HardwareX 9 (2021) e00199

Contents lists available at ScienceDirect

HardwareX

journal homepage: www.elsevier.com/locate/ohx

Hardware Article

Open-hardware wireless controller and 3D-printed pumps for efficient liquid manipulation

Alain Gervasi [a], Pierre Cardol [b,*], Patrick E. Meyer [c,*]

[a] Genetics and Physiology of Microalgae, InBios/Phytosystems, BotaBotLab, Institut de Botanique, University of Liège, Belgium
[b] Genetics and Physiology of Microalgae, InBios/Phytosystems, Institut de botanique, University of Liège, Belgium
[c] Bioinformatics and Systems Biology Lab, InBios/Phytosystems, Institut de botanique, University of Liège, Belgium

ARTICLE INFO

Article history:
Received 30 December 2020
Received in revised form 12 April 2021
Accepted 3 May 2021

Keywords:
Automation
MQTT
ESP32
Node-RED
3D-Printing
Liquid handling

ABSTRACT

Many routines in biological experiments require the precise handling of liquid volumes in the range of microliters up to liters. In this paper, we describe a new wireless controller that is adapted to liquid manipulation tasks, in particular when combined with the proposed 3D-printed pumps. It can be built from widely available electronic components and managed with open-source software. The use of peristaltic pumps enables to move volumes from milliliters to liters with a relative error below 1% or a syringe pump capable of injecting volumes in the range of milliliters with microliter accuracy. The system is remotely controllable over WiFi and easily automated using the MQTT communication protocol. The programming of the microcontroller is performed on the Arduino IDE. The WiFi settings and the calibration value can be easily modified, stored and exported in the form of a JSON file to create a user friendly, plug and play and easily scalable device. Additional sensors or actuators can be added, allowing the system to adapt to various usages. Finally, in addition to its low manufacturing cost and its capability to fit a large variety of tasks involving liquid handling, our system has been specifically designed for research environments where adaptability and repeatability of experiments is essential.

Specifications table

Hardware name	MQTT modular pump controller
Subject area	• Biological Sciences (e.g. Microbiology and Biochemistry) • Educational Tools and open-source Alternatives to Existing Infrastructure • General
Hardware type	Biological sample handling and preparation
Open-source License	CC BY-NC-SA 4.0
Cost of Hardware	50€ for one control board two peristaltic pumps and one syringe + 15-25€ for the server (one server can controls multiple boards)
Source File Repository	https://doi.org/10.17605/OSF.IO/ZP2S6

* Corresponding authors.
E-mail addresses: A.gervasi@uliege.be (A. Gervasi), Pierre.Cardol@uliege.be (P. Cardol), Patrick.Meyer@uliege.be (P.E. Meyer).

https://doi.org/10.1016/j.ohx.2021.e00199
2468-0672/

Hardware in context

3D Printing

3D printing is a type of additive manufacturing that consists of adding successive layers of material on top of each other in order to create a 3D object. The various existing technologies are mostly differentiated by the nature of the raw materials used as well as the method employed to fuse the successive layers. We used the FDM technology as recent developments and emergence of open-source alternatives have reduced the cost of 3D printers, thereby expanding its applications in schools, homes and laboratories [1,2]. Previously used for prototyping, this technology is mature enough to be used as a full-fledged manufacturing tool. Although FDM is slower and more expensive for large scale items compared to other technologies such as injection molding, it is perfectly suited for applications that require constant modifications and adjustments, and to produce custom made parts whether it is for manufacturing conventional labware [3–5] or building advanced tools and automation equipment [6,7]. All the parts used in our system that do not require to be sterilized are 3D printed. Virtually any materials (PLA, PETG, ABS.) and 3D printer brands can be used to reproduce our system since the small imperfections introduced by the manufacturing process will be minimized by the calibration. Subtractive manufacturing, such as CNC milling or laser cutting, could also be employed in order to achieve a higher accuracy.

Pipette, syringe and peristaltic pumps

A syringe pump is a system composed of a syringe and a linear actuator that pushes the syringe piston in order to deliver a definite amount of liquid at a controlled flow rate [8]. Many commercial systems exist with various accuracies, reliabilities and capacities that are guided by the field of use of the pump (medial, chemistry, microfluidic, biology). Medical syringe infusion pumps that are designed to inject intravenously fluids, drugs, or nutrients require active pressure feedback and other patient safety features [9] and generally offer an accuracy on the microliter range with capacities of 5–60 ml [10]. Designed for industrial, microfluidic, chemical or automation applications, syringe pumps have a simpler design, typically have 0.5 μl to 100 ml capacity and could reach a picoliter accuracy [11]. Peristaltic pumps on the other hand, are positive displacement pumps that achieve the pumping action by cyclically squeezing (through a circular or linear motion) a flexible tube [12] containing a fluid against a rigid housing [13]. As the liquid is never in direct contact with the pump's mechanism and is moved in a close system, this type of pump is perfectly adapted to transfer sterile or hazardous liquid with volumes ranging from milliliter to any desired amount with typical relative accuracy around 0.5% [14]. Peristaltic pumps allow a continuous operation and can transport large volumes at the cost of a pulsating flow and the need of calibration [15] contrary to the syringe pumps that are limited to a fixed volume but can inject a constant flow of liquid. Finally, the most versatile and commonly used liquid handling tool is the pipette. This tool relies on the movement of a piston that allows to aspirate and then dispense a precise amount of liquid ranging from a few tenths of a microliter to a few tens of milliliters with a typical accuracy below 3% [16].

Open-source and DIY automation

Syringe pumps, peristaltic pumps and micropipettes are the main liquid handling systems that allow automating tasks while offering a high repeatability. Professional systems focused on medical application and microfluidic can easily exceed a few thousand euros. Although these systems offer excellent accuracy and build quality, they mainly rely on proprietary hardware and software. Open-source and do it yourself (DIY) systems may offer a higher flexibility for a lower price. Such devices are not only tens to hundreds of times less expensive than their professional commercial equivalent but keep an acceptable precision and accuracy. Their source files, schematic and code are also publicly available for anyone to use, redesign, and resell [17,18]. One of the simplest syringe pump designs that could easily be 3D printed consist of a rigid support that maintains the syringe and a linear actuator, such as a threaded rod rotated by a motor. The linear motion of the pusher will be defined by the following formula: Linear displacement (mm) = (thread/mm) $\times$ Rotation. Peristaltic pumps can also be 3D printed and only require a motor and ball bearing. Other laboratories have already developed with success open-source peristaltic pumps [19,20] and syringe pumps [21–25] specifically designed to perform their experiments, as well as designed open-source pipettes [26–28]. Advanced devices also implement these technologies in order to conduct autonomous experiments [29–44].

Hardware description

Implementation and design

The system is composed of a control board that drives three motors used to handle liquids via a peristaltic pump system or a syringe pusher (Fig. 1). The electronic components have been specifically chosen for their simplicity, adaptability and wide availability. "Non-standard" components such as pump parts or syringe support are 3D printed.

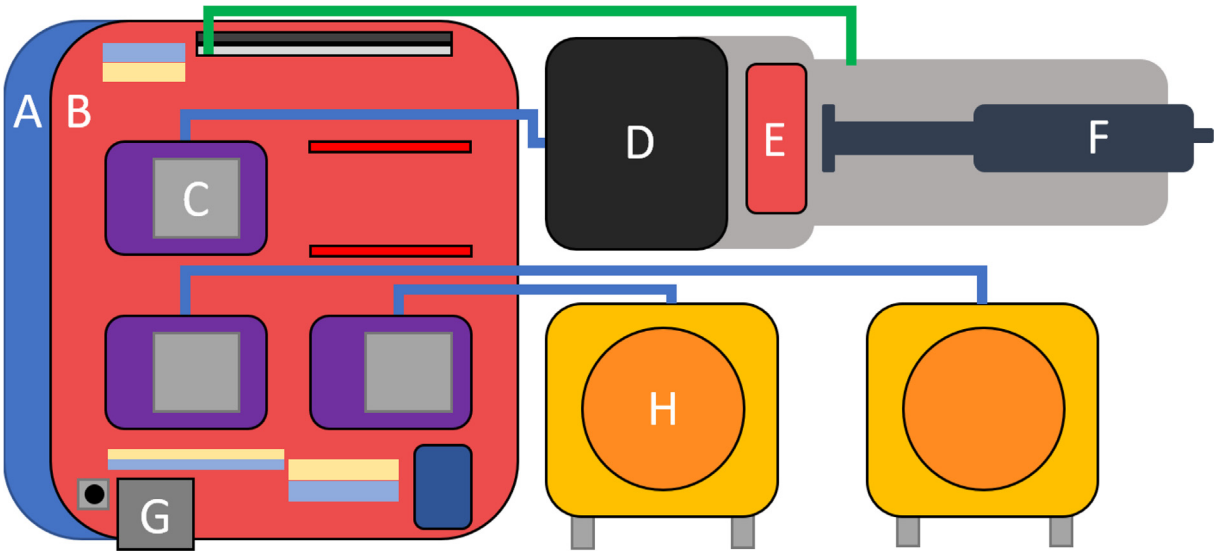

Fig. 1. Overall implementation. (A) ESP32 UNO connected to an Arduino UNO CNC shield (B) including 3 Polulu DRV8825 stepper motor drivers (C). The 3 drivers control the NEMA17 stepper motors (D) of the peristaltic pumps (H) or the syringe injectors (F). The ESP 32 and the shield are powered by a 12 V 1.5A power supply via a 5.5 mm DC jack connector (G). The pumps and syringe are calibrated using a limit switch (E).

DIY micropipettes are difficult to control in an automated way because those require 3D movements in addition to the suction/injection action. As a result, our system focuses on peristaltic pumps and syringe pumps. Furthermore, both pumps are able to handle large amounts of liquid thereby making them adapted to a wider range of lab tasks. However, instead of focusing on one specific application, we decided to develop a general-purpose design that put the accent on accessibility and adaptability. This makes it ideally suited for the rapid and intuitive implementation of experiments while being easily scalable. Users can choose between a peristaltic pump or a syringe pump depending on their needs and are also free to use a wide variety of tubing diameters and syringe capacities. The required hardware is low-cost and can easily be found on any electronic store. Our system (a WiFi control board, two peristaltic pumps and a syringe injector) can be built at a very competitive price using 3D printing for example, the 3D models of the pumps used in our specific setting are provided below. We have taken care to select the most user-friendly components and software to facilitate the integration of our machine in a wide range of research applications (i.e., biology, chemistry...).

For example, our wireless controller together with the 3D printed pumps can be used to sample aliquots, fill multi-well plates, add reagents to timely initiate a chemical reaction or to automatically prepare culture media by mixing all the required solutions with peristaltic pumps (if the volume is in the range of hundreds of milliliters) and syringe pump (to add oligoelements, vitamins or adjust the pH). Fundamentally, our device is suited to any task requiring a liquid handling step that can be automated. It could also be coupled with other machines such as a bioreactor in order to carry out a continuous culture. Furthermore, we used our pump system combined with a DIY turbidimeter to successfully build an automated phototurbidostat able to automatically monitor and adjust the turbidity of microalgae culture [45].

Microcontroller

We used a chip made by Espressif, the ESP32 [46]. The ESP32 microcontroller is cheaper than the well-known Arduino UNO while being more powerful with two cores clocked at 160-240Mhz, 520 KB of SRAM and up to 4 MB of external SPI RAM and plenty of peripheral interface (ADC, DAC, touch sensor, SPI, I^2S, I^2C, UART, CAN, PWM, hall effect sensor) and WiFi/BLE capabilities allowing to have a cheap, powerful, modular and remotely controllable system. Moreover, the ESP32 can be programmed in the Arduino IDE and is compatible with the majority of code and libraries made for the Arduino boards.

For our device we will only use the following functionalities of the ESP32:

- Digital output: to control the stepper motors and the debug LEDs.
- Digital input: to read the limit switch.
- WiFi: to control the board with the MQTT protocol.
- SPIFFS memory: to store the WiFi and calibration settings.
- UART: to control the board over a serial communication.

Other GPIO (touch sensors or analog input) or communication protocols (SPI, I^2C, CAN) can be used to add sensors to improve the board and create new functionalities (see chapter "reuse potential and adaptability")

MQTT

MQTT (Message Queuing Telemetry Transport) is a publish/subscribe machine to machine messaging invented by IBM [47]. Initially designed to retrieve information from resource constrained devices (like as microcontroller) over an unreliable, high latency and low bandwidth connection (satellite connection for instance) [48,49] this protocol is nowadays mainly used for IOT (Internet Of Things) applications where these constraints still apply. We decided to use the MQTT protocol for several reasons:

- Lightweight: in terms of computing resources as well as bandwidth consumption.
- Bi-directional: it allows to send commands to the device and receive debugging information.
- Scalable: a virtually infinite amount of devices (up to millions) can be connected to the same broker.
- Secured: by encryption of messages using TLS.
- Reliable: it offers three levels of QoS (Quality of Service).
- Simple: the topic/payload system allows to easily connect and control many robots in parallel or individually or to hierarchically sort the tools.
- Versatile: it is compatible with the ESP32 microcontroller used to build the board, and with almost all programming languages: python, C, C++, java, JavaScript, lua, ruby, rust, ...

To further facilitate the use of the device, we developed an intuitive set of commands inspired by the GCODE syntax [50]. For instance, the "X10″ command will allow the injection of 10 ml using the tool connected to the "X" slot. Error and info messages are also sent in MQTT as well as in Serial (to facilitate debugging in case of problems with the MQTT/WiFi connection).

A typical use of the device is schematized in the Fig. 2 and require four components:

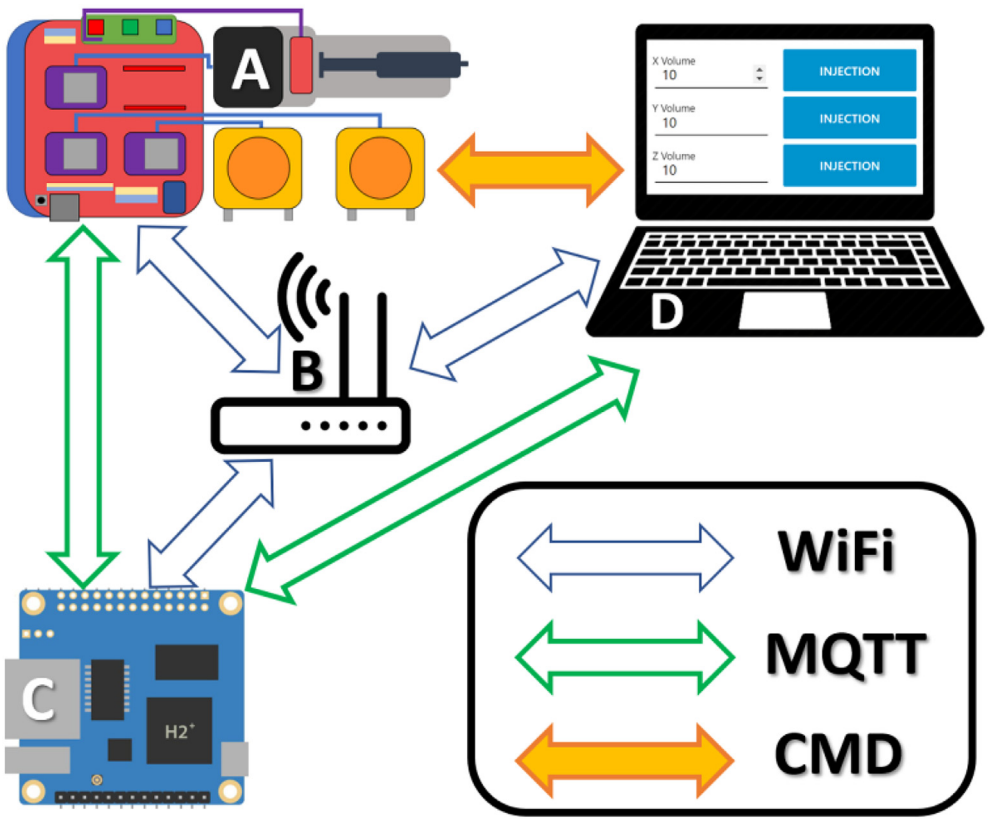

Fig. 2. Typical wireless connection between one controller (client), an MQTT broker (server) and a computer (sender). All devices must be connected to the same network in order to operate. The injector (A), the server (C) and client (D) are thus connected on the same router (B). The blue arrows represent the WIFI connections. An Orange pi Zero is used as an MQTT broker and also creates a graphical user interface (D). When a command is entered on the graphical interface, it is sent to the router, relayed to the MQTT broker before being sent to the injector via the router. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

- A controller board (A on the Fig. 2): the MQTT client that will subscribe to specific MQTT topics and control the motors.
- A WiFi router (or a WiFi device ins "access point" mode): to exchange the MQTT messages among the devices.
- A computer, smartphone or automation software (D on the Fig. 2): the MQTT publisher that will send the command (payload) to a specific tool on a control board (the topic).
- A server (C on the Fig. 2): the MQTT broker that makes the link between the publisher and the subscriber (here we used an Orange Pi zero or a Raspberry Pi but any Linux machine could perform this task).

In most company or institution networks some ports might be blocked and will prevent the MQTT (that runs on port 1883 by default) to operate. It might be necessary to use a router for which the user has full access to the administrator/firewall settings, to create a new local subnetwork, use a VPN or create a simple access point using an ESP8266/32, computer or smartphone. An internet connection is not mandatory and the unique requirement is to connect all the devices on the same network with the port 1883 unblocked.

Two pieces of information are required to control the system using the MQTT protocol: a Payload (the commands) and a Topic to address the message to the desired device. Topics can be hierarchized; a receiver can subscribe to multiple topics and a client can operate both as receiver and sender.

The following scenario (Table 1) demonstrates the uses of the MQTT protocol to control two or more devices connected on the same network. Here we imagine that we would want to control two boards that will act both as received (to get commands) and publisher (to send debug information) and are thus subscribed and will publish at the following topics.

To send a command to the Board 1, the user should publish the payload that contain the "**Command**" on the topic "**robot/room01/cmd/01**" while being subscribed to the topic "**robot/room01/debug/01**" (to retrieve debug information like the confirmation of the injection).

Two wildcard characters; "**+**" and "**#**" can be used to retrieve multiple payloads at the same time.

The single level wildcard "**+**" can replace one topic level. If we have four boards that operate in two different rooms that sends debug information on the topics "**robot/room01/debug/01**" and "**robot/room01/debug/02**" (for the boards 1 and 2 on the room 1) and "**robot/room02/debug/01**" and "**robot/room02/debug/02**" (for the boards 1 and 2 on the room 2). It is possible to monitor the boards 1 of both rooms by subscribing to only one topic instead of two (or more) by using the following topic: "**robot/+/debug/01**"

The multi-level wildcard "**#**" can replace multiple topic levels and must be placed at the end of the topic. Considering the previous example, this wildcard could allow to monitor all the boards in the room 1 by subscribing to the topic: "**robot/-room01/debug/#**"

Both wildcards can be combined to monitor all the boards at the same time using the topic: "**robot/+/debug/#**"

Control and GUI

Virtually any programming language (C, C#, C++, Java, JavaScript, Python. . .) or MQTT compatible software can be used to control the proposed device. We used Node-RED, a "low-code" visual flow-based development tool [51,52] built on Node.js that is capable of implementing MQTT and allowing quick and easy development of a graphical interface and complex protocols (such as injection sequences and automation). Moreover, the graphical interface is available via a web interface running on any devices (PC, smartphones, Single Board computers) and operating systems (Linux, Windows, Mac) without requiring the installation of a dedicated application. It is therefore possible to program an experiment on a computer and then monitor and control it via a smartphone from anywhere in the world. The use of Node-RED is not mandatory, we chose this open-source software because it combines all the desired features to build an automation server while being highly user friendly thanks to its "drag and drop" block programming. Programming of more advanced functionalities using the JavaScript programming language in customs "functions" nodes is also possible. Moreover, thousands of open-source libraries are available and free to use to interface the server with other applications or API.

The Node-RED menu (Figs. 3 and 4) can be accessed on any web browser by typing the IP address of the server with the port 1880 on the address bar (i.e. 192.168.1.100:1880 or 127.0.0.1:1880 if directly accessed from the computer that host the Node-RED server).

Flows can be imported in the form of a simple JSON array that can be pasted from the clipboard (in text format) or in the form of a .json file. New nodes can be installed from the "Manage palette" menu to meet the dependencies requirements of the imported flow.

Table 1
Example of hierarchized MQTT topics.

	Board 1	Board 2
Command	robot/room01/cmd/01	robot/room01/cmd/02
Debug	robot/room01/debug/01	robot/room01/debug/02

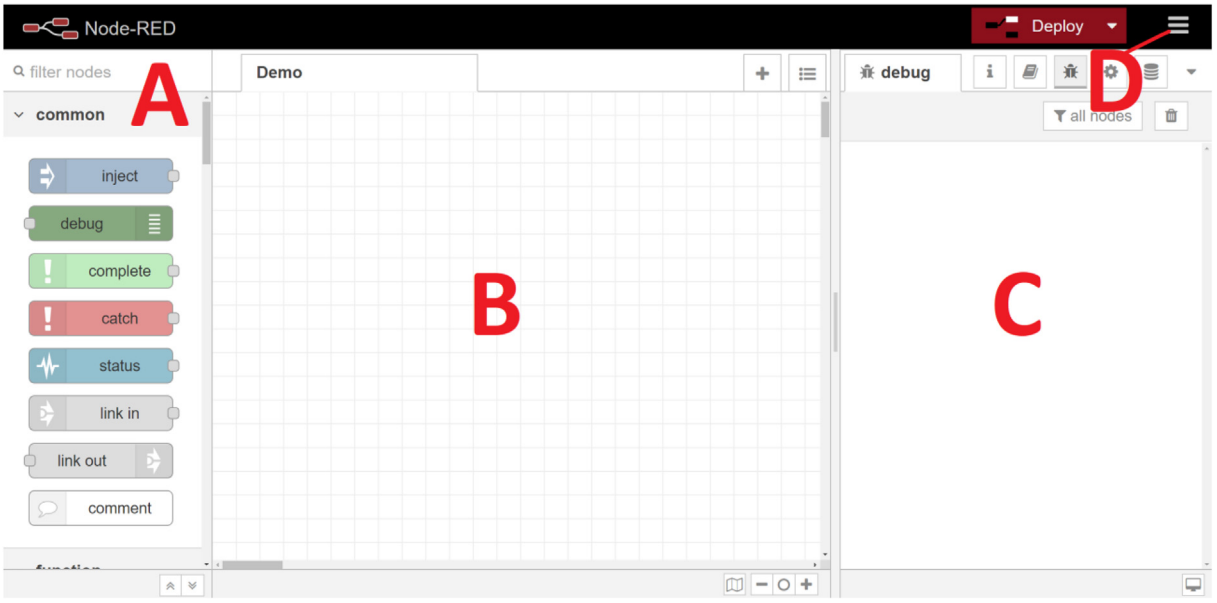

Fig. 3. Node-RED main menu. The main menu of Node-RED is composed of three panels; the node menu (A) allows to drag and drop "nodes" that have different functions on the flow panel (B) some are simple (delay, map, format, debug, . . .) other are able to perform advanced task while requiring simple settings (the MQTT node only requires a broker IP and a topic all the networking part is handled by Node-RED) or more advanced tasks can be programmed in JavaScript using the "function" node. The last panel (C) contains the debug menu or the documentation of the nodes. The top right menu (D) allows to import or export flows or install new nodes. A typical usage of Node-RED will consist of dragging and dropping node from the node menu to the flow, setting up / programming the nodes, connecting them to each other in order to perform the desired automation task, deploying the flow to save and apply the modification and use the debug menu to monitor the good working of the flow. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

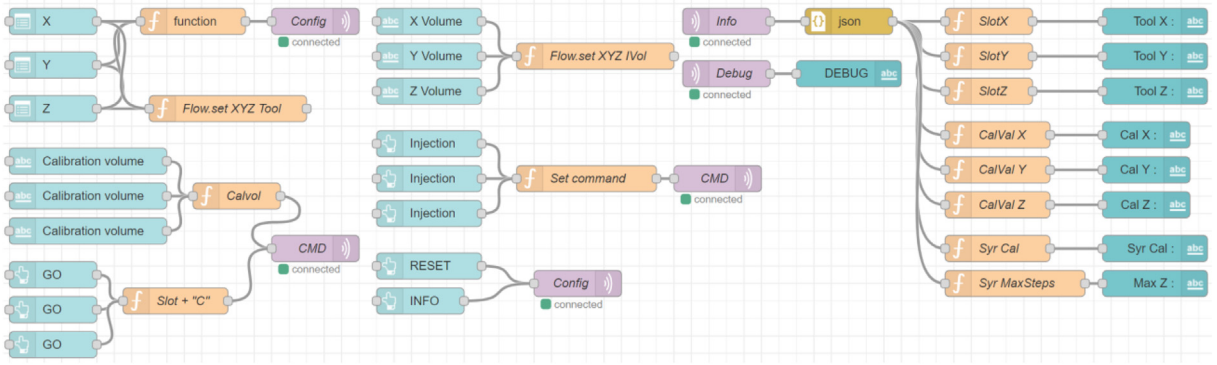

Fig. 4. Example of Node-RED flow that is able to create a graphical user interface to control the system using the "node-red-dashboard" palette, function nodes and MQTT. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

The setup of the different nodes presented on the Fig. 5 (accessible by double clicking on a node) is simple and intuitive and comprehensive information about the input, output data structure can be found on the information panel (C on the Fig. 3). The function nodes provide a syntax correction as well as semantic code highlighting. This provides all the necessary documentation and tools to use Node-RED even without any internet connection. However, a more complete documentation can be obtained in the "Documentation" section of Node-RED.org the website.

As mentioned previously, Node-RED is also able to create a graphical user interface (Fig. 6). After the installation of the "node-red-dashboard" palette, the GUI can be accessed by adding "/ui" on the address bar of the web browser (i.e.

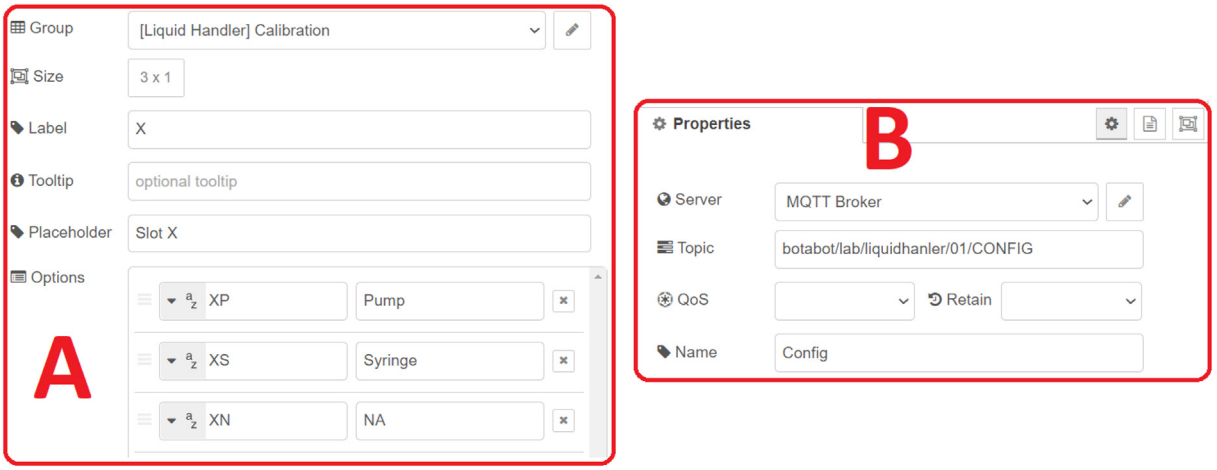

Fig. 5. Example of settings for a dashboard (A) or MQTT (B) nodes.

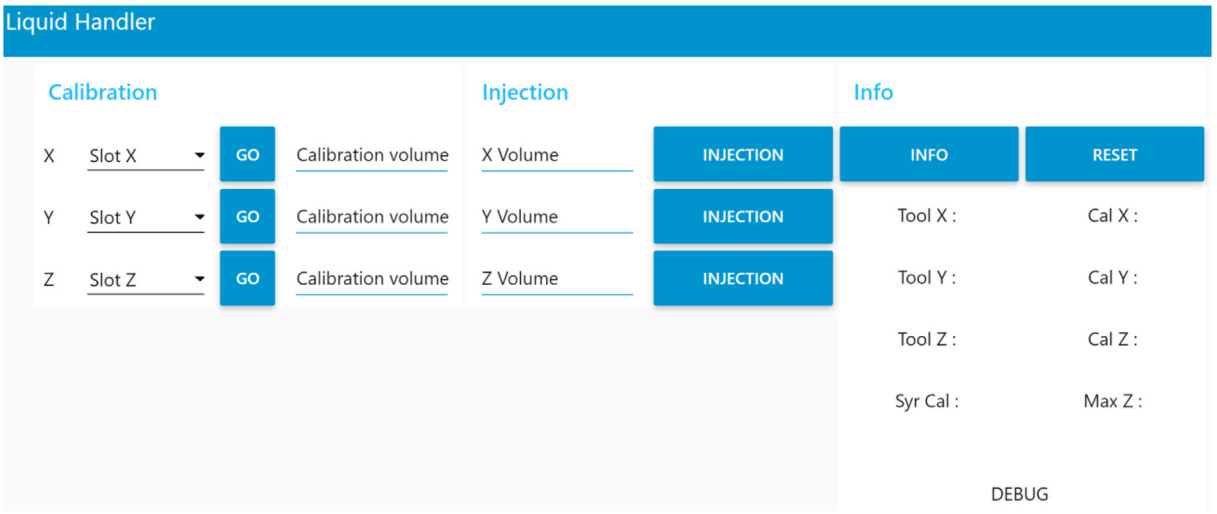

Fig. 6. Node-RED graphical user interface. That results from the flow presented on the Fig. 5. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

192.168.1.100:1880/ui or 127.0.0.1:1880/ui). The GUI is responsive (the panels size and position will automatically adjust to fit on the display) and works with tactile displays.

Setups and calibration exportation

To facilitate the setup of WIFI and MQTT (SSID, password, broker IP, topic ID) we use the Serial Peripheral Interface Flash File System memory (SPIFF) of the ESP32. The chip indeed includes a persistent flash memory that can someway act like an integrated SD card and thus allow to modify files without having to recompile all the source code. The most likely modified parameters as well as the calibration settings are both stored in two distinct JSON files that can be imported via the Arduino IDE [53]. These configuration files remain in the memory when the instrument is powered off and can be exported or modified via MQTT or Serial.

Modularity and scalability

The control board has three DRV8825 [54] stepper drivers. These three slots (X, Y and Z) can receive indiscriminately the two different tools. Arrangements such as tree syringes, tree peristaltic pumps, two syringes and a pump are therefore possible. When a new tool is attached in a slot, the user has to send a simple command via MQTT (for instance: "XP" = Pump on the slot X, "YS" = syringe on slot Y) to edit the settings stored in the SPIFF memory. Although the firmware of the control board was primarily designed to work with these tools (automatic calibration of the peristaltic pumps, conversion of threads/mm to motor steps) the code can be easily adapted to work with other tools involving steppers motors.

The system is also highly scalable. If an experiment requires more tools or if several experiments need to be performed, the use of two or more boards in parallel controlled by the same script or GUI is possible. Many kits can be connected to the same MQTT broker (if their hostnames are different) and be controlled at the same time. Finally, our system only uses some of the General-Purpose Input Output (GPIO) pins of the ESP32. Many GPIO remain available allowing the addition of other sensors or actuators (Analog reading, PWM, Touch, SPI).

We propose to use low cost and widely available hardware (ESP32 UNO, Nema17, Arduino UNO CNC shield) to ensure the long-term reproducibility of the device. In addition, all components can be substituted without altering the operating principle of the system if a specific component is not available or if the user wants to upgrade the system.

- The ESP32 UNO can be replaced by an ESP8266 or even a microcontroller that does not feature WiFi capabilities (as the board can also be controlled via a serial communication).
- The CNC shield can be replaced by a DIY PCB (Printed Circuit Board) or perfboard as its only function is to facilitate the connection between the microcontroller and the stepper drivers.
- The DRV8825 can be replaced by an A4988 or any steppers drivers that are controlled using a "step" and "direction" interface.
- The Nema17 can be replaced by any bipolar stepper motors. The step angle (steps/rotation) can be modified in the firmware.

Design files

Design file name	File type	open-source license	Location of the file
Peristaltic pump rotor	CAD file (F3D)	CC BY-NC-SA 4.0	available with the article
Peristaltic pump stator 7 mm	CAD file (F3D)	CC BY-NC-SA 4.0	available with the article
Syringe Pump	CAD file (F3D)	CC BY-NC-SA 4.0	available with the article

Bill of materials

Bill of materials

Designator	Component	Number	Cost per unit (€)	Total cost (€)	Source of materials	Material type
Stepper motor, CNC shield and limits switches	*CNC Kit*	1	45€	45€	https://www.aliexpress.com/item/33039557882.html	Other
ESP32 Uno	ESP32	1	3€	3€	https://www.aliexpress.com/item/33052923558.html	Other
Neopixel LEDs	WS2812B 1 m 30 IP30	1 m (only three LEDs will be cut from the 1 m strip)	1€	1€	https://www.aliexpress.com/item/32682015405.html	Other
3D printer	Prusament	1	30€	30€	https://shop.prusa3d.com/	Polymer

(*continued*)

Designator	Component	Number	Cost per unit (€)	Total cost (€)	Source of materials	Material type
filament (PETG, ABS, PC or PLA)					en/prusament/801-prusament-petg-prusa-orange-1kg.html	
Epoxy glue	Epoxy	1	1.5€	1.5€	https://hobbyking.com/en_us/hobbyking-4–6min-epoxy-glue.html	Polymer
M4X20 (screws + nuts)	M4X20	3	0.15€	0.45€	Hardware store	Other
M4X20 (screws + nuts)	M4X25	4	0.15€	0.6€	Hardware store	Other
M3X15 (screws + nuts)	M3X15	2	0.08€	0.16€	Hardware store	Other
M3X10 (screws + nuts)	M3X10	1	0.08€	0.16€	Hardware store	Other
M8 threaded rod	M8	1	0.5€	0.5€	Hardware Store	Other
Ball bearings	624ZZ 4x13x5mm Ball bearings	6	0.4€	2.4€	https://www.aliexpress.com/item/32903042053.html	Other
Raspberry Pi Zero W	Single Board Computer	1	11.5€	11.5€	https://www.kiwi-electronics.nl/raspberry-pi/raspberry-pi-zero/pi-zero-boards-packs/raspberry-pi-zero-w	Other
Orange Pi Zero	Single Board Computer	1	10.85€	10.85€	https://aliexpress.com/item/4000049806939.html	Other

Build instructions

The control board uses an ESP32 UNO as well as a CNC Shield for Arduino UNO. Both components are widely available in many online electronics stores. Although the footprint of the ESP32 UNO is similar to the Arduino UNO and the DRV8825 are compatible with 3.3 V logic levels, a slight modification of the shield is required to be compatible with the ESP32 UNO. The Pin connecting to the IO12 (Fig. 7) must be disconnected. The adjacent pin (corresponding to IO13) must then be bridged by soldering. To be able to supply both the stepper driver and the ESP with the same 12 V power supply a jumper is made between the jack connector of the ESP and the input of the Shield (each stepper motor requires around 500 mA in order to work properly and with a sufficient torque). A cable has to be soldered to the + of the ESP (connector furthest from the jack) and the - (connector furthest to the right) the wires can then be passed through the pre-existing holes in the PCB and connected to the shield by referring to the following picture.

All the 3D printed parts were printed using Prusament PETG at 235 °C (and 60 °C for the bed) using a 0.4 mm nozzle, a 0.2 mm layer height and a 50% gyroid filling. A 0.6 mm nozzle (and a 0.3 mm layer height) has been used with the same filament and filling, reducing the printing time without affecting the accuracy of the pumps. We also printed with success two other kits in ABS (250 °C, 100 °C bed, 0.4 mm nozzle, 0.2 mm layer height, 50% rectilinear filling) and PLA (250 °C, room temperature bed, 0.4 mm nozzle, 0.2 mm layer height, rectilinear filling). However, we recommend the use of PETG for its ease of printing temperature resistance and durability.

The peristaltic pump is composed of five 3D printed parts; two for the rotor (rotor base and rotor top) and three for the stator (stator top stator bar and stator lock) (Fig. 8). The stator is built by placing six 4x13x5mm ball bearings between the two 3D printed parts with M4x20 screws and nuts. The stator base should be fixed on the Nema17 motor with two M3x15 screws. The rotor is then placed on the Nema17 shaft and a 7 mm silicone tubing could be inserted on the pump and locked by the stator top and lock with four M4X25 screws.

The syringe pump requires six 3D printed parts to operate with a 10 ml syringe; “main support”, “piston pusher”, “motor coupling”, “endstop”, ”syringe lock A” and “syringe lock 10 ml” (Fig. 9). Two adapters are needed (“syringe lock 1 ml” and

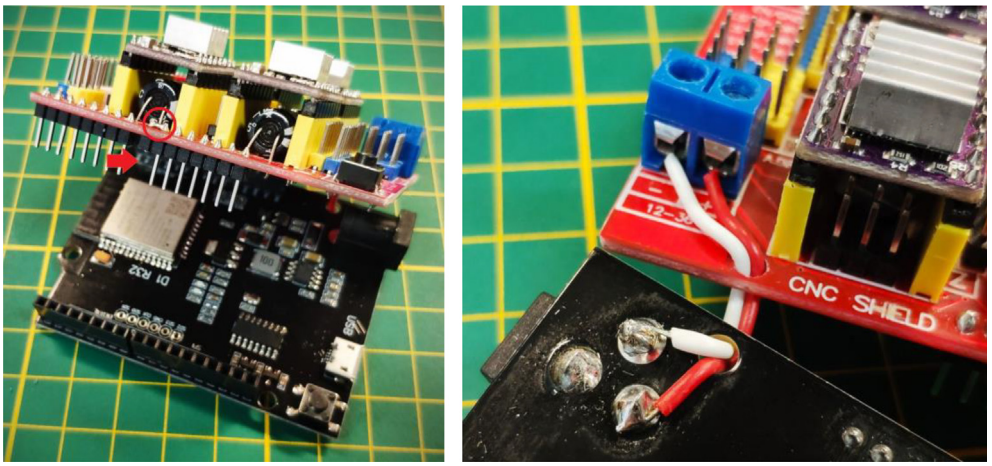

Fig. 7. Modifications of the ESP32 and the CNC shield (each square on the grid is 1 cm by 1 cm).

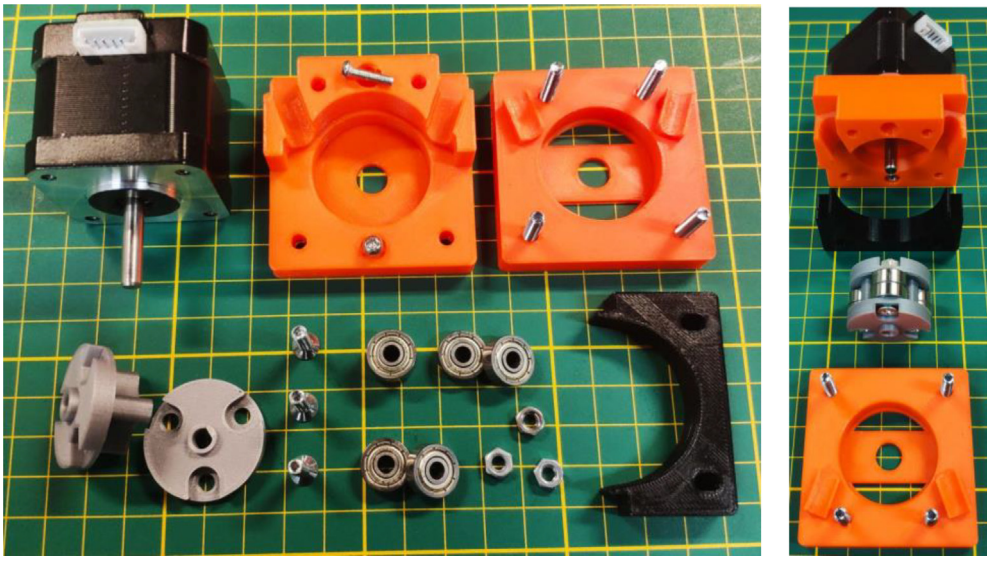

Fig. 8. Peristaltic pump assembly (each square on the grid is 1 cm by 1 cm).

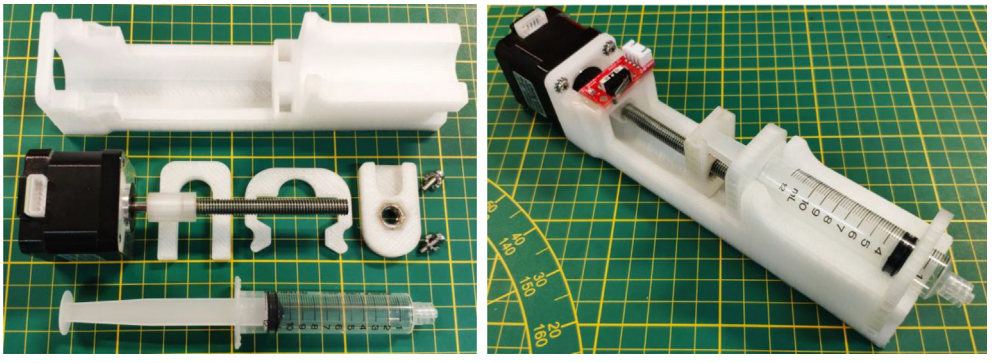

Fig. 9. Syringe pump assembly (each square on the grid is 1 cm by 1 cm).

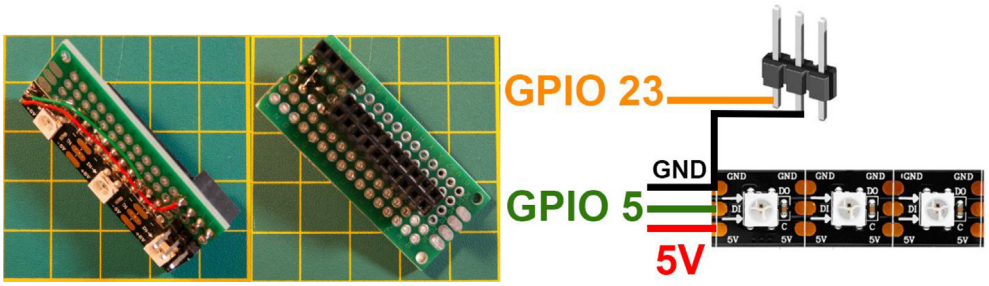

Fig. 10. Optional debug board (each square on the grid is 1 cm by 1 cm).

"syringe pusher 1 ml") in order to adapt the system for a 1 ml syringe. An adhesive is needed (epoxy resin, cyanoacrylate) to glue the M8 threaded rod to the motor coupler as well as the M8 nut in the syringe pusher and the microswitch for the endstop. The Nema 17 is then fixed on the main support with two M3x10 screws.

An optional board can be built to facilitate the debug and the endstop connection using a perfboard, three WS2812B LEDs, two male pin headers and twenty-four female pin headers (Fig. 10). The board can operate without this module as debugging information is also sent over serial and MQTT and the endstop can be directly plugged on the CNC shield or soldered to the ESP323. The LEDs will be used to get information on the status of the control board (WiFi, buttons, motors, calibration, error) while the male pin headers will facilitate the connection of the endstop used for the calibration of the pumps. The LEDs should be connected to the GPIO 5 of the ESP32 (Y + pin on the CNC shield) and the endstop to the GPIO 23 (Z + on the CNC shield)

Software

Compiling and uploading code to the microcontroller can be done using the Arduino IDE. The GUI presented in this paper is made on Node Red that requires NodeJs 8.x, 10.x or 12.x (recommended) installed on the server and uses Mosquitto as MQTT broker. All these software are available on Windows, Mac OS or Linux. However, we recommend using a small Linux server composed by a Single Board Computer like a Raspberry Pi that will offer a perfect balance of size, price, power consumption. We also installed with success the broker and the Node-RED server on a VPS server to facilitate the remote access of the system. In this case, a password protection of the Node Red flow and dashboard as well as the use of MQTT TLS is mandatory (also recommended for a local use).

The installation of the required software on a Debian/Ubuntu machine would only requires the following commands:

- sudo apt install mosquitto
- sudo apt install curl
- curl -sL https://deb.nodesource.com/setup_12.x | sudo -E bash -
- sudo apt install nodejs
- sudo npm install -g --unsafe-perm node-red

Dependencies

Most of the necessary libraries are already pre-installed in the IDE, however, it is necessary to install the two following libraries (available on the library manager of the Arduino IDE):

- ArduinoJson6 (v6.14.1)
- FastLED (v3.3.2)

By default, the Arduino IDE does not support ESP cards. It is thus necessary to add support for ESP32 by pasting the following line in **File > Preference > Additional Boards Manager URLs**: "https://dl.espressif.com/dl/package_esp32_index.json". Then by selecting ESP32 in **Tools > Board > Boards Manager**. The code can finally be uploaded to the board after selecting the correct COM port as well as the **ESP32 Dev Module card**.

Access to SPIFF memory is also not integrated in the Arduino IDE. The ESP32 filesy stem uploader can be downloaded from the following link: https://github.com/me-no-dev/arduino-esp32fs-plugin/releases/. The files can be unzipped in the **Tools** folder in the Arduino IDE directory (the folder structure should look like this < home_dir>/Arduino-<version> /tool s/ESP32FS/tool/esp32fs.jar). After restarting the IDE, the **ESP32 Sketch Data Upload** option should appear in the **Tools** tab and can be used to upload the "wifi.json" and "config.json" files in the internal memory of the board. The config files can be uploaded to the board at any time to change the settings without requiring to recompile the firmware.

The GUI developed on Node Red requires the installation of the "Dashboard" node package that can be installed from the "Manage palette" option on the Node Red interface or via the "npm install node-red-dashboard" command. Then the flow can be loaded by importing the JSON config available on the repository.

ESP32 setup

The firmware available on the repository should contain seven ".ino" files and one "data" folder that contain two ".json" files. The main ino file; "ESP32_Liquid_Handler.ino" should be opened with the Arduino IDE and the two JSON files can be modified with any text editor.

The file "config.json", described on the Table 2, contains all the calibrations information which will automatically be edited and saved by the board. It is nonetheless possible to edit and upload these informations to manually change some settings or calibration values.

The file "wifi.json", described on the Table 3, contains the WiFi and MQTT settings and should be modified in order to connect the device to your WiFi network and MQTT broker.

Operation instructions

Safety

In order to make the machine as user friendly and safe as possible, we have added three LEDs. Calibrate or command a Tool to inform the user about the status of the device. Calibration and injection procedures have been optimized to prevent user errors. As the system has to handle liquids, the control box can be placed remotely from the pumps to limit any risk of a short circuit. Additional sensors (such as flame, humidity and current) can also be connected to improve safety, especially if the machine is intended to be left unsupervised.

The moving mechanical parts (threaded rod, rotor) must not be obstructed. The limit switch must imperatively be placed on the syringe before the calibration. Even though the motors used are Stepper motors which do not have a gearbox and are therefore not likely to be damaged in the event of a blockage (the motors will simply skip steps), the plastic parts could be damaged if used incorrectly.

The electronic parts must remain within an acceptable range of temperature and humidity and the control box must not be confined in a box preventing a minimum air flow required to dissipate the heat of the stepper driver.

Calibration

A calibration is required when a new tool is connected to the control box. To avoid user error, the system will prevent the use of the syringe or pump until it has been calibrated. Once the calibration has been performed, the calibration data are stored in the SPIFF memory and will persist after a restart or power failure. It is also possible to export or re-import the calibration values. The tools connected in each slot and their respective calibration values (number of ml/revolutions in the case of a peristaltic pump and relative position of the piston in the case of the syringe) are stored in the JSON. Since the syringe module can accommodate various sizes (currently 1 ml or 10 ml) or if a threaded rod other than the recommended one (M8) is used, two other calibration values are required to determine the number of ml injected per revolution and the maximum number of steps. Contrary to peristaltic pumps where the number of ml/revolutions can vary according to the diameter of the tubing used and the force applied on the rotor, the number of ml/revolutions for the syringe remains fixed. Thus, only the position of the piston is reset to zero during the homing procedure.

Syringe calibration

The syringe homing is fully automated and can be done with or without a syringe inserted. The two fixed calibration values (ml/turn and thread/mm) must first be entered in the JSON and imported into the SPIFF memory. The limit switch must imperatively be placed in its slot before proceeding to the calibration as shown on Fig. 11. Node-RED users may fill the GUI with the slot on which the tool is connected and press "GO".

User can also send a RAW MQTT command to setup the system with the following structure at the topic "TopicCONFIG":

Table 2
Description of the config.json file.

Name	Description	Possible values
X / Y / Z	Tool installed on the slot (X, Y or Z)	"N", "P" (pump) or "S" (syringe)
XC / XC / ZC	Calibration value	(float) ml/turn (pump) or (int) piston position in steps (syringe)
SC	Syringe thread/mm	(float) depend of the threaded rod used (1 or 1.25 for a M8)
SM	Syringe max	(int) maximal injection (steps)

Table 3
Description of the wifi.json file.

Name	Description
SSID	WiFi SSID
PASS	WiFi Password
SERVER	MQTT broker IP
MQTTUser	MQTT user (if the MQTT authentication is enabled)
MQTTPass	MQTT password (if the MQTT authentication is enabled)
TopicINFO	Current config of the device (ESP -> Control App)
TopicDEBUG	Error message and debug informations (ESP -> Control App)
TopicCONFIG	Modification of the device config (Control App -> ESP)
TopicCMD	Topic for injection commands (Control App -> ESP)
HostName	HostName of the ESP

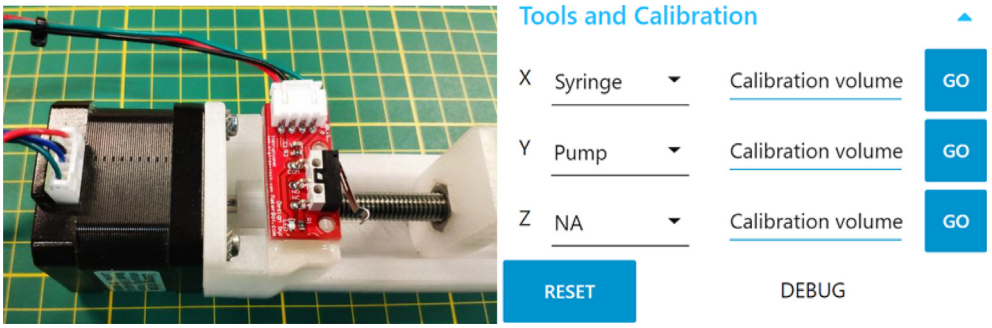

Fig. 11. Syringe pump limit switch an Node-RED GUI. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

- Tool setup: "Slot" (X, Y, Z) "S". Ex: XS, YS, ZS,
- To start the calibration: "Slot" (X, Y, Z) "C". Ex: XC, YC, ZC

Commands sent by serial start with "C" then follow the same structure as the setup (eg: "CXS" instead of "XS" via MQTT) and by a "C" for the calibration (eg: "CXC instead of "XC" via MQTT). The motor will then run until the limit switch is activated. The LEDs will indicate the status of the calibration. During calibration the LED 0 will light up orange and the motor will activate and pull the piston. The direction of the connector on the control board can be reverted if the motor is rotating in the wrong direction. Once the piston holder activates the limit switch, LED 3 will light blue to confirm its activation, the direction of rotation of the axis will then reverse to push the piston back to the neutral position (LED 0 will light blue). Once the homing is completed, LED 0 will light green and a confirmation message will be sent in MQTT and serial. It is recommended to repeat multiple injections/homings to ensure that the piston always remains in the same position. It is not necessary to perform this calibration before each manipulation, but it is advised to perform this manipulation at least once after printing and assembling the part.

Pump calibration

The calibration process is done by connecting the pump on the shield and selecting the tool "Pump" and its corresponding slot on the GUI. A silicone tube is then inserted in the pump and its input in placed in the liquid that will be pumped when the output is connected to a container placed on a precision scale (with an accuracy of at least 0.01 g) the density of the liquid should be known in order to convert the measured weight into a volume. The pump is then activated by clicking on the "GO" button on the calibration panel of the GUI. Users can also send a RAW MQTT command at the topic "TopicCONFIG" to set up the system with the following structure.

- Tool setup: "Slot" (X, Y, Z) "P". Eg: XP, YP, ZP
- Start calibration: "Slot" (X, Y, Z) "C". Eg: XC, YC, ZC

Commands sent by serial start by a "C" then follow the same structure as for MQTT. Eg: "CXC", "CYC" or "CZC".

When the calibration is started, the pump will activate and transfer the liquid, the LED 0 will light up orange to confirm that the calibration is in progress. If the pump rotates in the wrong direction, reverse the direction of the connector on the control board.

If the LED is Orange but the motor is not running, make sure that the motor is correctly connected, the stator is not too tight, and the potentiometer on the stepper driver is adjusted to send enough power to the motor. The user can stop the pump at any moment (preferably when a volume close to the volumes that will be transferred during the experiment is reached) by pressing the limit switch previously used for the syringe pump. The motor should stop and LED 0 will turn green.

The calibration volumes (obtained by converting the weight measured on the balance to a volumes) is finally entered on the GUI or send via the following command:

- MQTT: "Slot"(X, Y, Z) "C" "Volume". Eg: XC10, YC50, ZC25.5
- Serial: "C" "Slot"(X, Y, Z) "C" "Volume". Eg: CXC10, CYC50, CZC25.5

A message will be sent in MQTT and serial to confirm the calibration and inform the user about the ml/rev value.

In order to ensure proper calibration, it is recommended to empty and then inject a specific amount of liquid into the vessel to verify the accuracy of the transferred volume. Peristaltic pumps can present different problems in case of continuous use. In the event of major variations in volume, make sure that the tubing is not obstructed (that it does not rest on the bottom of the container or against the wall, the vacuum at the time of aspiration could stick to the wall reducing the flow of the liquid and altering the dispensed volume). If the liquid backflows when the pump is stopped it means that the compression of the pipe between the stator and the rotor is insufficient. If the volume of liquid transferred decreases over time this may be caused by insufficient tightening of the two rotor clamping screws. Since the ball bearing continually rubs against the tubing, wear and leaks are possible at the rotor. The quality and flexibility of the pipes used will have a great impact on the durability of the system.

When calibration is completed, the values are automatically stored in the ESP memory and can continue to be used as long as the pumps and the syringe remain connected to the same slot, the potentiometers of the stepper drivers are not modified, the type of syringe used remains the same and the length and diameter of the pipes do not vary and the level of tightening of the rotors remains the same. If one of these parameters is modified, the calibration of the tool in question must be performed again.

The following tables (Tables 4 and 5) summarize the complete command set of the two modes of control.

MQTT

A check of the stability of the network connection and the MQTT broker may also be necessary. LED 2 informs the user of the integrity of the connection (Green connected to the WIFI and MQTT and turns red if one or the other is disconnected) in case of disconnection, make sure that the module is close enough to the WIFI router or that each device has a host name in case several separate injection modules operate on the same network.

Injection

Once the modules are installed and calibrated. The injection can be controlled through the GUI or by sending the following command in MQTT on the topic "TopicCMD" or via serial by adding a "C" in the beginning of the command as summarized on the Table 6.

The Table 7 provides an example of instructions sequence and their effect on the device to inject 0.5 ml with a 1 ml syringe (57 mm / ml) installed on slot X, and 50 ml with a peristaltic pump installed on slot Y (calibrated with 100 ml).

Validation and characterization

Pumps validation

The volume delivered by the peristaltic pump has been measured periodically during successive activations to ensure the accuracy and reliability over time of the system as well as its wear resistance (Fig. 12). A 7 mm diameter silicone tubing has been installed in the stator and connected to a flask containing water on the input and a graduated cylinder at the output. The pump was then calibrated by the semi-automatic procedure. The volume to dispense has been set to 25 ml and five samples were extracted and measured. A sequence of twenty successive injections has then been programmed on Node Red. A delay of two seconds was applied between each injection to ensure that a constant stop and restart of the motor will not

Table 4
Command set to attach a tool to a specific slot on the board.

Slot	Pump (MQTT) TopicCONFIG	Syringe (MQTT) TopicCONFIG	Pump (Serial)	Pump (Serial)
X	"XP"	"XS"	"SXP"	"SXS"
Y	"YP"	"YS"	"SYP"	"SYS"
Z	"ZP"	"ZS"	"SZP"	"SZS"

Table 5
Set of commands to calibrate a tool attached on a specific slot on the board.

Slot	Pump/ Syringe (MQTT) TopicCONFIG	Pump / Syringe (Serial)
X	"XC"	"CXC"
Y	"YC"	"CYC"
Z	"ZC"	"CZC"

Table 6
Command set to inject a liquid with a peristaltic pump or a syringe.

Slot	Pump (MQTT) topicCMD	Syringe (MQTT) topicCMD	Pump (Serial)	Syringe (Serial)
X	"X[Volume(ml)]"	"X[Distance(mm)]"	"SX[Volume(ml)]"	"CX[Distance(mm)]"
Y	"Y[Volume(ml)]"	"Y[Distance(mm)]"	"SY[Volume(ml)]"	"CY[Distance(mm)]"
Z	"Z[Volume(ml)]"	"Z[Distance(mm)]"	"SZ[Volume(ml)]"	"CZ[Distance(mm)]"

Table 7
List of commands required to attach, calibrate and activate a syringe and peristaltic pump.

Command	MQTT topic	Effect	LED
"XS" or "SXS"	TopicCONFIG	Attach the syringe to the slot "X"	
"YP" or "SYP"	TopicCONFIG	Attach the pump to the slot "Y"	
"XC" or "CXC"	TopicCONFIG	Homing of the syringe	Orange > Blue > Green
"YC" or "CYC"	TopicCONFIG	Activation of the pump for the calibration	Orange > Blue
"YC100" or "CYC100"	TopicCONFIG	Conversion of the volume to ml/turn	Green
"X28.5" or "CX28.5"	TopicCMD	Injection of 0.5 ml with the syringe pump	Blue
"Y50" or "CY50"	TopicCMD	Injection of 50 ml with the peristaltic pump	Blue

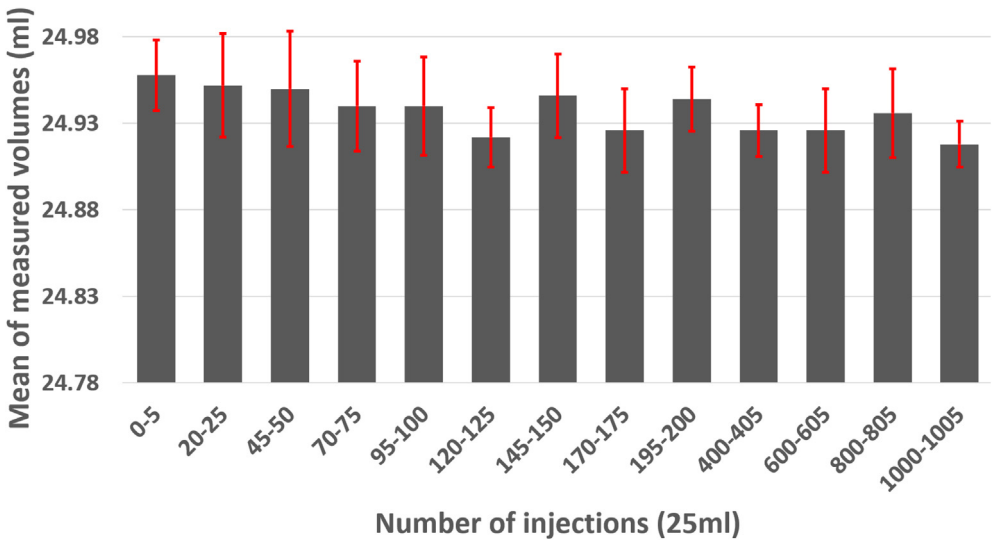

Fig. 12. Validation of the accuracy over time of the peristaltic pump during 1000 successive injections of 25 ml.

affect the accuracy. The volume was again measured five times. This cycle was repeated to reach one thousand successive injections for a total of 25 L (much more than typical volumes handled by these pumps, most of the cases presented in this article will not require the injection more than 1 L for the whole experiment).

During the one thousand activations of the pump, the volume remained relatively constant with an average of 24.94 ml and a standard deviation of 0.02 ml. No leak was observed. Despite constant activation, neither the motor nor the stepper driver produced excessive heat and the MQTT connection remained stable. The pump was then left inactive for 30 days (with the pipes placed in the system) to ensure that the constant pression of the rotor on a single point will not deform the tubing and affect the calibration. Five sampling of 25 ml has been performed after this period of inactivity and resulted in an average volume of 24.91 ml with a standard deviation of 0.02 ml. Although it is not recommended to leave the tubing in place when the pump is not used for a long period of time and then reused without performing a new calibration, the system seems to be able to remain accurate during multiple activations or a period of inactivity.

Another experiment has been conducted in order to determine the accuracy of the pump for volumes ranging from 100 μl to 100 ml. The pump was first calibrated with three different calibration volumes (25 ml, 50 ml and 100 ml) to ensure that the calibration volume did not affect the calibration value. All three volumes returned the same calibration value of 0.82 ml/-turn. Twenty samples for each test volume (100 μl, 500 μl, 1 ml, 5 ml, 10 ml, 20 ml, 50 ml and 100 ml) were extracted and measured as shown in Table 8.

A peristaltic pump with a 7 mm silicone hose does not seem to be able to offer an acceptable level of accuracy for small volumes and high accuracy applications. Although the average sample volume is close to the commanded value, the variation from one sample to another makes the system unreliable. However, samples larger than 1 ml have a small deviation from the command and are therefore validating the ability of the device to operate with volumes of the range of tens to hundreds of milliliters. Nevertheless, the weakness of the peristaltic pump could be complemented by the syringe pump that should be more efficient in handling small liquids as discussed in the following section.

Syringe validation

A similar calibration has been performed with the syringe pump. Unlike the peristaltic pump, this type of pump is less affected by external factors. The repeatability of the piston positioning is important and a homing procedure is necessary to allow the system to register and track the relative position of the pusher. A beforehand calibration of the dispensed volume is not mandatory since both the pusher mechanism and the syringe body are rigid and remain constant (unlike silicone tubes of variable diameters and flexibility as well as variable stator tightening of the peristaltic pump).

Backlash is a lost motion caused by a gap between the parts of a mechanism. It can be defined as "the maximum distance or angle through which any part of a mechanical system may be moved in one direction without applying appreciable force or motion to the next part in mechanical sequence" [55]. For the syringe pump, the small gap between the threaded rod and its nut could cause a loss of accuracy when the direction of the movement is reversed (Fig. 13). To minimize this effect the syringe pump should always be tensioned and only operate on "injection" mode, bi-directional operations are not recommended.

As the homing requires to move the pusher in two directions, a tensioning step (Fig. 14) is added in the procedure. The first step of the calibration pulls the pusher until the activation of the limit switch and then reverts the rotation of the motor to counteract the backlash. The syringe can then be inserted in the pump and used to proceed to injections. We applied this procedure over fifty successive homing injections cycles to test the reliability of the homing as well as the repeatability of the piston positioning.

Table 8
validation of the accuracy of the peristaltic pump with 20 samples of volumes from 100 μl to 100 ml.

Command (ml)	0.1	0.5	1	5	10	20	50	100
Average (ml)	0.11	0.5	1	5	10	19.94	49.83	99.79
Stand. Dev. (%)	12.08	4.15	3.53	0.6	0.43	0.19	0.07	0.06

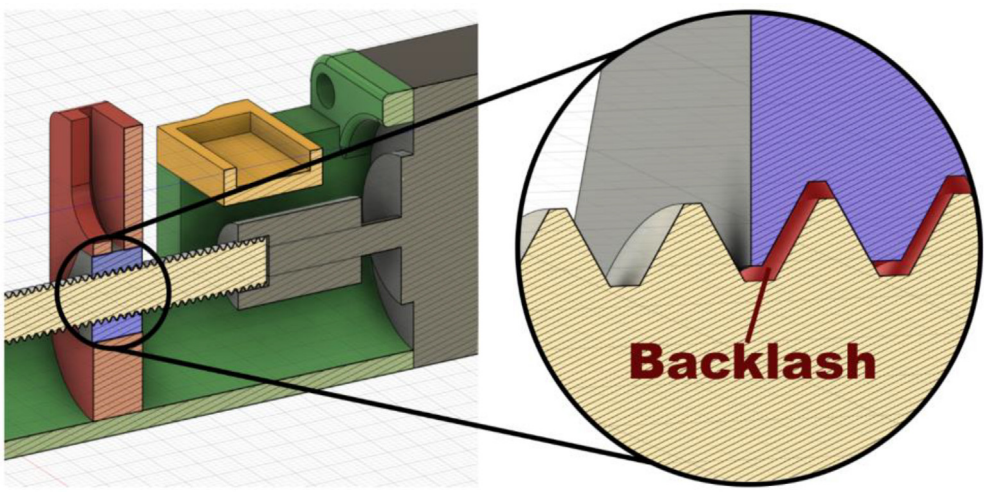

Fig. 13. Backlash.

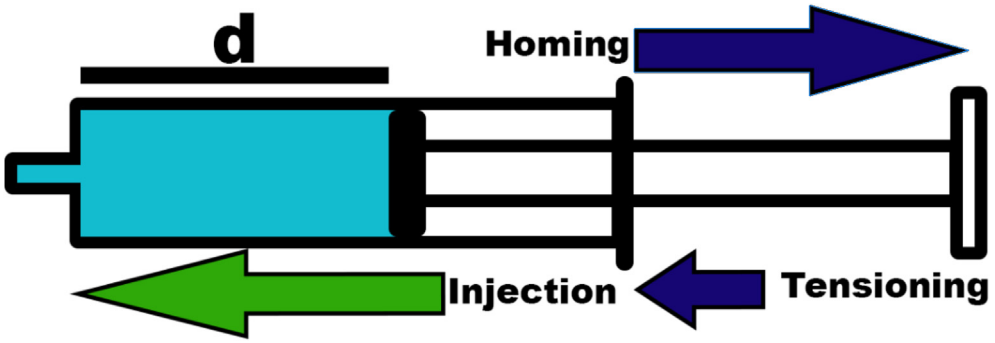

Fig. 14. Homing procedure and reliability measurement. A syringe has been placed on the system and homed. The blue arrows represent the homing procedure that consists of pulling then pushing the pusher after triggering the limit switch. The green arrow represents the arbitrary injection that reflects the typical use of the pump; a large volume up to the complete injection of the syringe or the successive injection of small volumes. The distance "d" has been measured after each cycle. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

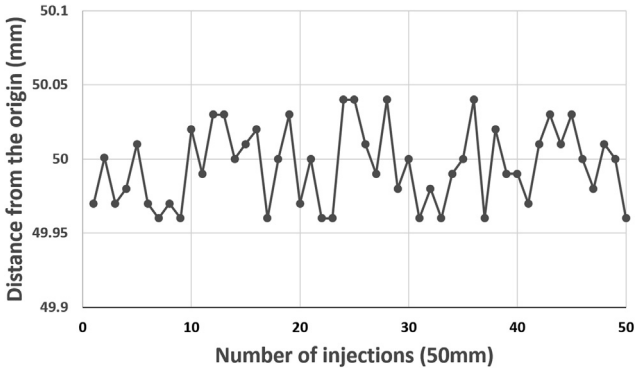

Fig. 15. Evaluation of the repeatability of the syringe piston positioning over fifty successive homing and activation of 50 mm.

Table 9
Measured pusher position during ten cycles of homing and successive push from 10 mm to 50 mm.

Command (mm)	10	20	30	40	50
Average (mm)	9.99	19.98	30.00	39.99	50.00
Stand. Dev. (mm)	0.02	0.02	0.02	0.02	0.04
Equivalent volume with 10 ml syringe (200 µl/mm)	1998 µl ± 4 µl	3996 µl ± 4 µl	6000 µl ± 4 µl	7998 µl ± 4 µl	10000 µl ± 8 µl
Equivalent volume with 5 ml syringe (125 µl/mm)	1248.7 µl ± 2.5 µl	2557.4 µl ± 2.5 µl	3750 µl ± 2.5 µl	4998.7 µl ± 2.5 µl	NA
Equivalent volume with 1 ml syringe (17.5 µl/mm)	174.8 µl ± 0.3 µl	349.6 µl ± 0.3 µl	525 µl ± 0.3 µl	699.8 µl ± 0.3 µl	875 µl ± 0.6 µl

The design of our pump is based on a compromise between ease of printing and assembly and accuracy. This compromise is not a concern for numerous manipulations implying water or liquids with equivalent viscosity as shown in the following validation results. The spindle/nut used for the pump is a regular M8 (with 1.25 thread/mm) galvanized steel that can be found on any hardware store. Users are free to change those parts as the firmware allows to automatically convert any threading to a linear displacement (the "SC" expressed in thread/mm in the config.json has to be modified to correspond to the used threading). Adding a gearbox to the motor can also highly improve the accuracy and the "resolution". The firmware is also able to accommodate the addition of a gearbox simply by dividing the thread/mm value by the gear ratio. By default, the TPM value is set to 1.25, if a 2:1 gearbox is used, the value should be changed to 0.625 as the motor will now have to complete two full rotations in order to achieve the same linear displacement. The lack of guiding rods could lead to a decrease of accuracy if a liquid more viscous than water is used as the piston pusher could bend over the higher force. However, as mentioned previously, the system is not restricted to the use of the proposed pumps. Any syringe pusher controlled by a stepper motor could replace our design if stringent operating conditions are required.

During fifty successive homing and injections of 50 mm (Fig. 15) the pusher remained in the acceptable range that should guarantee adequate liquid injection with an average measured distance of 49.99 mm and a standard deviation of 0.03 mm.

The previous experiment validated the ability to fully empty a syringe in one injection. A second calibration has been performed to determine the accuracy of five different command activations from 10 mm to 50 mm and proves the absence of deviations over time. The homing injection procedure has been repeated ten times and gave results ranging from 9.99 mm +- 0.02 mm to 50 mm +- 0.04 mm (Table 9)

MQTT stability

We left the machine turned on and connected for more than a month without observing any disconnection of the WIFI or the MQTT broker. We tried to saturate the MQTT connection by sending hundreds of messages per second without observing any disconnection or latency. The typical latency between the sending of a command to a VPS server running the MQTT broker as well as the Node-RED server and the reaction of the device was around 120 ms.

Biological use assessment

Accuracy is of course important, but other parameters such as sterility, reliability of the connection, the ability to operate autonomously and durability in general should not be neglected. Experiments have been thus carried out to ensure the reliability of our system to perform liquid handling in a long-term biological experiment that requires sterility.

Our pumps use 7 mm autoclavable silicone tubing (Brand ™ 143361). As peristaltic pumps operate in a closed system, the liquid is not in contact with the external environment, and the risks of contamination are low. To ensure sterility, it is necessary to autoclave the installation with the pipes already connected and make sure that the system is watertight. The following installation (Fig. 16) has been made to confirm the long-term sterility of the system.

After sterilization, 1 ml of vitamins for *E. gracilis* (biotin 10–7%, B12 vitamin 10–7% and B1 vitamin 2 × 10–5% (w/v)) [57] has been added to the Erlenmeyer flask and the Falcon and the Erlenmeyer flask was inoculated with a culture of *E. gracilis* (Fig. 17). To prevent algae from rising through the tubes into the Falcon containing the fresh media, the inlet pipes are not in direct contact with the liquid (see red circle in Fig. 16).

The culture was maintained for one week. Periodically the pump (OUT) removed 10 ml of culture followed by the IN that added the same volume of fresh medium. The aim of the experiment was not to maintain constant turbidity but to ensure that no contamination appeared as the pumps were activated. This setup also allows us to confirm that the pump remains watertight and accurate during a long-term experiment and correctly activates at a specific timing. The experiment was repeated two times, and no contamination was ever observed after microscopic observation of the cultures, the pumps worked correctly and with sufficient accuracy to keep the volume constant. No leak was observed.

The device has been implemented on a previous publication [45] (Fig. 18). The aim was to design an open source automated phototurbidostat that is easily reproducible. The turbidity was maintained using two 3D printed peristaltic pumps and a simple analog turbidimeter designed in our lab. The pumps were controlled by an Arduino MEGA combined with a RAMPS 1.4. The system was controlled remotely via a graphical interface developed on NodeRed. The Arduino MEGA incorporating no WiFi chip, an ESP8266 module was connected to the circuit and transmitted commands to the Arduino via serial communication. The current control board is more advanced and use an ESP32 combined with a CNC shield allowing a more compact integration while being more secure thanks to the addition of safety protocols and debug LED. However, the operating principle remains the same.

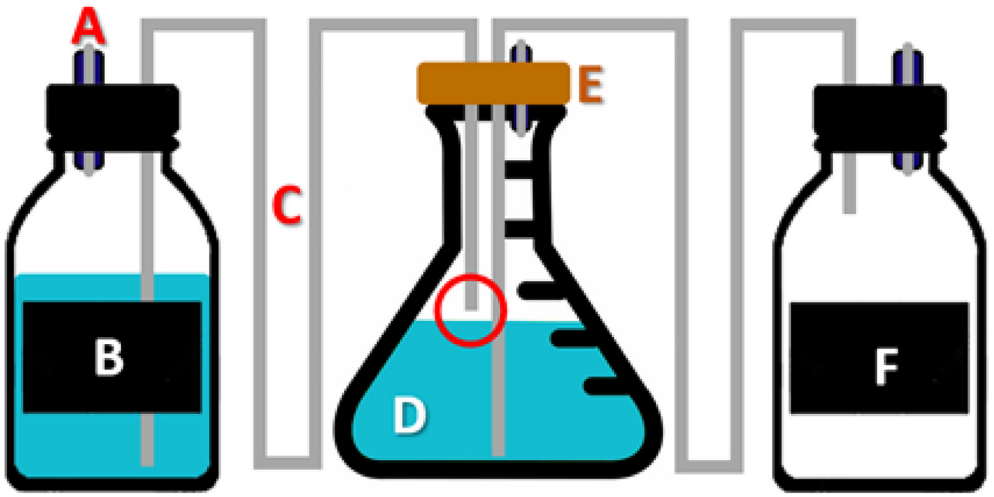

Fig. 16. Pre-sterilization installation. A 1L Falcon (B) and a 3L Erlenmeyer flask (D) were filled with 1L of TAP culture medium [56]. The second 1L (F) Falcon was empty. The two falcons were closed using their plug pierced with two holes, allowing the silicone hose to pass through, a glass rod filled with cotton wool (A) was inserted in a second hole to keep the pressure at the interior of Falcon constant while preserving the sterility of the air. The Erlenmeyer flask is also closed by a rubber stopper allowing the silicone pipes to pass through as well as a glass rod filled with cotton wool allowing gas exchanges with the outside. The installation was then autoclaved, sterilizing the inside of the pipes and the culture medium.

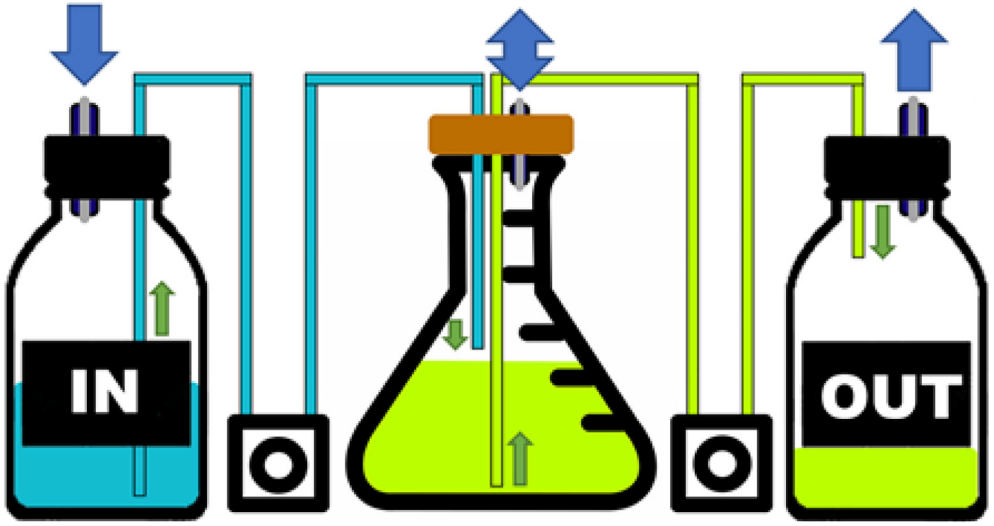

Fig. 17. Continuous culture. The blue arrow corresponds to the airflow inside of the system. When liquid is extracted from the input or output bottles the pressure is equalized via the filtered opening on the cap. The cap of the culture flask also integrates a filter that allows gaseous exchange between the liquid and the surrounding air. The flow of liquid responsible for the continuous culture is represented by green arrows. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

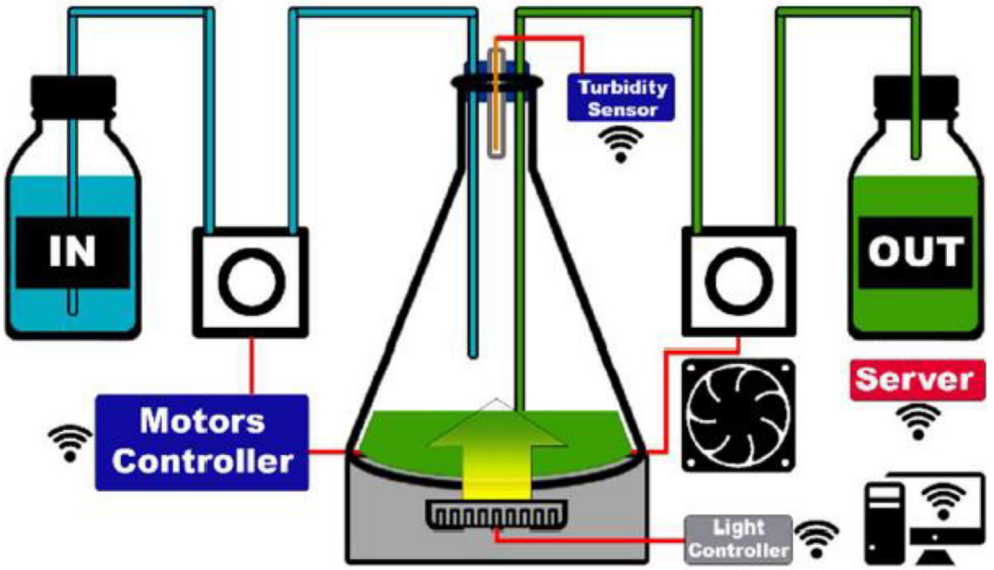

Fig. 18. Phototurbidostat installation. 800 ml culture done in a 2L Erlenmeyer placed on 100RPM Heidolph Instruments Unimax1010 shaker and sealed by a Deutsch & Neumann 47-55-40 mm rubber plug including 4 holes. Two for the liquid inlet and outlet, one for the air exchange and the last one for the turbidimeter composed by a GL5528 photoresistor previously validated for this application. The illumination was provided by a ring of 241 WS2812B LEDs [58].

Reuse potential

This system could easily be adapted for other lab applications or even for domestic use and requires little or no modification of the original design and firmware. With a relative error of + -1% (with a 7 mm tube) the system could replace a micropipette to load culture media or samples into a multiwell plate, falcons, etc. Combined with sensors the system could be used for other automation applications as shown in the Fig. 19.

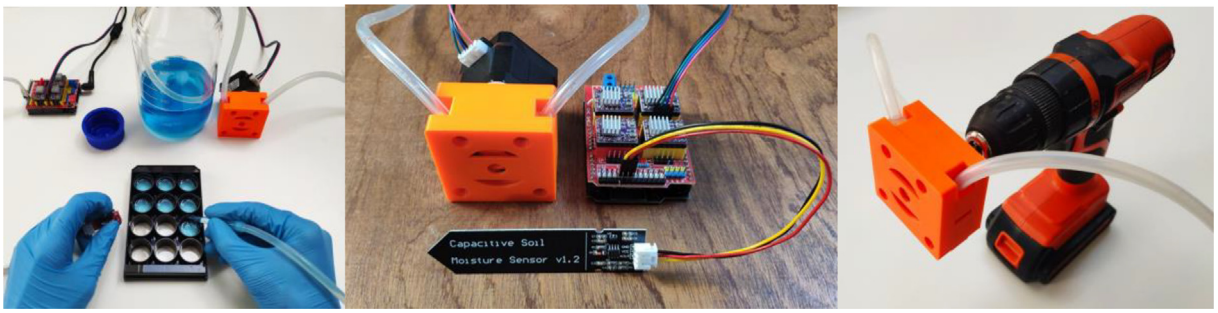

Fig. 19. Reuse potential of the system as a multiwell plate filler (A) automatic watering system (B) when coupled with a capacitive moisture sensor or the use of the 3D printed peristaltic pump with an electric drill (C).

Conclusion

Our liquid handling system has already proven to be functional for several lab applications. It not only improves the speed and repeatability of experiments but it is also convenient. Both the peristaltic pump and the syringe pump offers a degree of precision that is suitable for many lab applications. Our liquid handling system has proven to be reliable during repeated use over a long period of time while being easy to manufacture, low cost and also scalable. Moreover, the design offers a lot of freedom regarding the components used, the manufacturing tolerances and quality of the tools employed (3D printer, plastic type, silicone diameter, threaded rod. . .). It was also successfully implemented in a photo-turbidostat device to maintain an algae culture at constant turbidity [45]. The system has the potential to be reused and adapted to plenty of other applications thanks to its modularity and also the user-friendly interface. Developers could even use the great hardware and software flexibility in order to develop more advanced protocols involving new sensors and actuators.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

P.C. acknowledges financial support from the Belgian Fonds de la Recherche Scientifique F.R.S.-FNRS (PDR T.0032) and European Research Council (ERC, H2020-EU BEAL project 682580). PC is Senior Research Associate from F.R.S. – FNRS.

We would like to thank Gwenaël Gain for providing cell cultures and Nicolas Juste for his participation in the device development and validation during his internship.

References

[1] T.D. Ngo, A. Kashani, G. Imbalzano, K.T.Q. Nguyen, D. Hui, David Hui, Additive manufacturing (3D printing): a review of materials, methods, applications and challenges, Compos. B Eng. 143 (2018) 172–196, https://doi.org/10.1016/j.compositesb:2018.02.012.
[2] C. Schubert, M.C. van Langeveld, L.A. Donoso, Innovations in 3D printing: a 3D overview from optics to organs, Br. J. Ophthalmol. 98 (2) (2014) 159–161, https://doi.org/10.1136/bjophthalmol-2013-304446.
[3] M. Coakley, D.E. Hurt, 3D printing in the laboratory: maximize time and funds with customized and open-source Labware, J. Lab. Autom. 21 (4) (2016) 489–495, https://doi.org/10.1177/2211068216649578.
[4] K.C. Dhankani, J.M. Pearce, Open-source laboratory sample rotator mixer and shaker, HardwareX 1 (2017) 1–12, https://doi.org/10.1016/j.ohx.2016.07.001.
[5] C. Zhang, N.C. Anzalone, R.P. Faria, J.M. Pearce, A.G. de Brevern, Open-source 3D-printable optics equipment, PLoS One 8 (3) (2013) e59840, https://doi.org/10.1371/journal.pone.005984010.1371/journal.pone.0059840.g00110.1371/journal.pone.0059840.g00210.1371/journal.pone.0059840.g00310.1371/journal.pone.0059840.g00410.1371/journal.pone.0059840.g00510.1371/journal.pone.0059840.g00610.1371/journal.pone.0059840.g00710.1371/journal.pone.0059840.g00810.1371/journal.pone.0059840.g00910.1371/journal.pone.0059840.g01010.1371/journal.pone.0059840.t001.
[6] V. Dragone, V. Sans, M.H. Rosnes, P.J. Kitson, L. Cronin, 3D-printed devices for continuous-flow organic chemistry, Beilstein J. Organ. Chem. 9 (2013) 951–959, https://doi.org/10.3762/bjoc.9.109.
[7] Y. He, G.-H. Xue, J.-Z. Fu, Fabrication of low cost soft tissue prostheses with the desktop 3D printer, Sci. Rep. 4 (2014) 6973, https://doi.org/10.1038/srep06973.
[8] Choosing the Right Pump for Your Application & Budget. Harvard apparatus. http://www.harvardapparatus.com/media/harvard/pdf/Pump%20Selection%20Guide.pdf, (accessed 20.07.2020)
[9] International Electrotechnical Commission. Medical electrical equipment—part 2–24, Particular requirements for the basic safety and essential performance of infusion pumps and controllers. IEC 60601–2-24:2012. Geneva: International Electrotechnical Commission; 2.0 ed; 2012.
[10] Syringe pumps. Medical expo. https://www.medicalexpo.com/medical-manufacturer/syringe-pump-2083-_3.html, 2020 (accessed 17.07.2020).
[11] Microfluidic syringe pumps. Darwin microfluidic. https://darwin-microfluidics.com/collections/syringe-pump?page=2, 2020 (accessed 17.17.2020).
[12] Peristaltic pump tubing. Elkay. https://www.elkaylabs.com/pages/files/Elkay_Documents/07_Pump_Tubing.pdf, (accessed 20.06.2020).

[13] F. Esser, T. Masselter, T. Speck, Silent pumpers: a comparative topical overview of the peristaltic pumping principle in living nature, engineering, and biomimetics, Adv. Intell. Syst. 1 (2) (2019) 1900009, https://doi.org/10.1002/aisy.v1.210.1002/aisy.201900009.
[14] Peristaltic pumps. Medical expo. https://www.medicalexpo.com/medical-manufacturer/laboratory-peristaltic-pump-13781.html, 2020 (accessed 07.17.2020).
[15] K. József, K. Levente, Peristaltic pumps – a review on working and control possibilities, in: IEEE 12th International Symposium on Applied Machine Intelligence and Informatics, 2014, pp. 191–194, https://doi.org/10.1109/SAMI.2014.6822404.
[16] Christian Rohrer, How Things Work — Pipettors. laboratorymedicine 32(9) (2014) 514-517. 10.1309/R229-A3G9-4DGF-PCUE.
[17] The open-source Definition. https://opensource.org/osd, (accessed 06.20.2020).
[18] A. Gib, Building open-source Hardware: DIY Manufacturing for Hackers and Makers, Addison-Wesley (2015) 253–277.
[19] M.R. Behrens, H.C. Fuller, E.R. Swist, J. Wu, M.M. Islam, Z. Long, W.C. Ruder, R. Jr Steward, Open-source, 3D-printed peristaltic pumps for small volume point-of-care liquid handling, Sci Rep. 10 (2020) 1543, https://doi.org/10.1038/s41598-020-58246-6.
[20] Precise Peristaltic Pump. https://www.thingiverse.com/thing:2619479, 2017 (accessed 08.08.2020).
[21] J.Z. Milanovic, P. Milanovic, R. Kragic, M. Kostic, "Do-It-Yourself" reliable pH-stat device by using open-source software, inexpensive hardware and available laboratory equipment, PLos One 13 (3) (2018) e0193744, https://doi.org/10.1371/journal.pone.0193744.
[22] B. Wijnen, E.J. Hunt, G.C. Anzalone, M. Pearce Joshua, Open-source syringe pump library, PLos One 9 (9) (2014) e107216, https://doi.org/10.1371/journal.pone.0107216.
[23] Jian Wern Ong, Dwayne Chung Kim Chung, Eric Shen Lin, Hassan Ali Abid, Oi Wah Liew, Tuck Wah Ng. Syringe infusion pump with absolute piston displacement control. Rev. Sci. Instrum. 90 (2019) 076108. 10.1063/1.5099271
[24] M.S. Cubberley, W.A. Hess, An inexpensive programmable dual-syringe pump for the chemistry laboratory, J. Chem. Educ. 94 (2016) 72–74, https://doi.org/10.1021/acs.jchemed.6b00598.
[25] R. Lake John, C. Heyde Keith, C. Ruder Warren, Low-cost feedback-controlled syringe pressure pumps for microfluidics applications, Plos One 12 (4) (2017) e0175089, https://doi.org/10.1371/journal.pone.0175089.
[26] J. Bravo-Martinez, Open-source 3D-printed 1000 µL micropump, HardwareX 3 (2017) 110–116, https://doi.org/10.1016/j.ohx.2017.08.002.
[27] D. Brennan Martin, F. Bokhari Fahad, T. Eddington David. Open Design 3D-Printable Adjustable Micropipette that Meets the ISO Standard for Accuracy. Micromachines (Basel) 9(4) (2018) 191. https://dx.doi.org/10.3390%2Fmi9040191
[28] E. Lee, B. Kim, Sungyoung Choi, An open-source programmable smart pipette for portable cell separation and counting, RSC Adv. (2019) 71, https://doi.org/10.1039/C9RA08368E.
[29] A. Faiña, B. Nejati, K. Stoy, EvoBot: an open-source, modular, liquid handling robot for scientific experiments, Appl. Sci. 10 (3) (2020) 814, https://doi.org/10.3390/app10030814.
[30] M.C. Carvalho, Portable open-source autosampler for shallow waters, HardwareX 8 (2020) e00142, https://doi.org/10.1016/j.ohx.2020.e00142.
[31] V. Klar, J.M. Pearce, P. Kärki, P. Kuosmanen, Ystruder: open-source multifunction extruder with sensing and monitoring capabilities, Hardware X 6 (2019) e00080, https://doi.org/10.1016/j.ohx.2019.e00080.
[32] M. O'Brien, L. Konings, M. Martin, J. Heap, Harnessing open-source technology for low-cost automation in synthesis: flow chemical deprotection of silyl ethers using a homemade autosampling system, Tetrahedron Lett. 58 (25) (2017) 2409–2413, https://doi.org/10.1016/j.tetlet.2017.05.008.
[33] C. Zhang, B. Wijnen, J.M. Pearce, Open-source 3-D platform for low-cost scientific instrument ecosystem, J. Lab. Autom. 21 (4) (2016) 517–525, https://doi.org/10.1177/2211068215624406.
[34] J.M. Pearce, N.C. Anzalone, C.L. Heldt, Open-source Wax RepRap 3-D printer for rapid prototyping paper-based microfluidics, J. Lab. Autom. 21 (4) (2016) 510–516, https://doi.org/10.1177/2211068215624408.
[35] M.C. Carvalho, R.H. Murray, Osmar, the open-source microsyringe autosampler, HardwareX 3 (2018) 10–38, https://doi.org/10.1016/j.ohx.2018.01.001.
[36] C.N. Takahashi, A.W. Miller, F. Ekness, M.J. Dunham, E. Klavin, A low cost, customizable turbidostat for use in synthetic circuit characterization, ACS Synth Biol. 4 (1) (2014) 32–38, https://doi.org/10.1021/sb500165g.
[37] K. Chan, M. Coen, J. Hardick, C.A. Gaydos, K.-Y. Wong, C. Smith, S.A. Wilson, S.P. Vayugundla, S. Wong, F. Romesberg, Low-cost 3D printers enable high-quality and automated sample preparation and molecular detection, PLoS One 11 (6) (2014) e0158502, https://doi.org/10.1371/journal.pone.015850210.1371/journal.pone.0158502.g00110.1371/journal.pone.0158502.g00210.1371/journal.pone.0158502.g00310.1371/journal.pone.0158502.g00410.1371/journal.pone.0158502.g00510.1371/journal.pone.0158502.g00610.1371/journal.pone.0158502.t00110.1371/journal.pone.0158502.s00110.1371/journal.pone.0158502.s00210.1371/journal.pone.0158502.s00310.1371/journal.pone.0158502.s00410.1371/journal.pone.0158502.s00510.1371/journal.pone.0158502.s006.
[38] M.D. Symes, P.J. Kitson, J. Yan, C.J. Richmond, G.J.T. Cooper, R.W. Bowman, T. Vilbrandt, L. Cronin, Integrated 3D-printed reactionware for chemical synthesis and analysis, Nat. Chem. 4 (5) (2012) 349–354, https://doi.org/10.1038/nchem.1313.
[39] P.J. Kitson, S. Glatzel, L. Cronin, The digital code driven autonomous synthesis of ibuprofen automated in a 3D-printer-based robot, Beilstein J. Organ. Chem. 12 (2016) 2776–2783, https://doi.org/10.3762/bjoc.12.276.
[40] G. Gome, J. Waksberg, A. Grishko, I. Y. Wald, O. Zuckerman, OpenLH: Open Liquid-Handling System for Creative Experimentation with Biology. In Proceedings of the Thirteenth International Conference on Tangible, Embedded, and Embodied Interaction; TEI '19; ACM: New York, NY, USA, (2019) 55-64. 10.1145/3294109.3295619.
[41] L.C. Gerber, A. Calasanz-Kaiser, L. Hyman, K. Voitiuk, U. Patil, I.H. Riedel-Kruse, Liquid-Handling Lego Robots and Experiments for STEM Education and Research, PLoS Biol. 15 (3) (2017) e2001413, https://doi.org/10.1371/journal.pbio.2001413.
[42] F. Barthels, U. Barthels, M. Schwickert, T. Schirmeister, FINDUS: an open-source 3D printable liquid-handling workstation for laboratory automation in life sciences, SLAS Technol. 25 (2) (2019) 190–199, https://doi.org/10.1177/2472630319877374.
[43] Opentrons | Open-source Pipetting Robots for Biologists, 2020 https://opentrons.com/ (accessed 07.30.2020).
[44] Sebastian Eggert, Pawel Mieszczanek, Christoph Meinert, Dietmar W Hutmacher, OpenWorkstation: a modular open-source technology for automated in vitro workflows, HardwareX 8 (2020) e00152, https://doi.org/10.1016/j.ohx.2020.e00152.
[45] Alain Gervasi, Pierre Cardol, Patrick E. Meyer, Designing an Open-hardware Remotely Controllable Phototurbidostat for Studying Algal Growth, in: ICCBB '19: Proceedings of the 2019 3rd International Conference on Computational Biology and Bioinformatics, 2019, pp. 13–19, https://doi.org/10.1145/3365966.3365969.
[46] ESP32 Overview. https://www.espressif.com/en/products/socs/esp32/overview, 2019 (accessed 02.10.2021).
[47] U. Hunkeler, H.L. Truong, A. Stanford-Clark, MQTT-S A publish/subscribe protocol for Wireless Sensor Networks, in: 3rd International Conference on Communication Systems Software and Middleware and Workshops (COMSWARE, 2008, p. '08), https://doi.org/10.1109/comswa.2008.4554519.
[48] MQTT. http://mqtt.org/, (accessed 04.20.2020).
[49] MQTT Version 5.0. https://docs.oasis-open.org/mqtt/mqtt/v5.0/mqtt-v5.0.html, (accessed 02.10.2021).
[50] G-code. https://reprap.org/wiki/G-code, 2019 (accessed 05.19.2020).
[51] Node-RED. https://Node-RED.org/, (accessed 08.29.2020).
[52] Nick Heath, How IBM's Node-RED is hacking together the Internet of things. https://www.techrepublic.com/article/node-red/, 2014 (accessed 04.20.2020).
[53] Arduino IDE. https://www.arduino.cc/en/main/software, 2020 (accessed 08.20.2020).
[54] DRV8825. https://www.pololu.com/product/2133, 2020 (accessed 08.20.2020).
[55] V.S. Bagad, Mechatronics. Technical Publications Pune (2008) 1–8.
[56] D.S. Gorman, R.P. Levine, Proc. Natl. Acad. Sci. USA 54 (1965) 1665–1669.

[57] E. Perez, M. Lapaille, H. Degand, L. Cilibrasi, A. Villavicencio-Queijeiro, P. Morsomme, P. Cardol, The mitochondrial respiratory chain of the secondary green alga Euglena gracilis shares many additional subunits with parasitic Trypanosomatidae, Mitochondrion 19 (2014) 338–349, https://doi.org/10.1016/j.mito.2014.02.001.
[58] WS2812B Intelligent control LED integrated light source. https://www.kitronik.co.uk/pdf/WS2812B-LED-datasheet.pdf (accessed 04.20.2020).