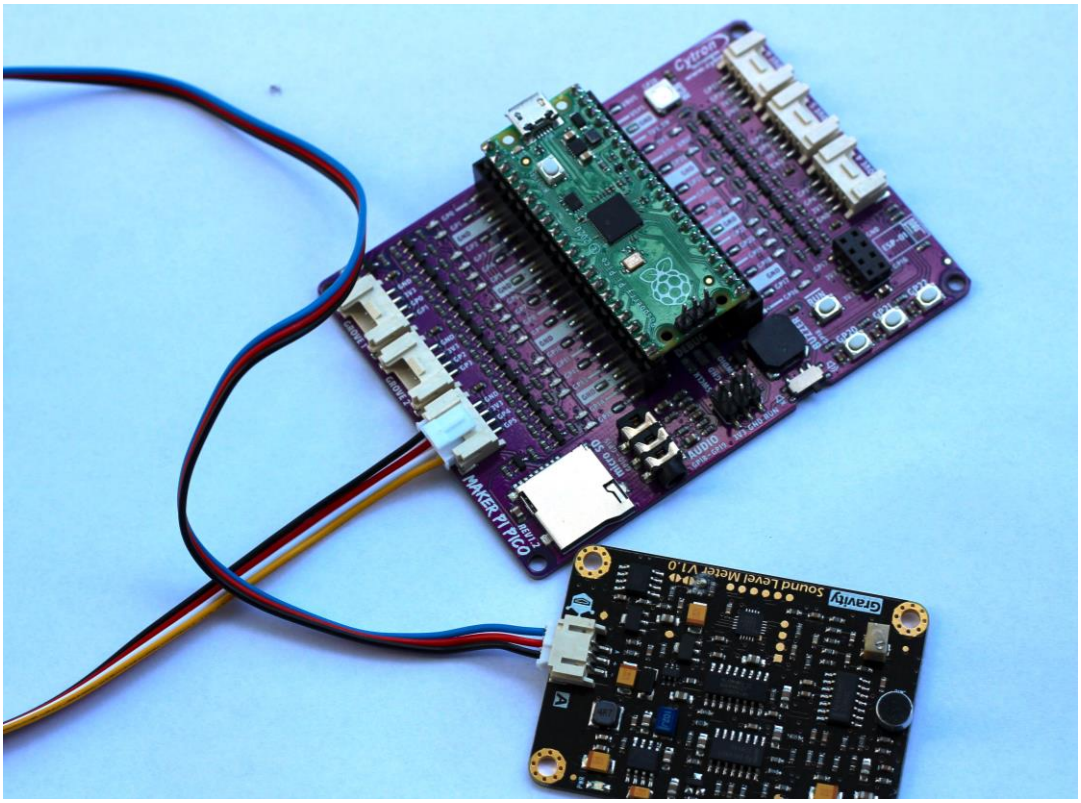
Materials required

- Grove male wire (or 3x Dupont male-male wire)
- micro usb 2.0 high speed (15cm or 30 cm)
- paper for note

Software required

-Thonny (thonny.org)
-Sound level meter doesn't require custom library for Micropython
-LCD 1602 display library:
https://files.seeedstudio.com/wiki/Grove_Shield_for_Pi_Pico_V1.0/Libraries.rar

Setup Hardware



Because **Sound Level Meter** sensor is **analog**, then we must connect them to GP26, GP27 or GP28- analog ports of Raspberry Pi Pico. In our case the best way is select GP27 or GP26 (we have grove socket on the shield Maker Pico connected to these pins). Below

Sensor pins	Raspberry Pi Pico pins	Dupont	Grove wire
signal (A)	**GP27**	blue	**yellow**
power (+)	**3V3**	red	**red**
ground (-)	**GND**	black	**black**

The Pico use 3.3V power (not 5!!!). Usually we connect red wire to pin 3V3 and black wire to the ground (GND)

Setup software

The script doesn't use special libraries, but only standard libraries from Micropython, custom libraries not required.. The script is very simple, very easy to understand, like scripts, which read data from analog sensors.

Python script for measure sound level which use Analog Digital Converter (12-bit ADC)

```
import utime
import machine
# Dfrobot Sensor SoundLevelmeter connected to ADC1 (GP27)
# other options: pin26- ADC0, pin28- ADC2
analog_value = machine.ADC(1)
#conversion_factor = 3.3/ 65535
# in Mocropython ADC can give us 65535 levels (2^16-1)
while True:
    sound_lev = analog_value.read_u16()
    print("raw:", sound_lev)
    #print(" minivolt:", minivolt)
    voltageValue = sound_lev / 65535.0
    print(voltageValue,"V")
    dbValue = voltageValue * 50.0 * 3.3
#if we use other microcontroller, which use 5V, then we must change 3.3 to 5.
#Pico: only 3.3V
    print(dbValue,"dB")
    utime.sleep(0.5)
#You can change sleep time to get better results
```

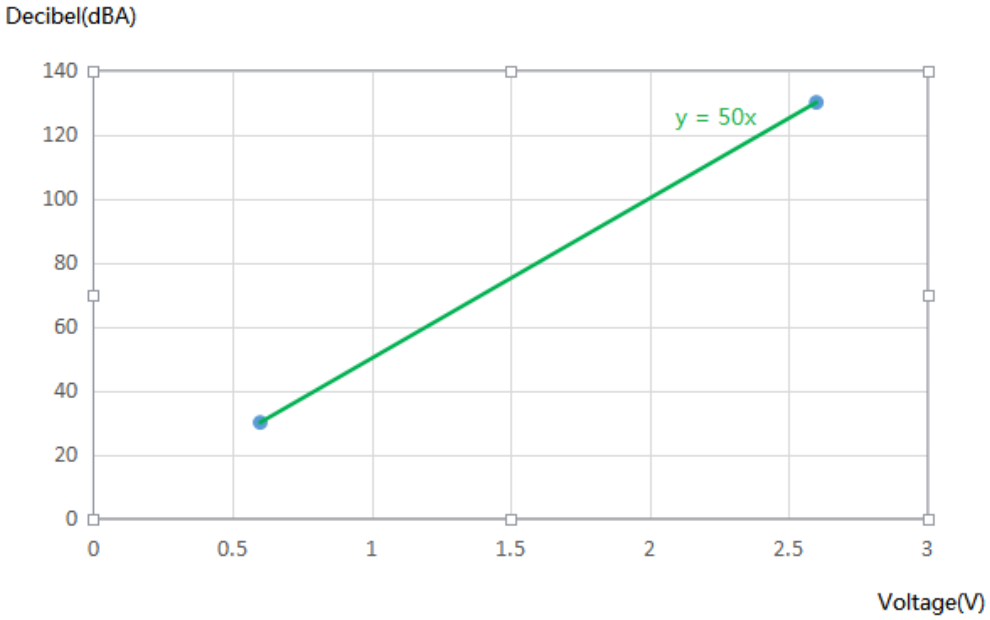
You can use Shall window, to observe results or change the script and plot results in Plotter window (View/Plotter).

The principle of operation of Sound Level Meter

Sound level meter (also known as the decibel meter, noise meter) is a basic noise measurement instrument.We have launched a sound level meter, which is compatible with Arduino, and is plug-and-play. It can accurately measure the sound level of it's surrounding environment. This product uses an instrument circuit and a low noise microphone, which makes it highly precise. It supports 3.3-5.0V wide input voltage, 0.6-2.6V voltage output. The decibel value is linear with the output voltage, which leads to a simple conversion, and therefore does not need a complex algorithm. The connector is plug-and-play so this product can be easily used in your application.

Relation between Decibel Value and Voltage Output

For this product, the decibel value is linear with the output voltage.When the output voltage is 0.6V, the decibel value should be 30dB. When the output voltage is 2.6V, the decibel value should be 130dB. The calibration is done before leaving the factory, so there is no need to calibrate it. So we can get this relation: Decibel Value(dB) = Output



Voltage(V) × 50, as shown below.

Experiment Evaluation

System testing:
-Measure the loudness level in various situations: street noise, library, school corridor, scream, musical instruments
-Compare the obtained results with the results of the mobile application
-Change the delay (sleep) and observe the effect on the results
-Change the Python script to plot graph based on results (using internal ploter from Thonny App)

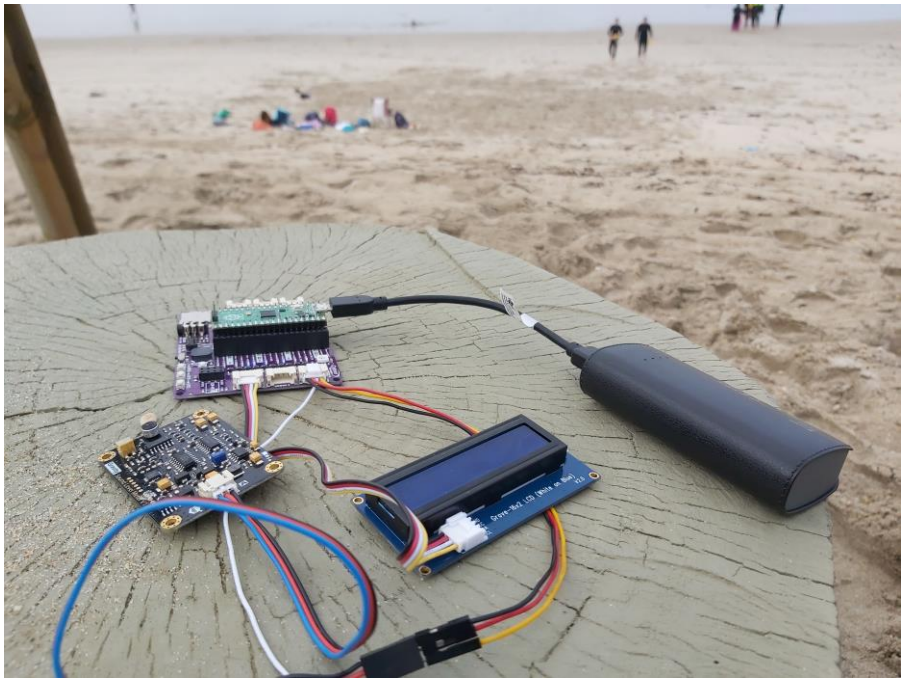
Write down Your results

situation	result (dba)	situation	result (dba)

Go further

- We can add to the script conditional instruction **IF**, which can display additional information depend on sound level,: level like on the street, level like in library, etc.
- To compare results, we can use free of charge mobile App from Google Play store: **Sound meter**.
- Supplementing the code with the operation of the LCD display
- Building a mobile version (with UPS) adding results logging to the microcontroller memory.

In the photo below, an autonomous system made by a Polish team that measures the sound level on the beach in Vila do Conde (Portugal)
The script was saved in Pico's memory under the name **main.py**.



Measure the sound level using a mobile application:

Topic: Waves Simple harmonic motion with ultrasonic sensor	Age >14	Country Poland	Date May 2022

Function, realisation

Construction of a circuit based on microcontroller and Python that measures distance from the sensor to the mass of the spring.Plotting the position in harmonic motion on the basis of indications of the distance from the ultrasonic sensor.

Hardware required

- Microcontroller Raspberry Pi Pico with Micropython 1.18
- Cytron Maker Pico docking station for Pico
- ultrasonic ranger HC-SR04P (special version for 3.3 V microcontrollers, we don't need to use resistor like in the case of standard HC-SR04)
- PC or Mac computer

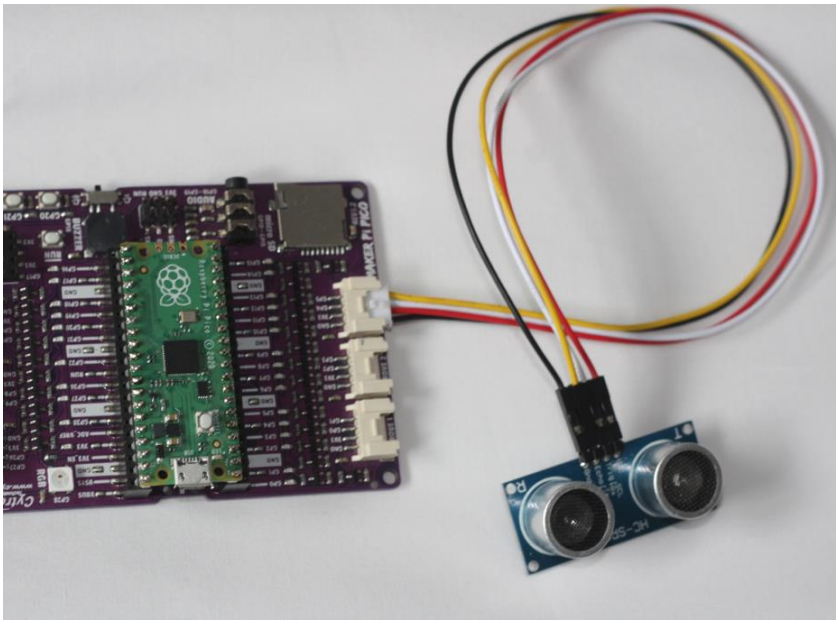
Materials required

- Grove male wire (or 4x Dupont male-male wire)
- micro usb 2.0 high speed (15cm or 30 cm)
- mass
- stand
- spring
- paper for note

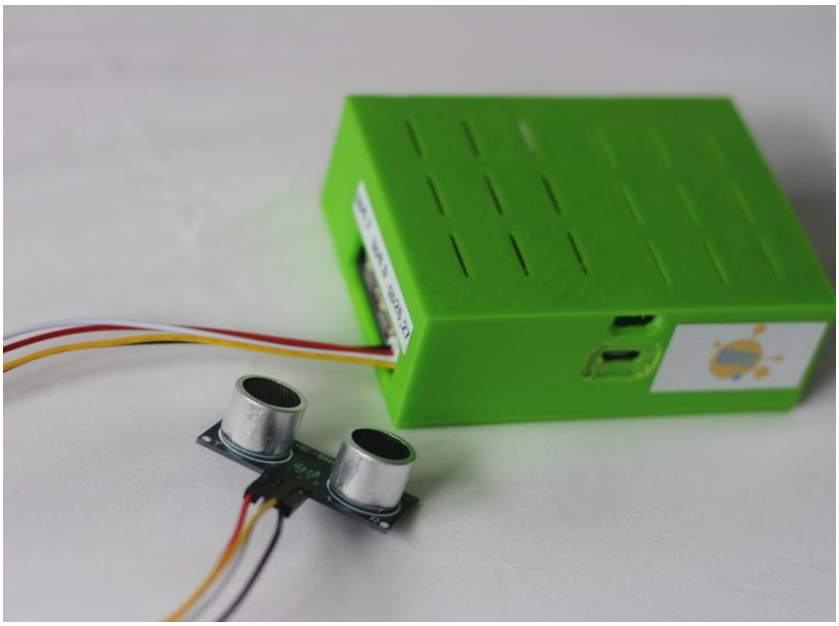
Software required

-Thonny (thonny.org)
- micropython library https://github.com/rsc1975/micropython-hcsr04

Setup Hardware



Below the same system, but with our 3D printed housing



In our Python script, we assume that the ultrasonic sensor HC-SR04P will use two signal pins: GP26, GP27

Sensor pins	Raspberry Pi Pico pins	Grove
Echo	**GP27**	**yellow**
Trig	**GP26**	**white**
power (+)	**3V3**	**red**
ground (-)	**GND**	**black**

The Pico uses 3.3V power (not 5!!!). Usually we connect red wire to pin 3V3 and black wire to the ground (GND)

Setup software

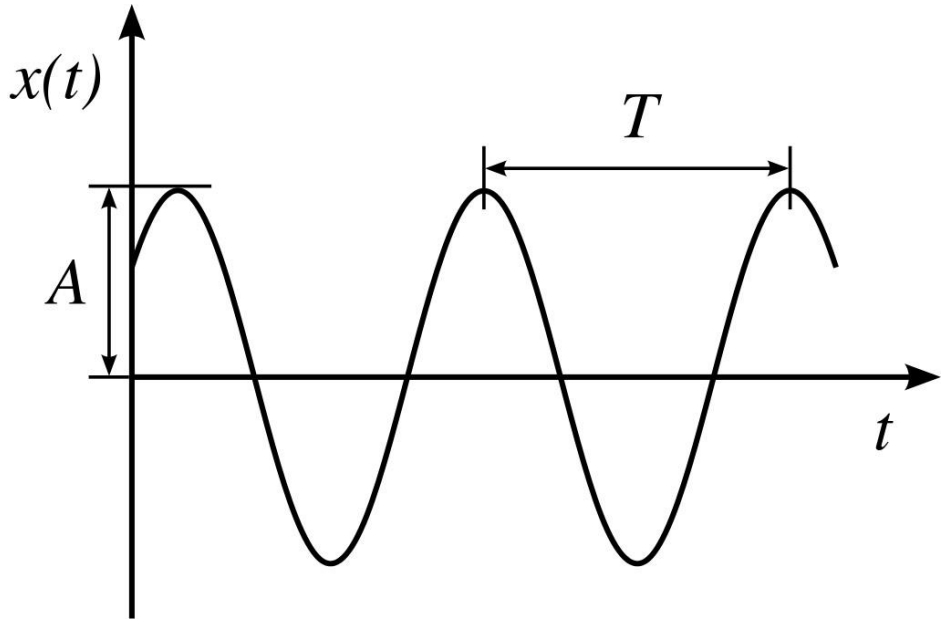
Before use of the system we must install libraries from the site: https://github.com/rsc1975/micropython-hcsr04
We must put the file hcsr04.py to folder with the name lib on our Raspberry Pi Pico. We can do this with Thonny software.
The script is very simple, very easy to understand and we can use 2 methods from library: **distance_cm** or **distance_mm**.

```
# Harmonic motion with HC-SR04P and Raspberry Pi Pico
# based on the methods from library
# https://github.com/rsc1975/micropython-hcsr04
from hcsr04 import HCSR04
from utime import sleep

sensor = HCSR04(trigger_pin=26, echo_pin=27, echo_timeout_us=1000000)
while True:
    distance = sensor.distance_cm()
    #print('Distance:', distance, 'cm')
    print(distance)
    sleep(0.05)
```

To look for the graph of distance over time, we must select from Thonny's menu **View/plotter.**

We can compare our experimental plot with the picture below



Displacement in Simple harmonic motion

Simple harmonic motion- type of periodic motion where the restoring force on the moving object is directly proportional to the magnitude of the object's displacement and acts towards the object's equilibrium (medium) position. It results in an oscillation which continues indefinitely, if

$$x(t) = A\cos(\omega t - \varphi),$$

uninhibited by friction or any other dissipation of energy.
x is its displacement from the equilibrium (or mean) position
A-amplitude.
omega= . . . . . .
fi= . . . . . .

Experiment Evaluation

System testing:

- measure displacement the mass from mean position
- comparison results of different mass and different springs
- change the delay (sleep) and observe the effect on the results

- change the Python script to plot graph based on results (using internal ploter from Thonny App)

Write down Your results:

.....

.....

.....

.....

.....

.....

.....

.....

.....

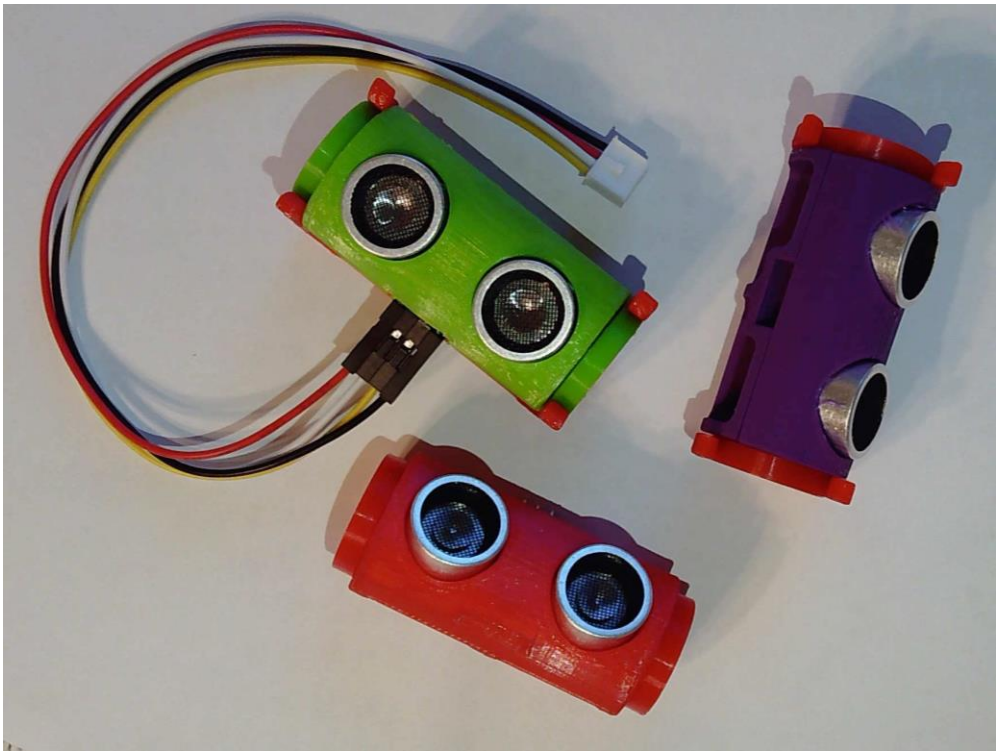
.....

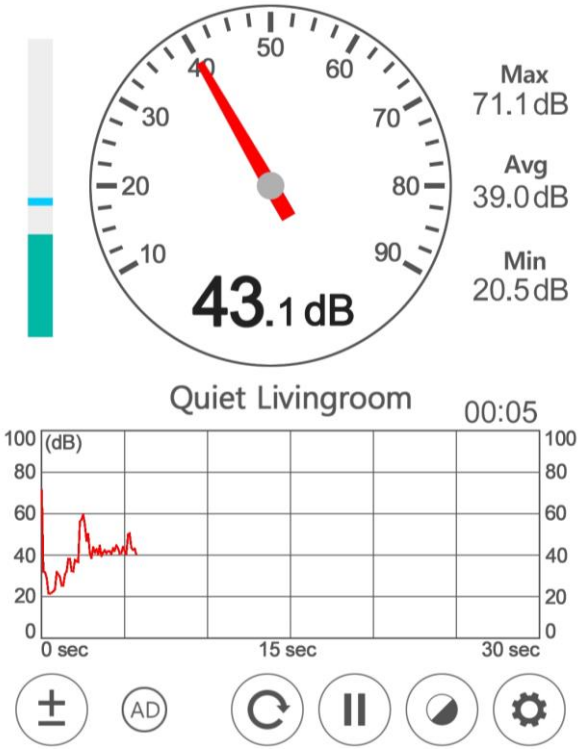
Go further

We can use this system in many other experiment and applications, e.g measure sound speed, measure distance and many other cases.
Use the circuit made to determine the spring constant elasticity

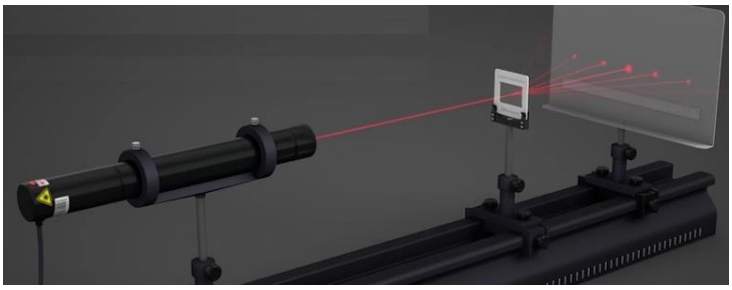
Supplementing the code with the operation of the LCD display
Building a mobile version (with UPS) adding results logging to the microcontroller memory.

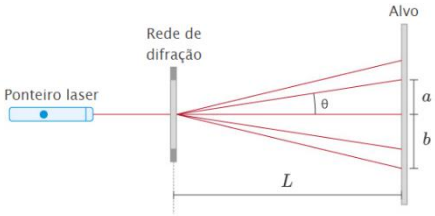
In the photo below - housings for ultrasonic distance sensors HCSR04P, printed by a Polish team on a 3D printer. The housings allow easy mounting and convenient rotation of the sensor.

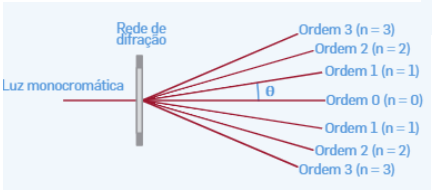



10
20
30
40
50
60
70
80
90
43.1 dB
Max
71.1 dB
Avg
39.0 dB
Min
20.5 dB
Quiet Livingroom
00:05
(dB)
100
80
60
40
20
0
0 sec
15 sec
30 sec
AD

Topic	Age	Country	Date
Estimating the thickness of a hair by light interference	≥14	Portugal	May 2022







$$n\lambda = d \sin\theta$$

Part I – Test it!

L=_____ n=_____ *a* =_____ *d*=?_____ (300 lines/mm)

$\lambda_{measured}$	$\lambda_{fabricant}$	% accuracy

Part II – Use it!

L=_____ n=_____ *a* =_____ λ=_____

d=?
(estimate the thickness of the hair)