Klar, Ville; Pearce, Joshua M.; Kärki, Pyry; Kuosmanen, Petri

**Ystruder: Open source multifunction extruder with sensing and monitoring capabilities**

*Published in:*
HardwareX

*DOI:*
10.1016/j.ohx.2019.e00080

Published: 01/10/2019

*Document Version*
Publisher's PDF, also known as Version of record

*Published under the following license:*
CC BY

*Please cite the original version:*
Ystruder: Open source multifunction extruder with sensing and monitoring capabilities

Ville Klar a,⇑, Joshua M. Pearce b,c, Pyry Kärki a, Petri Kuosmanen a

aDepartment of Mechanical Engineering, Aalto University School of Engineering, Espoo, Finland
bSchool of Electrical Engineering, Aalto University, Espoo, Finland
cDepartment of Materials Science & Engineering and Department of Electrical & Computer Engineering, Michigan Technological University, Houghton, MI, USA

A R T I C L E   I N F O

Article history:

Keywords:
Open hardware
Open source hardware
Syringe pump
Extruder
3D printing
Additive manufacturing
Paste extruder
Monitoring
Sensing

A B S T R A C T

Syringe pumps are widely used in a multitude of tasks where precise volumes of an extrudate need to be delivered at a specific flow rate. In the past decade various open source syringe pump designs have accelerated scientific research and exploration by reducing costs and introducing new ideas. To further expand the capabilities of open source syringe pumps we introduce a novel syringe pump design, the Ystruder. It features a load cell to monitor the piston force. This capability enables clog detection as well as development of advanced dosing algorithms. The Ystruder can be monitored wirelessly through a browser-based interface that is integrated into the embedded system. The design is modular and simple which facilitates different syringe and motor configurations, to meet a wide range of use cases. Finally, the Ystruder is not limited to functioning solely as a pump as it can be integrated into a wide range of devices such as three-dimensional motion systems. Here the dosing accuracy and repeatability of the Ystruder are quantified, and we demonstrate its functionality both as a syringe pump and a paste extruder for 3D printing.

© 2019 The Authors. Published by Elsevier Ltd. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

Specifications table:

<table>
<thead>
<tr>
<th>Hardware name</th>
<th>Ystruder</th>
</tr>
</thead>
</table>
| Subject area  | • Engineering and Material Science  
• Educational Tools and Open Source Alternatives to Existing Infrastructure  
• Measuring physical properties and in-lab sensors  
• Field measurements and sensors  
• Mechanical engineering and materials science |
| Hardware type |          |
| Open source license | GNU GENERAL PUBLIC LICENSE, (GPL) 3.0 |
| Cost of hardware | $150 |
| Source file repository | https://doi.org/10.17605/OSF.IO/T9HPE |

⇑ Corresponding author.
E-mail address: ville.klar@aalto.fi (V. Klar).

https://doi.org/10.1016/j.ohx.2019.e00080
2468-0672/© 2019 The Authors. Published by Elsevier Ltd. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).
1. Hardware in context

It is now well-established that the use of free and open source hardware (OSHW) development and distributed digital fabrication using three-dimensional (3D) printing reduces costs while accelerating innovation [1–7]. Reducing costs of high-performance tools is particularly important for providing access to both medical and scientific equipment in the developing world [6,8,9]. Low-cost and open scientific equipment can provide capabilities in both scientific experimentation as well as research and development. For example, syringe pumps, which dispense precise volumes of fluid over a set amount of time, have a wide variety of applications across fields including medicine, engineering and fabrication.

Owing to the versatility and ubiquity of syringe pumps, several OSHW syringe pump designs have been published in the past decade. Wijnen et al. developed an open source syringe pump library [10] using OpenSCAD1 (an open-source computer aided design (CAD) software). These pumps can be operated using an open source firmware called Franklin [11]. The models in the library can be used to produce most of the mechanical components for the syringe pump. These components can be manufactured using fused filament fabrication (FFF)-based desktop 3D printers. One key factor contributing to the proliferation of such printers is their capability to self-replicate. The development of such FFF printers has been spearheaded by the RepRap (self replicating rapid prototyper) project [12–14]. The cost of the OSHW syringe pump designed by Wijnen et al. [10], including the controller and web-based control interface, is on the order of 5% or less of comparable commercial syringe pumps. The design file library has been downloaded thousands of times saving the international scientific and medical community millions of dollars [15]. It provides an example of the high return on investment (ROI) for funding open hardware development [16].

Recently more feature-rich OSHW designs have broadened the application range of these devices. For example, Lake et al. reported on an OSHW design capable of feedback control to regulate pressure in microfluidic chips [17]. Garcia et al. published a dual syringe design featuring a touchscreen based customizable graphical user interface (GUI) [18]. In addition, open source syringe pumps have been a part of a larger device, such as an auto-sampler [19], a fluid handling scientific robotic platform [20] and a pH-stat device [21]. OSHW syringe pump design has also been used in a case study regarding the design principles of open source bioinstrumentation [22].

While syringe pumps can be used on their own for precise dosing, they can also be combined with 3D motion systems to produce paste extrusion (PE) or direct ink writing (DIW) 3D printers. For example, Pusch et al. [23] used the frame of an open source PrintrBot combined with a OSHW design to produce a large volume extruder suitable for printing of hydrogel structures. Similar configurations have been used for multi-material printing [24,25], bioproduct printing [26–28] and bioprinting [29–32]. OSHW based research equipment facilitates efficient and agile application-centric research of these manufacturing processes. For example, OSHW based research equipment has been utilized in the research of slot die coating systems [33,34], multi-material 3-D printing of bendable sensors [35], wax-printing for paper-based microfluidics [36], and soft robotics actuators [37].

In order to continue to improve OSHW syringe pump designs and assist in broadening their applications, this paper describes the Ystruder. The Ystruder is an open source multifunction extruder with sensing and monitoring capabilities. It can be manufactured for approximately $US 150 and offers more capabilities than commercial syringe pumps systems that cost more than ten times as much. The Ystruder is shown in Fig. 1a.

The compact and lightweight design of the Ystruder enables its use either as a syringe pump or as an extruder for 3D printing. It can be mounted on simple lab stands or on various types of 3D printer frames. Fig. 1b shows the Ystruder mounted on a Prusa RepRap – style frame using tubing and Fig. 1c shows it mounted on a 43 mm spindle mount. When coupled with a precise 3D motion system, the resulting printer can compete with bioprinting systems costing in excess of $US 100 000. The modular structure also promotes extensibility. Advanced functionality such as in situ UV curing [28], heating [36] or mixing can be easily added to the Ystruder. The control electronics of the Ystruder have been designed in a way that they can be interfaced with a wide range of equipment types such as robotic arms, delta robots and box frames. Furthermore, the simple structure and modular design of the Ystruder makes it possible to change both the syringe volume as well the motor size to meet a wide range of dosing and extrusion requirements in terms of thrust, speed, volume and resolution.

Another distinct feature of the Ystruder design is the piston load monitoring capability. Instead of relying solely on the open-loop accuracy of stepper motors, the piston load is monitored continuously during extrusion. Load data can be used for clog detection or it can be incorporated into the dosing algorithm to improve dosing both in terms of latency and accuracy as demonstrated by Lake et al. [17].

Similarly, to the design by Wijnen et al. [10], the Ystruder has a browser based interface for monitoring its performance. The interface features a time series plot of the load cell readings. In addition to displaying the data, the measurements are stored in the browser session and they can be downloaded for further analysis.

1 http://www.openscad.org/.
In summary, the Ystruder is an OSHW syringe pump design that introduces novel features regarding monitoring, modularity and usability. It is both cost-effective and versatile and the approachable design encourages development of new hardware, electronic and software modules to augment its capabilities.

2. Hardware description

The Ystruder consists of mechanical, electronic and software components. Most of the mechanical parts can be 3D printed using desktop FFF printers the remaining mechanical parts (fasteners, bolts, inserts, etc.) are readily available. The design files for the printed circuit boards (PCB) have been designed using the open source electronics design automation suite KiCad. The PCB designs are also provided in a format (Gerber) which can be sent directly to inexpensive PCB fabrication. The firmware of the Ystruder is written using the Arduino C++ abstraction layer and Platformio development environment. All the components of Ystruder are open source and licensed under GNU Public license version 3 (GPL 3). Additionally, the Python based testing and analysis software used in the development of the Ystruder has been open sourced under the same license to encourage further development, validation and testing.

---

2 http://kicad-pcb.org/.
3 https://www.arduino.cc/.
4 https://platformio.org/.
2.1. Mechanics

The linear actuation of the syringe pump is produced with a non-captive linear stepper motor. This type of motor reduces the component count as the lead screw is directly integrated into the motor. The motor is mounted with 3D printed components on 20 by 20 mm aluminum profile, which provides sufficient rigidity and straightness for the assembly. The motor lead screw is connected to a linear rail and block (MGN9H) that completes the linear actuator.

A wide range of syringe sizes and types can be used with the Ystruder design. By default, it uses standard 10 mL syringes (ISO 7886-2:1996) and the syringes can be easily inserted and removed using an interlocking geometry. Different syringe mounts and attachments can be designed and 3D printed. The theory of operation and dimensioning instructions for a screw based linear stepper motor driven syringe pump are presented in detail in the Supplementary section of this paper. The use of luer-lock syringes enables efficient testing of various orifice sizes.

In contrast to most conventional syringe pumps, the Ystruder features load sensing capabilities. The piston load measurement makes it possible to characterize the extrudability of various liquids directly during extrusion or 3D printing. Users can, therefore, assess and identify suitable extrusion parameters more efficiently. Such pressure driven designs are available on the market at a high price.

The piston of the syringe is attached to the load cell to enable measurement of both positive (push) and negative (pull) loads. The load cell is secured in place using threaded brass inserts. While the use of a load cell is recommended, it can be omitted to reduce costs.

2.2. Electronics

The Ystruder electronics design is based on two subsystems on separate PCBs. We denote these subsystems as the Teensy-board and the ESP32-board. The Teensy-board is housed inside the main Ystruder assembly and it is based on a Teensy 3.2 development board. It is connected to a ESP32-board, which serves as the primary user interface for the Ystruder. The ESP32-board is based on the ESP32 microcontroller. The Teensy-board receives instructions from and relays measurement data to the ESP32-board via a universal asynchronous receiver-transmitter (UART). The main functional components of the two Ystruder subsystems are illustrated in Fig. 2a.

The Teensy-board has four main tasks. Firstly, it controls of the linear stepper motor via the TMC2130 driver. Unlike stepper motor drivers commonly used in conventional RepRap 3D printers, the TMC2130 driver features software configurability as well as advanced monitoring capabilities. The TMC2130 stepper motor driver is capable of monitoring the counter-electromotive force of the motor that can be used in the control algorithm. Additionally, the Ystruder does not require limit switches for homing as sensorless homing features of the TMC2130 are used. Secondly, the Teensy reads the load cell values from the ADS1232 analog-to-digital converter (ADC) and relays this data to the ESP32-board. Thirdly, it monitors three button inputs: stop, jog up and jog down. These inputs are connected directly to the Teensy-board in order to ensure responsiveness. Finally, in printing mode the Teensy-board monitors the step and direction signals and performs the extrusion based on them.

---

Fig. 2. a) Block diagram of the main electronic components of the Ystruder; b) Screenshot of the Ystruder browser interface.
The ESP32-board has two main tasks. Firstly, it runs the user interface displayed on the thin-film-transistor (TFT) display. A rotary switch encoder is used as the primary input device. The TFT display breakout also contains a memory card slot that is connected to the ESP32 via a serial peripheral interface (SPI) bus. Secondly, it hosts a Hypertext Transfer Protocol (HTTP) server, which displays the time series measurement data. Unlike previous designs, the browser interface is implemented directly into the embedded system and as such does not require a separate computer to run a server. The interface features a continuous plotting of a sensor data stream based on websocket communication. The HTML (Hyper-Text Markup Language), CSS (Cascaded Style Sheet) and JS (JavaScript) are stored in the serial peripheral interface flash file system (SPIFFS) and they are served to a client upon connection. The measurement data is sent to the client via a websocket connection and plotted using the Chart.js framework. Plotting and rendering are performed on the client side and thus they do not place substantial computational load on the ESP32. The Ystruder browser interface is available for anyone on the same network and consequently no inputs affecting motor movement are accepted from it by default. Such functionality can be incorporated into the firmware, but network security aspects should be taken into account in this development work. The browser interface is shown in Fig. 2b.

3. Design files

3.1. Design files summary

Table 1
Mechanical design files.

<table>
<thead>
<tr>
<th>Design file name</th>
<th>File type</th>
<th>Open source license</th>
<th>Location of the file</th>
</tr>
</thead>
<tbody>
<tr>
<td>Front motor mount</td>
<td>STL &amp; STEP</td>
<td>GNU GPL v3.</td>
<td><a href="https://osf.io/t9hpe/">https://osf.io/t9hpe/</a></td>
</tr>
<tr>
<td>Rear motor mount</td>
<td>STL &amp; STEP</td>
<td>GNU GPL v3.</td>
<td><a href="https://osf.io/t9hpe/">https://osf.io/t9hpe/</a></td>
</tr>
<tr>
<td>Electronics cover</td>
<td>STL &amp; STEP</td>
<td>GNU GPL v3.</td>
<td><a href="https://osf.io/t9hpe/">https://osf.io/t9hpe/</a></td>
</tr>
<tr>
<td>Syringe holder</td>
<td>STL &amp; STEP</td>
<td>GNU GPL v3.</td>
<td><a href="https://osf.io/t9hpe/">https://osf.io/t9hpe/</a></td>
</tr>
<tr>
<td>Syringe body retainer</td>
<td>STL &amp; STEP</td>
<td>GNU GPL v3.</td>
<td><a href="https://osf.io/t9hpe/">https://osf.io/t9hpe/</a></td>
</tr>
<tr>
<td>Syringe piston retainer</td>
<td>STL &amp; STEP</td>
<td>GNU GPL v3.</td>
<td><a href="https://osf.io/t9hpe/">https://osf.io/t9hpe/</a></td>
</tr>
<tr>
<td>Syringe piston pusher</td>
<td>STL &amp; STEP</td>
<td>GNU GPL v3.</td>
<td><a href="https://osf.io/t9hpe/">https://osf.io/t9hpe/</a></td>
</tr>
<tr>
<td>Shaft load cell coupler</td>
<td>STL &amp; STEP</td>
<td>GNU GPL v3.</td>
<td><a href="https://osf.io/t9hpe/">https://osf.io/t9hpe/</a></td>
</tr>
<tr>
<td>Shaft retainer</td>
<td>STL &amp; STEP</td>
<td>GNU GPL v3.</td>
<td><a href="https://osf.io/t9hpe/">https://osf.io/t9hpe/</a></td>
</tr>
<tr>
<td>Display enclosure</td>
<td>STL &amp; STEP</td>
<td>GNU GPL v3.</td>
<td><a href="https://osf.io/t9hpe/">https://osf.io/t9hpe/</a></td>
</tr>
</tbody>
</table>

Table 2
Electronics design files.

<table>
<thead>
<tr>
<th>Design file name</th>
<th>File type</th>
<th>Open source license</th>
<th>Location of the file</th>
</tr>
</thead>
<tbody>
<tr>
<td>Teensy PCB</td>
<td>KiCad project &amp; Gerber</td>
<td>GNU GPL v3.</td>
<td><a href="https://osf.io/t9hpe/">https://osf.io/t9hpe/</a></td>
</tr>
<tr>
<td>ESP32 PCB</td>
<td>KiCad project &amp; Gerber</td>
<td>GNU GPL v3.</td>
<td><a href="https://osf.io/t9hpe/">https://osf.io/t9hpe/</a></td>
</tr>
</tbody>
</table>

Table 3
Firmware files.

<table>
<thead>
<tr>
<th>Design file name</th>
<th>File type</th>
<th>Open source license</th>
<th>Location of the file</th>
</tr>
</thead>
<tbody>
<tr>
<td>Teensy firmware</td>
<td>.ino,.cpp &amp; .h</td>
<td>GNU GPL v3.</td>
<td><a href="https://osf.io/t9hpe/">https://osf.io/t9hpe/</a></td>
</tr>
<tr>
<td>ESP32 firmware</td>
<td>.ino,.cpp &amp; .h</td>
<td>GNU GPL v3.</td>
<td><a href="https://osf.io/t9hpe/">https://osf.io/t9hpe/</a></td>
</tr>
<tr>
<td>Measurement scripts</td>
<td>.py</td>
<td>GNU GPL v3.</td>
<td><a href="https://osf.io/t9hpe/">https://osf.io/t9hpe/</a></td>
</tr>
<tr>
<td>Analysis scripts</td>
<td>.py</td>
<td>GNU GPL v3.</td>
<td><a href="https://osf.io/t9hpe/">https://osf.io/t9hpe/</a></td>
</tr>
</tbody>
</table>

9 https://www.chartjs.org/.
4. Bill of materials

Tables 4 and 5.

<table>
<thead>
<tr>
<th>Part</th>
<th>Component number</th>
<th>Amount</th>
<th>Cost per unit (USD)</th>
<th>Total cost</th>
<th>Source of Materials</th>
<th>Material Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Nema 17 Non-captive Linear stepper motor</td>
<td>17LS19-1684N-200D</td>
<td>1</td>
<td>37.06</td>
<td>37.06</td>
<td>OMC-Stepperonline</td>
<td>Other</td>
</tr>
<tr>
<td>Miniature load cell, 50 kg</td>
<td>DYMH-103</td>
<td>1</td>
<td>39.94</td>
<td>39.94</td>
<td>AE*: XNQ Electric Company Store</td>
<td>Other</td>
</tr>
<tr>
<td>Linear rail MGN9, 150 mm</td>
<td>MGN9-150</td>
<td>1</td>
<td>5.02</td>
<td>5.02</td>
<td>AE*: RDC Official Store</td>
<td>Steel</td>
</tr>
<tr>
<td>Linear block, MGN9H</td>
<td>MGN9H</td>
<td>1</td>
<td>4.02</td>
<td>4.02</td>
<td>AE*: RDC Official Store</td>
<td>Steel</td>
</tr>
<tr>
<td>2020 Aluminium profile, 200 mm</td>
<td>–</td>
<td>1</td>
<td>3.48</td>
<td>3.48</td>
<td>AE**: ZHUHAI Store</td>
<td>Aluminium</td>
</tr>
<tr>
<td>Luer Lock syringe (10 ml), 100 pcs</td>
<td>613-2045</td>
<td>1</td>
<td>35.00</td>
<td>35.00</td>
<td>VWR</td>
<td>Polypropylene</td>
</tr>
<tr>
<td>M5 Flanged threaded Inserts 10 pcs</td>
<td>–</td>
<td>2</td>
<td>8.39</td>
<td>16.78</td>
<td>EB**: t7design</td>
<td>Brass</td>
</tr>
<tr>
<td>M3, sliding t-nut, 100 pcs</td>
<td>20-M3</td>
<td>2</td>
<td>9.99</td>
<td>19.98</td>
<td>AE*: Transkoot</td>
<td>Steel</td>
</tr>
<tr>
<td>M6 × 20 screw</td>
<td>–</td>
<td>2</td>
<td>0.05</td>
<td>0.10</td>
<td>Würth</td>
<td>Steel</td>
</tr>
<tr>
<td>M6 wide washer</td>
<td>–</td>
<td>2</td>
<td>0.05</td>
<td>0.10</td>
<td>Würth</td>
<td>Steel</td>
</tr>
<tr>
<td>M3 × 8 screw</td>
<td>–</td>
<td>2</td>
<td>0.05</td>
<td>0.10</td>
<td>Würth</td>
<td>Steel</td>
</tr>
<tr>
<td>M3 × 20 screw</td>
<td>–</td>
<td>4</td>
<td>0.05</td>
<td>0.20</td>
<td>Würth</td>
<td>Steel</td>
</tr>
<tr>
<td>M3 × 25 screw</td>
<td>–</td>
<td>8</td>
<td>0.05</td>
<td>0.40</td>
<td>Würth</td>
<td>Steel</td>
</tr>
<tr>
<td>M3 × 30 screw</td>
<td>–</td>
<td>2</td>
<td>0.05</td>
<td>0.10</td>
<td>Würth</td>
<td>Steel</td>
</tr>
<tr>
<td>M3 × 16, thumb screw</td>
<td>–</td>
<td>2</td>
<td>0.20</td>
<td>0.40</td>
<td>AE*: nainine Official Store</td>
<td>Steel</td>
</tr>
<tr>
<td>M3 nut</td>
<td>–</td>
<td>4</td>
<td>0.05</td>
<td>0.20</td>
<td>Würth</td>
<td>Steel</td>
</tr>
</tbody>
</table>

* AE: Aliexpress.
** EB: Ebay.

Table 5
Electronics components.

<table>
<thead>
<tr>
<th>Part</th>
<th>Component number</th>
<th>Amount</th>
<th>Cost per unit (USD)</th>
<th>Total cost</th>
<th>Source of Materials</th>
<th>Material Type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Teensy 3.2</td>
<td>TEENSY32</td>
<td>1</td>
<td>19.80</td>
<td>19.80</td>
<td>Watterott</td>
<td>Development board</td>
</tr>
<tr>
<td>Wemos MINI D1 ESP32</td>
<td>–</td>
<td>1</td>
<td>4.57</td>
<td>4.57</td>
<td>AE*: WorldChips</td>
<td>Development board</td>
</tr>
<tr>
<td>IL9341 2.4&quot; TFT-Display</td>
<td>–</td>
<td>1</td>
<td>4.44</td>
<td>4.44</td>
<td>AE*: WorldChips</td>
<td>Module</td>
</tr>
<tr>
<td>Rotary encoder module (5 pcs)</td>
<td>EC11</td>
<td>1</td>
<td>1.72</td>
<td>0.344</td>
<td>AE*: Eti Electronics</td>
<td>Module</td>
</tr>
<tr>
<td>ADS1232</td>
<td>ADS1232IPWR</td>
<td>1</td>
<td>4.46</td>
<td>4.46</td>
<td>Digikey</td>
<td>Integrated circuit</td>
</tr>
<tr>
<td>Linear regulator</td>
<td>AMS1117-5.0</td>
<td>2</td>
<td>0.1289</td>
<td>0.2578</td>
<td>Digikey</td>
<td>Integrated circuit</td>
</tr>
<tr>
<td>2.1 mm DC Barrel Jack</td>
<td>PJ-002A</td>
<td>1</td>
<td>0.60</td>
<td>0.60</td>
<td>Digikey</td>
<td>Connector</td>
</tr>
<tr>
<td>Stepper motor connector</td>
<td>SMR-04 V-B &amp; SMP-04V-B</td>
<td>1</td>
<td>0.39</td>
<td>0.39</td>
<td>Digikey</td>
<td>Wire to Wire connector</td>
</tr>
<tr>
<td>Load cell connector</td>
<td>B4B-XH-(LF)(SN) &amp; XHP-4</td>
<td>1</td>
<td>0.31</td>
<td>0.31</td>
<td>Digikey</td>
<td>Wire to Board connector</td>
</tr>
<tr>
<td>Board connector</td>
<td>SMR-06 V-B &amp; SMP-06V-B</td>
<td>1</td>
<td>0.39</td>
<td>0.39</td>
<td>Digikey</td>
<td>Wire to Wire connector</td>
</tr>
<tr>
<td>Stop button</td>
<td>PS10244RED</td>
<td>1</td>
<td>1.21</td>
<td>1.21</td>
<td>Digikey</td>
<td>Button</td>
</tr>
<tr>
<td>Jog switch</td>
<td>M2018SS1W01</td>
<td>1</td>
<td>5.21</td>
<td>5.21</td>
<td>Digikey</td>
<td>Switch</td>
</tr>
<tr>
<td>Resistor, 10 k</td>
<td>SMD-R 1206</td>
<td>2</td>
<td>0.1</td>
<td>0.2</td>
<td>Digikey</td>
<td>Passive component</td>
</tr>
<tr>
<td>Capacitor, 22μF</td>
<td>SMD-C 1206</td>
<td>2</td>
<td>0.1</td>
<td>0.2</td>
<td>Digikey</td>
<td>Passive component</td>
</tr>
<tr>
<td>Capacitor, 0.1μF</td>
<td>SMD-C 1206</td>
<td>2</td>
<td>0.1</td>
<td>0.2</td>
<td>Digikey</td>
<td>Passive component</td>
</tr>
</tbody>
</table>

* AE: Aliexpress.

5. Build instructions

5.1. Mechanical assembly

Tools required

- M3 Allen wrench
- M5 Allen wrench
- M6 thread tap and tap holder
- Hand drill
- Band saw or hacksaw
- Marker
- Soldering iron

A rendering of the Ystruder assembly is shown in Fig. 3a and an exploded view of the Ystruder assembly with component numbering is shown in Fig. 3b. The part naming with the component numbering is shown in Table 6.

Fig. 3. a) Ystruder assembly, 3D printed parts shown in red color b) Exploded view of the Ystruder assembly with part numbering using the same color scheme. For associating numbers with respective component names, the reader is referred to the web version of this article.
Steps for assembly:

1. 3D print all the files listed in the Design Files Table.
2. Cut a 240 mm length of the aluminum profile. The cutting surfaces should be straight. Thus, the use of a chop saw and secure mounting is advised. Many vendors also sell aluminum profile cut to length. In case a different motor size is used, the aluminum profile should be cut to length according to the motor dimensions.
3. Tap both ends of the aluminum profile with the M6 thread tap. The use of a suitable cutting fluid is recommended. The threads should be at least 12 mm deep at both ends.
4. Attach the rear motor mount with the M6 × 20 screw.
5. Attach the linear stepper motor to the front stepper motor mount using four M3 × 20 screws and slide it in place against the rear motor mount. The rear motor mount has a slot for the stepper motor wires and the motor should be oriented accordingly.
6. The front motor mount can be further secured in place with the M3 t-nut underneath the aluminum profile.
7. Insert M3 t-nuts using M3 × 8 screws at both ends of the linear rail and slide it in place. Be careful not to remove the linear slide from linear rail as this may result in the ball bearings to drop out. Insert the syringe holder temporarily to position the linear slide so that it mates with the surface of the syringe holder. Tighten the M3 screws to secure it on aluminum profile.
8. The wire of load cell needs to point upwards and thus the orientation needs to be marked onto the M5 threaded insert. Screw the threaded insert on the load cell firmly and mark the orientation of the wire onto it.
9. Insert the threaded insert into the shaft load cell coupler using a soldering iron heated up to 250°C. Make sure that the previously made marking faces up.
10. Screw in the load cell on the shaft load cell coupler and attach it to the linear rail using two M3 × 25 screws.
11. Slide the shaft load cell coupler in place so that the circular surface mates with the screw of the linear stepper motor.
12. Attach the shaft retainer through the shaft load cell coupler to the linear rail using two M3 × 25 screws so that its circular surface mates with the stepper motor shaft’s surface.
13. Screw the syringe retainer on the syringe holder with two M3 × 30 screws and nuts.
14. Slide the syringe holder on the aluminum profile and secure it in place with the M6 × 20 screw.
15. Assemble and test the Ystruder PCBs (see Section 5.2 for details).
16. Screw the stop button and jog switch on to the electronics cover.

Table 6

Components in exploded view.

<table>
<thead>
<tr>
<th>Number</th>
<th>Part name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Rear motor mount</td>
</tr>
<tr>
<td>2</td>
<td>Linear stepper motor</td>
</tr>
<tr>
<td>3</td>
<td>Front motor mount</td>
</tr>
<tr>
<td>4</td>
<td>M3 × 20 screw</td>
</tr>
<tr>
<td>5</td>
<td>20 by 20 aluminium profile, 240 mm long</td>
</tr>
<tr>
<td>6</td>
<td>T-nut, M3</td>
</tr>
<tr>
<td>7</td>
<td>Linear block</td>
</tr>
<tr>
<td>8</td>
<td>Linear rail</td>
</tr>
<tr>
<td>9</td>
<td>M3 × 30 screw</td>
</tr>
<tr>
<td>10</td>
<td>Syringe body retainer</td>
</tr>
<tr>
<td>11</td>
<td>Syringe holder</td>
</tr>
<tr>
<td>12</td>
<td>M6 × 20 screw and wide washer</td>
</tr>
<tr>
<td>13</td>
<td>M3 nut</td>
</tr>
<tr>
<td>14</td>
<td>Syringe body</td>
</tr>
<tr>
<td>15</td>
<td>Syringe piston</td>
</tr>
<tr>
<td>16</td>
<td>M3 × 8 screw</td>
</tr>
<tr>
<td>17</td>
<td>M3 thumb screw</td>
</tr>
<tr>
<td>18</td>
<td>Syringe piston retainer</td>
</tr>
<tr>
<td>19</td>
<td>Threaded brass insert, M5</td>
</tr>
<tr>
<td>20</td>
<td>Syringe piston pusher</td>
</tr>
<tr>
<td>21</td>
<td>Load cell</td>
</tr>
<tr>
<td>22</td>
<td>Shaft load cell coupler</td>
</tr>
<tr>
<td>23</td>
<td>M3 × 25 screw</td>
</tr>
<tr>
<td>24</td>
<td>Shaft retainer</td>
</tr>
<tr>
<td>25</td>
<td>Teensy PCB</td>
</tr>
<tr>
<td>26</td>
<td>Stop button</td>
</tr>
<tr>
<td>27</td>
<td>Jog switch</td>
</tr>
<tr>
<td>28</td>
<td>Electronics cover</td>
</tr>
</tbody>
</table>
17. Connect the stepper motor and load cell to the Teensy PCB and insert it into to the electronics cover. The Teensy-board snaps in place with the tabs on the sides of the electronics cover.
18. Attach the electronics cover on top of the motor mounts using the captive M3 nuts and M3 × 25 screws.
19. Follow the calibration instructions to finalize the assembly.

5.2. PCB assembly

While the PCBs can be manufactured using techniques such as toner transfer or isolation milling using open source computer numerical control (CNC) milling machines [38], it is recommended to order them from a fabrication service using the Gerber files. The assembly features some surface mount devices (SMD), which may be challenging for inexperienced electronics fabricators. Basic soldering skills are required. The components are annotated on the silkscreen of the PCB for ease of assembly.

Tools required
- Soldering iron
- Solder wick

Teensy board
Renderings of the top and bottom layers of the Teensy-PCB are shown in Figs. 4a and 4b respectively.

1. Solder on the SMD components. Drag soldering and cleaning up with solder wick where necessary is recommended. A reflow oven and solder paste can also be used.
2. Solder six lead wire on the Teensy-PCB. Refer to the schematic and layout for further instructions regarding the connections. The six lead wire is inserted through the oval slit in the Teensy-PCB and solder joints should be on the top layer of the Teensy-PCB. A minimum wire length of 1 meter is advised, but the length can be chosen to fit the application. Lengths exceeding 2 meters are not recommended.
3. Attach the rectangular connector (SMR-06V-B female) housing to the other end of the six lead cable using crimping tools.
4. Solder the stop button and jog switch leads to the Teensy PCB according to the schematic.
5. Solder the through hole components (Teensy 3.2, TMC2130 breakout, load cell connector and 100 μF capacitor).
6. Attach a heat sink on TMC2130. Heat sinks of up to 12 mm in height fit inside the electronics case.

Fig. 4. Teensy-PCB rendering, a) Top view b) Bottom view.
7. Test fit the circuit board on the assembly and cut the load cell wires to length. Make sure the full range of motion can be covered.

8. Cut the stepper motor wires to approximately 40 mm measured from the end of the stepper motor.

9. Strip the ends of the stepper motor wires and attach a rectangular connector (SMR-04V-B male) housing to them.

10. Solder the cut wires onto the Teensy PCB according to the schematic and attach a rectangular connector housing (SMR-06V-B female) to them.

11. Strip the end of the load cell wire and attach the connector (XH-4A female) housing to them.

12. Flash the firmware following the instructions in Section 5.3.

**ESP32 board**

Renderings of the top and bottom layers of the ESP32-PCB are shown in Figs. 5a and b, respectively.

1. Solder the SMD components on the PCB.
2. Solder the six lead cable wires according to the schematic.
3. Attach the rectangular connector (SMR-06V-B male) housing to the other end of the six lead cable using crimping tools.

![ESP32-PCB rendering](image)

*Fig. 5. ESP32-PCB rendering, a) Top view b) Bottom view.*
4. Solder on the through hole components. Use female 0.1” headers with the display module to allow sufficient space for the solder joints beneath it.

5. Flash the firmware following the instructions in section 5.3.

The fully assembled electronics of the Ystruder are shown in Fig. 6.

5.3. Flashing the firmware

Tools required

- Raspberry Pi single board computer, alternatively a computer with a Debian based distribution installed.
- Micro USB cable

These instructions are based on using the Raspberry Pi (RPI) single board computer (SBC) to flash the firmware. Users can use a computer with another Linux or FreeBSD distribution as well, but the instructions will be slightly different based on the particular operating system and distribution. For a Debian-based Linux distribution\textsuperscript{10} the instructions are the same as below.

1. Install the Raspbian distribution.\textsuperscript{11} The Lite version is sufficient, but installation can be performed using a desktop environment as well.
2. Log in and connect to the Internet.

\textsuperscript{10} https://www.debian.org/.
3. Open a terminal window and update the package repository and upgrade to the latest software.
   ```bash
   sudo apt-get upgrade && sudo apt-get update
   ```

4. Install python pip. 
   ```bash
   sudo apt-get install python-pip
   ```

5. Install platformio.
   ```bash
   sudo pip install platformio
   ```

6. Add the teensy rules to udev and reload them.
   ```bash
   wget https://www.pjrc.com/teensy/49-teensy.rules
   sudo mv 49-teensy.rules /etc/udev/rules.d
   sudo udevadm control --reload-rules && sudo udevadm trigger
   ```

7. Download the firmware files from OSF and unzip them.
   ```bash
   wget -O YStruder_FW.zip https://osf.io/sp4yn/?action=download
   unzip YStruder_FW.zip
   ```

8. Change to the firmware folder.
   ```bash
   cd YStruder_FW
   ```

9. Disconnect the stepper motor from the Teensy-board and the boards from each other. Connect the Teensy-board via the USB.

10. Upload the Teensy firmware using make.
    ```bash
    make -C Teensy/ upload
    ```

11. Once uploaded disconnect the Teensy-board and connect the ESP32-board.

12. Upload the filesystem and firmware to the ESP32.
    ```bash
    make -C ESP32/ uploadfs && make -C ESP32/ upload
    ```

5.4. Calibration of load cell

It is necessary to perform a calibration routine prior to operation to set correct gain and offset values for the ADS1232 ADC. A mass with a known weight (1 kg) is needed for the calibration routine. We recommend using a water-filled bottle weighed with a precision scale. For best performance the calibration routine should be performed at regular intervals.

1. Remove the load cell from the Ystruder assembly and attach it to the calibration stand using two M5 nuts. The calibration stand (shown in Fig. 7) is composed of two identical 3D printed parts that use captive M5 nuts to thread onto the load cell.

![Calibration stand](image)

**Fig. 7.** Calibration stand.

---

12 https://pypi.org/project/pip/.
2. Measure 1 kg of water (approximately 1 L) into a container of adequate size. Make sure to weigh the bottle and account for the weight of the container itself as well as the weight of the top part of the calibration stand.

3. Choose the calibration option in the Ystruder Settings menu. Once you have clicked on the menu entry, you have 10 s to place the weight on the calibration stand.

4. The Ystruder will measure the weight and calculate the necessary gain and offset values from it. These calibration values are stored in non-volatile memory of the Teensy.

5. If the calibration was successful, the ‘Force’ field on the display should read approximately 9.81 N.

6. **Operation instructions**

   The Ystruder can be used either as a syringe pump or as an extruder. Furthermore, the Ystruder platform has been designed in a way that it can be modified and integrated into a wide range of processes and test procedures. The operation will vary slightly based on use case, but a common syringe pump as well as a common 3D printing use case are described.

6.1. **Syringe pump**

   1. Turn the Ystruder on by connecting it to 12V DC power supply via the barrel jack connector on ESP32-board.
   2. Use the encoder switch to home the Ystruder with the "Home" command on the main menu. The linear actuator will drive the load cell coupler to the top position.
   3. Load the sample material into the syringe. Use degassing methods if necessary.
   4. Insert the syringe piston into the syringe body.
   5. Attach the syringe retainer to the syringe piston.
   6. Twist the syringe in place onto the syringe holder. There should be a slight interference fit on the syringe holder and the syringe body flanges.
   7. Equip the syringe with the desired luer-lock tip or connector.
   8. Jog (jog switch is on the side of the main assembly) the linear carriage down until it pushes the syringe piston and material starts to extrude.
   9. Attach the load cell mount to the syringe piston retainer with the M3 thumb screws.
   10. Choose “Pump” in the “Mode” menu.
   11. Insert the desired dose (mL) and dosing speed in the menu (mL/min) in the Pump menu.
   12. An upper force limit can be set for the dosing process in the “Pump menu”.
   13. Choose the “Pump” command on the main menu.
   14. The pump will extrude the set dose and amount. The dosing can be stopped by pressing the red stop button on the main assembly. Pressing it again will resume the interrupted dosing process.
   15. Once the dosing process is completed it can be repeated by pressing the “Pump” command on the main menu.

6.2. **Extruder**

   The Ystruder can be mounted in a direct-feed configuration or via tubing. With high-viscosity liquids the tubing can result in substantial increases in necessary piston force for a given flow rate. The mounting and installation of the Ystruder depends on chosen printer frame type. Installation instructions for a Prusa I3 frame and a 43 mm spindle mount are provided in the Supplementary information.

   Ystruder can interface with most common stepper driven 3D printers as it can extrude based on the step & direction signals that are commonly used as inputs for stepper motor drivers. However, to retain pressure control capability the Ystruder stepper motor driver is not directly connected to these signals. Instead, a retriggerable monostable multivibrator (RMM) circuit (e.g. 74HC123) converts the high frequency step pulses into a single pulse that is high during extrusion or retraction and low otherwise. The output of the RMM is monitored on the Teensy-board with external interrupts and used to control the stepper motor in extruder mode. The printer signal monitoring and interfacing is described in further detail in the Supplementary section.

   1. Follow steps 1–9 described in Section 6.1.
   2. Choose the “Extrude” option in the “Mode” menu.
   3. Calculate the extrusion speed based on syringe orifice diameter, layer height and layer width. See the Supplementary Section for the calculation procedure.
   4. Enter the calculated extrusion speed along with an optional upper pressure limit value
   5. Load the gcode to the host software and run it. The Ystruder will monitor incoming extrusion and retraction signals and print accordingly.
   6. The Ystruder can be stopped at any point of the print by pressing the stop button. Pressing it again will resume the extrusion.
7. Validation and characterization

The performance of the Ystruder was validated and characterized experimentally by constructing a test bench. The test bench features measurement of syringe pressure with a pressure sensor (SMC PSE560), extruded mass with a precision scale (Mettler Toledo PG1000-3) and stepper motor movement with a digital indicator (Sylvac μS229). The Ystruder was operated in serial mode where it reports back its target position and measured force. A RPi (Model 3B+) SBC was used to acquire the data from the sensors via a serial and I2C bus. A schematic illustration of the testing equipment is shown in Fig. 8a and a photograph of the testing equipment is shown in Fig. 8b. Both the measurement and analysis software are written in Python and are also released under the GNU GPL v3 license.

Two tests were run with the developed test bench. Firstly, the maximum thrust capability of the Ystruder was established by measuring the movement force and pressure with a plugged syringe. Secondly, the dosing accuracy and repeatability of the Ystruder was determined by running a test series with varying process parameters. Additionally, 3D printing tests with different paste materials and geometries were conducted. All dosing and printing tests were performed using ISO 7886-2:1996 standard 10 mL syringes (Henke-Sass, Wolf GmbH, Germany).

7.1. Maximum thrust

The maximum thrust capability of the Ystruder was determined by measuring the force, pressure and movement of plugged syringe filled with silicone oil (1000 cSt kinematic viscosity). Fig. 9 shows the actual (measured with the digital indicator) and target distance signal as well as the load cell and pressure sensor readings as a function of time in a blocked syringe.

As shown by Fig. 9 the motor starts to skip steps or stall at approximately 200 N. While the maximum force of the linear stepper in the tested speed range is 320 N, step loss started to occur at a lower force due to a lower driver voltage (12 V) and current limit (900 mA). Other driver configurations also contribute to the achievable torque of a stepper motor. Stalling related issues should be avoided by adequately sizing the stepper motor and syringe (cross-sectional area) and choosing suitable stepper motor driver parameters. Calculations and design considerations regarding the thrust in different speed ranges and configurations are provided in the Supplementary information of this paper.

Fig. 9 also shows that as the motor moves the force and pressure increase, and they correlate well which suggests that measuring the piston force can be used to infer the extrusion pressure without needing to substantially correct for frictional forces. Depending on the speed and pressure range as well as syringe type, frictional forces may have a more significant influence on the correlation of piston force to syringe pressure.

Another error source shown in Fig. 9 is indicated by the deviation of the slope between target position and measured position at forces exceeding 50 N. The motor does not stall completely but it does not move according to the step signals sent to...
the stepper motor driver. This behaviour was attributed to microstep associated loss of incremental torque. Microstep modes in stepper motor actuation result in a loss of incremental (per microstep) torque [39]. Such microstep loss may be difficult to observe without measuring the movement of the linear actuator. The utilized stepper motor driver (TMC2130) has features such as adaptive excitation current control which may be used to prevent such behaviour. Future revisions of the Ystruder should implement these features to avoid microstep loss and maximize torque.

Given the modular design of the Ystruder the stepper motor could be replaced with a finer pitch version and a higher current stepper motor driver. Naturally, higher stepper driver currents require adequate cooling to function properly. Additionally, a higher pitch screw requires higher step signal frequency (pulses per second). These factors influence the dynamic capabilities of the syringe pump. Future revisions of the Ystruder design should quantify the performance of different configurations. Further information on sizing of the stepper motors is given in the Supplementary information.

7.2. Accuracy and repeatability

To estimate the accuracy and repeatability of the Ystruder, silicone oil of two different kinematic viscosities (1000 and 10,000 cSt) was dosed with two different flow rates (0.2 and 1 mL/min) through two different syringe orifice diameters (0.5 and 1.8 mm) resulting in a total of eight dosing parameter sets. Each parameter set was run for a 2 min dosing cycle with a minimum post dosing period of 1 min to allow the higher viscosity fluid to flow out and syringe pressure to equalize.

Herein, the terms extruded volume and flow rate refer to the volume derived from the mass (scale output). The density of the utilized silicone oils was determined a precision scale (Mettler) and a 10 mL graduated measuring cylinder. Displacement refers to the volume calculated from the linear movement and cross-sectional area of the syringe. The cross-sectional area was measured with a digital caliper (Mitutoyo) in an unpressurized state. Speed refers to the rate of change of displacement. The flow rate and speed are determined by fitting a one term polynomial on the linear section of the curve and calculating its slope. The results shown are derived from 10 parallel measurements.

The piston force ranged from 2.1 N to 223 N depending on the viscosity, syringe diameter and flow rate. The highest extruded volume error (41%) was measured with 10,000 cSt silicone oil extruded through a 0.5 mm orifice at 1 mL per minute. Given the audible skipping during these tests, the error was contributed to exceeding the maximum torque of the stepper motor resulting in stalling. Apart from the test run where the maximum torque was exceeded, all the displacement errors were below 1%.

Syringe pumps have a slow response time especially at low flow rates and the response time is dependant on the flow behaviour of the extrudate, the flow restriction of the circuit it is connected and the dimensions of the utilized syringe [40–42]. Fig. 10 shows the relationship between displacement and extruded mass in four different parameter sets with different viscosities and syringe sizes. The piston force is plotted in the same figure on a second vertical axis.

As shown in Fig. 10a the extruded volume signal corresponds well to the movement in situations where the extrusion pressures (i.e. flow restrictions) and extrudate viscosity are low. The difference in the initiation of movement and the initiation of flow is negligible. With increased flow constraint and slower speeds the delays increase. This can be seen in Fig. 10b–d. In other words, the system accumulates pressure until the flow initiates. Similarly, once the piston seizes the move, the pressure dissipates slowly as the material oozes out.

Fig. 10c demonstrates how a slow extrusion with a high flow restriction and viscosity results in a high flow rate error. Given a 3 min dosing cycle with two minutes of movement and 1 min post dosing period, the system does not achieve a steady state as shown by the lack of saturation in the force signal. Despite a low displacement error (0.51 ± 0.17%) the system does not attain a steady state and the flow rate determined from the linear part of the curve has a relative error of 30.5 ± 3.8% compared to the target flow rate. Similar behaviour has been observed by Banovic et al. [30].
Fig. 11 shows the mean relative error and standard error of the mean thereof of the displacement, extruded volume, flow rate and speed of test runs where the extrusion attained a steady state. The low displacement error indicates that the stepper motor is accurately performing the desired movement in cases where the maximum torque is not exceeded. However, the errors regarding displacement and speed do not directly correspond to the errors of extruded volume and flow rate. While there is a correlation between error values between movement and extruded volume, the mass derived error values are higher. This indicates that the relationship between displacement and extruded volume is not ideal. As Fig. 11 also demonstrates, the flow rate error (rate of change in mass) has the highest error compared to other sources.

The compliance of the syringe assembly influences the accuracy of the syringe pump. The syringe components (piston, gasket and body) are manufactured out of thin polypropylene or rubber, and they exhibit elastic deformation even under moderate loads. Syringe and infusion line compliance is a well-known issue in precise drug administration and fluid therapy regulation [43–45]. Increasing the rigidity of the syringe and improving the gasket geometry, as suggested by Banovic et al. [30], can be used to mitigate issues stemming from syringe compliance. Plastic three component syringes are not rated for high pressure applications. A typical maximum pressure for a polypropylene and natural rubber syringe is 125 psi (8.6 bar).
While the pressure can be increased further the deformation (both elastic and plastic) will decrease the dosing accuracy and repeatability. The linear actuator itself exhibits inaccuracies. As most screw mechanisms, the linear stepper motor also exhibits backlash. The low errors in displacement suggest that in comparison to stalling, step loss, syringe compliance and extrudate flow behaviour related errors, the errors stemming from the screw mechanism are substantially lower. Scale based calibration and tuning of the stepper motor driver parameters is strongly recommended to achieve higher dosing accuracies.

7.3. 3D printing

To validate the performance of the Ystruder in 3D printing it was attached to a CNC-machine frame via a 43 mm spindle (configuration shown in Fig. 1c). Various models printed from different raw materials are shown in Fig. 12. The printing and slicing parameters used in printing are listed in the Supplementary information.

Printing with paste-like materials is inherently limited by the slow solidification of the extrudate. Consequently, printing even moderate overhangs is challenging. Printing errors resulting from this behaviour can be seen in Fig. 12a. Such issues have been successfully addressed with e.g. UV-curing during extrusion [28] or printing into support baths [23] in past research. Fig. 12b and c are simpler geometries produced from 2-dimensional vectors which are easier to manufacture using paste based 3D printing. Consequently, they exhibit less printing imperfections.

In summary, the Ystruder performs well both as a syringe pump and as a paste extruder for 3D printing. The load monitoring facilitates a better understanding of the extrusion and enables development of more sophisticated dosing algorithms. Such dosing algorithms are especially promising in the context of 3D printing where extrusion occurs intermittently and extrudates may exhibit non-Newtonian behaviour.

8. Discussion and future work

The syringe pump remains one of the most popular devices in OSHW. Despite its conceptual simplicity, inspecting the operation of a syringe pump more closely reveals several factors which influence its performance. In this work, we demonstrated how force measurement during extrusion can be used both to characterize the extrudates and to evaluate the influence of piston force on performance.

The versatile design presented here can be used both as a pump and as an extruder in a 3D printing system. Ystruder has the potential to be used in demanding applications such as 3D bioprinting. While the design is already sufficient for a wide variety of extrusion tasks, there are both hardware and software modules that require additional development. We distinguish four key areas of improvement, error quantification, control development, interface improvement and extension of modularization.

Firstly, as shown in Fig. 11 there are both systematic and stochastic dosing errors. Evident errors such as stalling of the stepper motor are easy to detect but issues such as the flow rate error in transient stages (as shown in Fig. 10) are more
elusive. In continuous pumping scenarios the errors associated with such transients are minor, and they can be reduced by applying simple compensation factors in the stepper motor actuation. The relative errors regarding displacement and extruded volume are below 1% and respective error values regarding speed and flow rate are between 1 and 4%. The higher error of the rate values is likely caused by microstep loss and compliance of the syringe. Further measurements are required regarding error and uncertainty quantification.

Secondly, in more dynamic extrusion tasks, such as 3D printing, hybrid velocity-force control could be used to improve extrusion accuracy and latency. Such control schemes have been developed for 3D printing paste-like materials in the context of freeze-form extrusion fabrication [47–49]. These control methods improve both the latency and precision of dosing by incorporation both the extrusion force and piston velocity into a model-based control algorithm. The Ystruder design and accompanying measurement and analysis utilities facilitate development of such control schemes.

Thirdly, from a usability perspective, both the display and browser interfaces require improvement. The browser interface features basic monitoring and data acquisition. Future work should aim to extend its functionality. For example, tuning of the dosing parameters and compensation factors regarding error minimization could be performed interactively in the browser. Ideally, these tuned parameters could be shared openly between different users.

Finally, the modularization of the Ystruder design should be extended. The design can already be easily modified by e.g., replacing the stepper motor with a finer pitch version or upgrading to a higher current capacity stepper motor driver. The sizing information provided in the Supplementary information is intended to accommodate adapting the design to different extrusion requirements. However, further development is required in parametrization of the CAD models to facilitate such adaptations. This parametrization work should also involve development of a standardized add-on system for the Ystruder regarding e.g., heating and UV-curing.

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

This project was supported by the Aalto University, School of Engineering Doctoral programme.

Human and animal rights

This work does not involve the use of any human or animal subjects.

Appendix A. Supplementary data

Supplementary data associated with this article can be found, in the online version, at https://doi.org/10.1016/j.ohx.2019.e00080.

References
