Journal of Visualized Experiments
Manufacturing, control and performance evaluation of a gecko-inspired soft robot
--Manuscript Draft--

Article Type:	Invited Methods Article - JoVE Produced Video
Manuscript Number:	JoVE61422R2
Full Title:	Manufacturing, control and performance evaluation of a gecko-inspired soft robot
Section/Category:	JoVE Engineering
Keywords:	mobile soft robots, fast pneu-nets, gecko-inspired robot, climbing robot, soft robotics, soft robot applications, natural machine motion
Corresponding Author:	Lars Schiller Technische Universitat Hamburg Hamburg, Hamburg GERMANY
Corresponding Author's Institution:	Technische Universitat Hamburg
Corresponding Author E-Mail:	lars.schiller@tu-harburg.de
Order of Authors:	Lars Schiller
	Arthur Seibel
	Josef Schlattmann
Additional Information:	
Question	**Response**
Please indicate whether this article will be Standard Access or Open Access.	Open Access (US$4,200)
Please indicate the **city, state/province, and country** where this article will be **filmed**. Please do not use abbreviations.	Hamburg, Hamburg, Germany

Dear Editorial Staff,

Please find attached the protocol *Manufacturing, control and performance evaluation of a gecko-inspired soft robot*.

Best regards
the authors

TITLE:

Manufacturing, Control and Performance Evaluation of a Gecko-Inspired Soft Robot

AUTHORS AND AFFILIATIONS:

Lars Schiller[1], Arthur Seibel[2], Josef Schlattmann[1]

[1]Workgroup on System Technologies and Engineering Design Methodology, Hamburg University of Technology, Hamburg, Germany

[2]Fraunhofer Research Institution for Additive Manufacturing Technologies IAPT, Hamburg-Bergedorf, Germany

Corresponding Author:

Lars Schiller (lars.schiller@tuhh.de)

Email Address of Co-authors:

Arthur Seibel (arthur.seibel@iapt.fraunhofer.de)
Josef Schlattmann (j.schlattmann@tuhh.de)

KEYWORDS:

mobile soft robots, fast pneunets, gecko-inspired robot, climbing robot, soft robotics, soft robot applications, natural machine motion

SUMMARY:

This protocol provides a detailed list of steps to be performed for the manufacturing, control and evaluation of the climbing performance of a gecko-inspired soft robot.

ABSTRACT:

This protocol presents a method for manufacturing, control and evaluation of the performance of a soft robot that can climb inclined flat surfaces with slopes of up to 84°. The manufacturing method is valid for the fast pneunet bending actuators in general and might, therefore, be interesting for newcomers to the field of actuator manufacturing. The control of the robot is achieved by means of a pneumatic control box that can provide arbitrary pressures and can be built by only using purchased components, a laser cutter, and a soldering iron. For the walking performance of the robot, the pressure angle calibration plays a crucial role. Therefore, a semi-automated method for the pressure-angle calibration is presented. At high inclines (> 70°), the robot can no longer reliably fix itself to the walking plane. Therefore, the gait pattern is modified to ensure that the feet can be fixed on the walking plane.

INTRODUCTION

The interaction between humans and machines is becoming constantly closer. The increasing robot density in companies and households poses new challenges for the robot technology. Frequently, dangers are excluded by separation methods, but in many areas, especially in households, this is not a satisfactory solution. Soft robotics tackle this problem by using properties of soft materials and structures to develop new types of machines that behave like living organisms[1], which is why soft robots are often inspired by biological models[2]. Most soft robots can be classified into two different types: mobile robots and robots designed for gripping and manipulation[3]. For soft mobile robots, typical locomotion principles are crawling, walking, running, jumping, flying, and swimming[4]. Another interesting field of application for soft robots is climbing – a combination of locomotion and adhesion[5]. Soft machines are very robust and cannot damage their surroundings due to their softness. This characteristic predestines this robot class for climbing, as they can easily survive a fall. Consequently, the literature offers several examples of soft robots capable of climbing[6,7,8].

The goal of this protocol is to provide a method to manufacture, control and evaluate the performance of a gecko-inspired, climbing soft robot[9]. Its design is based on the use of fast pneunet soft bending actuators[10] made of elastomer. However, another soft actuator design and/or material could also be used. The literature offers a wide range of different designs of soft actuators[11] and suitable materials[12]. The presented manufacturing method is similar to existing methods[13] but includes some modifications that result in increased repeatability and robustness, at least in the case of the soft climbing robot[9]. The method is valid for fast pneunet bending actuators in general and might, therefore, be interesting for newcomers to the field of actuator manufacturing.

For controlling pneumatic actuated soft robots, the literature provides different solutions. It ranges from low-cost and easy-to-replicate control boards[13] to powerful but more complex boards[14] which cannot be rebuilt without special tools. Here, a brief description is provided for building a pneumatic control box by only using a laser cutter and a soldering iron. The control box allows the supply of any pressure and offers real-time sensory feedback, which is especially important for robotics applications. However, it can also be used for many other applications.

PROTOCOL:

1. Printing of molds

1.1. Download the *.stl data for molds from the **Supplementary Data 1** “CAD/Moulds/”.

1.2. Use the printer-specific slicing software to convert the 3D models into a print job.

1.3. Print the molds using a 3D printer.

1.4. Clean the printed molds by putting them for 15 min in an ultrasonic bath.

1.5. Put the molds for at least 3 h in a UV chamber.

2. Preparing the elastomer

2.1. Gather the following before starting this step: elastomer (part A and part B), spatula, plastic cup, mold, weight scale, plastic syringe, screw clamps (or similar), acrylic glass plate with two corresponding holes, cutter knife.

2.2. Mix part A and part B of the elastomer in a 1:9 ratio in a cup. Place the cup on a weighing machine. First add 5 g of part B (dark red). Then, using a spatula, add 45 g of part A (white and viscous)

NOTE: Ensure the accuracy of weighing is to 1 g. 50 g is enough for one leg. The best way for portioning part A is to take a spatula and let it drain. Approximately 6 g per draining operation is possible with the spatula used.

2.3. Keep stirring until no more white or red areas are visible at the edge of the cup.

2.4. Put the cup for 15 min in a vacuum chamber to remove the air that was trapped in the elastomer due to the stirring process.

2.5. Fill the mixed elastomer into a plastic syringe. This allows the elastomer to be positioned much more precisely.

NOTE: **Supplementary Figure 1** illustrates the processing steps described in this section.

3. Manufacturing of upper part (base part)

3.1. Clamp an acrylic glass plate with two corresponding holes onto the mold. Insert the syringe into the lower hole and press the elastomer into the mold.

3.2. Apply force to the syringe by pushing the plunger until the mixed elastomer emerges from the upper hole.

3.3. Loosen the screw clamps and pull off the acrylic glass plate sideways.

NOTE: It is important to pull it off to the side and not upwards. Otherwise, the elastomer will be pulled out of the mold as well.

3.4. Puncture the rising air bubbles with a sharp tool. Do not puncture too deeply as this will create new air bubbles rather than removing the existing ones. It is especially important to pierce the larger bubbles as these will later affect the functionality of the actuator significantly.

NOTE: Optionally, evacuate the filled mold in the vacuum chamber to remove any air still trapped. When doing so, however, it can happen that the rising air bubbles get stuck on the mold on their way to the surface and create holes in the casting at functionally relevant areas. **Supplementary Figure 2** illustrates this phenomenon.

3.5. Put the mold into the oven at 65 °C for 30 min.

3.6. Check after 10 min if the level of the elastomer has fallen significantly. This happens if the mold is not completely tight or has bent slightly due to frequent use. If the level has dropped more than 1 mm, refill the elastomer. Then, continue curing.

3.7. After a total of 30 min in the oven, take out the mold and cut off the extruded elastomer with a cutter knife.

3.8. Open the mold by levering apart with a screwdriver being careful to not damage surfaces relevant for casting.

3.9. Remove the almost finished actuator from the part of the mold to which it had stuck in the previous step.

NOTE: A first visual check can be made here to see if casting was successful. If irreparable defects are found (see **Supplementary Figure 3**), the manufacturing process is to be stopped here. Smaller holes can be repaired later. It is also important that the sealing lip is as pronounced as possible over its entire circumference.

3.10. Cut off any protruding burrs with a cutter knife. This is sometimes very laborious, but essential for a good final result.

NOTE: **Supplementary Figure 4** illustrates the processing steps described in this section. The described steps are valid for casting the four legs (mold can be found in **Supplementary File 1** "CAD/Moulds/small_leg_schwalbe*.stl") and the two base parts of the torso ("CAD/Moulds/small_belly*.stl"). To cast the suction cups (feet of the robot, to be found in "CAD/Moulds/suctionCup*.stl") or the bottom part of torso ("CAD/Moulds/small_torso_base1*.stl"), carry out the same process steps, with the exception of steps 3.1 and 3.3, as these molds for casting have a built-in port for the syringe and therefore no additional acrylic glass plate is required. In total, build four base parts of leg, two base parts of torso, one bottom part of torso and four suction cups.

4. Manufacturing of lower part (bottom part)

4.1. Push a silicone tube through the holes provided for this purpose in the mold of the bottom part, see **Supplementary Figure 5**.

4.2. Fill the mold of the base part with elastomer and distribute it with the small spatula up to the corners.

NOTE: The level of the elastomer should be no higher than 5 mm and no lower than 4 mm and must completely cover the embedded tube. The mold for the bottom part of the legs can be found in **Supplementary File 1** "CAD/Moulds/small_base_schwalbe.stl".

4.3. Put the mold into the oven for 15–20 min for curing. For the following steps, it is necessary that the bottom part remains in the mold for the time being joined with the top part.

5. Joining the base and bottom part

5.1. Fill the mold of the bottom part with elastomer so that the level is 1–1.5 mm above the already hardened elastomer.

5.2. Insert a butterfly cannula into the base part and mark the puncture site so that it can be found more easily later. This step is necessary to allow the expanding air in the oven to escape.

5.3. Place the base part into the bottom mold and press only the sides slightly into the elastomer bath.

5.4. Put the actuator into the oven for 10–15 min and remove the mold afterwards.

NOTE: It should be easy to remove the actuator from the mold. If it fails to do so, either the elastomer is not yet fully cured (in this case, increase the curing time by 10 more min) or the bottom part is stuck in the mold (in this case, it should be pulled harder). But in general, it is a bad sign if the actuator cannot be released easily.

5.5. Connect a pressure source by using the puncture site from step 5.2 and perform the final leakage test, see **Supplementary Figure 6**.

NOTE: If small leaks are present, they can be repaired. Application of a little elastomer with a small spatula and 10 min in the oven should fix the leak. If all leaks are fixed, the actuator is ready. **Supplementary Figure 6** illustrates the processing steps described in this section and **Supplementary Figure 7** illustrates the entire process described in Sections 3–5. For joining the base and bottom part of the torso, perform the same steps, with exception of step 5.1, where you do not fill the mold but the bottom part directly.

6. Joining of all limbs

6.1. Fix the parts to be joined with a pin needle on a wooden board so that they can be held together in the following process step.

6.2. Cover the joining surface with elastomer as shown in **Supplementary Figure 8A**. Make sure that the joining surface is clean and free of fat. Otherwise, the parts will delaminate at this point.

6.3. Put the assembly (see **Supplementary Figure 8B**) for 10–15 min into the oven.

7. Mounting of supply tube inlets

7.1. Widen the insertion point of the butterfly cannula from step 5.2 further by using a 1 mm

Allen key.

7.2. Place the end of a silicone tube with a maximum outer diameter of 3 mm over the hole and press it in with the Allen key.

7.3. Seal the inlet with a little elastomer. This also protects against mechanical stress.

7.4. Put the assembly for 10 min into the oven.

NOTE: **Supplementary Figure 9** illustrates the processing steps described in this section.

8. Building the control box

8.1. Download the corresponding *.dxf drawings for the housing from **Supplementary Data 1** “CAD/ControlBox/” and cut them out on a laser cutter.

8.2. Assemble the “User Interface Unit” on the front panel according to **Supplementary Figure 10A** and **Supplementary Figure 11**.

8.3. Build the six “Valve Units” according to **Supplementary Figure 10B** and **Supplementary Figure 12**.

8.4. Assemble the six “Valve Units” and the “User Interface Unit” on the bottom panel according to **Supplementary Figure 10C, Supplementary Figure 13** and **Supplementary Figure 14**. Assemble the two side panels and the back panel. Last, assemble the top panel.

8.5. Configure the two single-board computers embedded in the control box according to **Supplementary File 1** and upload the complete folder “Code” (including all subfolders) provided in the **Supplementary Data 2** onto both boards.

8.6. Upload the script “Code/arduino_p_ctr.ino” provided in the **Supplementary Data 2** onto the six micro-controllers embedded in the control box.

9. Building a test bench with embedded measurement system

9.1. Download the corresponding *.dxf drawing of the camera holder from **Supplementary Data 1** “CAD/TestBench/” and cut it out on a laser cutter.

9.2. Download the corresponding *.stl files of the clamps from **Supplementary Data 1** “CAD/TestBench” and print them on a 3D-printer.

9.3. Assemble the camera holder with the clamps on a DIN-A1 poster panel according to **Supplementary Figure 15** and mount the camera and a single-board computer at the intended location.

9.4. Configure the ethernet interface and the SSH settings of the single-board computer according to sections 4–5 of **Supplementary File 1** and upload the complete folder "Code" (**Supplementary Data 2**) onto the board.

10. Setting up the entire system

10.1. Create a local network and assign the correct IP address from the script "Code/main.py" to all single-board computers and the computer used for monitoring – or rewrite the script accordingly.

10.2. Insert pin needles into both ends of the torso as shown in **Supplementary Figure 16**, so that the robot only contacts the walking plane with the pins and its feet (suction cups).

10.3. Print out the visual markers[15] provided in **Supplementary File 2** on a DIN-A4 sheet and cut them out using a scissor.

10.4. Attach the markers to the robot using pin needles according to **Supplementary Figure 17**.

10.5. Connect the robot to the control box.

NOTE: **Figure 1** illustrates the wiring of the entire system.

11. Running the control box

11.1. Power on the main switch of the control box and wait until everything is booted.

11.2. Log into the main single-board computer as "root" using SSH, browse to the folder "Code", and start the control box by the command "*root@beaglebone:~# python3 main.py*". At the same time, start the monitor on the personal computer by the command "*user@pc:~ python2 monitor.py*".

NOTE: Both programs must start more or less at the same time. The program "main.py" running on the single-board computer in the control box tries to connect to the personal computer used for monitoring. If there is no listening port at the personal computer (triggered by the script "monitor.py"), the monitor will not start. Except of "monitor.py", all programs/scripts used in this protocol are intended to run with python3.

11.3. Connect a pressure source to the control box (max. 1.2 bar).

11.4. Connect a vacuum source to the control box.

12. Calibrating the robot

12.1. Place the robot on the walking plane of the test bench. For steep inclines, attach a string between the robot's front and the top of the walking plane in order to hold the robot in place.

12.2. On the control box, activate the "**pattern reference**" mode by pushing the "**mode 2**" button as shown in **Supplementary Figure 18**.

12.3. Scroll through the menu displayed on the LCD by using the up and down buttons until you find the entry "clb". Then, push the enter button.

12.4. Scroll through the next menu up to the entry "mode_4.csv" and push the "enter" button.

12.5. On the monitor, press the "**record**" button as shown in the **Supplementary Figure 19**.

NOTE: Pressing the "**record**" button will automatically create a *.csv file on the monitoring computer at the location specified in "Code/Src/GUI/save.py:save_last_sample_as_csv()", which is the folder "current_exp" (example measurements are provided in **Supplementary Data 3**).

12.6. On the control box, press the "**function 1**" button to start the calibration procedure.

12.7. After calibration, press the "**record**" button on the monitor to stop recording and the "**function 1**" button on the control box to stop the pressure controller.

12.8. Rename the automatically created "current_exp/*.csv" file so that it can be uniquely identified later.

12.9. Run the script "Calibration/eval_clb.py" provided in the **Supplementary Data 4** and store the output (coefficients of the polynomial fit) in the file "Code/Src/Controller/calibration.py" as an entry with the keyword "[robot version]" within the existing dictionary.

13. Creating a gait pattern

13.1. Run the script "Code/Patterns/create_pattern.py" and store the outputted *.csv file(s) in the folder "Code/Patterns/[robot version]/".

NOTE: This script converts the predefined gait pattern for straight gait[8] (see **Supplementary Figure 20A** or **Supplementary Animation 1**) formulated in angle references into robot-specific pressure references. To generate a gait pattern for steep inclines, modify the script by uncommenting line 222. This will generate a pattern according to **Supplementary Figure 20B** or **Supplementary Animation 2**. The interface for pattern references provided by the control box consists of *.csv files where each row defines a discrete set point for all actuators. Therein, the first eight columns define the reference pressures, the next four columns define the references for the direct acting valves, and the last column defines the time this set point should be hold.

13.2. Synchronize the single-board computer in the control box with the personal computer, i.e., upload the folder “Code/Pattern/*” onto the board. For this purpose, the program “main.py” has to be interrupted (Ctrl+C).

14. Carrying out the climbing experiment

14.1. Perform the steps 11–13 for each inclination to be tested.

14.2. Place the robot at the marked point on the walking plane.

14.3. Select a pattern reference as described in steps 12.2–12.4, but select in the first menu the desired “robot version” (instead of “clb”), and in the second menu the pattern reference according to the current inclination (instead of “mode_4.csv”).

14.4. Start recording as described in step 12.5.

14.5. Push the “**function 1**” button to activate the pressure controller.

14.6. Let the robot walk/climb for at least 6 cycles.

14.7. Stop recording by pushing the “**record**” button on the monitor (like in step 12.7).

14.8. Make sure the robot will not fall when executing the next step.

14.9. Stop the pressure controller by pushing again the “**function 1**” button. This will also stop the vacuum supply, and consequently the robot will fall.

14.10. Move the recorded *.csv file into the folder “ExpEvaluation/[robot version]/[pattern type]/[inclination]/”.

NOTE: Repeat each run at least five times in order to have a solid base for the next step.

15. Evaluating the experiment

15.1. Run the script “ExpEvaluation/eval_vS11_adj_ptrn.py” provided in **Supplementary Data 5** to automatically mean over all measurement data.

NOTE: This script outputs the track of all feet, the applied pressure over time, the measured bending angle of all limbs over time, the velocity of the robot over time, the orientation of the robot over time, the mean velocity over inclination (cf. **Figure 2A**), and an approximation of the energy used over inclination (cf. **Figure 2B**).

REPRESENTATIVE RESULTS:

The presented protocol results in three things: a soft climbing robot, a universally applicable control box, and a control strategy for the robot's straight motion that increases its ability to climb and at the same time decreases its consumed energy. The control box described in section 8 enables a continuous supply of any desired pressure level on up to six channels (expandable to eight) and additionally on four channels the supply of vacuum (expandable as required). The "User Interface Unit" enables the user to easily operate the control box at runtime and the interface to the monitor allows the measured data to be directly viewed and saved as a csv-file. The pattern-reference mode of the control box provides the user with an intuitive interface to loop predefined patterns. This can be the gait pattern of the robot, as in this protocol, or it can be used for actuator fatigue testing, or any other application that requires cyclic loading. **Figure 1** depicts all hardware components assembled in the control box and the measurement system and how they are connected.

The gait pattern for the robot's straight motion is formulated in angular references[8]. To operate the robot, those angular references must be converted into pressure references. The control strategy used in this protocol is based on a prior angle-pressure calibration. Each method of calibration results in a different alpha-pressure curve. Therefore, it is necessary to adapt the calibration procedure to the real operating conditions as far as possible. When changing the inclination angle of the walking plane, the operating conditions change as well. Therefore, the angle-pressure curve must be re-calibrated for each inclination. **Figure 2A** shows the velocity of the robot for various inclines with an unchanged calibration and a re-calibrated angle-pressure curve. The experiment clearly shows the effectiveness of the re-calibration. The re-calibrated robot was not only way faster, it was also able to climb steeper inclines (84° instead of 76°) while consuming less energy[9] as depicted in **Figure 2B**. In **Figure 3,** a series of photographs of the robot's motion is shown for an inclination of 48°. The figure clearly illustrates that the climbing performance with re-calibration shown in **Figure 3B** is much better than with unchanged calibration shown in **Figure 3A** as the shift in position within the same time interval is almost twice as large. This robot can move very fast compared to other soft robots. Qin et al.[7] summarize the forward velocity of various soft robots. Without payload and in the horizontal plane, the robot described in this protocol was five times faster in relation to the body length than the fastest robot in Ref.[7].

FIGURE AND TABLE LEGENDS:

Figure 1: **Diagram of hardware components assembled in the control box.** Therein $\acute{p}_i$ denotes the pressure reference for the *i*-th channel, $\acute{\alpha}_i$ the angular reference for the *i*-th channel, u_i the control signal of the *i*-th proportional valve, α the vector containing the angle measurements, x the vector containing the position measurements, and f the vector containing control signals for the direct-acting solenoid valves, i.e. the fixation states of the feet. UI is short for "User Interface Unit", BBB is an abbreviation for BeagleBone Black, i.e., the single-board computer used in the control box and RPi is short for Raspberry Pi, i.e., the single-board computer used in the measurement system.

Figure 2: Evaluation of climbing performance. Dashed curves show the values for constant and solid curves for recalibrated pressure references. (**A**) Forward velocity of the robot for various

inclination angles. (**B**) Energy consumption for various inclination angles. This figure is adapted from ref.[9].

Figure 3: Series of photos of the robot's motion at an inclination of 48°. The time elapsed between each photo is 1.2 s. (**A**) The motion for constant pressure references and (**B**) the motion for recalibrated pressure references.

SUPPLEMENTARY LEGENDS:

Supplementary Figure 1: Preparation of the elastomer.

Supplementary Figure 2: Comparison of air bubble formation during evacuation before and after casting. (**A**) Evacuation of the elastomer was performed only before casting. Trapped air bubbles stay in place, but they were more in the area of the bumps, which does not greatly affect the actuator's functionality. (**B**) Evacuation was performed before and after casting. Trapped air bubbles rise but get stuck again on the upper side of the struts and create holes in the actuator which can affect the functionality.

Supplementary Figure 3: Examples of successful and not successful cured castings. Upper row shows successful examples and lower row unsuccessful examples. If the defect was not clearly recognizable, it was marked with a green circle.

Supplementary Figure 4: Manufacturing of the base part.

Supplementary Figure 5: Scheme for manufacturing the bottom part. A tube (which is later used as the supply tube for the suction cup) was clamped into the mold before casting. Then, the mold was filled with liquid elastomer.

Supplementary Figure 6: Joining of base and bottom part.

Supplementary Figure 7: Lamination casting of a soft bending actuator. Liquid elastomer is represented in red, cured elastomer in light red, and the strain-limiting layer, as well as the molds in black. (**A**) Mixed elastomer was poured into two separate molds – one for the base part and one for the bottom part. Thereby, the bottom part was only half filled. A strain-limiting layer (supply tube) was then inserted into the bottom part mold. (**B**) The parts were cured and the base part was demolded. (**C**) The bottom part mold was filled to the top with liquid elastomer. (**D**) The base part was dipped into this mold. (**E**) The two parts were cured together. (**F**) The actuator was demolded. This figure is based on ref.[13].

Supplementary Figure 8: Joining of all limbs. (**A**) Covering the surfaces to be joined with fluid elastomer. (**B**) Rendered view of the complete assembly.

Supplementary Figure 9: Mounting the supply tube inlets.

Supplementary Figure 10: Photographs of the control box. (**A**) Front view of the User Interface Unit for enabling the user to interact with the robot. (**B**) Detail view of a Valve Unit. (**C**) Top view of the entire control box.

Supplementary Figure 11: Circuit diagram of the User Interface Unit.

Supplementary Figure 12: Circuit diagram of the Valve Unit.

Supplementary Figure 13: Simplified circuit diagram of the entire control box.

Supplementary Figure 14: Diagram of used pins of the single-board computers embedded in the control box. (**A**) Used pins of the board needed for user communication. (**B**) Used pins of the board needed for robot control.

Supplementary Figure 15: Rendered view of the walking plane with installed measurement system.

Supplementary Figure 16: Visualization of the lifting effect. Pin needles with 6 mm heads were inserted into both ends of the torso. This minimizes friction during walking and causes the suction cups to have full contact with the walking plane.

Supplementary Figure 17: Assembly of the visual markers. The markers were mounted on the robot by using pin needles. Marker 0 was mounted at the front left foot, marker 1 at the torso's front, marker 2 at the front right foot, marker 3 at the rear left foot, marker 4 at the torso's back, and marker 5 at the rear right foot. For the assembling of marker 4, three pin needles were used This figure is adapted from ref.[9].

Supplementary Figure 18: Legend of buttons of the control box.

Supplementary Figure 19: Legend of buttons of the Graphical User Interface.

Supplementary Figure 20: Gait patterns for straight movement of the robot. Fixed feet are indicated by filled circles and unfixed feet by unfilled circles. (**A**) Gait pattern for low and moderate inclination angles (<70°). (**B**) Gait pattern for high inclinations (>70°). Vacuum was applied to red and black filled feet. Black filled feet were fixed to the ground, whereas red feet do not necessarily have to be. In order to secure the fixation, the foot to be fixed was swinged back and forth once. This figure is adapted from ref.[9].

Supplementary Figure 21: Rendered explosion view of the soft climbing robot. Dovetails were located at the legs and corresponding keyways at the torso's ends. This makes the joining process much more precise This figure is adapted from ref.[9].

Supplementary Figure 22: Different calibration procedures for the determination of the pressure-angle curve. Each subfigure shows the qualitative pressure course and snapshots of the corresponding robot pose. (**A**) Each actuator was inflated continuously beginning from 0 bar up to 1 bar, while all others remain pressureless. (**B**) A pressure plateau was applied to a single actuator for 3 s; then, it was deflated completely for 2 s. In the next round, the level of the pressure plateau was increased by the increment until the plateau reaches 1 bar. This was done for each actuator individually. (**C**) Same procedure as in mode 2, but here, the same plateau was applied to actuators (0,3,4), respectively actuators (1,2,5), at the same time. (**D**) Same procedure as in mode 3, but plateaus for actuators (0,3) were starting at 0 bar (like before) and ending at 1.2 bar (instead of 1 bar). Basically, the increment for actuators (0,3) was slightly increased, while the increments for the other actuators remain the same.

Supplementary Figure 23: Angle-pressure curves for different calibration procedures.

Supplementary Animation 1: Animation of the robot's straight gait.

Supplementary Animation 2: Animation of the robot's climbing gait.

Supplementary File 1: Instructions for configuring the single-board computers.

Supplementary File 2: Print template for the visual markers.

Supplementary Data 1: CAD files. This zip-compressed folder contains the *.stl-files for printing the molds, the *.dxf-files for laser cutting the housing of the control box, the *.stl-files for printing the clamps used for the measurement system, and *.dxf-file for laser cutting the frame of the measurement system.

Supplementary Data 2: Code to run on the single-board computers. This zip-compressed folder contains the programs and their sources running on the board used for the "User Interface Unit", the board used for robot control, and the board used for image processing. Upload the complete folder to all three boards.

Supplementary Data 3: Exemplary measurement data. This zip-compressed contains two *.csv files generated during the calibration procedure.

Supplementary Data 4: Calibration script. This zip-compressed folder contains the python script and its sources for evaluating the measurement data generated during the calibration procedure.

Supplementary Data 5: Evaluation script. This zip-compressed folder contains two python scripts and their sources for evaluating the measurement data generated during the climbing experiment. In addition, it contains all the measurement data used for the generation of **Figure 2**.

DISCUSSION:

The presented protocol includes many different aspects related to the climbing soft robot, including manufacturing, control, calibration, and performance evaluation. In the following, the pros and cons resulting from the protocol are discussed and structured according to the aspects mentioned above.

The presented manufacturing method is strongly based on the existing literature[10,13]. A substantial difference is the design of the actuator. To join the individual limbs, dovetail guides were inserted at the appropriate points, as shown in **Supplementary Figure 21**. This results in a much more precise and robust connection between the limbs compared to previous designs of the robot. Furthermore, the supply tubes were embedded in the bottom part of the actuators. This integrated design allows the suction cups to be supplied with vacuum and at the same time makes the bottom layer no longer stretchable, which significantly increases the performance of the actuator. Another difference to the procedure described in the literature is that the mixed elastomer is evacuated only once (immediately after mixing). Many sources recommend evacuating the elastomer twice: once after mixing and once after it has been filled into the mold. It may happen that air remains trapped in very small spaces. In the vacuum chamber, this air expands and in the best case rises to the surface. Often enough, however, these air bubbles get stuck on their way, creating unpleasant holes in the finished casting. Here, a decision must be made as to what is more important: perfect contours on the bottom side of the base part or as little risk as possible of producing a non-functional actuator, (see **Supplementary Figure 2)**. In this protocol, no second evacuation was performed. In the procedure presented, the height of the bottom part may vary as it is filled manually, and, unlike for the base part, there is no possibility of cutting it to a uniform height after curing. In order to ensure that the height of the bottom part is as uniform as possible, it is recommended to use a syringe when filling the mold of the bottom part and to measure the volume poured in. However, depending on how much time has elapsed since mixing, the flow properties of the elastomer change significantly. Therefore, it is recommended to always use freshly mixed elastomer. Joining the base and the bottom part of the actuator involves the largest process uncertainty. If the elastomer bath is too high, the air channel between the chambers will most likely be covered as well. Then, the actuator is no longer usable. If the elastomer bath is too low, the sealing lip may not be covered in its entire circumference and the actuator would leak. Therefore, it takes a certain amount of practice to dose the elastomer bath correctly. Important for joining in general is a fat-free joining surface. If the joining surface is too contaminated, the finished actuator may delaminate. Therefore, it is essential to ensure that the parts are only touched on surfaces that are not to be joined. A major limitation of the manufacturing method is the number of pieces to be realized. The production of a single actuator takes at least two hours in total. Although it is possible to work with several molds in parallel, more than four is not recommendable due to time constraints. The pot life of the elastomer is too short to be able to fill even more molds. In addition, the molds only withstand a limited number of production cycles (approx. 10–20) before they become very deformed or break. A further limitation is the process uncertainties already discussed. Since almost all process steps are performed manually, each actuator is a little different. This can lead to two robots that are identical in construction but show two very different behaviors.

With the control box a method was provided to control the robot. Nevertheless, for each pneumatic system the control gains of the script "Code/arduino_p_ctr.ino" must be determined individually. This was not covered in the protocol. However, the "pressure reference mode" of the control box allows a playful handling of the robot, so that controller tuning can be made without writing several scripts. Another limitation of the control box is its cost, as the material costs about 7000 US$ in total. The literature[11] offers a building instruction for a control box that costs only about 900 US$ and with some upgrades could also be used to operate the robot.

Critical for the calibration of the individual actuators is the choice of the calibration procedure. **Supplementary Figure 22** shows the qualitative course of the pressure references over time for four different procedures and **Supplementary Figure 23** shows the resulting alpha-pressure curves. As can be seen in the latter, each method of calibration results in a different angle-pressure curve. This shows that the relationship between pressure and angle is highly dependent on the load acting on the actuator. Therefore, the calibration procedure must reflect the real load case as best as possible. Consequently, it is necessary to adapt the calibration procedure to the real operating conditions as far as possible. The best walking performance was obtained with calibration procedure 4. However, as can be seen in **Figure 3B**, the subsequent poses in the series are not completely symmetrical, which is an indicator for the potential of improvement in calibration.

Critical to the measuring system is the assembly of the visual markers[15] in section 10. Since they cannot be mounted directly at the desired points (because the tubes interfere), the measured points must be shifted artificially. Special care must be taken when determining this offset vector (in pixel coordinates of the camera); otherwise, the entire measurement will have significant systematic errors. It must also be ensured that the tags do not displace with time. If this happens, for example, due to a downfall of the robot, the corresponding tag must be remounted in the exact same place. In any case, it should be checked regularly whether the measuring system still produces reliable output.

The limiting factor in the experiment was the fixation of the feet. In order to be able to climb even steeper inclinations, the fixation mechanism must, therefore, be reconsidered. Currently, the robot is not able to actively push its feet against the walking plane, and for high inclines, the normal force caused by gravity is too small to bring the suction cups close enough to the walking plane to ensure reliable suction.

The presented manufacturing method can be transferred to any fluidic elastomer actuator and could, therefore, be interesting for future applications. The presented control box enables the control of any pneumatic system consisting of six individual actuators (expandable up to eight), including robotic platforms as they require fast sensory feedback. Therefore, it could be used as a universal platform for testing and control future robots. Finally, the presented calibration method can be transferred in principle to any feed-forward controlled pneumatic system. In summary, all presented methods are universal within the discussed scope.

ACKNOWLEDGMENTS:
The authors like to thank Fynn Knudsen, Aravinda Bhari, and Jacob Muchynski for the helpful discussions and inspiration.

DISCLOSURES:

The authors declare that they have no competing financial interests.

REFERENCES:

1. Majidi, C. Soft robotics: a perspective—current trends and prospects for the future. *Soft Robotics*. **1** (1), 5-11 (2014).
2. Kim, S., Laschi, C., Trimmer, B. Soft robotics: a bioinspired evolution in robotics. *Trends in Biotechnology*. **31** (5), 287–294 (2013).
3. Rus, D., Tolley, M. T. Design, fabrication and control of soft robots. *Nature.* **521** (7553), 467–475 (2015).
4. Calisti, M., Picardi, G., Laschi, C. Fundamentals of soft robot locomotion. *Journal of the Royal Society Interface.* **14** (130), 20170101 (2017).
5. Chu, B., Jung, K., Han, C.-S., Hong, D. A survey of climbing robots: locomotion and adhesion. *International Journal of Precision Engineering and Manufacturing.* **11** (4), 633–647 (2010).
6. Gu, G., Zou, J., Zhao, R., Zhao, X., Zhu, X. Soft wall-climbing robots. *Science Robotics*. **3** (25), eaat2874 (2018).
7. Qin, L. et al. A versatile soft crawling robot with rapid locomotion. *Soft Robotics*. **6**(4) 6, 455–467 (2019).
8. Seibel, A., Schiller, L. Systematic engineering design helps creating new soft machines. *Robotics and Biomimetics.* **5** (1), 5 (2018).
9. Schiller, L., Seibel A., Schlattmann, J. Toward a gecko-inspired, climbing soft robot. *Frontiers in Neurorobotics.* **13** (1), 106 (2019).
10. Mosadegh, B. et al. Pneumatic networks for soft robotics that actuate rapidly. *Advanced Functional Materials*. **24** (15), 2163–2170 (2014).
11. Elango, N., Faudzi, A. A. M. A review article: investigations on soft materials for soft robot manipulations. *The International Journal of Advanced Manufacturing Technology.* **80** (5-8), 1027-1037 (2015).
12. Natarajan, E., Razif, M. R., Faudzi, A., Palanikumar K. Evaluation of a suitable material for soft actuator through experiments and FE simulations. *International Journal of Manufacturing, Materials, and Mechanical Engineering (IJMMME)*. **10** (2), 64-76. (2020)
13. Soft Robotics Toolkit [software]. https://softroboticstoolkit.com (2020).
14. PneumaticBox [software]. https://www.robotics.tu-berlin.de/menue/software_and_tutorials/pneumaticbox/ (2020)
15. Wang, J., Olson, E. Apriltag 2: Efficient and robust fiducial detection. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS):* 4193–4198. Daejeon, South Korea (2016).

Figure 1

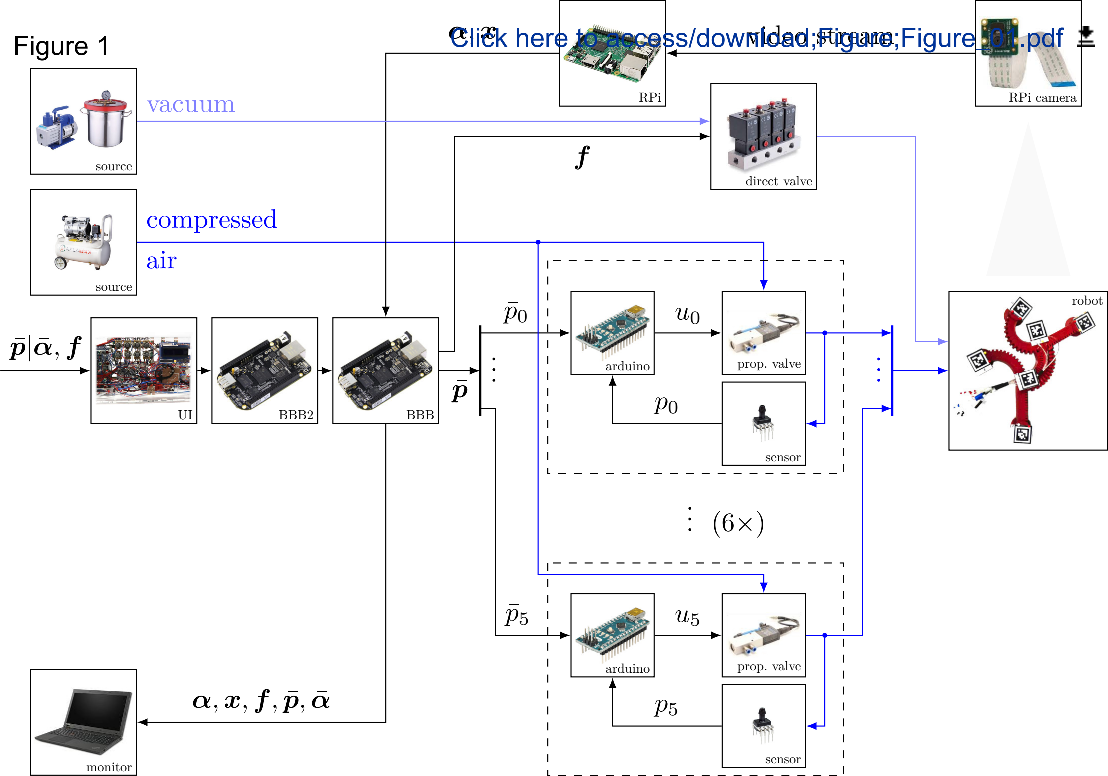

RPi
RPi camera
source
vacuum
direct valve
$\boldsymbol{f}$
source
compressed
air
robot
$\bar{\boldsymbol{p}}|\bar{\boldsymbol{\alpha}}, \boldsymbol{f}$
UI
BBB2
BBB
$\bar{\boldsymbol{p}}$
$\bar{p}_0$
arduino
u_0
prop. valve
p_0
sensor
(6×)
$\bar{p}_5$
arduino
u_5
prop. valve
p_5
sensor
$\boldsymbol{\alpha}, \boldsymbol{x}, \boldsymbol{f}, \bar{\boldsymbol{p}}, \bar{\boldsymbol{\alpha}}$
monitor

Figure 2

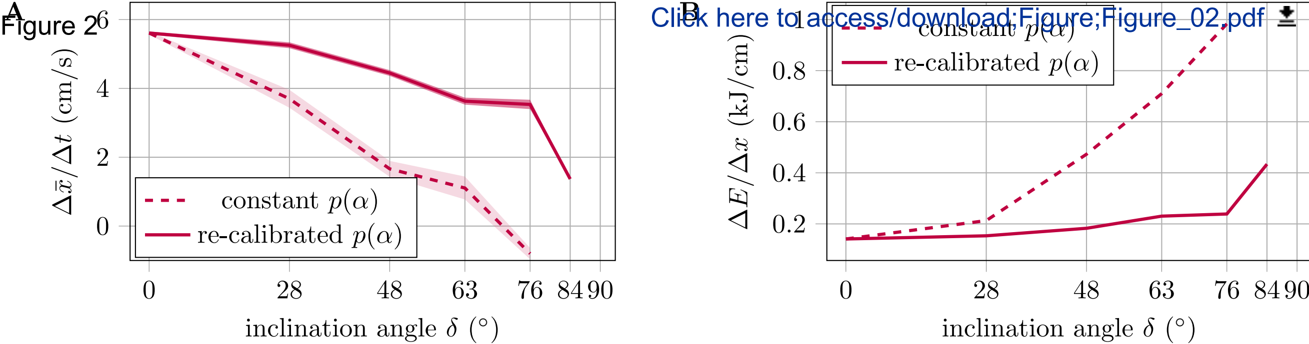
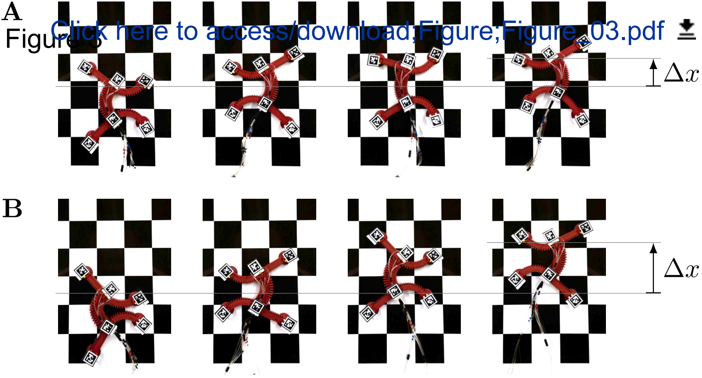

A
6
4
2
0
$\Delta\bar{x}/\Delta t$ (cm/s)
constant $p(\alpha)$
re-calibrated $p(\alpha)$
0
28
48
63
76
84
90
inclination angle δ (°)
B
1
0.8
0.6
0.4
0.2
$\Delta E/\Delta x$ (kJ/cm)
constant $p(\alpha)$
re-calibrated $p(\alpha)$
0
28
48
63
76
84
90
inclination angle δ (°)

Figure 3



straight gait animated

Click here to access/download

Video or Animated Figure

Supplementary_file_01.gif

climbing gait animated

Click here to access/download
Video or Animated Figure
Supplementary_file_02.gif

Table of Materials

Click here to access/download;Table of Materials;v1_JoVE_Materials.xls

Name of Material/ Equipment	Company	Catalog Number
3D Printer	Formlabs	Form 2
acrylic glass plate with two holes	-	
acrylic glass back panel	-	
acrylic glass bottom panel	-	
acrylic glass front panel	-	
acrylic glass side panel	-	
acrylic glass top panel	-	
Arduino Nano	Arduino	A000005
Allan Key 1mm		
BeagleBone Black	beagleboard	BBB01-SC-505
butterfly cannula	B. Braun Melsungen AG	5039573
clamp 1 for measurement system	-	
Clamp 2 for measurement system	-	
cutter knife		
direct acting solenoid valves	Norgren	EXCEL22 DM/49/MDZ83J/T4
elastomer	Wacker Chemie	ELASTOSIL M4601
frame measurement system part 1	-	
frame measurement system part 2	-	
laser cutter	Trotec	SP500
LED	RND COMPONENTS	RND 210-00013
LCD	JOY-IT	SBC-LCD16X2
mould bottom part leg	-	
mould bottom part torso 1	-	
mould bottom part torso 2	-	
mould leg 1	-	
mould leg 2	-	
mould torso 1	-	
mould torso 2	-	
oven	Binder	ED 115
Plastic Cup		
Plastic syringe		

poster panel	Net-xpress.de (distributor)	10620232
Potentiometer	VISHAY	P16NM103MAB15
Power Supply	Pulse Dimension	CPS20.241-C1
pressure sensor	Honeywell	SSCDANN150PG2A5
Pressure Source	EINHELL	4020600
proportional valves	Festo	MPYE-5-1/8-LF-010-B
Raspberry Pi	RASPBERRY PI	RASPBERRY PI 3B+
Raspberry Pi Cam	RASPBERRY PI	RASPBERRY PI CAMERA V2.1
resin	formlabs	grey resin 1l
screw clamps	VELLEMAN	3935-12
silicon tube 2mm	Festo	PUN-H-2X0,4-NT
silicone Tube 2.5mm	Schlauch24	n/a
Switches	MIYAMA	MS 165
ultrasonic bath	RND LAB	605-00034
UV chamber	formlabs	Form Cure
Vacuum chamber + pump	COPALTEC	PURE PERFEKTION
weight scale	KERN-SOHN	PCB 2500-2

Comments/Description

for casting, see Supplementary
see Supplementary
see Supplementary
see Supplementary
see Supplementary
see Supplementary

available in every workshop

see Supplementary
see Supplementary
available in every workshop

see Supplementary
see Supplementary

see Supplementary
see Supplementary
see Supplementary
see Supplementary
see Supplementary
see Supplementary
see Supplementary

available in every supermarket
available in every pharmacy

as walking plane

6x

for connecting robot to control box
for supply tube inlet (https://www.ebay.de/itm/281761715815)

min. resolution 1g

Rebuttal letter - JoVE article v2

Dear editorial staff,

thank you for your patience. The JoVE-specific format is very different to us. We have included your comments. Below you will find our response.

Best regards

the authors

Statement on the editorial comments

- *Please proofread the manuscript well. Please use American English throughout.*

  A: We have proofread the manuscript and have used American English throughout.

--

- Step 1.1: *Please include the details about the supplementary file... supplementary file name or number.*

  A: We have structured the Supplementary as follows:

  1. **Supplementary Animations** containing animations
  2. **Supplementary Data** containing the following folders:
     1. *CAD*: CAD Data for:
        1. ControlBox
        2. Molds
        3. TestBench
     2. *Code*: Code to run on the single board computers and the monitor
     3. *current_exp*: Exemplary measurement files
     4. *Calibration*: Scripts used for calibration procedure
     5. *ExpEvaluation*: Scripts used for performance evaluation
  3. **Supplementary Figures** containing figures
  4. **Supplementary Files** containing an instruction for configuring the BBB and a print template for the visual markers

--

- Step 1.2: *What kind of supporting structures are added? Please include details.*

  A: We have specified the corresponding step.

--

- Step 1.4: *How is this done?*

  A: We have added an explanation.

--

- Step 3.4: *Please explain what is being referenced here.*

  A: We have added an explanation in the paragraph. Additionally, the Supplementary

Figure 2 Legend explains the phenomenon.

--

- Step 3.9: *Like? please include some examples.*

  A: We have added the Supplementary Figure 3 to illustrate good and bad examples of the base part.

--

- Supplementary Figure 6: *Please remove the legend from the figure and move it to the figure legend section of the manuscript.*

  A: We have done so.

--

- Step 8.1: *Please include the name of the file.*

  A: We have done so.

--

- Step 8.2: *All figures need to be referenced in the order. So figure 12 should be referenced after 8,9, 10, 11. Please reorganize and renumber accordingly.*

  A: We have reorganized the figures, such that they appear in the order.

--

- Supplementary File 1: *This is commercial. If using commercial commands to perform this procedure, please remove this and include as per the manufacturer's recommendation (and reference the document either in the table of materials or in the references as citation)*

  A: What do you mean with "This is commercial"? The file describes how to configure the BeagleBone and was created by one of the authors (Lars Schiller). It uses standard programming language.

--

- Step 8.5: *Please name and number this file as it is confusing otherwise.*

  A: We have done so.

--

- Step 9.1: *Please name and number this file as it is confusing otherwise.*

  A: We have done so.

--

- Step 9.2: *Please name and number this file as it is confusing otherwise.*

  A: We have done so.

--

- Step 9.4: *Please expand.*

A: We have done so.

--

- Step 11: *If steps are highlighted, please highlight the subheading as well.*

  A: We have removed all highlights from the headings. Only subheading (or actual process steps) are highlighted now. This is valid for the editor we use (LibreOffice Writer on Debian 10) and might be different on another machine.

--

- Representive Results: *How do you check this looking at the figure.*

  A: We have added a explanation and modified Figure 3 in order to highlight the performance improvement.

--

- Figure and Table Legends: *Please include the legends for all the supplementary figures to be included with the manuscript 1-20. Please include the legend for animated figures and reference them in the manuscript.*

  A: We have included Legends for all Supplementary Figures, Supplementary Animations, Supplementary Files and Supplementary Data.

--

- Figure 2: *Please include a one liner title.*

  A: We have done so.

--

- Discussion: *Please include limitations and future application of the protocol.*

  A: We have added a discussion of limitations of the manufacturing method at the end of the first paragraph. We have added a discussion of limitations of the control box as a second paragraph. We have moved the discussion of limitations of the climbing experiment from the Representative Results section to the Discussion section. Last, we have added a final paragraph discussing the future applications of each presented method.

--

- Discussion/Apriltags: *Where is this done in the protocol.*

  A: We have renamed "apriltags" to visual markers and references to the corresponding section.

--

- Discussion/Calibration: *Please do not make subpoints. Please write in paragraph style.*

  A: We have moved the listing to the Supplementary Figure Legend.

Highlighted Content

Please find attached the file "v2_JoVE_Manuscript_yellow_proof.docx". This document contains only the highlighted content. In the text editor we use (LibreOffice Writer on Debian

10), this is less than 2.75 pages.

Track of changes

Please find attached the file "v2_JoVE_Manuscript_track_of_changes.docx". In this document all changes with respect to the last submitted version are marked blue.

Lars Schiller received the Bachelor's degree in General Engineering Science and Master's degree in Theoretical Mechanical Engineering from Hamburg University of Technology, Hamburg, Germany, in 2014 and 2017, respectively.

He is a Research Assistant with Workgroup on System Technologies and Engineering Design Methodology, Hamburg University of Technology, Hamburg, Germany. His research interests include design and control of soft robots.

Josef Schlattmann received the Diploma and Ph.D. degrees in Theoretical Engineering Science from the University of Paderborn, Germany, in 1979 and 1994 respectively.

Since 2001, he is Professor with Workgroup on System Technologies and Engineering Design Methodology, Hamburg University of Technology, Hamburg, Germany. He holds an honorary doctorate from the University of Tirana, Albania, and an honorary professorship from the University of Peking, China.

Arthur Seibel received the Diploma and Ph.D. degrees (both with distinction) in Mechanical Engineering from Hamburg University of Technology, Hamburg, Germany, in 2011 and 2015, respectively.

He is a Research Associate with Workgroup on System Technologies and Engineering Design Methodology, Hamburg University of Technology, Hamburg, Germany. His research interests include soft robotics, parallel robotics, and design methodology.

Figure 2 is adapted from:

Schiller, L., Seibel A., and Schlattmann, J. Toward a Gecko-Inspired, Climbing Soft Robot. *Frontiers in Neurorobotics* **13**(1), 106 (2019).

This publication is distributed under the CCBY-Licence:

Configuration of BeagleBone Black

Lars Schiller (lars.schiller@tuhh.de)

April 9, 2020

Contents

1 Install Operating System

It is recommended to use one the following images (download: `http://beagleboard.org/latest-images`):

kernel 8.7 `bone-debian-8.7-iot-armhf-2017-03-19-4gb.img`
kernel 9.14 `bone-debian-9.9-iot-armhf-2019-08-03-4gb.img`

Install OS on SD card:
To install it on an `8GB` Micro-SD card, follow the instructions:

- You can use Etcher (`https://etcher.io/`).

OR (on debian):

- Decompress and write on SD card (needs to be `su` and make sure the security locker of SD adapter is in writing mode):

```
$ xz -d bone-debian-**.img.xz
$ dd if=./bone-debian-**.img of=/dev/sdX
```

  Here, `sdX` is the mounted empty SD card. It can be found with multiple use of the command `mount` or `df`.

Then, push the SD card with installed operating system into the BeagleBone Black (BBB).

2 Log into BBB for the first time

Assuming you are called `user` and your PC is called `pc`, your BBB is called `beaglebone` and the default user on BBB is called `debian`, then the following sythax is correct.

- Connect your PC with a MicroUSB cable to the BBB.
- Open a terminal and ssh into BBB as `debian` and then get superuser to configure the board.

```
user@pc:~ ssh debian@192.168.7.2
temppwd
debian@beaglebone:~ su
root
root@beaglebone:~#
```

- Note that the default passwords are:

temppwd	for `debian`
root	for `root`

3 Change static IP of USB port

`https://stackoverflow.com/questions/23805457/changing-the-static-ip-of-beagle-bone-black-usb0`

- To change the static ip of BBB's usb0 interface from default 192.168.7.2 to ...5.2:

```
root@bbb:~# nano /etc/network/interfaces

iface usb0 inet static
        address 192.168.5.2
        netmask 255.255.255.0
        network 192.168.5.0
        gateway 192.169.5.1
```

- (I also edited the file `/opt/scripts/boot/am335_evm.sh`. Maybe it had an effect...)

4 Configure Ethernet

`https://groups.google.com/forum/#!msg/beaglebone/AS2US9rtNd4/8y0mZ3LxAwAJ`

- Assuming you want to configure `eth0` like this:

address	134.28.136.51 (ask administrator for your personal IP)
netmask	255.255.255.0
dns-nameservers	134.28.205.14
gateway	134.28.136.1

- Plug in LAN cable.
- Get the name of the LAN connection:

```
su
root@beaglebone:/etc/network# connmanctl services
*Ac Wired                 ethernet_689e19b50543_cable
```

- Using the appropriate ethernet service, tell `connman` to setup a static IP address for this service.

```
connmanctl config <service> --ipv4 manual <ip_addr> <netmask> <gateway> --nameservers <
    dns_server>
```

In our case:

```
connmanctl config ethernet_689e19b50543_cable --ipv4 manual 134.28.136.51 255.255.255.0
    134.28.136.1 --nameservers 134.28.205.14
```

- Reboot and you are done.
- You can revert back to a DHCP configuration simply as follows:

```
$ connmanctl config ethernet_689e19b50543_cable --ipv4 dhcp
```

5 Configure SSH

https://askubuntu.com/questions/115151/how-to-set-up-passwordless-ssh-access-for-root-user

- If your Board crashed, and you were forced to reinstall the OS, there already exist a ssh-key. This you have to remove first (this is for USB cable):

```
user@pc:~ ssh-keygen -f "/home/user/.ssh/known_hosts" -R 192.168.7.2
```

- Generate a new key:

```
user@pc:~ ssh-keygen -f "/home/user/.ssh/key_user"
```

  When you are prompted for a password, just hit the enter key and you will generate a key with no password.

- Allow to log in as root with a password on the server, in aim to transfer the created key to it:

```
root@beaglebone:# nano /etc/ssh/sshd_config
```

  Make sure you allow root to log in with the following syntax

```
PermitRootLogin yes
PasswordAuthentication yes
```

  Restart the ssh-server:

```
root@beaglebone:# service ssh restart
```

- Now you are able to transfer the key to the server:

```
user@pc:~ ssh-copy-id -i /home/user/.ssh/key_user root@192.168.7.2
```

- Check if its work:

```
user@pc:~ ssh root@192.168.7.2
```

- Now disable root login with password on server (for safety):

```
root@beaglebone:# nano /etc/ssh/sshd_config
```

  And modify the Line:

```
PermitRootLogin without-password
PasswordAuthentication yes
```

  This will allow to login as root with valid key, but not with a password. All other users can further login with a password. Restart the ssh-server and you are done:

```
root@beaglebone:# service ssh restart
```

6 Configure Device Tree (enabling all PWM pins)

In order to enable `P9.28` as pwm pin, you have to load `cape-universala`.

Debian 9 / Kernel v4.14.71-ti-r80:
`https://elinux.org/Beagleboard:BeagleBoneBlack_Debian#U-Boot_Overlays`

- Note: you might need to disable HDMI with `disable_uboot_overlay_video=1` in `/boot/uEnv.txt` if the pins are already in use.
- update bootloader (check version 19-08-07):

```
root@beaglebone:~$ cd /opt/scripts/tools/
root@beaglebone:/opt/scripts/tools$ git pull
root@beaglebone:/opt/scripts/tools$ ./version.sh | grep bootloader
bootloader:[eMMC-(default)]:[/dev/mmcblk1]:[U-Boot 2016.01-00001-g4eb802e]:[location: dd MBR]
```

To upgrade your version of U-Boot:

```
root@beaglebone:~$ cd /opt/scripts/tools/developers/
root@beaglebone:/opt/scripts/tools/developers$ ./update_bootloader.sh
root@beaglebone:/opt/scripts/tools/developers$ reboot

...

root@beaglebone:/opt/scripts/tools$ ./version.sh | grep bootloader
bootloader:[microSD-(push-button)]:[/dev/mmcblk0]:[U-Boot 2019.04-00002-gbb4af0f50f]:[location: dd MBR]
bootloader:[eMMC-(default)]:[/dev/mmcblk1]:[U-Boot 2016.01-00001-g4eb802e]:[location: dd MBR]
```

Delete the old version:

```
root@beaglebone:/opt/scripts/tools$ dd if=/dev/zero of=/dev/mmcblk1 bs=1M count=10
```

Also make sure the bb-cape-overlays package is upto date

```
apt update
apt install --only-upgrade bb-cape-overlays
```

Debian 8 / Kernel version v4.4.54
`https://groups.google.com/forum/#!topic/beagleboard/EYSwmyxYjdM`

- `/boot/uEnv.txt` should be looking something like this:

```
root@beaglebone:# cat /boot/uEnv.txt | grep -v "#"

uname_r=4.4.54-ti-r93
cmdline=coherent_pool=1M quiet cape_universal=enable
```

Edit it with:

```
root@beaglebone:# nano /boot/uEnv.txt
```

Add the following lines, such that `/boot/uEnv.txt` looks like:

```
root@beaglebone:# cat /boot/uEnv.txt | grep -v "#"

uname_r=4.4.54-ti-r93
dtb=am335x-boneblack-overlay.dtb
cmdline=coherent_pool=1M quiet cape_universal=enable
cape_enable=bone_capemgr.enable_partno=cape-universala
```

- Reboot and you should be able to configure with:

```
root@beaglebone:# config-pin P9_28 pwm
```

7 Set I2C Bus to FastMode (400kHz)

Kernel version 4.14.xx:

- Backup the original `.dtb`:

```
root@beaglebone: /boot/dtbs/4.14.71-ti-r80# cp am335x-boneblack.dtb am335x-boneblack.dtb
    .orig
```

- Generate source device tree (`.dts`) from binary block device tree (`.dtb`) with device tree compiler (`dtc`):

```
root@beaglebone: /boot/dtbs/4.14.71-ti-r80# dtc -I dtb -O dts -o am335x-boneblack.dts
    am335x-boneblack.dtb
```

- There are 3 diffrent i2c-buses in the `.dts`:
  - i2c0: 0x44E0B000 (not available as Pins)
  - i2c1: 0x4802A000 (not enabled by default)
  - i2c2: 0x4819C000 (the actual one for configured i2c-1 in Linux-Debian, although the register name/-expansion port is i2c2)

  We want to increase the speed of the `i2c2` bus. Therefore modify the `.dts` with nano:

```
i2c@4819c000 {
compatible = "ti,omap4-i2c";
#address-cells = <0x1>;
#size-cells = <0x0>;
ti,hwmods = "i2c3";
reg = <0x4819c000 0x1000>;
interrupts = <0x1e>;
status = "okay";
pinctrl-names = "default";
pinctrl-0 = <0x35>;

#clock-frequency = <0x186a0>;
clock-frequency = <0x61a80>;

linux,phandle = <0xa1>;
phandle = <0xa1>;
```

  The `clock-frequency = <0x186a0>` is the frequency, `0x186a0 = 100000 = 100kHz` here is the default i2c-1 (Expansion port i2c2) frequency for stock BeagleBone Black image. `0x61a80 = 400000 = 400kHz` is the highest frequency possible for i2c-devices. This we gonna use.

- Generate the `.dtb` from this modified .dts:

```
root@beaglebone: /boot/dtbs/4.14.71-ti-r80# dtc -I dts -O dtb -o am335x-boneblack.dtb
    am335x-boneblack.dts
```

- Reboot and check:

```
root@beaglebone:# dmesg | grep i2c
```

  Something like

```
...
omap/i2c@4819c000 is enabled at 400kHz
...
```

  should be the output.

Kernel version <4.4.xx:

- For kernel version < 4.4.xx replace `am335x-boneblack.dtb` with `am335x-boneblack-overlay.dtb`

8 Disable unused programs

- Webserver:

```
root@beaglebone: systemctl stop apache2.service
root@beaglebone: systemctl disable apache2.service
```

- NodeJS:

```
root@beaglebone: systemctl stop bonescript-autorun.service
root@beaglebone: systemctl disable bonescript-autorun.service
```

9 Install Software

In order to run the software on the BBB, install following packages:

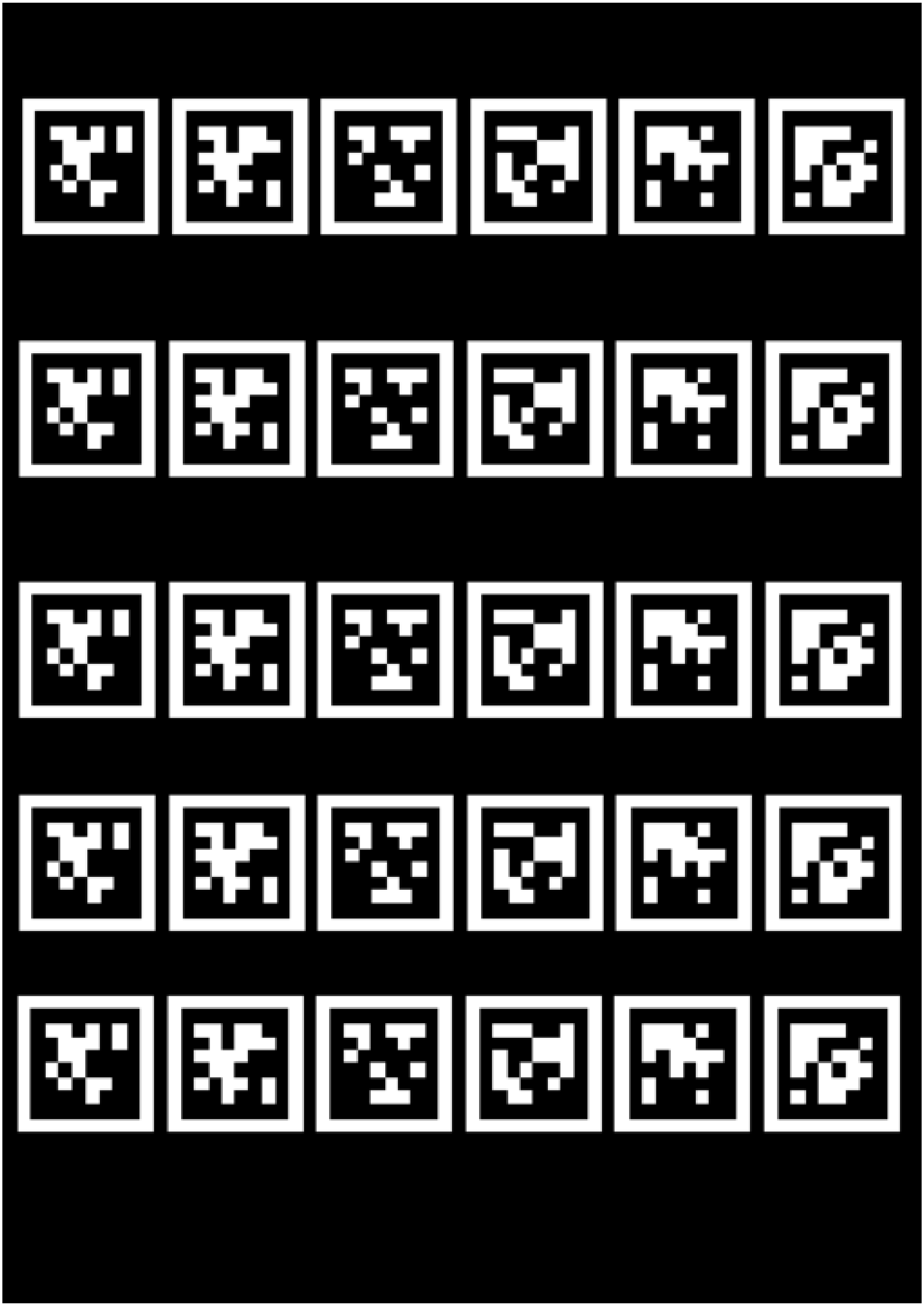
- **python3**: on BBB as `su`

```
root@beaglebone:# apt-get update
root@beaglebone:# apt-get install ntpdate
root@beaglebone:# ntpdate pool.ntp.org
root@beaglebone:# apt-get install build-essential python3-pip python3-scipy python3-numpy -y

root@beaglebone:# pip3.5 install Adafruit_BBIO Adafruit_GPIO board Adafruit-Blinka adafruit-circuitpython-charlcd


root@beaglebone:~# mkdir Git
root@beaglebone:~# cd Git
root@beaglebone:~/Git/# git clone https://github.com/larslevity/GeckoBot.git
```

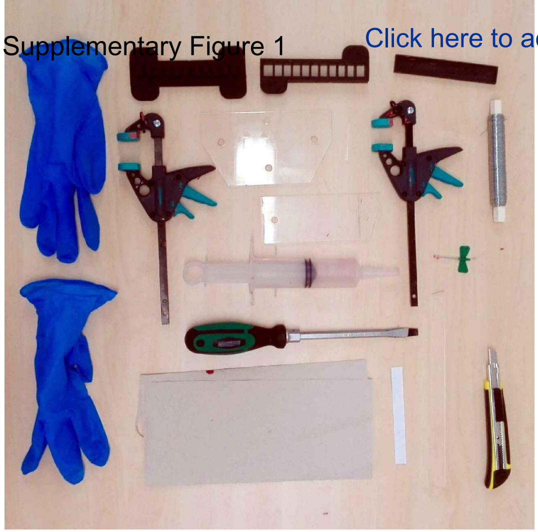
- **python2**: on BBB as `su`

```
root@beaglebone:# apt-get update
root@beaglebone:# apt-get install ntpdate
root@beaglebone:# ntpdate pool.ntp.org
root@beaglebone:# apt-get install build-essential python-dev python-pip -y
root@beaglebone:# pip install --upgrade pip
root@beaglebone:# pip install Adafruit_BBIO
root@beaglebone:# pip install Adafruit_GPIO
root@beaglebone:# pip install termcolor
root@beaglebone:# pip install numpy

root@beaglebone:~# mkdir Git
root@beaglebone:~# cd Git
root@beaglebone:~/Git/# git clone https://github.com/larslevity/GeckoBot.git
```

Supplementary File 2

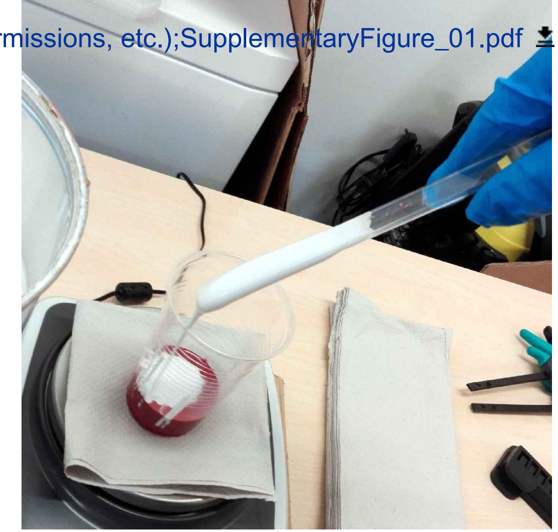


Supplementary Figure 1



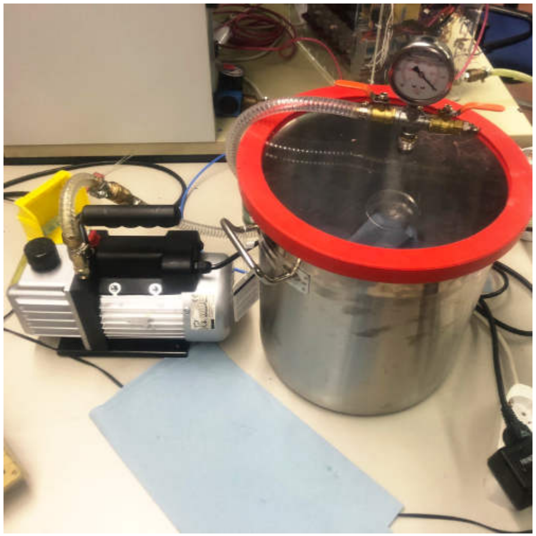
step 2.1

step 2.2

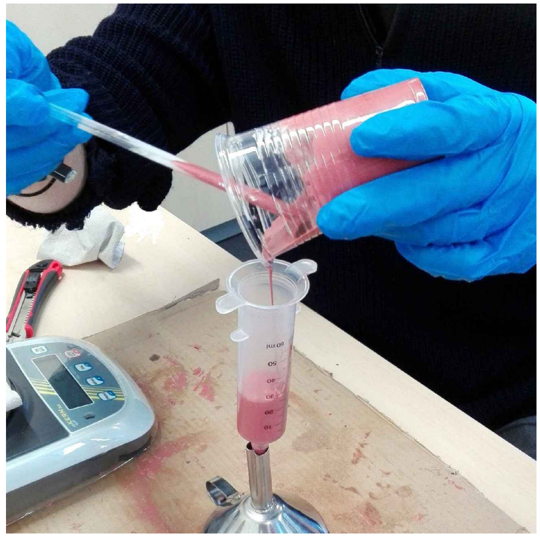


step 2.3

step 2.4



step 2.5

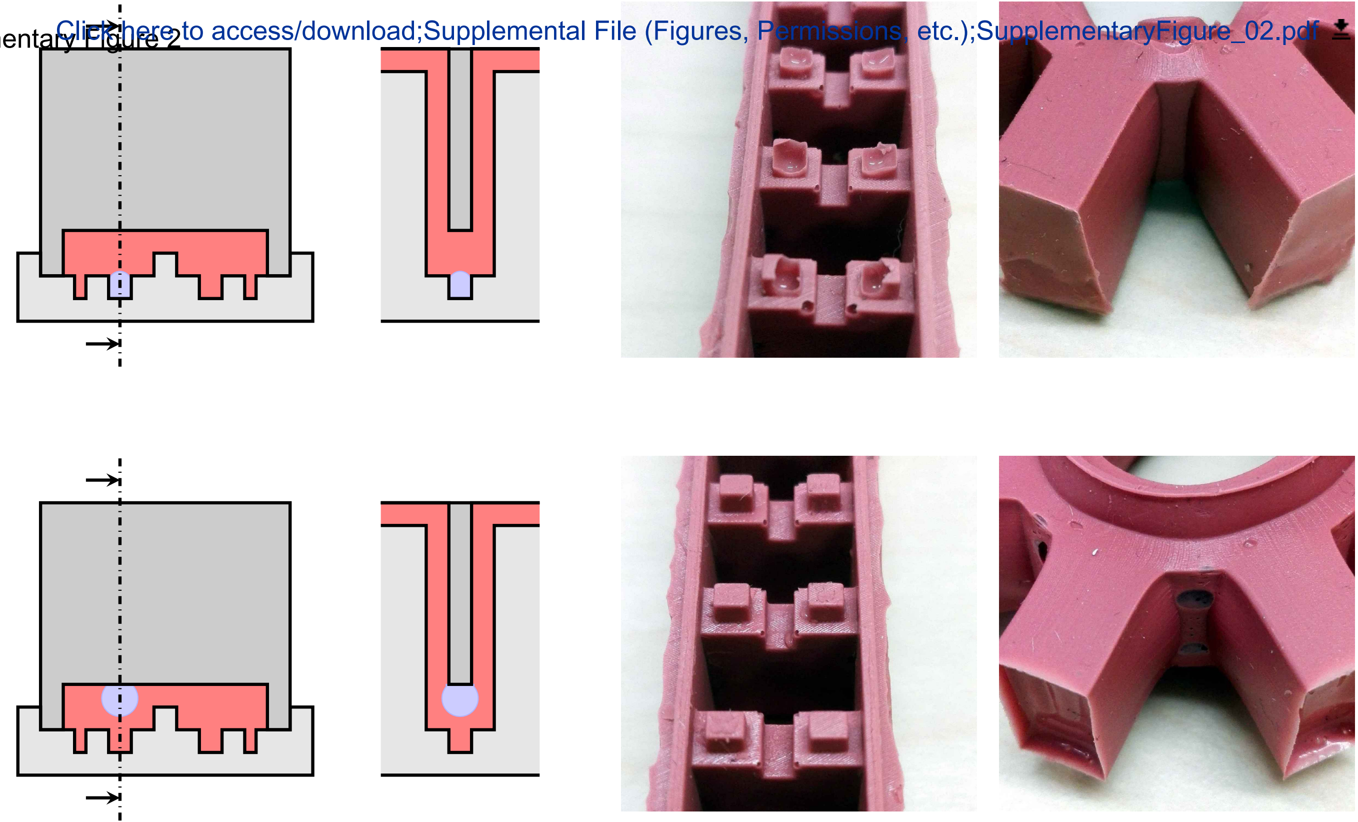


step 2.6

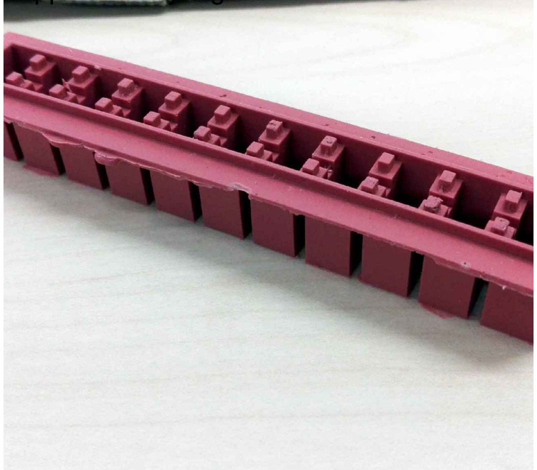
Supplementary Figure 2

A

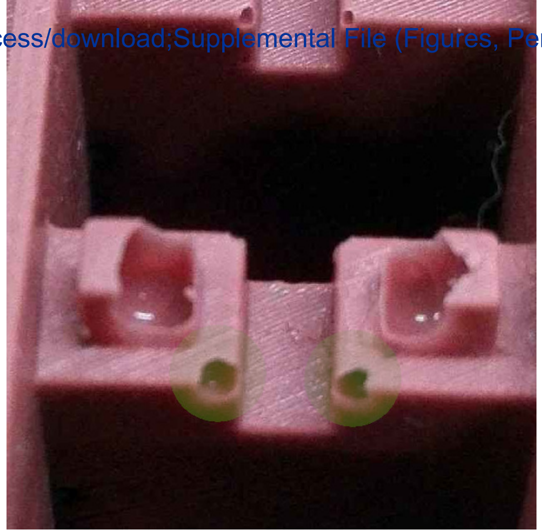
B



Supplementary Figure 3



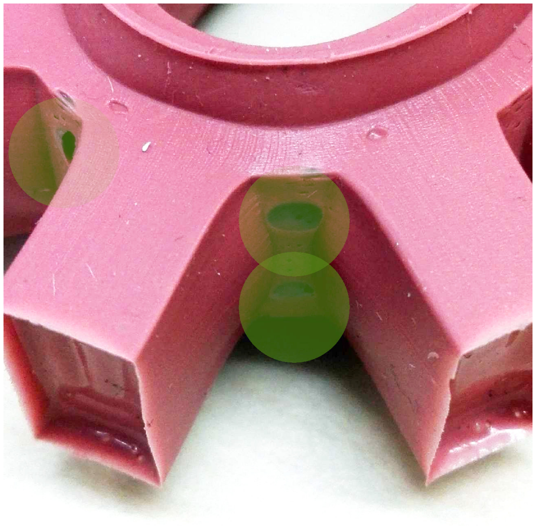
fine



okay

okay

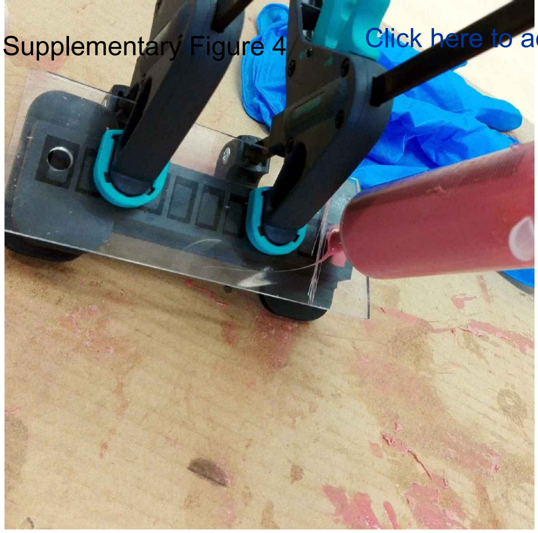
too much air trapped in the elastomer



holes at functionally relevant areas

mold was opened too early

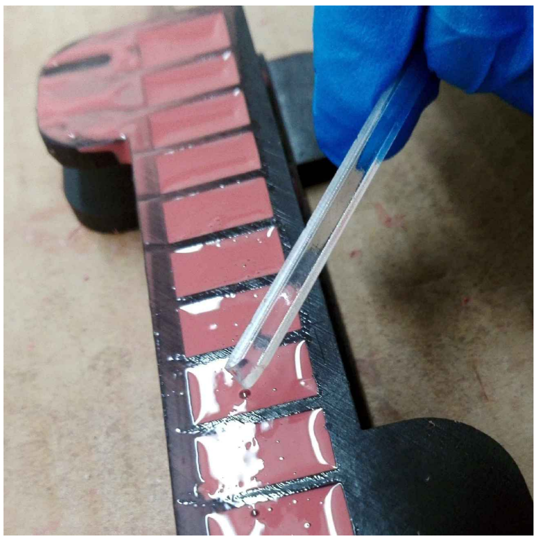
Supplementary Figure 4



step 3.1

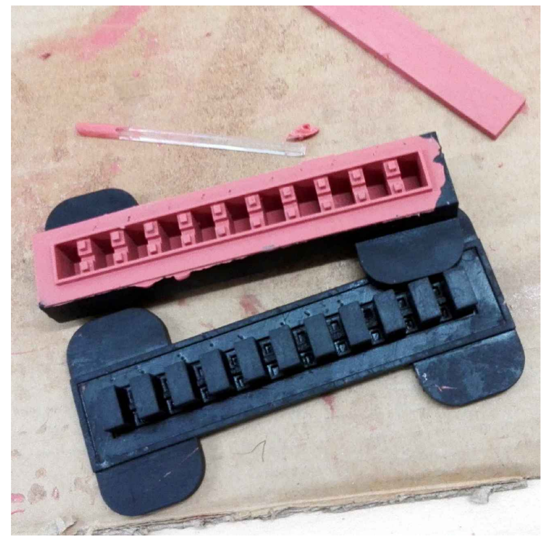
step 3.2

step 3.3

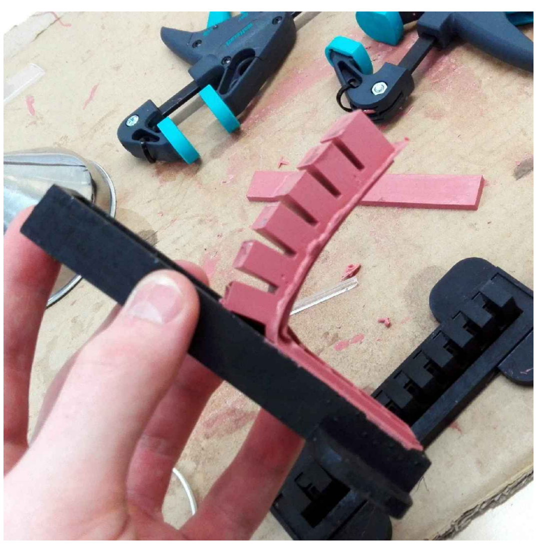


step 3.4

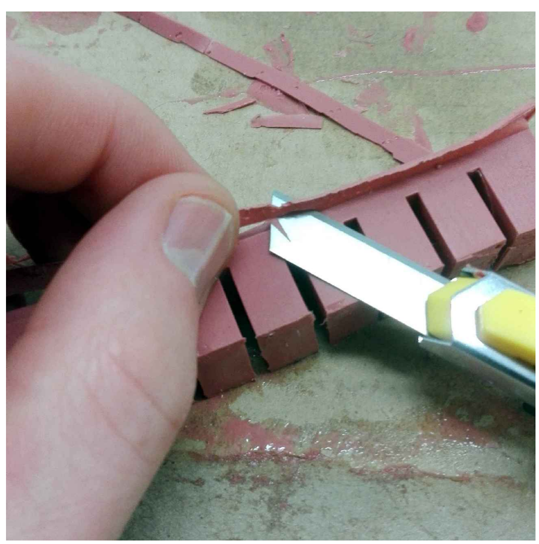
step 3.5



step 3.8

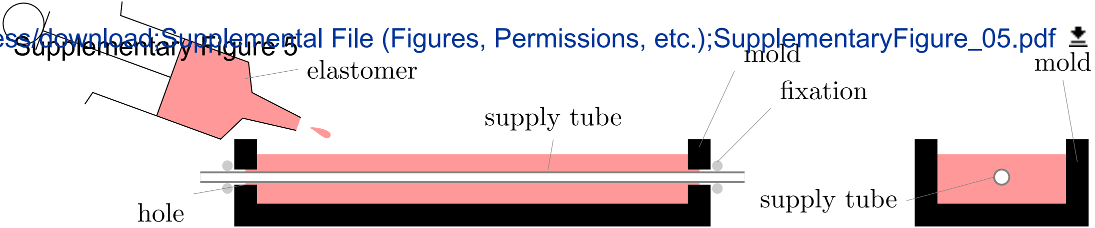


step 3.9



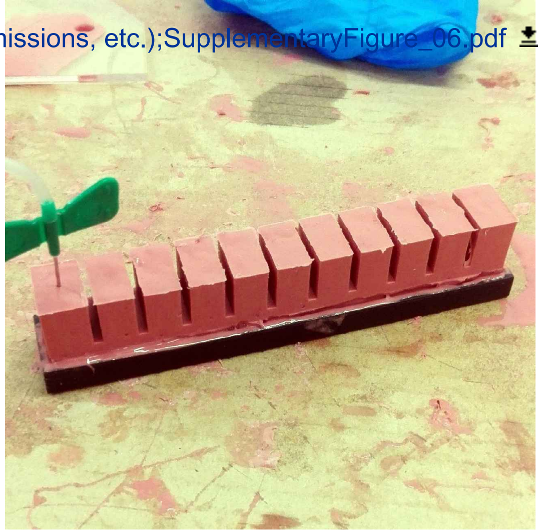
step 3.10

Supplementary Figure 5

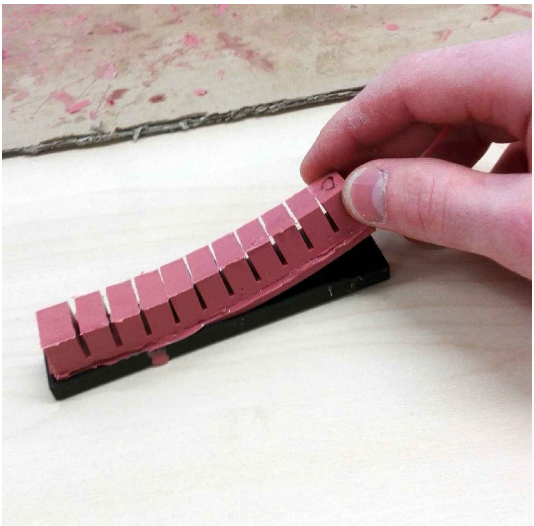


Supplementary Figure 6

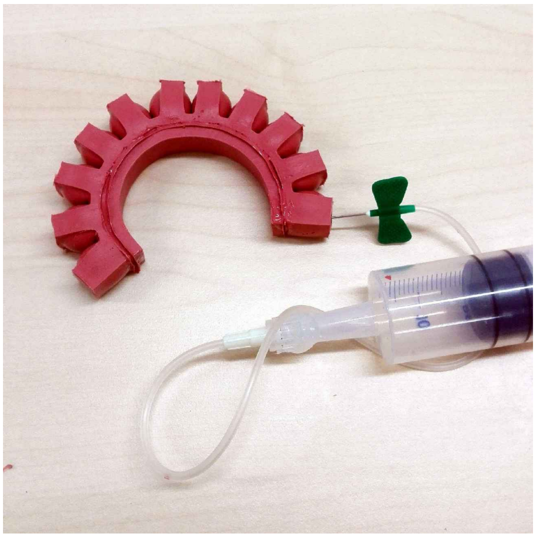
steps 5.1 and 5.2



step 5.3

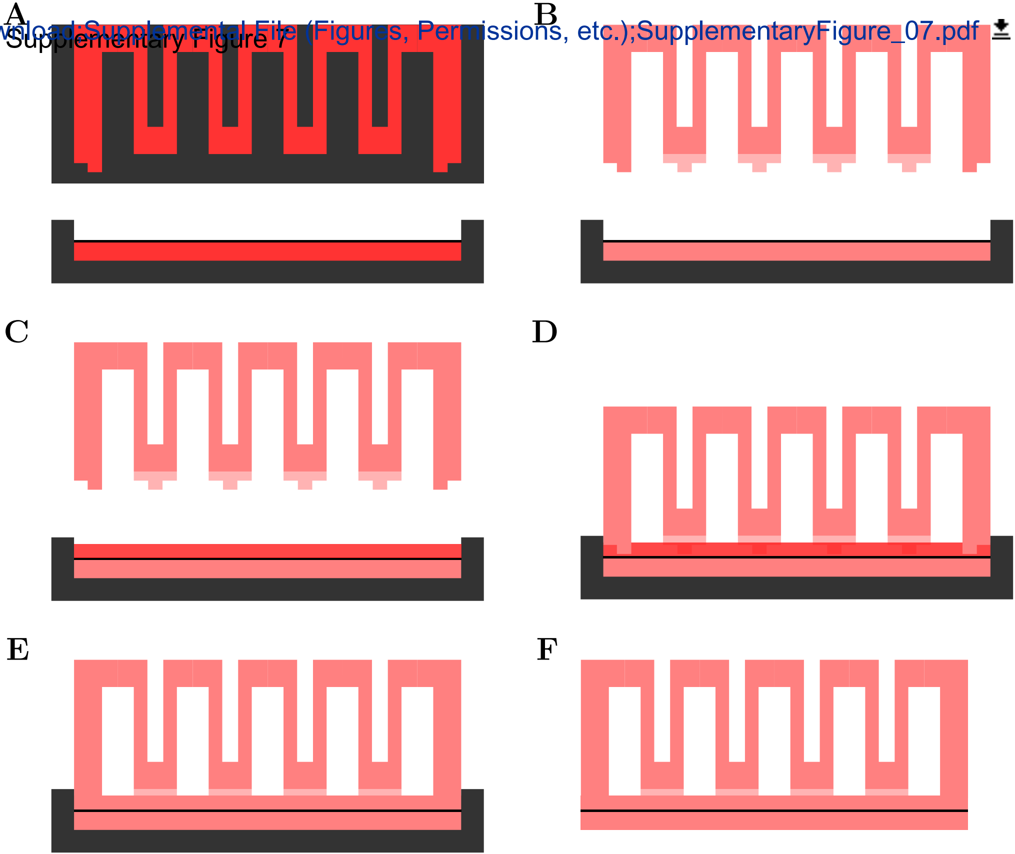


step 5.4

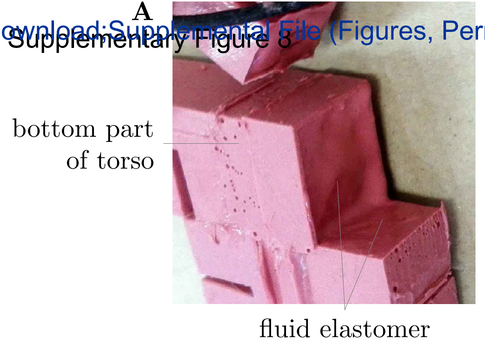


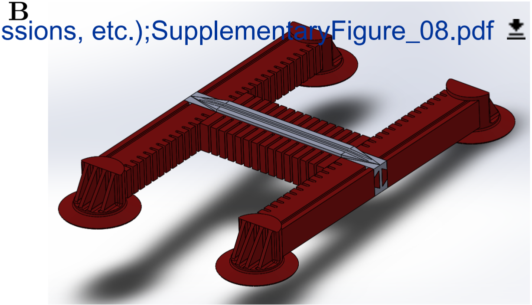
step 5.5

Supplementary Figure 7

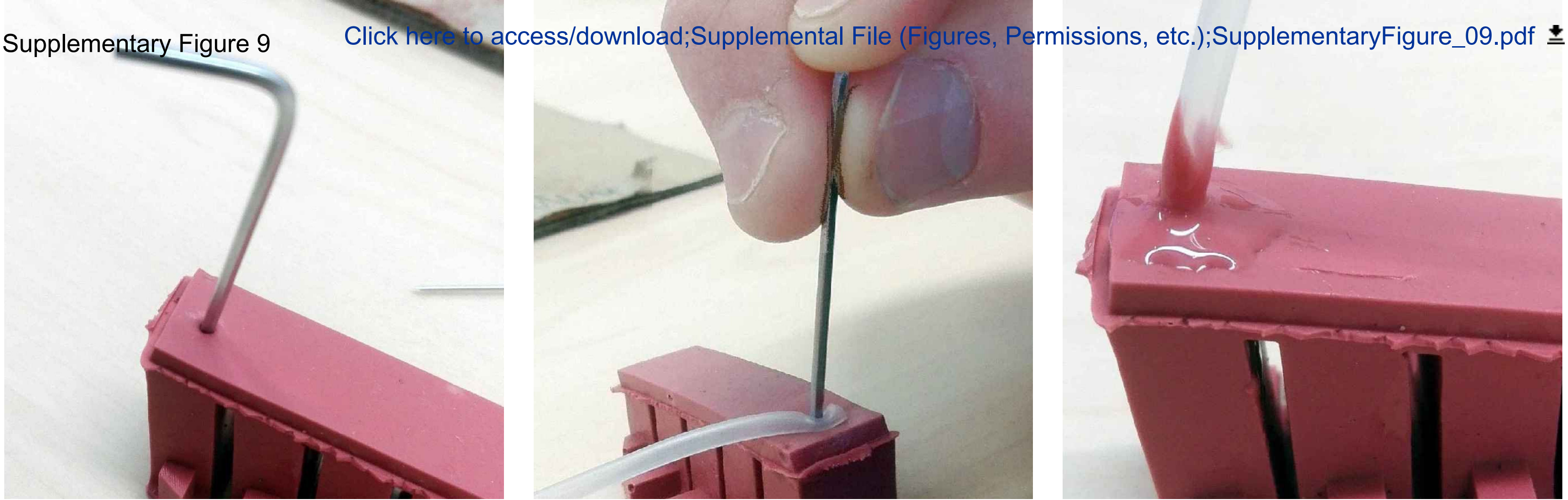


Supplementary Figure 8



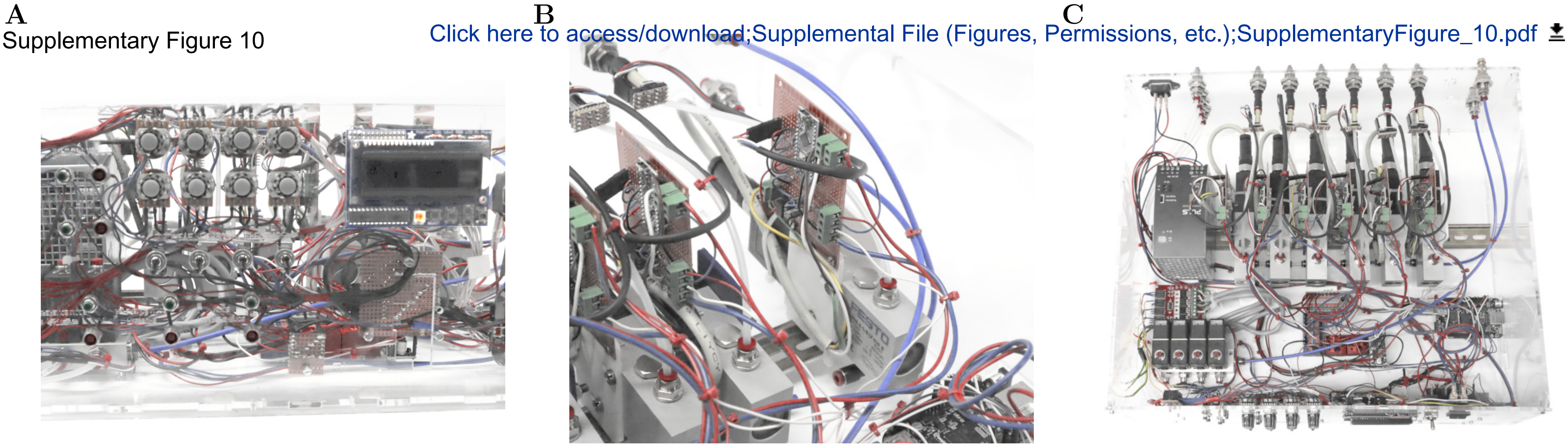


Supplementary Figure 9

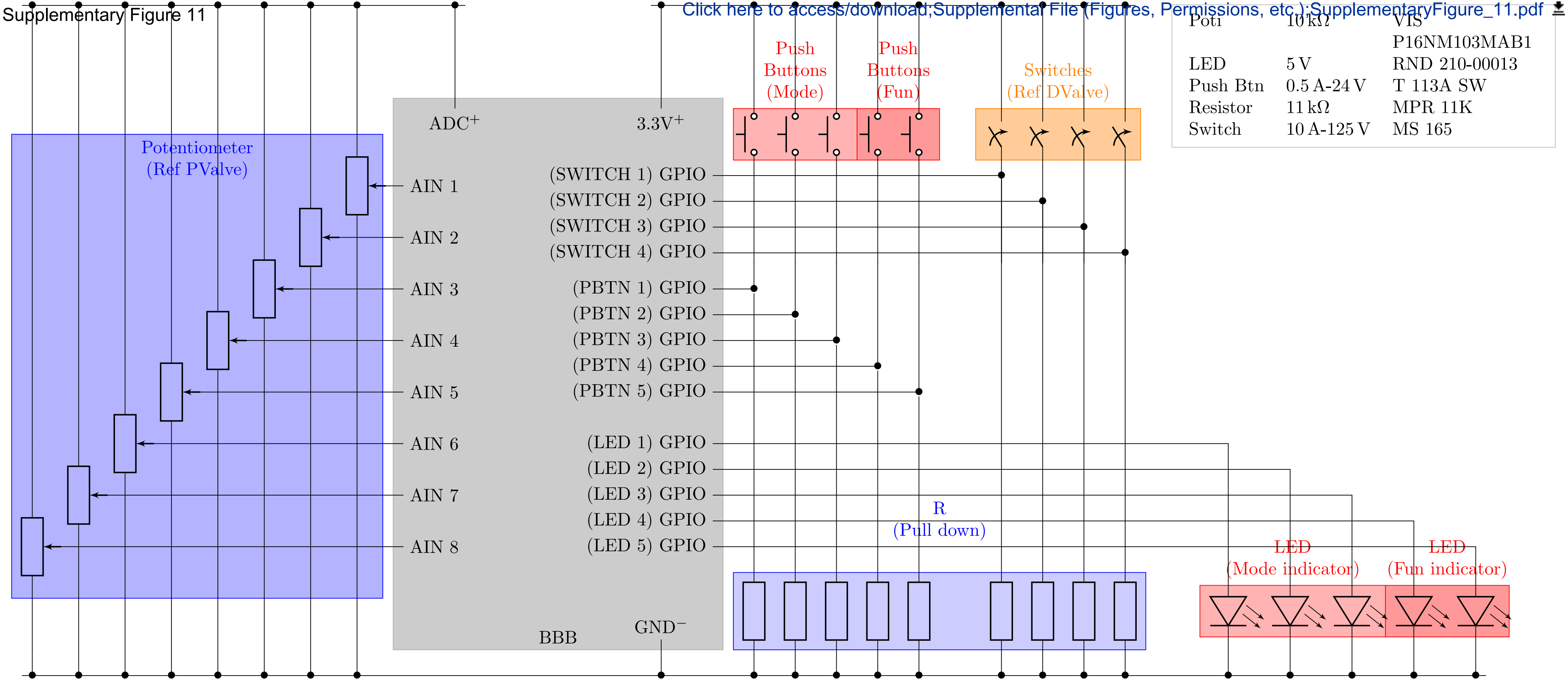


step 7.1 step 7.2 step 7.3

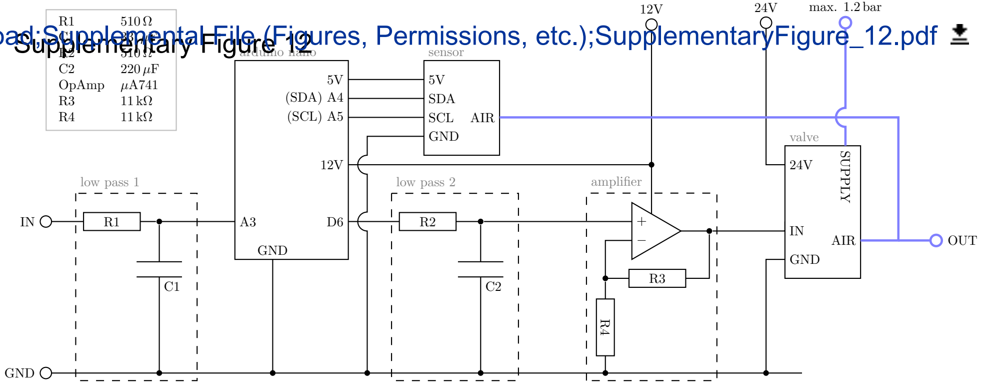
Supplementary Figure 10



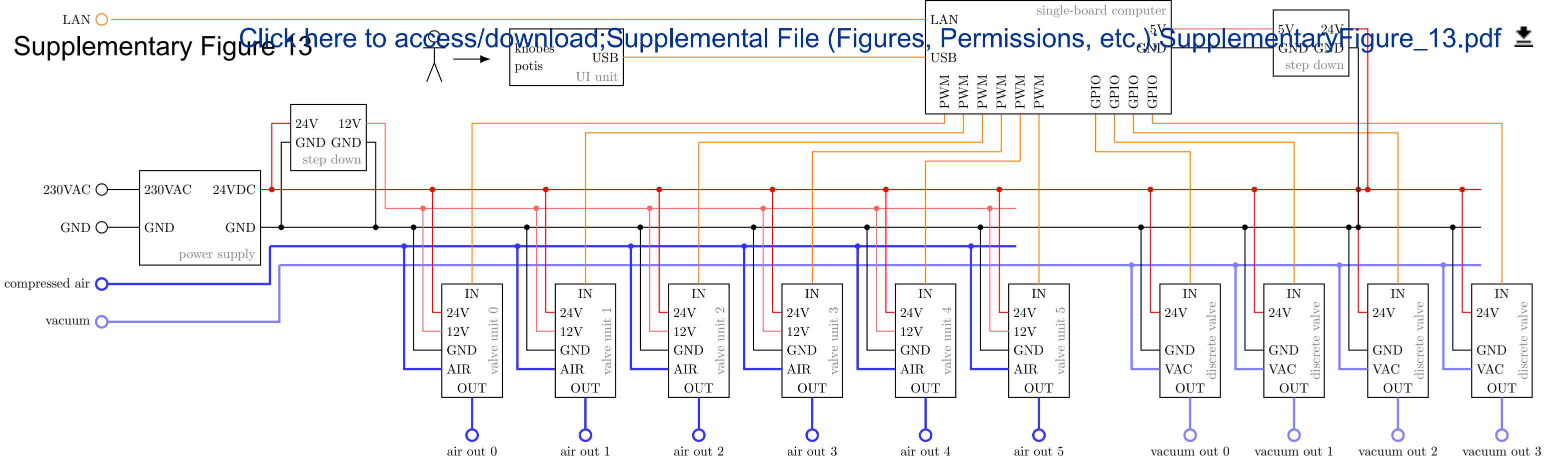
Supplementary Figure 11



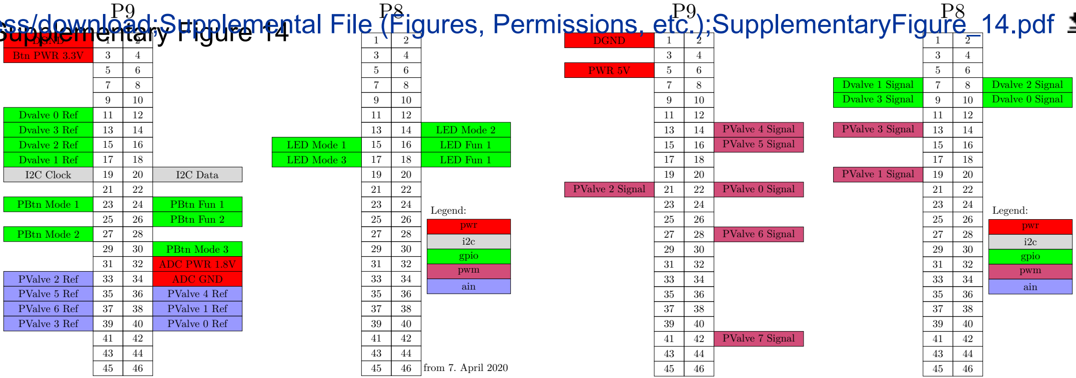
Supplementary Figure 12



Supplementary Figure 13



Supplementary Figure 14

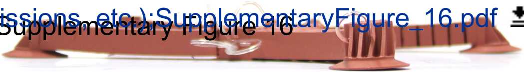


A User Interface BBB

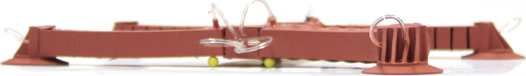
B Main BBB

Supplementary Figure 15

Supplementary Figure 16

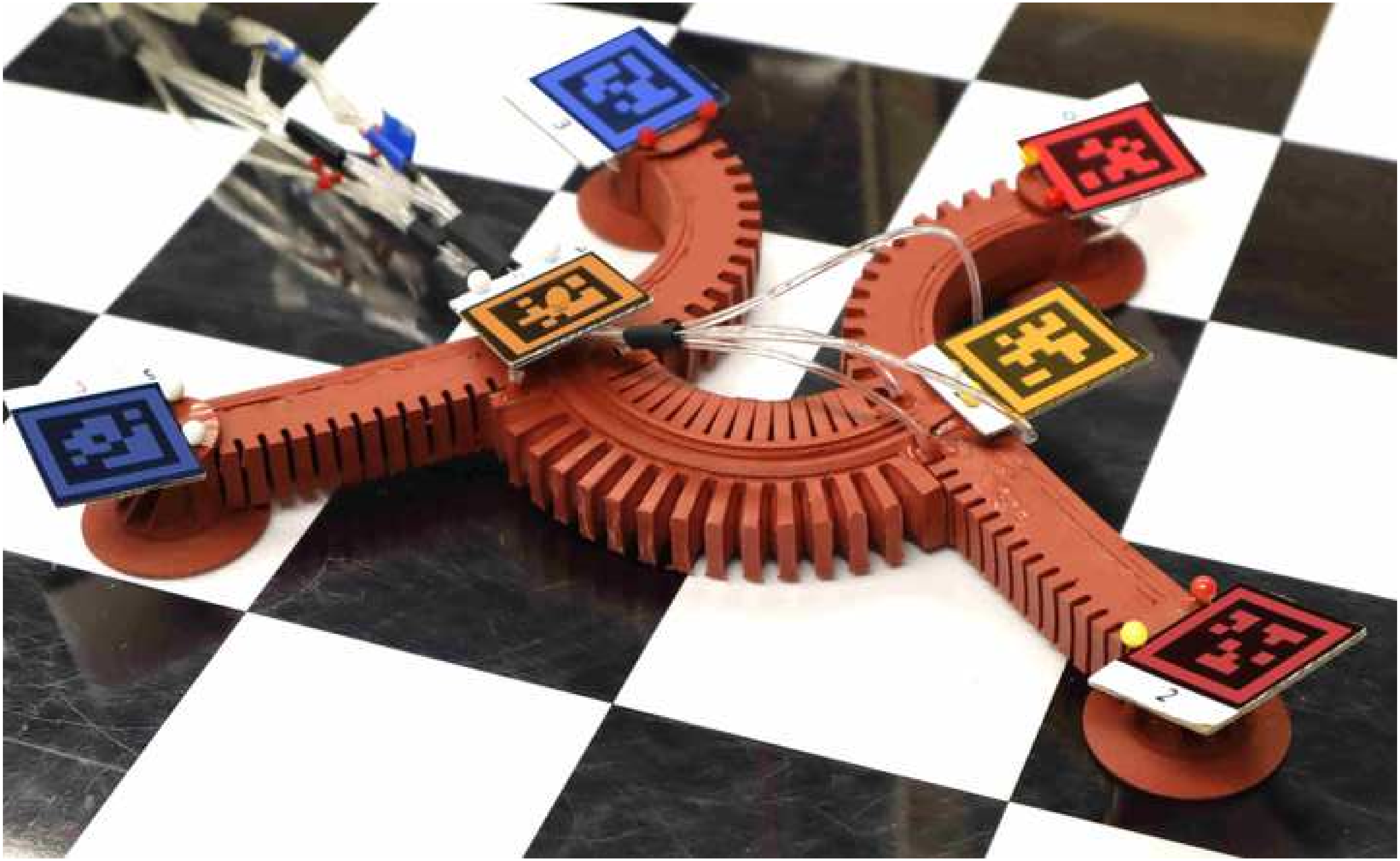


without pins

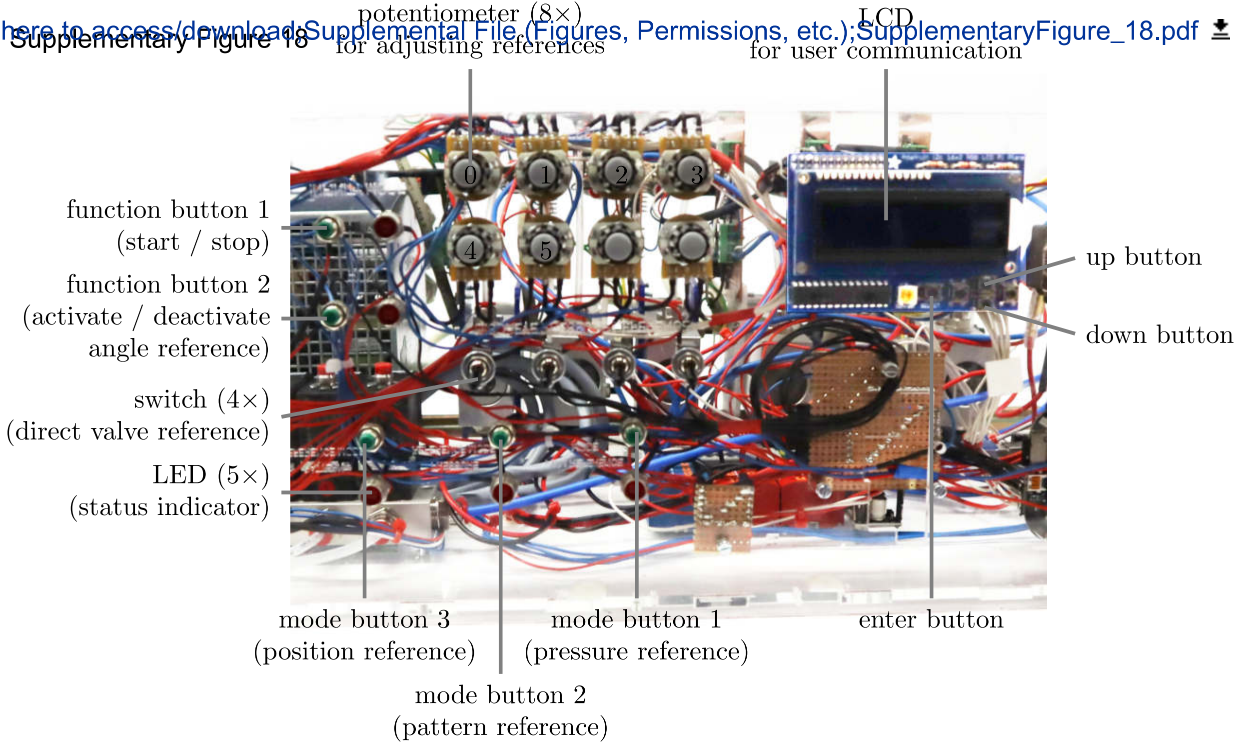


with pins

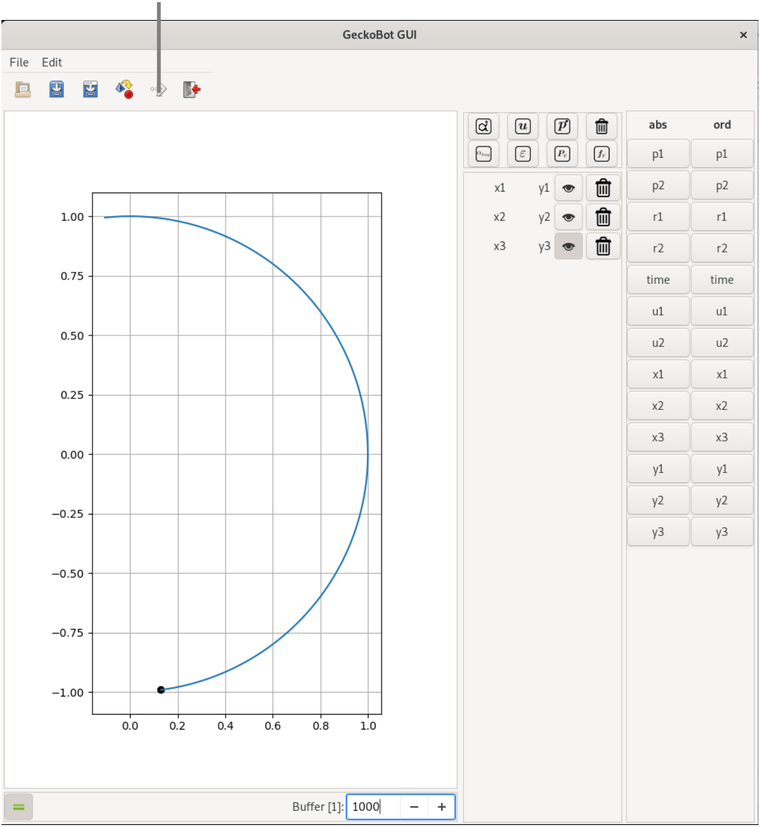
Supplementary Figure 17



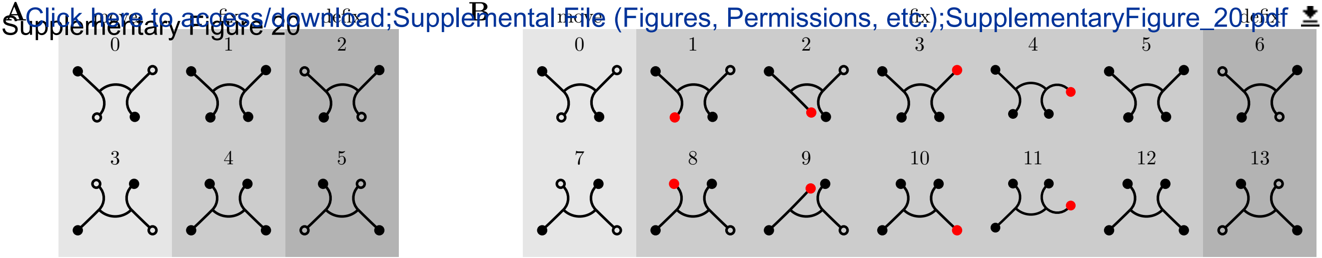
Supplementary Figure 18



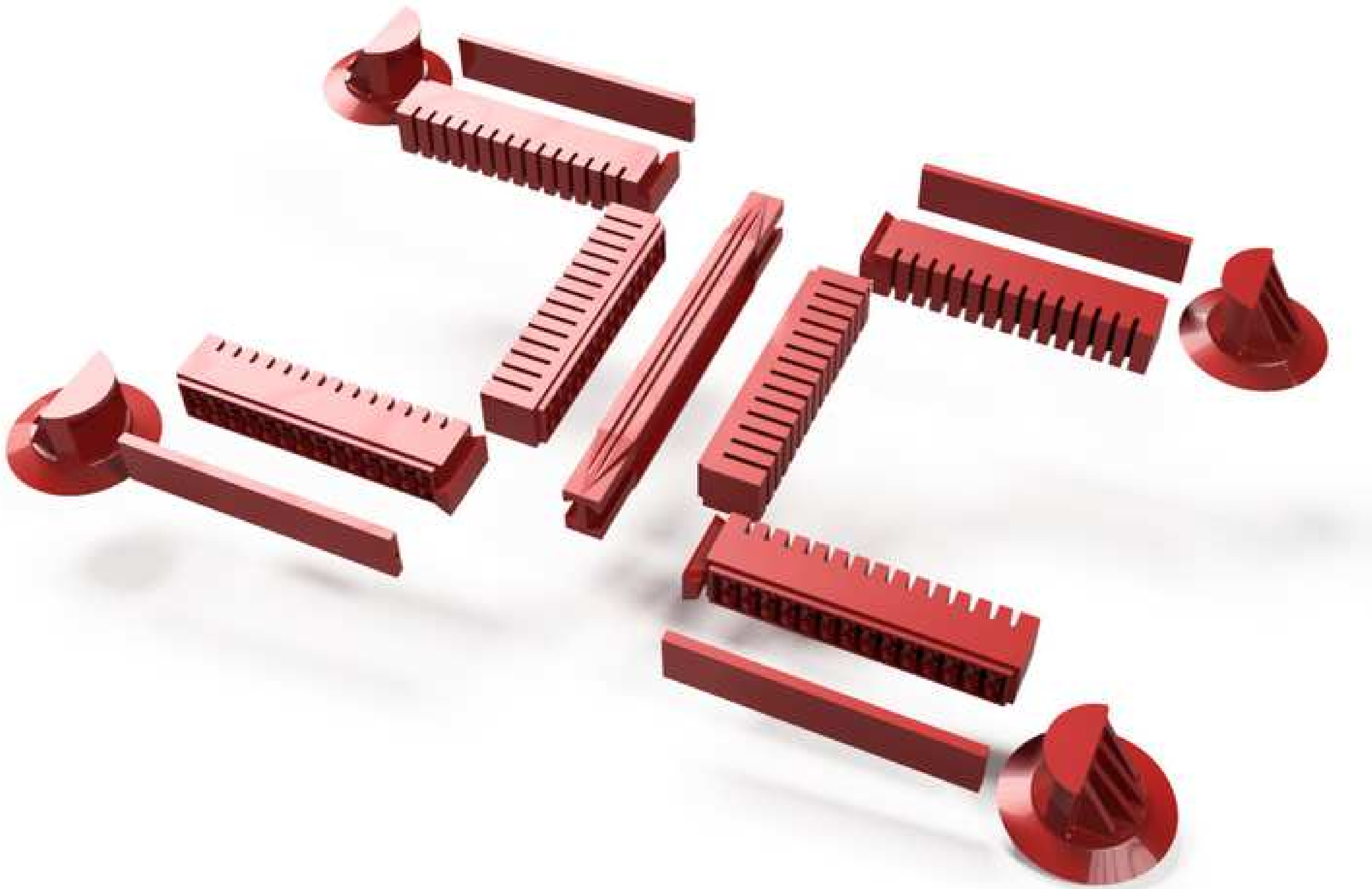
Supplementary Figure 19



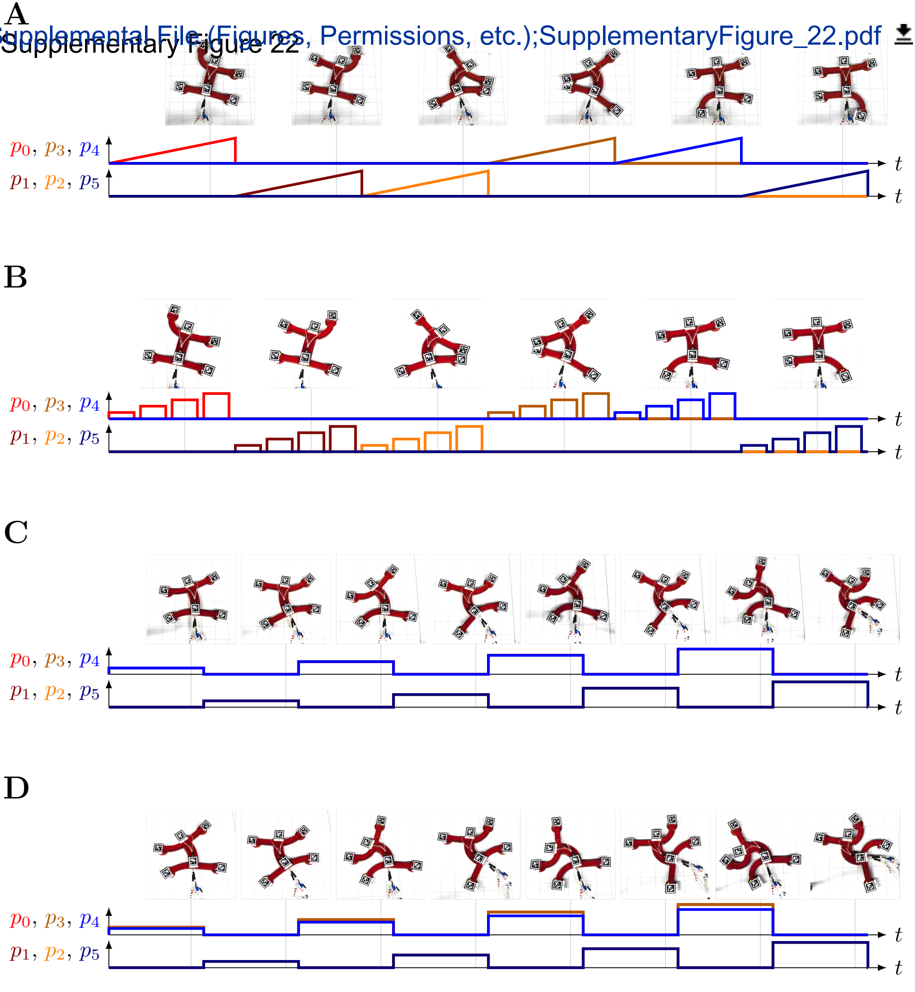
Supplementary Figure 20



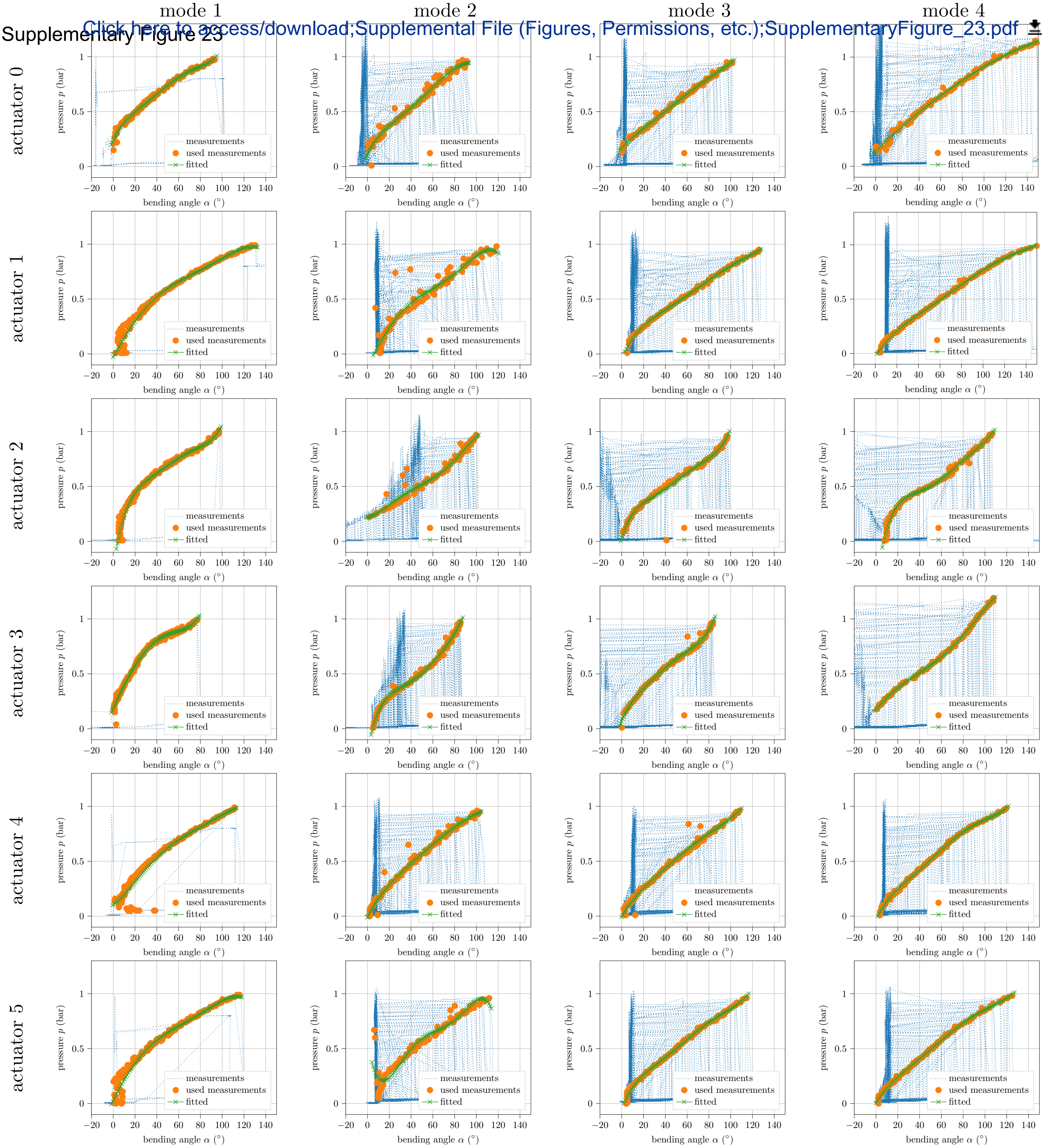
Supplementary Figure 21



Supplementary Figure 22



Supplementary Figure 23



CAD Data

Click here to access/download

Supplemental Coding Files

CAD.zip

Coding File: Code

Click here to access/download
Supplemental Coding Files
Code.zip

Coding File: current_exp

Click here to access/download

Supplemental Coding Files

current_exp.zip

Coding File: Calibration

Click here to access/download
Supplemental Coding Files
Calibration.zip