Supporting Information

for

A Soft Tube-climbing Robot

Mohit S. Verma[1+], Alar Ainla,[1+] Dian Yang[1,2+], Daniel Harburg[1], Zhigang Suo[2,4], and George M. Whitesides[1,3,4]*

[1] Department of Chemistry and Chemical Biology, Harvard University, 12 Oxford Street, Cambridge, MA 02138, USA.

[2] School of Engineering and Applied Sciences, Harvard University, 29 Oxford Street, Cambridge, MA 02138, USA.

[3] Wyss Institute for Biologically Inspired Engineering, Harvard University, 60 Oxford Street, Cambridge, MA 02138, USA.

[4] Kavli Institute for Bionano Science and Technology, Harvard University, 29 Oxford Street, Cambridge, MA 02138, USA.

[+] Authors contributed equally.

(*) Author to whom correspondence should be addressed: gwhitesides@gmwgroup.harvard.edu

Supplementary Figures

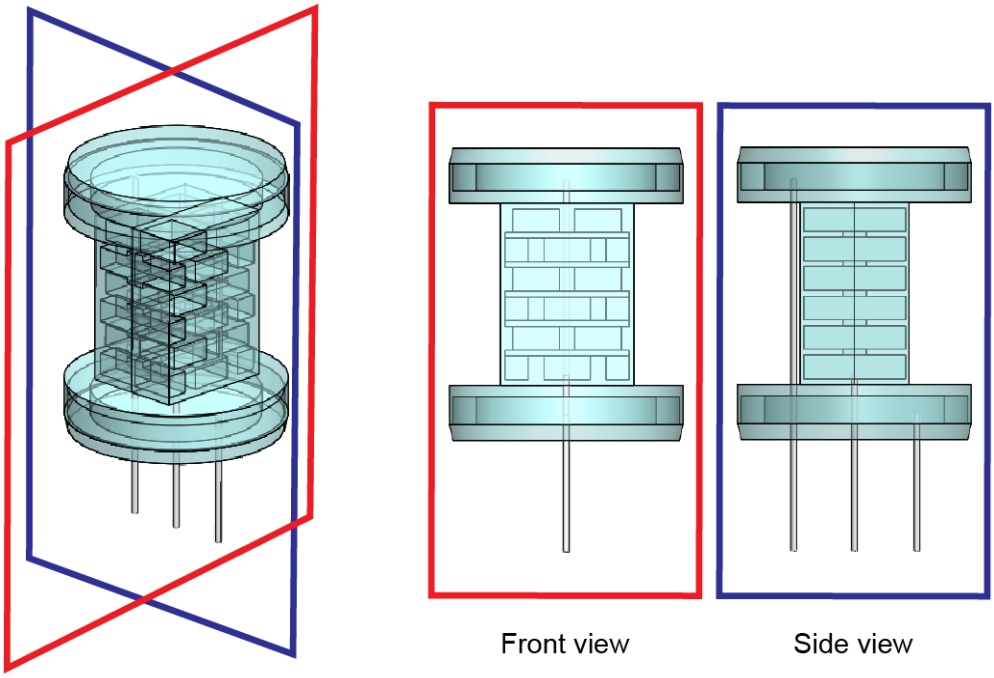

Figure S1. Structure of the tube climber. Computer-aided design (CAD) drawing showing 3D view and two cross-sectional views.

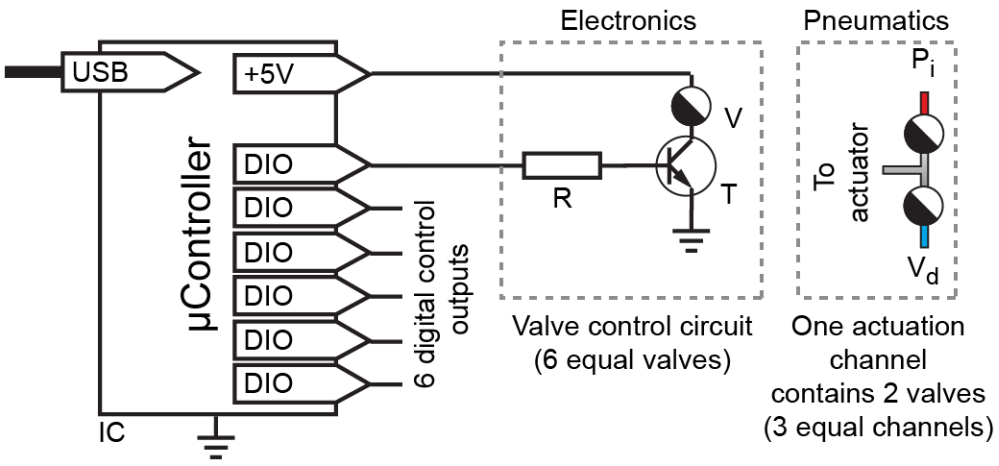

Figure S2. Design of the control electronics. All electronics were controlled and powered from a laptop computer through Arduino microcontroller board. The tube-climbing robot was composed of three pneumatic actuators: two expansion rings in both ends and one linear actuator. All were controlled by three identical pneumatic control channels. Pressure and gas flow in each channel was controlled by six solenoid valves (two per channel). Each valve is controlled through an NPN transistor using microcontroller digital output pins. Power was drawn from the power output pin (+5 V) of the microcontroller board. All components used in the controller are listed in Table S1. Microcontroller was programmed in Arduino software 1.5.6. Firmware is supplied in the supplementary text. Timing characteristics of the sequence were set through USB-virtual serial port interface. Execution of the sequence was timed using timer interruptions with a resolution of 1ms.

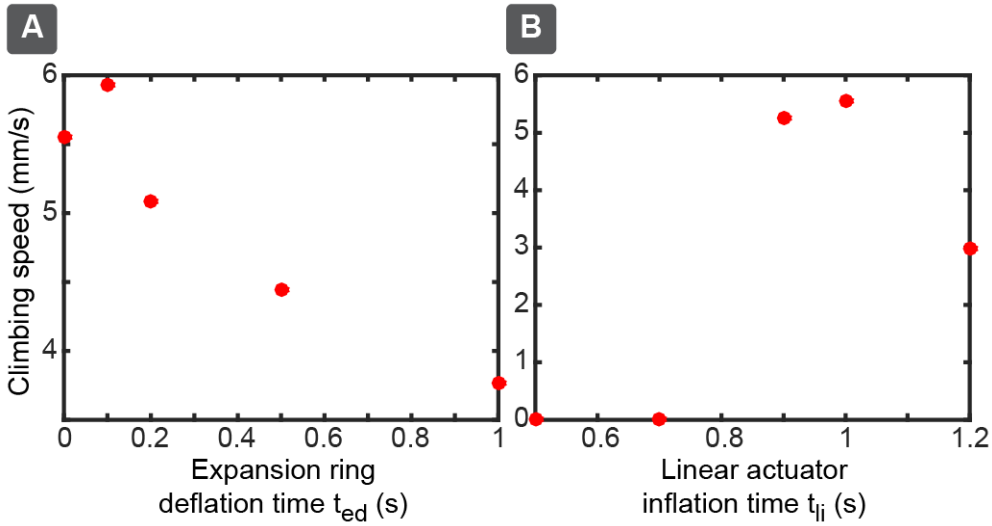

Figure S3. Optimization of the time periods used for operating the tube-climbing robot. Tube climber moves vertically without additional load (mass = 98 g). A) We varied the deflation time of the expansion ring t_{ed}, while t_{ei}=0.4 s, t_{li}=1 s and t_{ld}=2 s. Optimal time for t_{ed} was 100 ms for achieving maximum climbing speed. B) We varied inflation time of linear actuator t_{li}, while t_{ei}=0.4 s, t_{ed}=1 ms and t_{ld}=2 s. Optimal was 1 s. Optimized parameter set t_{ei}=0.4 s, t_{ed}=0.1 s, t_{li}=1 s and t_{ld}=2 s, yielded highest vertical speed of tube climber without any load. (See Figure 2 in the main text for images of the tube climber in operation at maximum speed.)

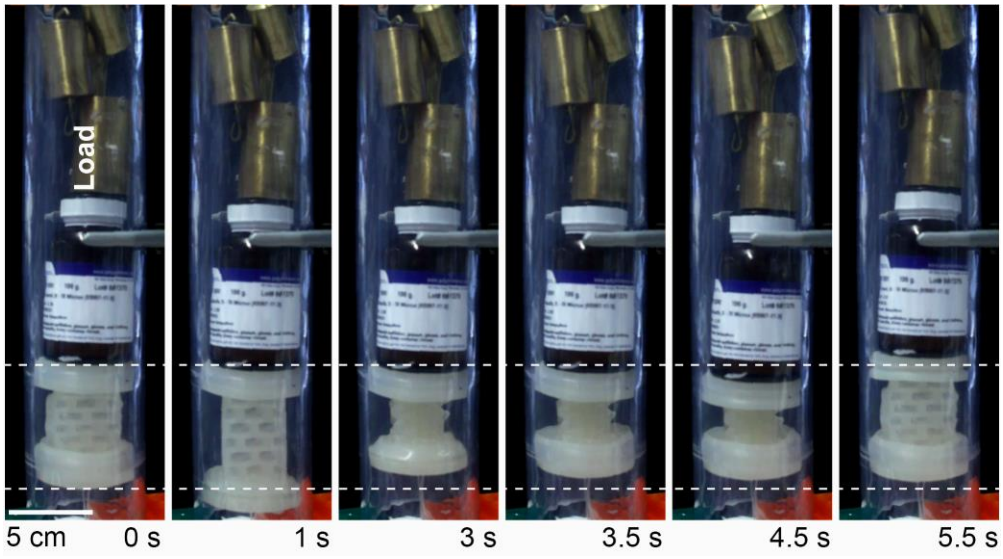

Figure S4. The operation of tube-climbing robot under additional load (m = 1381 g). Sequence parameters are: t_{ei}=0.5 s, t_{ed}=1 s, t_{li}=1 s and t_{ld}=2 s. We can see the sliding of both front and back expansion rings as the climber attempts to move up. We can also observe that the linear actuator does not achieve full actuation due to the heavy load. The large load also causes kick back of the back expansion ring when contracted (see the frame at 1s). The climber still manages to climb up slowly (~ 0.76 mm/s).

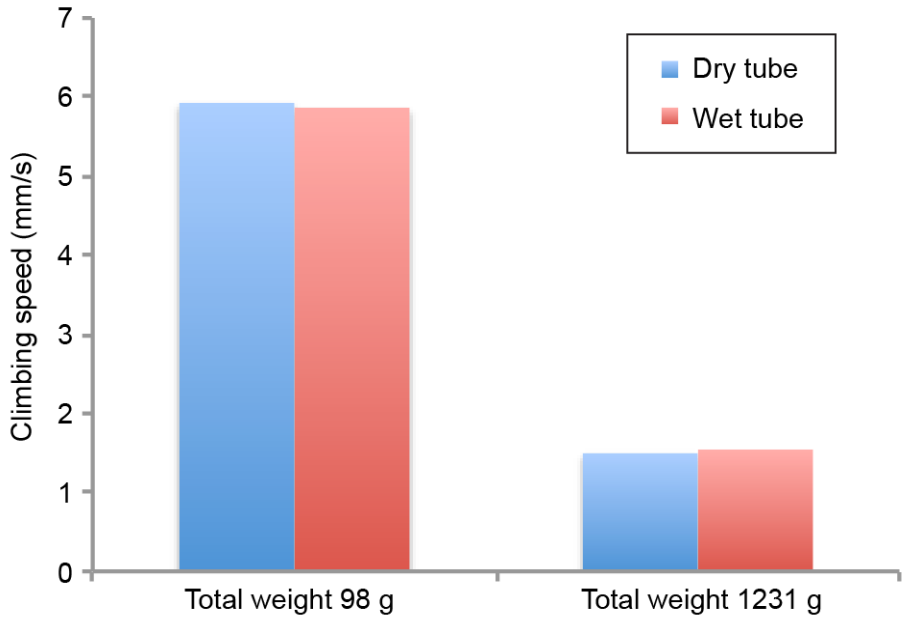

Figure S5. Comparison of climbing in dry and wet conditions (tube and climber were freshly soaked in water). Parameters: $t_{ei} = 0.4$ s, $t_{ed} = 0.1$ s, $t_{li} = 1$ s and $t_{ld} = 2$ s. Water does not have an observable effect on the climbing performance.

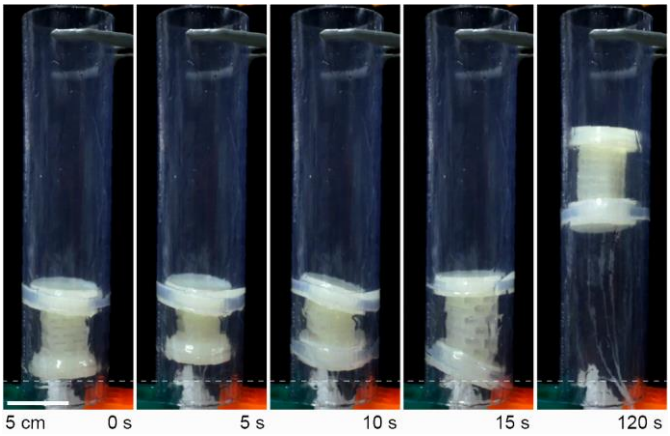

Figure S6. Climbing in oily tube. The tube and climber were covered with vegetable oil. The expansion rings lose grip and tend to slide. In an oily tube, speed of bare tube climber (mass = 98 g) drops 3.3 times (from 6 mm/s to 1.8 mm/s) compared to a dry tube. Also load carrying capability drops substantially and is below <200 g, while dry tube can carry >1200 g at the same settings.

Supplementary Table

Table S1. List of components used in the pneumatic controller

Symbol	Quantity	Name	Description
IC	1	Arduino UNO	Powered, controlled and programmed over USB
V	6	Lee LHDA0533115H	Solenoid microvalve, 5Vdc, -45 … 30 psid.
T	6	2N4124	NPN transistor, 30V, 200mA
R	6	1 kΩ	Resistor

Firmware for the controller

```
// ----------------------------------------------------------------------
// Tube-climber controller
// Alar Ainla, 2017 GMW Group Harvard
// ----------------------------------------------------------------------

//Timer configuration
long microseconds=1000;     //Timer period in microseconds. We use 1ms resolution
#define RESOLUTION 65536    // Timer1 is 16 bit

//IO pins definition
//If pressure valve is in state HIGH actuator is connected to pressure source, if LOW is isolated
//If vacuum valve is in state LOW actuator is connected to the vacuum source, if HIGH is isolated
#define FrontP 4
#define FrontV 3
#define MiddleP 6
#define MiddleV 5
#define BackP 2
#define BackV 7

//Timing the robot
int counter=0;  //Counts time steps in ms
int tA=1000;    //Inflation time of the expansion ring (both ends) in ms
int tB=500;     //Deflation time of the expansion ring (both ends) in ms
int tC=1000;    //Linear actuator (middle part) inflation time in ms
int tD=2000;    //Linear actuator deflation time in ms
int state=0;    //Actuation step: 0-all off (not running).
                 //1 -inflate front, 2-deflate back, 3-deflate-middle,
                //4-inflate back, 5-deflate front, 6-inflate middle.

//Set output step based on the state of the system
void setState()
{
    //First switch all off (disconnected state)
    digitalWrite(BackV, HIGH);
    digitalWrite(BackP, LOW);
    digitalWrite(FrontV, HIGH);
    digitalWrite(FrontP, LOW);
    digitalWrite(MiddleV, HIGH);
    digitalWrite(MiddleP, LOW);
    if(state==0) //All under vacuum
    {
      digitalWrite(BackV, LOW); digitalWrite(FrontV, LOW); digitalWrite(MiddleV, LOW);
    }else
    if(state==1) //Inflate front
    {
      digitalWrite(FrontP, HIGH);
    }else
    if(state==2) //deflate back
    {
      digitalWrite(BackV, LOW);
    }else
    if(state==3) //deflate middle and back
    {
      digitalWrite(MiddleV, LOW);
      digitalWrite(BackV, LOW);
    }else
    if(state==4) //inflate back
    {
      digitalWrite(BackP, HIGH);
    }else
    if(state==5) //deflate front
    {
      digitalWrite(FrontV, LOW);
    }else
    if(state==6) //inflate middle and deflate front
    {
      digitalWrite(MiddleP, HIGH);
      digitalWrite(FrontV, LOW);
    }else;
}

//Timer interupt service - configured for 1ms
ISR(TIMER1_OVF_vect)
{
  if((state==1)&&(counter>tA)){ state=2; counter=0; setState(); }else
  if((state==2)&&(counter>tB)){ state=3; counter=0; setState(); }else
```

```
  if((state==3)&&(counter>tD)){ state=4; counter=0; setState(); }else
  if((state==4)&&(counter>tA)){ state=5; counter=0; setState(); }else
  if((state==5)&&(counter>tB)){ state=6; counter=0; setState(); }else
  if((state==6)&&(counter>tC)){ state=1; counter=0; setState(); }else;
  counter++;
}

//Initilize the timer interup service based on Timer1
void initTimer()
{
  long cycles = (F_CPU / 2000000) * microseconds; // the counter runs backwards after TOP,
                                                  //interrupt is at BOTTOM so divide microseconds by 2
  char oldSREG;
  unsigned char clockSelectBits;
  TCCR1A = 0;                 // clear control register A
  TCCR1B = _BV(WGM13);        // set mode 8: phase and frequency correct pwm, stop the timer
  if(cycles < RESOLUTION) clockSelectBits = _BV(CS10);       // no prescale, full xtal
  else if((cycles >>= 3) < RESOLUTION) clockSelectBits = _BV(CS11);              // prescale by /8
  else if((cycles >>= 3) < RESOLUTION) clockSelectBits = _BV(CS11) | _BV(CS10);  // prescale by /64
  else if((cycles >>= 2) < RESOLUTION) clockSelectBits = _BV(CS12);              // prescale by /256
  else if((cycles >>= 2) < RESOLUTION) clockSelectBits = _BV(CS12) | _BV(CS10);  // prescale by /1024
  else cycles = RESOLUTION - 1, clockSelectBits = _BV(CS12) | _BV(CS10);
        // request was out of bounds, set as maximum
  oldSREG = SREG;
  cli();         // Disable interrupts for 16 bit register access
  ICR1 = cycles; //pwmPeriod = cycles;       // ICR1 is TOP in p & f correct pwm mode
  SREG = oldSREG;
  TCCR1B &= ~(_BV(CS10) | _BV(CS11) | _BV(CS12));
  TCCR1B |= clockSelectBits;                     // reset clock select register, and starts the clock
  TIMSK1 = _BV(TOIE1);
}

//Setup the micontroller
void setup()
{
  initTimer(); //Configure timer interupt for 1ms resolution
  //Configure all IO pins to OUTPUT and low
  for(int i=0; i<10; i++)
  {
    pinMode(i, OUTPUT);
    digitalWrite(i, LOW);
  }
  //Configure serial port
  Serial.begin(115200); //Init comport for data communication
  delay(10);
  //Ready for operation send instruction information
  Serial.println("Tube climber controller, FW: 1.0, Alar Ainla, GMW Group");
  Serial.println("Commands (all time parameters are in milliseconds):");
  Serial.println("R - run");
  Serial.println("X - halt");
  Serial.println("A= - set end inflation (default: 1000)");
  Serial.println("B= - set end deflation (default: 500)");
  Serial.println("C= - set middle part inflation (default: 1000)");
  Serial.println("D= - set middle part deflation (default: 2000)");
  Serial.println("-------------------------------------------------------------");
}

//This is main loop to service

String datain;   //Incoming data
char command;    //1 char long string command
int datain_param; //Numeric parameter of data in

void loop()
{
  if(Serial.available()>0)
  {
      datain=Serial.readString();
      if(datain.length()>0) //If length is larger than zero
      {
          command=datain.charAt(0); //Get command
          if(datain.length()>2) //Get numeric parameter
          { datain_param=datain.substring(2).toInt(); }
          //Process
          if(command=='A')
          {
              tA=datain_param;
              Serial.print("A=");
              Serial.println(tA);
          }else
          if(command=='B')
```

```
          {
              tB=datain_param;
              Serial.print("B=");
              Serial.println(tB);
          }else
          if(command=='C')
          {
              tC=datain_param;
              Serial.print("C=");
              Serial.println(tC);
          }else
          if(command=='D')
          {
              tD=datain_param;
              Serial.print("D=");
              Serial.println(tD);
          }else
          if(command=='X')
          {
              state=0;
              setState();
              Serial.println("HALT");
          }else
          if(command=='R')
          {
              state=1;
              counter=0;
              setState();
              Serial.println("RUN");
          }
      }
   }
}
// -- END --
```

Captions for supplementary videos

Video S1. Demonstration of the operation of tube-climbing robot in a vertical tube, in real time.

Video S2. Demonstration of the operation of tube-climbing robot in a tube at different angles, video is sped up 4x.

Video S3. The tube-climbing robot clears a piece of tissue paper from the tube, video is sped up 4x.

Video S4. The tube-climbing robot turns around a corner in a joint connecting two tubes, video is sped up 4x.

Video S5. The operation of the tube-climbing robot with weights, in real time.

Video S6. The vertical operation of tube-climbing robot through a water column, in real time.

Video S7. The operation of the tube-climbing robot in a horizontal tube submerged under water in real time.

Video S8. The operation of the tube-climbing robot in a vertical tube when the surface is covered in vegetable oil, in real time.