Supplementary File 2: WiFi Control via Blynk

To control the stretcher, we used a commercially available, customizable interface called Blynk (https://blynk.io/). Blynk is a platform designed for the Internet of Things (IoT) and provides the user with the flexibility to control hardware remotely and display/store data. To use Blynk one requires the following: hardware that is WiFi enabled (the ESP8266 in our system), internet access and a smartphone/tablet.

The major components of the Blynk platform are the Blynk application, Blynk server and Blynk libraries. The application builder (Blynk app) can be downloaded from applications stores on any iOS or Android devices. The application allows the user to build the user interface page through a simple drag and drop process from a directory of widgets (virtual buttons, displays and controls), designing the overall look of the application and controllers/displays of interest, as shown in Supplementary Figure 4. The application also provides the virtual interface with the project, which in our case is the stretcher. The Blynk libraries consist of commands that are included in the hardware code to allow for processing of any incoming or outgoing data. The Blynk server allows for communication between the hardware (ESP8266) and the smartphone.

Finally, the hardware code that is found below links the individual widget names (V0-V23) used in the application page to actions and calculations to be performed by the controller, ESP8266. We have segmented the code into two main divisions: cyclic stretch and static stretch codes. The following widgets were used: Numeric Input, Timer, value display, styled button and text input widgets. For cyclic stretching virtual pins V0-V3, V10-V16, while for static stretching virtual pins V4-V7, V17-23 were used. The details of initialization of these pins on the application are as follows:

Cyclic Stretch:

Numeric Input widgets:
V1: output 48-136 with 1 step increment
V2: output 0 – 20 with 1 step increment

Timer Widget:
V3: output 0-1

Value Display widgets:
V11-V15: Input: 0 – 1023
Refresh interval: 1 sec

V10: Input: 0 – 1023
Refresh interval: Push

Styled Button Widget:
V0: output 0-1
Mode: Switch

Text Input Widget:
V16: Character limit 10

Static Stretch:

Numeric Input widgets:
V5: output 48-136 with 1 step increment
V6: output 0 – 20 with 1 step increment

Timer Widget:
V7: output 0-1

Value Display widgets:
V17-V21: Input: 0 – 1023
Refresh interval: 1 sec
V23: Input: 0 – 1023
Refresh interval: Push

Styled Button Widget:
V4: output 0-1
Mode: Switch

Text Input Widget:
V22: Character limit 10

These setting must be initialized in the above manner for the code to run. Any changes to the output type and number (V) must be modified in the hardware code below.
The names of these widgets can be modified as desired as they do not affect the code.

Note: After building the application page of the stretcher using the Bylnk app, Blynk provides an authentication code (**1, below) that needs to be declared within the hardware code. This allows for communication between the hardware and the Blynk server, where your Blynk application widget details are stored.

Below is the hardware code for using the Blynk App to control the stretcher via WiFi. This program needs to be uploaded to the ESP8266 for the program to start running, this can be performed through the Arduino IDE.

Highlighted in yellow are user-specific inputs:

**1 – this is the authentication code provided by Blynk software

**2 – This is the WIFI username

**3 – This is the WIFI password

```
#define BLYNK_PRINT Serial
#include <ESP8266WiFi.h>
#include <BlynkSimpleEsp8266.h>
#include <Servo.h>
BlynkTimer timer;
Servo myServo;
// You should get Auth Token in the Blynk App.
// Go to the Project Settings (nut icon).
char auth[] = "**1";
// Your WiFi credentials.
// Set password to "" for open networks.
char ssid[] = "**2";
char pass[] = "**3";
int C_End_Position;
int C_Start_Position;
int S_End_Position;
int S_Start_Position;
float C_Percent;
float S_Percent;
int C_Time;
int S_Time;
int Cyclic_Switch_button;
int Static_Switch_button;
int servoPin=15; // The Pin number the actuator is connected to
String Running = "Running";
String Completed = "Completed";
String Stopped= "Stopped";
void Stretch() {
  Blynk.syncVirtual(V0);  // Check if switch still activated
```

```
  myServo.write(C_End_Position);
  Blynk.virtualWrite(V10, C_End_Position);
}
void Release() {
  timer.setTimeout(6500, Stretch);  // Run Servo loop
  Blynk.virtualWrite(V10, C_Start_Position);
  myServo.write(C_Start_Position);

}
void ServoDetach() {
  myServo.detach();
}
void C_STOP(int n){
  myServo.write(n);
  Blynk.virtualWrite(V0, 0);
  Blynk.virtualWrite(V16, Completed);
  ServoDetach();
}

void S_STOP(int n){
  myServo.write(n);
  Blynk.virtualWrite(V4, 0);
  Blynk.virtualWrite(V22, Completed);
  ServoDetach();
}
////////////////////////////// Cyclic Stretch input ////////////////////////////////////
BLYNK_WRITE(V1) // Start position data acquisition
{
  if (Cyclic_Switch_button==0){
  C_Start_Position= param.asInt();

  myServo.attach(servoPin);
  myServo.write(C_Start_Position);
  } else {
    Blynk.virtualWrite(V1, C_Start_Position);
  }
    Blynk.virtualWrite(V11, C_Start_Position);
}
BLYNK_WRITE(V2) // Percent movement data acquisition
{
  if (Cyclic_Switch_button==0){
  C_Percent= param.asInt();
       Blynk.virtualWrite(V14, C_Percent);
  float C_y= C_Percent/100;

  float C_z= (C_Start_Position*C_y);
       Blynk.virtualWrite(V12, C_z);
```

```
  float C_k= (C_Start_Position-C_z);
       Blynk.virtualWrite(V15 , C_k);
  C_End_Position= round(C_k);
       Blynk.virtualWrite(V13, C_End_Position);
  } else {
    Blynk.virtualWrite(V2, C_Percent);
  }

}
BLYNK_WRITE(V3) // Cyclic Timer data acquisition
{
  C_Time= param.asInt();
  if (C_Time == 1){
       Blynk.virtualWrite(V0, 1);
       Blynk.virtualWrite(V16, Running);
    myServo.attach(servoPin);
       Blynk.virtualWrite(V16, Running);
    timer.setTimeout(6500, Release);
   }else if (C_Time == 0){
   C_STOP(C_Start_Position);
  }
}
BLYNK_WRITE(V0) // Switch button to start Cyclic Stretch
{
  Cyclic_Switch_button = param.asInt();
  if (Cyclic_Switch_button == 1) {
    myServo.attach(servoPin);
    Blynk.virtualWrite(V16, Running);
    timer.setTimeout(6500, Release);  // Run motor loop every 6.5s

  } else {

    myServo.write(C_Start_Position);
      Blynk.virtualWrite(V16, Stopped);
      Blynk.virtualWrite(V10, C_Start_Position);
    ServoDetach();


  }
}

//////////////////////////// Static Stretch input ////////////////////////////////////////
BLYNK_WRITE(V5)// static initial motor position acquisition
{
  if (Static_Switch_button==0){
  S_Start_Position= param.asInt();
```

```
  myServo.attach(servoPin);
  myServo.write(S_Start_Position);
  } else {
    Blynk.virtualWrite(V5, S_Start_Position);
  }
    Blynk.virtualWrite(V17, S_Start_Position);
}
BLYNK_WRITE(V4) // Static Switch button to start static stretch
{
  Static_Switch_button = param.asInt();
  if (Static_Switch_button == 1) {
    myServo.attach(servoPin);
      Blynk.virtualWrite(V22, Running);
    myServo.write(S_End_Position);
      Blynk.virtualWrite(V23, S_End_Position);

  } else {
    myServo.write(S_Start_Position);
      Blynk.virtualWrite(V22, Stopped);
      Blynk.virtualWrite(V23, S_Start_Position);
      ServoDetach();
  }
}

BLYNK_WRITE(V6)// static Motor Percent movement acquisition
{
  if (Static_Switch_button==0){
  S_Percent= param.asInt();
       Blynk.virtualWrite(V18, S_Percent);
  float S_y= S_Percent/100;

  float S_z= (S_Start_Position*S_y);
       Blynk.virtualWrite(V19, S_z);
  float S_k= (S_Start_Position-S_z);
       Blynk.virtualWrite(V20 , S_k);
  S_End_Position= round(S_k);
       Blynk.virtualWrite(V21, S_End_Position);
  } else {
    Blynk.virtualWrite(V6, S_Percent);
  }

}
BLYNK_WRITE(V7)// Static timer acquisition
{
  S_Time= param.asInt();
  if (S_Time == 1){
       Blynk.virtualWrite(V4, 1);
```

```
      Blynk.virtualWrite(V22, Running);
   myServo.attach(servoPin);
      Blynk.virtualWrite(V22, Running);
   myServo.write(S_End_Position);
  }else if (S_Time == 0){
  S_STOP(S_Start_Position);
 }
}
////////////////////////////////////////////////////////////////////////////////////////////////
/////

void setup()
{
 // Debug console
 Serial.begin(9600);

 pinMode(LED_BUILTIN, OUTPUT);

 Blynk.begin(auth, ssid, pass);

}
void loop()
{
 Blynk.run();
 timer.run();
}
```