Supplementary Information

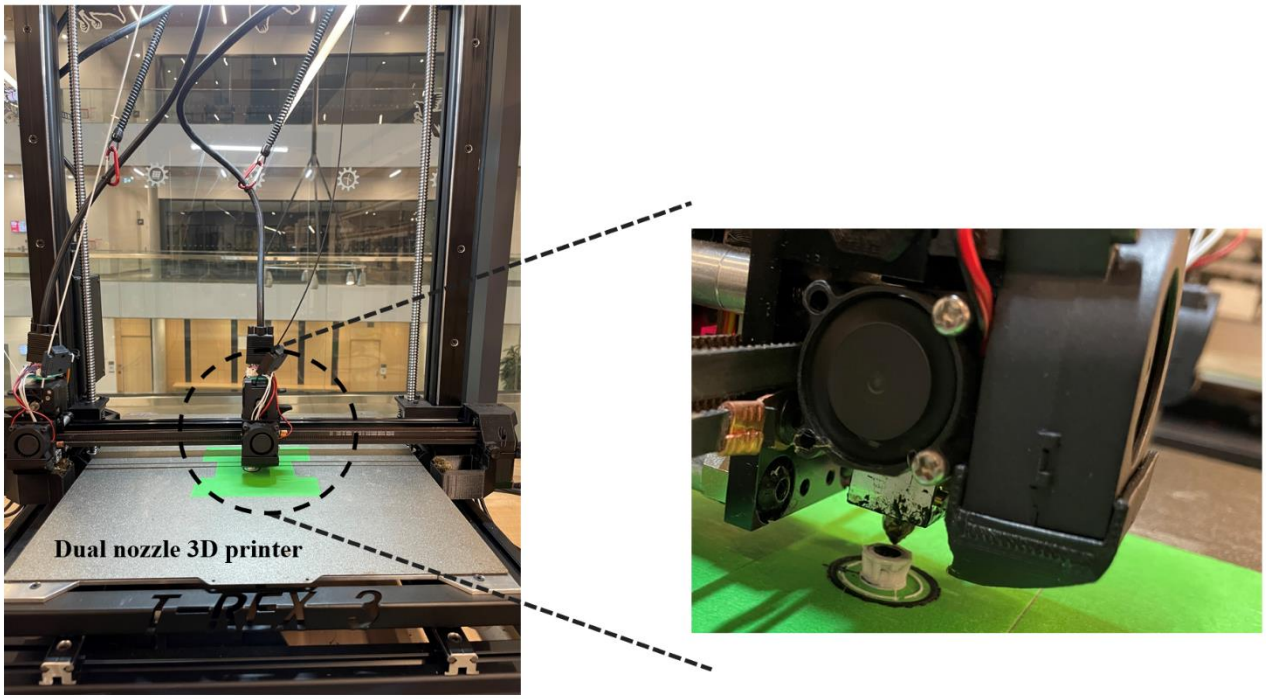
Wireless Pressure Monitoring System Utilizing a 3D-Printed Origami Pressure Sensor Array

Hadi Moeinnia[a], Danielle Jaye Agron[a], Carl Ganzert[b], Loren Schubert[b], Woo Soo Kim[a, *]

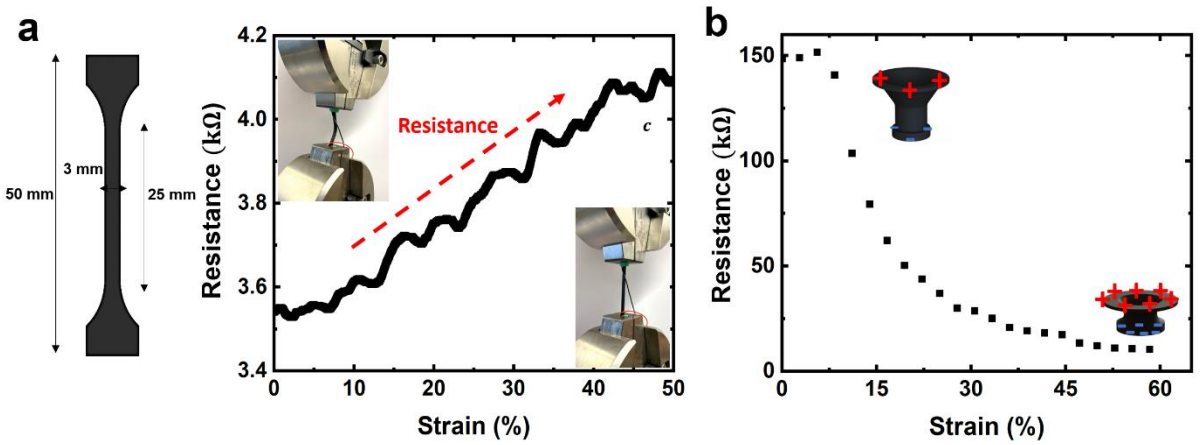
[a] School of Mechatronic Systems Engineering, Simon Fraser University, Surrey, V3T 0A3, Canada

[b] Hodgson Orthopedic Group, Coquitlam, BC V3K 6V7, Canada

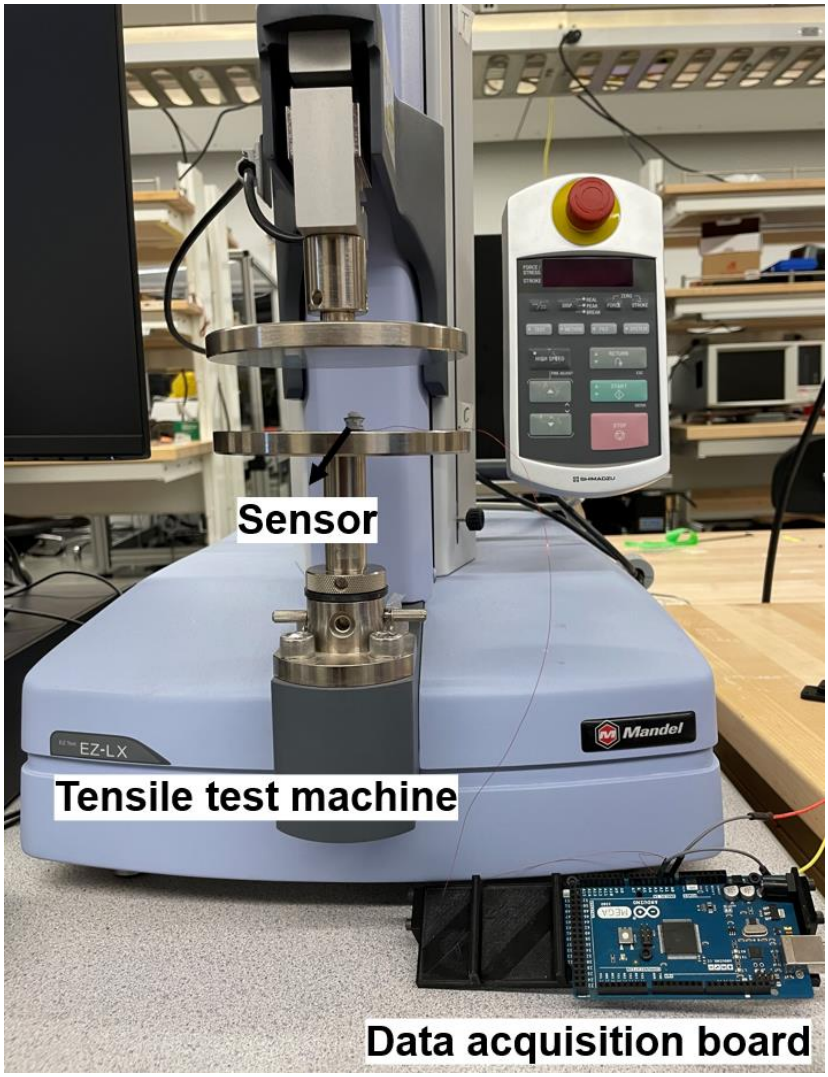
Supplementary Note A.



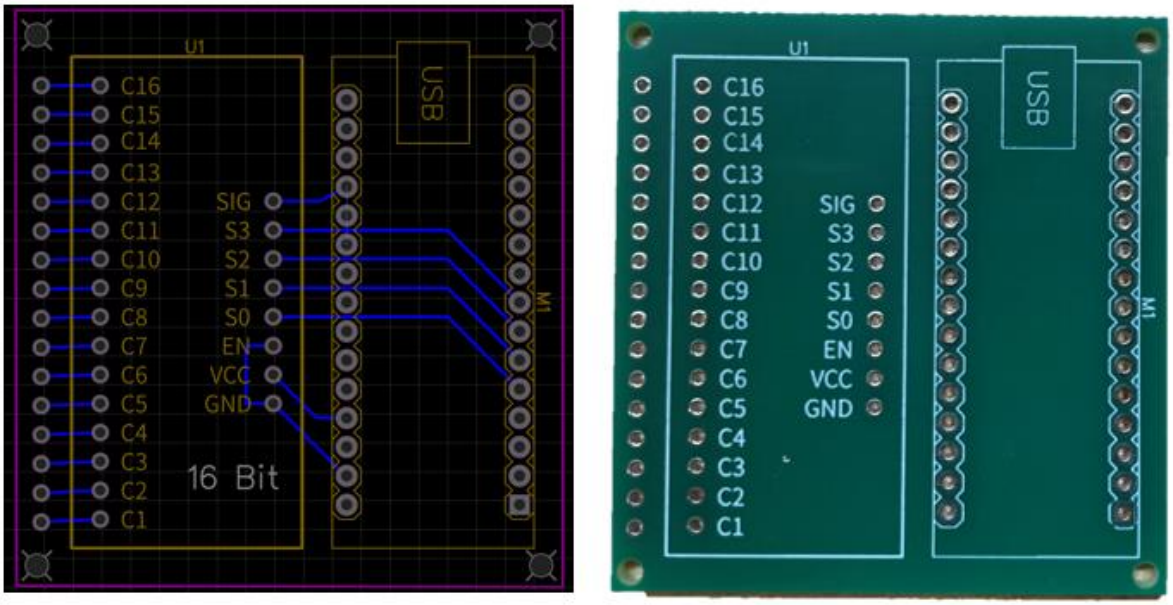
Supplamantery Fig. 1 Using dual nozzle T-Rex 3.0 printing machine to fabricate both the flexible base and dielectric structures of the Pillar-Origami pressure sensor.



Supplamantery Fig. 2 Flexible Conductive Filament Charactaristics. a The tensile test demonstrates a 10% increase in resistance in the dog-bone sample after 50% tensile strain. **b** The dielectric structure exhibits a 90% decrease in resistance after compression, leading to a significant change in the capacitance of the pressure sensor.

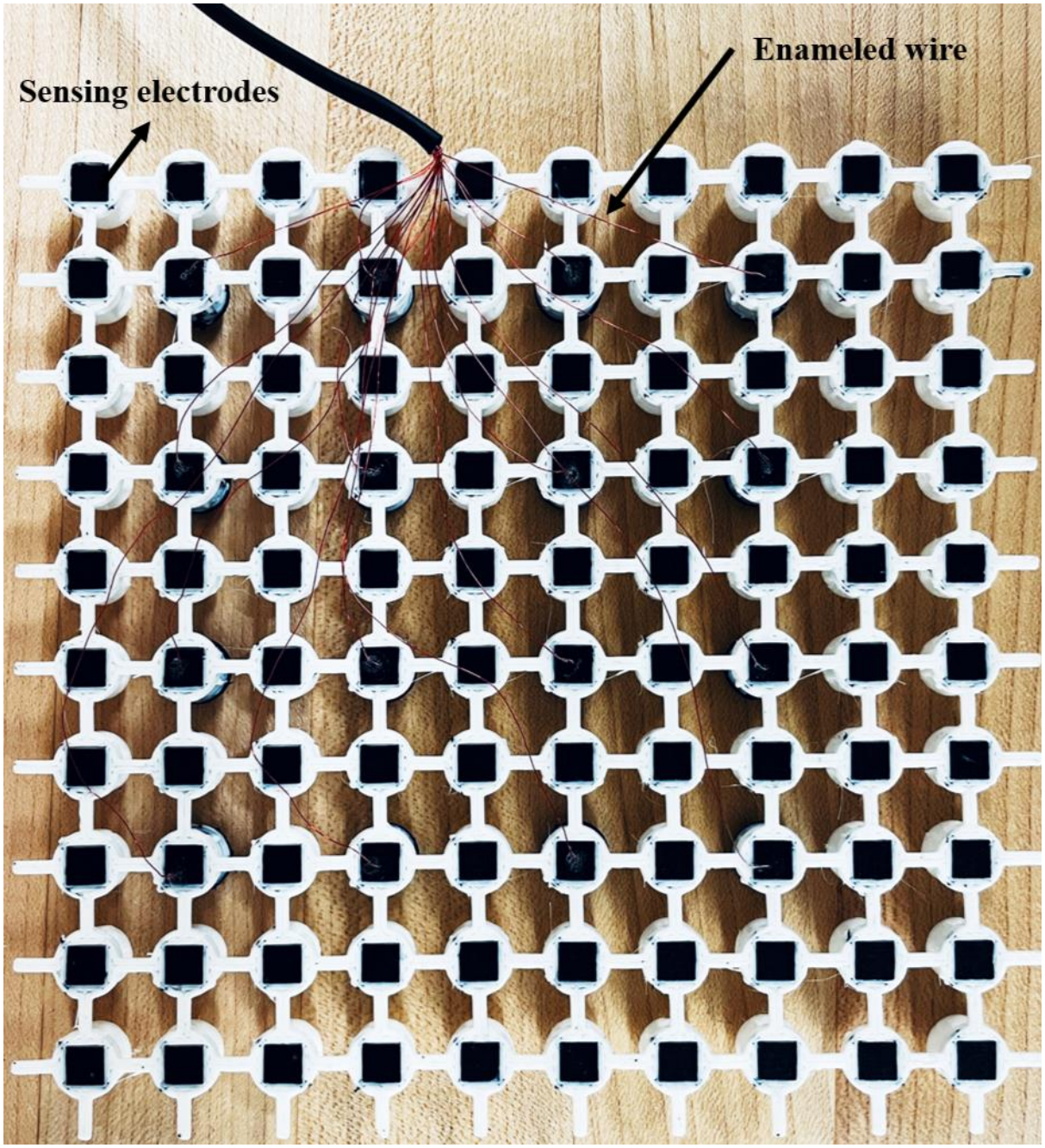


Supplamantery Fig. 3 The sensor performance experiment setup includes an EZ-LX tensile-compression machine and a data acquisition board for measuring the sensor's capacitance change during compression.

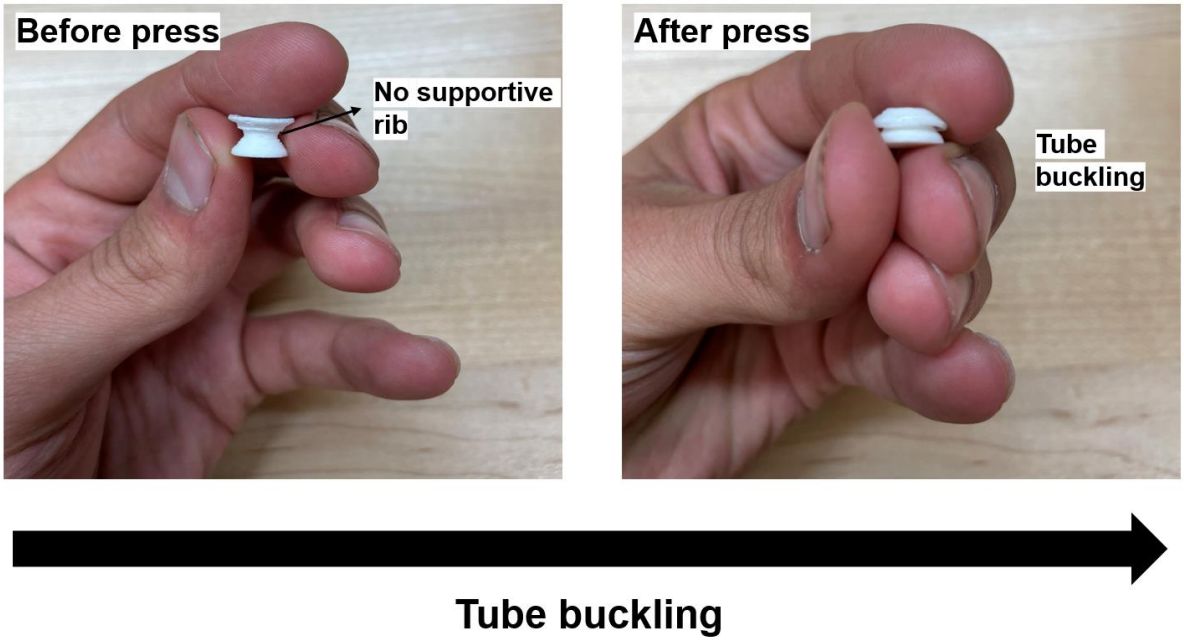


Circuit layout **Printed circuit board**

Supplamantery Fig. 4 Data acquisition board, including circuit layout and the printed PCB.



Supplamantery Fig. 5 Demonstration of the sensing array with the electrodes. Enameled wires are connected to the printed EEL electrodes to transmit the electrical signals to the data acquisition board.



Supplamantery Fig. 6 Buckling issues in the origami tube without supporting ribs.

Supplementary Note B.

Arduino codes for capacitance measurement and data transmission

```
#include <CapacitiveSensor.h>
#include <Capacitor.h>

#define muxS0Pin 2
#define muxS1Pin 3
#define muxS2Pin 4
#define muxS3Pin 5

#define muxSignalPin A2
#define capSenseSendPin A0

#define sampleRate 30
#define n_inputs 16

int touch[n_inputs] = {};

Capacitor cap1(muxSignalPin, capSenseSendPin);

int ValueMM(int channel) {
int controlPin[] = {muxS0Pin, muxS1Pin, muxS2Pin, muxS3Pin};

int muxChannel[16][4] = {
   {0, 0, 0, 0}, //channel 0
   {1, 0, 0, 0}, //channel 1
   {0, 1, 0, 0}, //channel 2
   {1, 1, 0, 0}, //channel 3
```

```
    {0, 0, 1, 0}, //channel 4
    {1, 0, 1, 0}, //channel 5
    {0, 1, 1, 0}, //channel 6
    {1, 1, 1, 0}, //channel 7
    {0, 0, 0, 1}, //channel 8
    {1, 0, 0, 1}, //channel 9
    {0, 1, 0, 1}, //channel 10
    {1, 1, 0, 1}, //channel 11
    {0, 0, 1, 1}, //channel 12
    {1, 0, 1, 1}, //channel 13
    {0, 1, 1, 1}, //channel 14
    {1, 1, 1, 1} //channel 15
  };

  //loop through the 4 contol pins
  for (int k = 0; k < 4; k ++) {
    digitalWrite(controlPin[k], muxChannel[channel][k]);
  }

return float(cap1.Measure());
}

void setup() {
  pinMode(muxS0Pin, OUTPUT);
  pinMode(muxS1Pin, OUTPUT);
  pinMode(muxS2Pin, OUTPUT);
  pinMode(muxS3Pin, OUTPUT);

  digitalWrite(muxS0Pin, LOW);
```

```
  digitalWrite(muxS1Pin, LOW);
  digitalWrite(muxS2Pin, LOW);
  digitalWrite(muxS3Pin, LOW);
  Serial.begin( 9600 );
}
void loop() {
  byte touchIndex = 0;
  for ( byte channel = 0; channel < n_inputs; ++channel ) {
        Serial.print(abs(ValueMM(0)));
        Serial.print(",");
        Serial.print(abs(ValueMM(1)));
        Serial.print(",");
        Serial.print(abs(ValueMM(2)));
        Serial.print(",");
        Serial.print(abs(ValueMM(3)));
        Serial.print(",");
        Serial.print(abs(ValueMM(4)));
        Serial.print(",");
        Serial.print(abs(ValueMM(5)));
        Serial.print(",");
        Serial.print(abs(ValueMM(6)));
        Serial.print(",");
        Serial.print(abs(ValueMM(7)));
        Serial.print(",");
        Serial.print(abs(ValueMM(8)));
        Serial.print(",");
        Serial.print(abs(ValueMM(9)));
        Serial.print(",");
        Serial.print(abs(ValueMM(10)));
```

```
  Serial.print(",");
  Serial.print(abs(ValueMM(11)));
  Serial.print(",");
  Serial.print(abs(ValueMM(12)));
  Serial.print(",");
  Serial.print(abs(ValueMM(13)));
  Serial.print(",");
  Serial.print(abs(ValueMM(14)));
  Serial.print(",");
  Serial.print(abs(ValueMM(15)));
  Serial.println();
  }
}
```

MATLAB code for wireless pressure mapping

```
clear all

close all

clc


% Define the location and values of sensors

Sensors = [600, 450, 300, 150, 600, 450, 300, 150, 600, 450, 300, 150; 150, 150, 150, 150, 300, 300, 300, 300, 450, 450, 450, 450; 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1];

X_Sensors = Sensors(1,:);

Y_Sensors = Sensors(2,:);

Z_Sensors = Sensors(3,:);


% Define parameters

Filtering_threshold = 100;

resolution = 10;


% Reading data from server

device = tcpclient('192.168.4.1',80,'Timeout',20)


%% Mapping and plotting

while 1

    Serial_Values = read(device,25);

    start_index = find(Serial_Values==255, 1);


    for k=1:25

        if Serial_Values(k) < Filtering_threshold

            Serial_Values(k) = 0;

        end

    end
```

```
    for k=1:12
        Z_Sensors(k) = Serial_Values(k + start_index)
    end

    % Making pressure mapping mat
    xv = linspace(0, 800, resolution);
    yv = linspace(0, 600, resolution);

    % Making the grid space for 2D mapping
    [Xm,Ym] = ndgrid(xv, yv);

    % Pressure interpolation from x-y grid
    f = scatteredInterpolant(X_Sensors', Y_Sensors', Z_Sensors', 'natural');
    Zm = f({xv,yv});
    contourf(Xm, Ym, Zm);
    colorbar
end
```

Matlab code to calculate l^*

```
clear all
clc

% Parameters
E = 20e6; % young's modolus
t = [0.5e-3 1e-3 1.5e-3]; % tickness
teta = [100 120 140]; % Fold length
v = 0.2; % poisson ratio
ln = [1.4 1.8 2.6]; % Number of ribs
d = 0.01;

% Origami stiffess obtained form experiments
K_Experiments = [2 26 91; 3.1 30 95; 3.7 39 100];

% Stiffness formula
K_Constant = (d*ln.*sin(3.14*teta/360))' * ((E*t.^3)/(12*(1 - v^2)));

% Calculating l*
l = K_Constant./K_Experiments;
plot (l);

% Final l*
l_star = mean(mean(l));
```