Assembly & user guide for the PingPongPID instrument

1. Main unit for hovering the ping-pong ball

Tools:

* 3D printer
* soldering iron
* drill
* screwdrivers
* CNC mill (e.g. Shapeoko Pro) / small saw (optional)

Materials and components:

* clear extruded acrylic tube, 50.00 mm outer diameter, 2.00 mm wall thickness, 1000 mm length
* 20 × 20 cm 11.5 mm thick plywood
* filament for 3D printer (e.g. PLA)
* superglue
* shrink tubing
* lead-free solder
* fan and motor from TRESemme 2000 W fast hair dryer (or similar)
* Tenma 72-2710 programmable DC power supply 30 V 5 A including power leads
* Adafruit VL53L4CD Time of Flight Distance Sensor (~1 to 1300 mm) (Adafruit product ID: 5396)
* several JST-SH cables (Qwiic, STEMMA QT, QW/ST) (optional)
* 6 pole DIN plug, DIN 45322, 4 A, 60 V AC, male, cable mount (RS Components, 692-5481)
* 50 pack colored ping pong balls 40 mm diameter (Amazon)
* 2× Silicone cover stranded-core wire - 2 m 26AWG Red (Pi Hut 102890)
* 2× Silicone cover stranded-core wire - 2 m 26AWG Black (Pi Hut 104353)
* 2× Silicone cover stranded-core wire - 2 m 26AWG Yellow (Pi Hut 104357)
* 2× Silicone cover stranded-core wire - 2 m 26AWG Green (Pi Hut 104358)
* 4× M2 5 mm screws (flat head self-tapping)
* 4× M4 20 mm pan head screws
* 4× M4 nuts

Assembly:

* 3D-print the various pieces (base for sensor.stl, air inlet unit.stl, lower adaptor for tube.stl, upper adaptor for tube.stl, holder for fan.stl, protector for fan.stl).
* Drill out the holes in the ‘base for sensor’ and ‘air-inlet base’ pieces with a 4 mm drill. CAD images of the various parts are shown in Fig. S1.

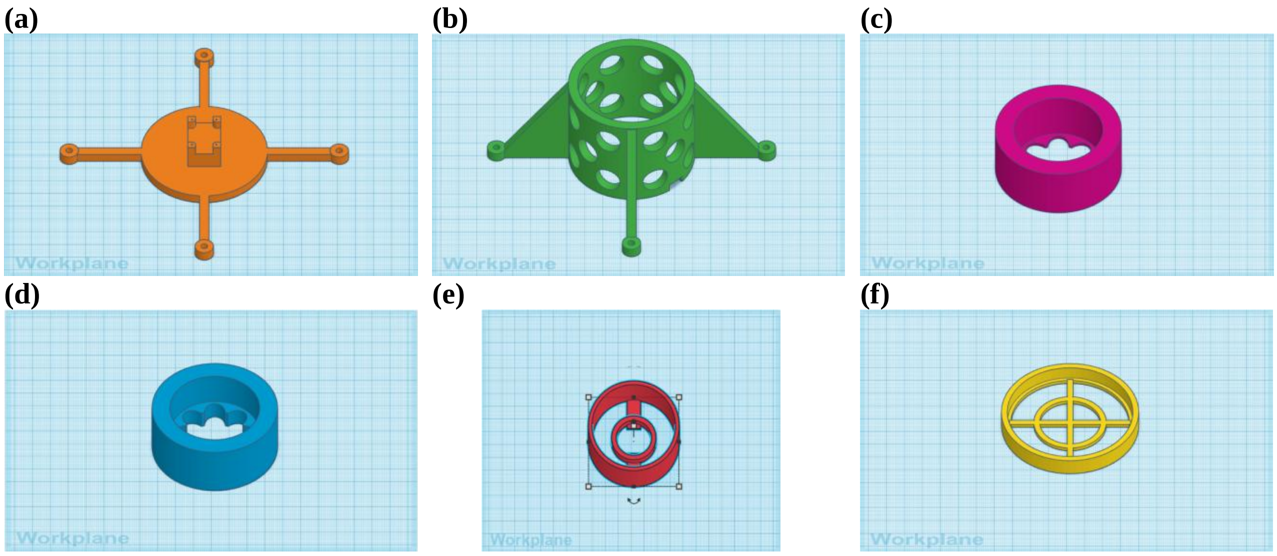

Fig. S1. CAD images of (a) base for sensor.stl, (b) air inlet unit.stl, (c) lower adaptor for tube.stl, (d) upper adaptor for tube.stl, (e) holder for fan.stl and (f) protector for fan.stl.

* CNC-mill wooden base plate (wooden base.c2d) and remove the bridges with a small saw. Alternatively, the base plate can be prepared using a saw and drill for creating the four holes needed for attaching the 3D-printed pieces. A rendering of the base plate is shown in Fig. S2.

Fig. S2. Rendering of a bottom view of the wooden base plate.

* Put the four M4 screws through the base plate from the bottom.
* Push the 3D-printed ‘base for sensor’ piece over the four M4 screws coming out of the wooden base.
* Solder red, black, yellow and green wires to distance sensor (Vin, GND, SCL, SDA).
* Alternatively, a JST-SH cable can be attached directly to the distance sensor, one end cut off and the four wires soldered to the four wires of the JST-SH cable.
* Tie the four wires together at regular intervals with shrink tube.
* Solder the four wires to a 6 pole DIN plug noting the types of wire on four of the pins. 4 or 5 pole DIN plugs could be used as well.
* Attach the distance sensor to the ‘base for sensor’ piece with 4 self-tapping M2 screws.
* Carefully remove Kapton foil from distance sensor with tweezers.
* Push ‘air inlet unit’ piece on top of the four M4 screws and secure the entire assembly with four M4 nuts. Make sure that the four wires from the sensor leave the unit through the small opening.
* Super glue the ‘lower adaptor for tube’ piece to the top of the ‘air inlet unit’ piece. The ‘crown’ part should be at the lower part of the adapter. Wait until the glue has fully hardened.
* Push the acrylic tube into the ‘lower adaptor for tube’ piece. This should be a tight fit.
* Open the hair dryer and extract the electric motor with fan.
* Solder black and red wires to the terminals of the motor. The negative terminal is the one connected to the housing of motor with a thin wire.
* Insert the motor with fan into the ‘holder for fan’ piece making sure that the cooling openings of the motor align with the holes in the holder. If a different fan is used, the ‘holder for fan’ piece may need to be redesigned.
* Tie wires together with shrink tube at regular intervals.
* Cut off croc clips from the power leads of the power supply and solder to the fan wires (red to red and black to black). Cover the soldered connections with shrink tube to prevent short circuits.
* Push the ‘protector for fan’ piece on top of ‘holder for fan’ piece which now contains the electric fan.
* Super glue the ‘upper adaptor for tube’ piece onto the bottom of ‘holder for fan’ making sure that the motor wires leave through the small opening. The ‘crown’ part should be towards the fan.
* Drop a ping-pong ball into the acrylic tube.
* Place the entire fan unit on top of the acrylic tube. This should be a comfortable fit so that it can be taken off and put back easily. Photos of the assembled base and fan units are shown in Fig. S3.

(a) (b) (c)

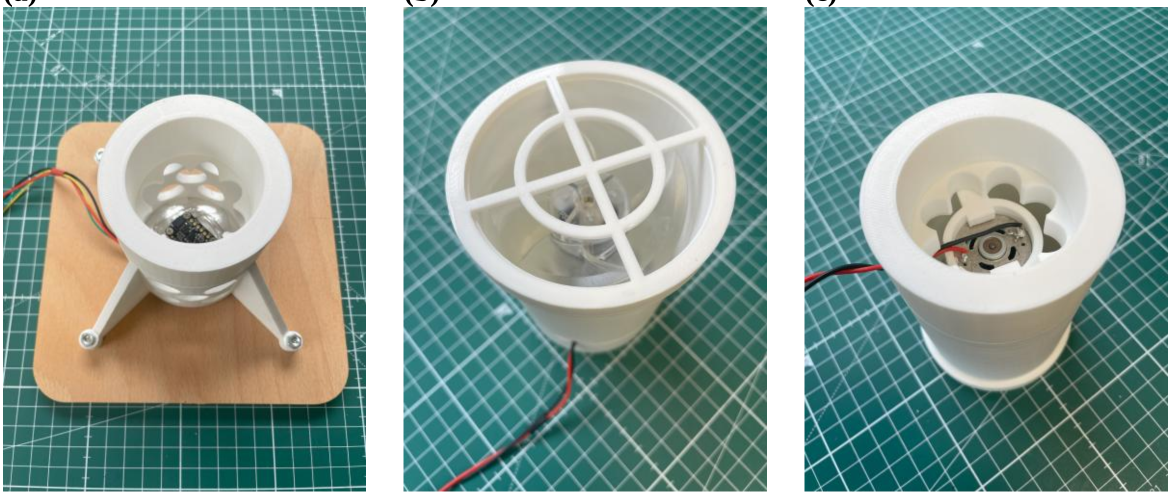

Fig. S3. Photos of (a) the complete base unit as well as (b) top and (c) bottom views of the fan unit.

2. Controller box

The controller box described here contains a few optional extras including two displays and a rotary encoder for changing the setpoint distance and starting/stopping runs. The controlling of the PingPongPID instrument and data recording can be carried out with a PC as described later.

Tools:

* soldering iron
* screwdrivers
* glue gun (optional)
* CNC mill (e.g. Shapeoko Pro) / small saw (optional)

Materials and components:

* METCASE Unicase Grey Aluminium Instrument Case, 180 × 185 × 65 mm
* 16 × 10 cm 1 cm thick PVC foam board
* Adafruit Feather M0 basic proto microcontroller (Adafruit product ID: 2772)
* 2× Adafruit 7 segment QT displays (Adafruit product ID: 881 or product ID: 880)
* Adafruit Rotary Encoder QT including knob, nut and washer for mounting (Adafruit product ID: 5880)
* DFR0077 RS232-TTL converter (DFROBOT)
* 6 Pole Din Socket, 4 A, 100 V ac, Female, Panel Mount (RS Components, 490-850)
* several JST-SH cables (Qwiic, STEMMA QT, QW/ST) (optional)
* Panel Mount USB Cable - B Female to Micro-B Male
* 1× M2.5 6 mm screw (pan head)
* 1× M2.5 nut
* 12× M2 5 mm screws (flat head self-tapping)
* 2× TE connectivity screw lock for use with D-Sub connector
* 2× 4-40 UNC imperial hexagon nuts (ANSI B18.6.3)
* filament for 3D printer (e.g. PLA)
* solid-core 22 AWG wires in different colours
* shrink tubing
* lead-free solder

Assembly:

* Assemble the instrument case following the instructions of the manufacturer.
* 3D-print ‘microcontroller base.stl’.
* Drill 2.5 mm hole at the center of the bottom of the instrument case.
* Attach ‘microcontroller base’ piece to instrument case with M2.5 6 mm screw and nut.
* CNC-mill a spacer for the two displays using the PVC foam board (box front spacer.c2d). The foam board was attached to the spoil board with small wood screws. Remove the bridges with a small saw.
* Bridge the A0 address pad on the back of one of the displays with solder. This changes its hardware address which means that both displays can be controlled independently by the microcontroller.

* Secure both displays with 4 self-tapping M2 5 mm screws each onto the spacer.
* Solder rotary encoder to its circuit board.
* Solder wires inside the controller box keeping approximate distances in mind. Connect from 3.3V, GND, SCL and SDA on the microcontroller to Vin, GND, SCL and SDA on the two displays and the rotary encoder. For convenience, JST-SH cables can be used to connect the displays and the rotary encoder.
* CNC-mill the openings of the front panel using 'box front.c2d'. Using the existing four holes, we attached the panel to the spoil board of the CNC machine with small wood screws. Alternatively, the panel openings can be cut out with a drill and metal saw.
* Hot-glue the PVC foam spacer to the back of the front panel making sure that the displays are not upside down.
* Attach rotary encoder to front panel using the washer and nut provided. Attach the plastic knob to the shaft of the rotary encoder at the front panel.
* Attach the entire front panel to the instrument case.
* CNC-mill the openings of the back panel using 'box back.c2d'. Using the existing four holes, we attached the panel to the spoil board with small wood screws. Alternatively, the panel openings can be cut out with a drill and a metal saw. Rendering of the three CNC-milled parts used for this part are shown in Fig. S4.

(a) **(b)** **(c)**

Fig. S4. Renderings of the CNC-milled parts including (a) the PVC foam spacer for the displays, (b) the aluminum front and (c) back panels.

* Attach the USB panel mount to the back panel using the screws provided. We had to use two additional M2.5 nuts to secure the panel mount because the threads were worn.
* Use two D-Sub connectivity screw locks and imperial hexagon nuts to attach the RS232-TTL converter to the inside of the back panel.
* Mount 6 pole DIN socket to the back panel and secure with the nut provided.
* Solder connections to the back panel. Ensure that the pins on the DIN socket correspond to the ones on the plug coming from the distance sensor. Ultimately, connect from 3.3V, GND, SCL and SDA on the microcontroller to Vin, GND, SCL and SDA on the distance sensor. Also, solder from 3.3V, GND, TX1 and RX0 on the microcontroller to VCC, GND, TXD and RXD on the RS232-TTL converter. Finally, connect the micro-USB plug from the USB panel mount to the microcontroller.
* Secure the microcontroller onto the 3D-printed base with 4 self-tapping M2 5 mm screws.
* Close the controller box with the top lid and secure with screws. Photos of the complete controller box are shown in Fig. S5.

(a) **(b)** **(c)**

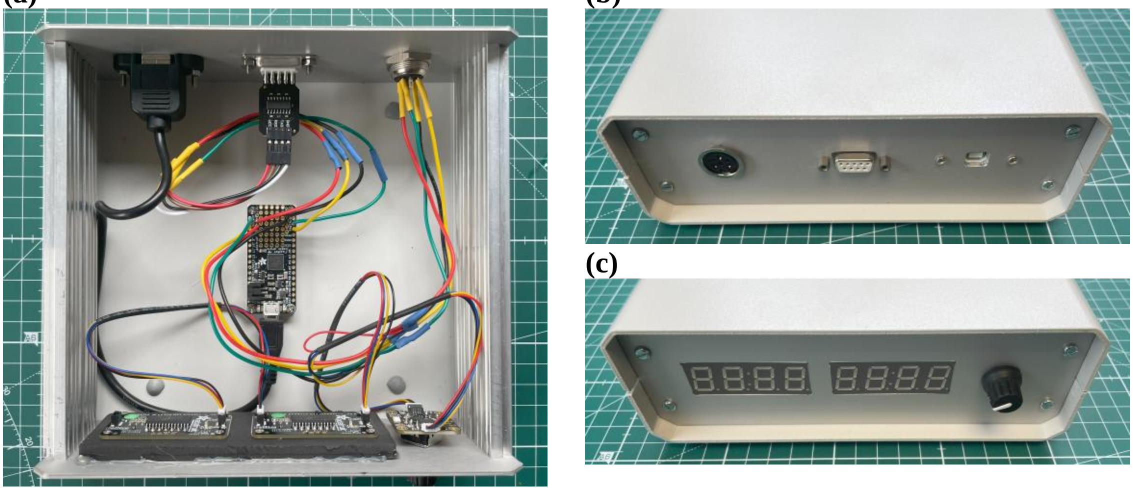

Fig. S5. Photos of the complete controller box including (a) a top view of the open box as well as (b) back and (c) front views of the closed controller box.

3. General assembly and software upload

Materials and components:

* 1.5 m RS232 serial male to male null modem cable (crossover for data communication)
* USB A to B cable

Assembly:

* Connect power supply to mains electricity.
* Connect power supply with controller box using the RS232 cable.
* Connect a computer with the controller box using the USB cable.
* Connect cable from main unit with the 6 pole DIN plug of the controller box. A photo of the complete setup is shown in Fig. S6.

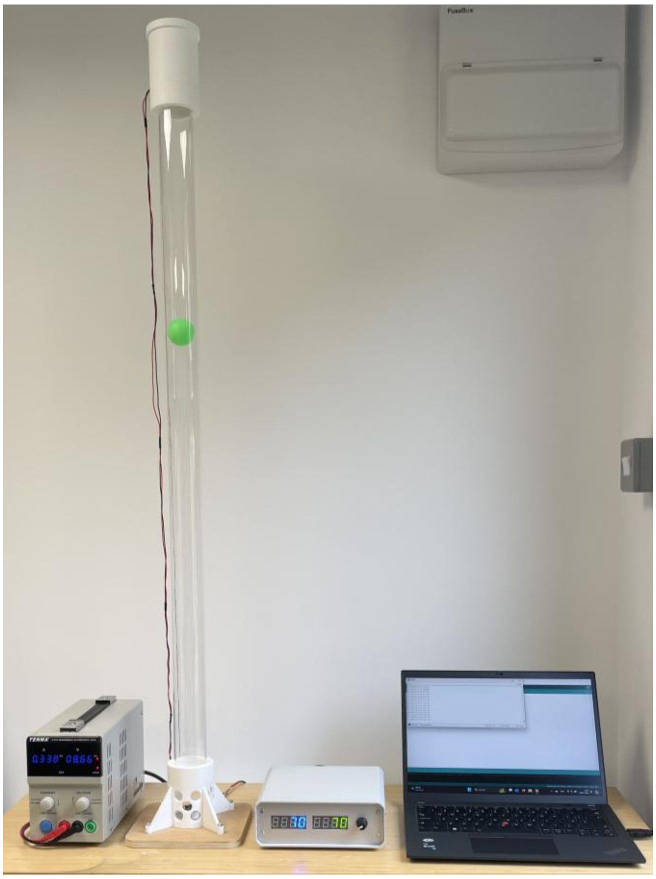

Fig. S6. Photo of the complete PingPongPID setup.

Software upload:

* Install the Arduino IDE on your computer (https://www.arduino.cc/en/software). We note that the version at the time of writing this document (2.3.2) has a problem with the Serial Monitor which will affect copying the data into other programs. The data shown in the main paper was recorded with the legacy version 1.8.19.
* Follow the installation instructions for the Adafruit Feather M0 microcontroller at https://learn.adafruit.com/adafruit-feather-m0-basic-proto/. Please note that other microcontrollers will probably work as well. The Feather M0 microcontroller is the only one tested by us.
* Install the library for the LED displays as described here: https://learn.adafruit.com/adafruit-led-backpack/
* Install the library for the rotary encoder as described here: https://learn.adafruit.com/adafruit-i2c-qt-rotary-encoder
* Install the library for the distance sensor as described here: https://learn.adafruit.com/adafruit-vl53l4cd-time-of-flight-distance-sensor
* Install the PID library (Brett Beauregard) as described here: https://www.arduino.cc/reference/en/libraries/pid/

* Download the zip file from our GitHub repository: https://github.com/UCL-salzmann/PingPongPID.git
* Open ‘PingPongPID.ino’ in the Arduino IDE.
* Make sure you define appropriate default parameters for the PID controller starting from line 105 including in particular the minimum (Vmin) and maximum voltages (Vmax) of your fan if you have used a different one compared to the one suggested above. Vmin should be defined somewhat below the voltage required for ‘take-off’ of the ping-pong ball. Applying Vmax to the fan should result in a rise time of about 1 or 2 seconds of the ping-pong ball from the bottom to the top of the tube.
* Select the correct COM port of the microcontroller in the Arduino IDE and compile/upload the program to the microcontroller.
* The PingPongPID instrument can be controlled either using line commands in the Arduino IDE or our Python graphical user interface (GUI). If you wish to use the Python GUI, close the Arduino IDE so that the COM port is freed and move on to chapter 5.

4. Running the PingPongPID setup with the Arduino IDE

* Open the Serial Monitor in the Arduino IDE under Tools. Select the ‘115200 baud’ rate and ‘Both NL & CR’.
* Make sure all cables at the back of the controller box are connected, the power supply is switched on and a ping-pong ball has been dropped into the tube of the main unit.
* Using the Serial Monitor, the following serial commands are possible:
  @BH XX sets the baseline distance in cm. Values between 0 and 100 cm are meaningful for XX.
  @KP XX sets the Kp constant as XX.
  @KI XX sets the Ki constant as XX.
  @KD XX sets the Kd constant as XX.
  @TP XX sets the time period in seconds of the oscillation as XX
  @AMP sets the amplitude in cm of oscillation as XX
  @PM returns the current values of Kp, Ki, Kd, baseline height, amplitude, and time period.
  @CS checks the distance sensor and returns its status.
  @DI returns the current distance in cm.
  @GO starts a new run based on the defined PID constants. During a run, the runtime at about 200 millisecond intervals, the current distance in cm, the setpoint distance in cm and the current output voltage will be returned as semicolon-separated values.
  @STOP stops an active run.
* The recorded values from the Serial Monitor can be saved as a CSV file with copy/paste and then opened with, for example, Microsoft Excel. The data can be copied directly into Origin from OriginLab using copy/paste.
* Before a new run, delete the data in the Serial Monitor with ‘Clear output’.
* For finding the optimal PID constants, please see the discussion in the main article.
* In addition to controlling the PingPongPID with the Serial Monitor, the knob at the front panel of the controller box can be used to change the setpoint distance by turning. Runs can be started or stopped by pushing the knob.

5. Running the PingPongPID setup with a Python graphical user interface

* Make sure all cables at the back of the controller box are connected, the power supply is switched on and a ping-pong ball has been dropped into the tube of the main unit.
* If Python is not already installed on your computer, download and install it from https://www.python.org/. During installation, make sure to check “Add Python to PATH”. If you already have Python installed but it is not in your PATH see the steps further below to add python to path manually.

* To check the correct installation of Python, open the command-shell of Windows by clicking the Windows Key + ‘R’, type ‘cmd’ and press enter. In the command prompt, run the following command:

```
python --version
```

If Python is correctly installed, the Python version will be returned.

* Run the following commands to install the pySerial, tk and matplotlib libraries:

```
pip install pyserial
```

```
pip install matplotlib
```

```
pip install tk
```

* To establish the serial port communication with the PingPongPID instrument, you must find out to which COM port your USB device is connected. To find this, you can use the Windows Device Manager. Click on the Windows Button and type ‘Device Manager’. Then, scroll to the COM ports drop-down menu and all the ports in use will be listed. You can identify which one is assigned to the PingPongPID instrument by disconnecting and reconnecting it. Take a note of the number of the COM port.
* Save ‘pingpongGUI.py’ from our GitHub repository (see above) on your computer.
* If the voltage of your fan is different from the one recommended above, you need to change this in the code in the “Graph plotting and serial data” section of the code using a text editor. Change the minimum and maximum voltages to match those of your fan:

```
ay.set_xlim(0, 60)
ay.set_ylim(5,13) #ADJUST VOLTAGE HERE
ay.set_title("Voltage vs Time")
ay.legend()
```

* Start the Python program by double-clicking on ‘pingpongGUI.py’.
* Check for any errors in the command prompt.

<u>Using the graphical user interface (GUI):</u>

* On the first tab ‘Connect COM port’, you will see all the COM ports available on your computer. You must select the correct port as determined earlier before running any experiment. Screenshots of the various tabs of the GUI are shown in Fig. S7.
* The ‘Parameters’ tab allows you to change the setpoint-distance parameters and PID constants of the instrument. The ‘Baseline Height’ slider defines the setpoint distance of the ping-pong ball. The setpoint distance can be changed according to a sine wave by defining an ‘Amplitude’ in centimeters and a ‘Period’ in seconds using the two sliders on the right. The K_p, K_i and K_d constant can be set by entering the numeric values. The various update buttons send the individual parameters to the PingPongPID instrument. These update buttons are only needed if the parameters are changed while the instrument runs.
* Upon pressing the ‘GO’ button on either the ‘Data plots’ or ‘Serial data’ tab, all parameters will first be sent and the PingPongPID instrument will start afterwards.
* On the ‘Data plots’ tab, the data will be plotted in real time including the actual and setpoint distances in the left plot and the fan voltage in the right plot. Each plot will show data from the last 60 seconds.

* On the ‘Serial data’ tab, all the data sent from the PingPongPID instrument will be recorded. The data will remain in the text window after stopping a run with ‘STOP’. However, when a new run is started, the content of the text window will be reset. Between runs, the data can be saved using the ‘SAVE’ button which will direct you a folder to save the data as a CSV file. At the bottom of the data, the used parameters for the run in question will also be written. The data can also be copied directly from the text window using copy/paste.

<u>Adding Python to PATH (if needed):</u>

* If you have installed Python, but did not add to path, you will have to modify the System Environment Variables in Windows. First locate the path of your python file by running this command-on-command prompt:

```
where python
```

* The system will display the path where Python is located and may look something like:
  Copy this line, but do not include \python.exe.

```
C:\Users\uccacpo\AppData\Local\Programs\Python\Python38-32\python.exe
```

* Using the windows search tool, open the ‘Edit Systems Environment Variables’ control panel. Click on Environment Variables and navigate to the ‘Path’ variable under the ‘System Variables’ header. Open and click ‘New’ to add a new variable. In the box paste the path copied from the command prompt previously. Add another new variable, paste the path and add ‘\Scripts’. It may look like this:

```
C:\\Users\\User\\AppData\\Local\\Programs\\Python\\Python38-32\\
C:\\Users\\User\\AppData\\Local\\Programs\\Python\\Python38-32\\Scripts\\
```

Make sure to save these changes. Now Python has been added to path and you should be able to open the code by double-clicking on it.

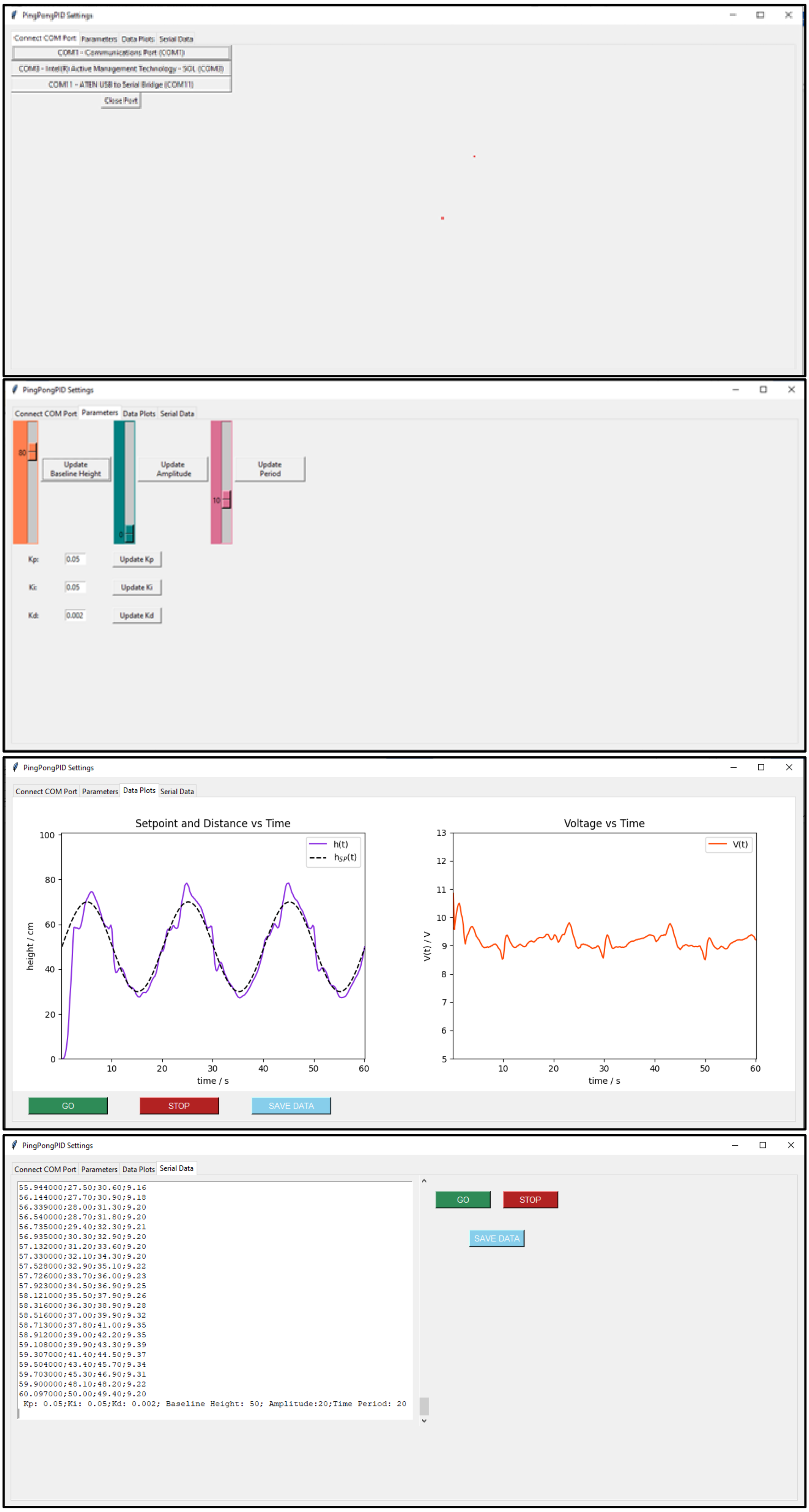

Fig. S7. Screenshots of the various tabs in the graphical user interface (GUI).