Twisted Nylon Muscle Manufacturing Process

William Donaldson

August 26, 2018

1 Introduction

Heat sensitive super coiled polymers (SCP) can be produced rapidly and cheaply using nylon fishing lines by inserting twist while maintaining a constant tension. This document outlines a process for manufacturing these muscles in a reproducible manner.

In order to make these muscles one must insert twist into a nylon filament under tension, and then anneal the twisted filament. Once this is done the muscle is put under tension (assuming it is wound in a homochiral manner) and when heat is applied the muscle will contract. When cooled the muscle will return to its original length.

Attributes of interest in these muscles include the filament diameter, coil diameter, spring index (ratio of coil to filament diameter), bias angle (angle between the coil and perpendicular plane), length and material. I will focus on muscles made of nylon as it has a higher melting point than the alternative, polyethylene, making the annealing process easier. One draw back of nylon is its tendency to absorb water, meaning that heating and cooling with hot/cold water is not a sustainable method of actuation.

Variables of interest that can be changed during the manufacturing process are; nylon length and diameter, counterweight mass during twisting and the annealing temperature and time.

The following 3 sections will outline the 3 stages in the manufacturing process.

2 Resistive Wire Wrapping

This stage of the manufacturing process is optional and can be skipped if you use other means of heating the twisted polymers.

2.1 Context

The purpose of winding resistive wire around the nylon fishing line before inserting twist is so that after the manufacturing process is complete, a current can be applied that will contract the muscle via Joule heating. This method has several advantages:

- Easy experimental setup; apply different temperatures to the muscle by rotating a potentiometer.
- Uniform global heating along the length of the muscle.
- Can easily be woven into complex geometries.

One hypothesized (untested) drawback is local hot-spots where the wire is in contact with the nylon. It is possible that after prolonged use, or periods of short but high temperature use, that the wire will begin to melt the area of nylon it is in contact with. I would advise experimental data collection and optical inspection of contact points to identify any potential damage.

There are two primary alternatives to Joule heating with resistive wire and they are the use of a heat gun, or hot and cold water running in tubes. Both of these have the disadvantage of requiring additional, bulky components. With the former, applying a constant temperature, uniformly along the length of the muscle may prove difficult. I hypothesis that the most reliable method for heating is to house the muscles in a tube and oscillate hot and cold water. However as mentioned this would be bulky, hard to weave with other muscles and over time nylon will absorb water.

2.2 Manufacturing Rig

The code for running this rig can be found in Appendix A.

This rig is run on an Arduino Mega with an external 12v power supply. The circuit consists of 3 stepper motors and their drives, a 1.8 inch LCD, limit switch, potentiometer and switch. The LCD, potentiometer and switch were meant to be used as a user interface to start/stop the rig and to set the coil density. Since the required functionality is minimal in the end I opted to hard code these variables into the code instead of using a user interface. I have left these components in the rig in case you wish to use them in the future. See Appendix D for circuit diagram.

This rig will spin the nylon filament while wrapping nichrome wire around it, operational procedure is as follows:

1. Cut a length of nylon fishing line and tie the ends to 2 metal paper clips with a fishermen's knot.

2. Attach one paper clip to the stepper motor on the left, the other to the spring on the stepper motor to the right. This spring is used to keep the nylon under tension during wrapping.

3. Set your chosen nichrome coil density as a variable in the code and upload the code onto the Arduino Mega. Disconnect the USB after upload and connect the external power supply.

4. Manually wind the nichrome wire around the left hand end of the nylon filament and the paper clip it is attached to. Wrap 2 inches of copper tape around this area ensuring good electrical connection with the paper clip and nichrome wire. This will act as an electrode later and will also stop the end of the nichrome wire unwinding.

5. Begin the winding process by pressing the switch. Wait while the nichrome winds down the length of the nylon filament. Use the switch a second time to stop the winding when the nichrome reaches the other paper clip.

6. Cut the nichrome with about 2 inches to spare and wrap this tail around the knot at the end of the nylon and through the paper clip. Again attach copper tape, ensuring good electrical connection between the tape, wire, and paperclip.

7. Remove the nylon from the rig and test electrical connectivity with a multimeter. Reset the position of the sliding spool component by pressing the button again and stopping it when it reaches the left hand side of the rig by pressing the button again.

2.3 Improvements

During the build of this rig I discovered some minor issues and suggested improvements:

- To ensure the nichrome spool unwinds at a constant rate, instead of relying on friction unwinding the spool, mount the nichrome spool on a stepper motor spinning at a constant RPM.

- I was unable to reliably source an 8mm guide rail of sufficient length that would ship in a timely manner. Instead I used some cold rolled rods from The Home Depot, if you remove these from the rig you will notice they are not perfectly cylindrical and this introduces unwanted friction, and consequently ruins the uniform application of the nichrome. As a temporary fix I have lubricated the bars with oil, but I suggest sourcing a perfectly cylindrical 8mm rod online.

- Currently the limit switch is not coded into the system as the process is largely manually controlled. If a second limit switch was attached at the other end, one could safely automate the entire process and use the limit switches for zeroing the position.

3 Twisting Rig

The code for running this rig can be found in Appendix B. Circuit diagram for this rig can be found in Appendix E.

This rig has a user interface consisting of an LCD, two potentiometers (the one on the left side for calibration, the one on the right side to adjust the LCD screen brightness) and two buttons, the left and right buttons will be referred to as button one and button two, respectively. More information about the command structure can be found in Appendix C.

This rig has two functions; measure the force applied by a hanging counterweight, and twist a nylon filament. Operational procedure as follows is divided into these two categories.

3.1 Force Measurement

This function can be chosen by pressing button one from the main screen.

By connecting the cord tethered to the side of the rig to the counterweight (with the load cell and HX711 sensor on the other end of the cord) can measure the applied force. It is important to note that recalibration may be required (see Appendix C for instructions). It is also important to remember that the load cell is only rated for 780 grams and this should not be exceeded.

While the force applied can be measured with a load cell, I found it easier and less time consuming to use the white table top scale located in the lab to determine the weight of the counterweight and use this value to calculate the applied force. Additionally friction between the rails adds to the applied force and can be reduced with lubrication.

3.2 Insert Twist

This function can be chosen by pressing button two from the main screen.

- Connect the fishing line from the previous step between the DC motor and the 3D printed sliding rail.
- Fill the water bottle with water to apply constant tension while twisting. Using a syringe can enable small increments of mass to be added at a time. The rated strength and diameter of the fishing line will determine how much mass is required and will vary across brands. For example when using Berkely Trilene Big Game, 0.71mm diameter fishing line, I found that 210grams was a suitable counterweight. Too little tension and the filament will frizzle when wound, too much and it will snap.
- Hang the counterweight water bottle over the edge of the table via the rotating bearings. Connect the external power source and using buttons 1 and 2 to navigate the user interface to find your desired application. See Appendix C command structure.
- When the twisting command is chosen the DC motor will begin spinning while the other end of the nylon, connected to the sliding rails will begin moving closer to the DC motor as the nylon is wound.
- When the nylon polymer is completely wound, stop the DC motor using button one. The number of revolutions is recorded using an optical encoder.

3.3 Improvements

During the build of this rig I discovered some minor issues and suggest some improvements:

- Replace the 8mm rods with cylindrical ones, or at least lubricate with oil (see previous comments in section 2.3).
- On the ends of the linear rails I have added dampening springs to prevent damage to the rig. These springs are a snug fit and if desired they could be exchanged for some larger diameter ones.
- At the ends of the rig are limit switches, currently these are not in use, but if it is desired they could be programmed into the code in Appendix B.
- Paracord connects the sliding frame to the counterweight, to reduce the mass and possibly friction this could be replaced with fishing line or similar.

4 Annealing

Annealing, or heat treating, the nylon muscles is the final stage of manufacturing. The purpose of doing this is to realign the nylon polymers so that when exposed to heat while under tension the muscle will return to its initial state.

Purchased in the lab is a toaster oven, to be used for this annealing process. I ran out of time to rigorously test suitable annealing conditions, and can only direct the reader to literature that varies from annealing with a heat gun to baking in an oven at temperatures ranging from 150 to 180 degrees Celsius, up to an hour. I hypothesis that the annealing temperature and time will depend on the filament diameter and the reader is encouraged to review literature and conduct experiments to find the optimal conditions for annealing.

5 Appendix A: Nichrome Winding Rig Code

```
/*
 * William Donaldson 2018
 * Code for a rig that winds nichrome wire around nylon fishing lines
 */


/* Uncomment following if you want to use the 1.8" LCD
#include <Adafruit_GFX.h>    // Core graphics library
#include <Adafruit_ST7735.h> // Hardware-specific library for ST7735
#include <Adafruit_ST7789.h> // Hardware-specific library for ST7789
#include <SPI.h>

#define TFT_CS     6
#define TFT_SCLK   5
#define TFT_MOSI   4
#define TFT_DC     3
#define TFT_RST    2
Adafruit_ST7735 tft = Adafruit_ST7735(TFT_CS, TFT_DC, TFT_MOSI, TFT_SCLK, TFT_RST);
*/

#include <AccelStepper.h>
AccelStepper rightStepper(1, 25, 24);
AccelStepper leftStepper(1, 27, 26);
AccelStepper beltStepper(1, 29, 28);

int togglePin=50;
int Direction=1;
bool Move=false;
int limitSwitch=21; //not currently being used, add in later if you want
int potPin=A0;      //not currently being used, add in later if you want

int MaxSpd=800;
int spd=80;         //ratio between these two speed variables will determine the nichro
int spinSpd=800;    //the following equality must be maintained: spinSpd<=MaxSpd

void setup() {
  pinMode(togglePin, INPUT);
  rightStepper.setMaxSpeed(MaxSpd);
  leftStepper.setMaxSpeed(MaxSpd);
  beltStepper.setMaxSpeed(MaxSpd);

  /* Following is not in use, but may be useful in the future
  pinMode(limitSwitch, INPUT);
  attachInterrupt(digitalPinToInterrupt(limitSwitch), trigger, CHANGE);
```

```
  pinMode(potPin, INPUT);

  tft.setRotation(3);
  testdrawtext("Lorem ipsum dolor sit amet, consectetur adipiscing elit. Curabitur ad
  delay(1000);
  */


}

void loop() {
   if(digitalRead(togglePin)==LOW){
    Serial.println(Direction);
    Direction*=-1;
    if (Move == true){
      Move=false;
    }
    else if (Move == false){
      Move=true;
      spd*=Direction;
    }
    delay(300);
   }

   if(Move==true){
    spinFilament(spinSpd);
    beltStepper.setSpeed(spd);
    beltStepper.runSpeed();
   }
   else if(Move==false){
    spinFilament(0);
    beltStepper.setSpeed(0);
    beltStepper.runSpeed();
   }
}

void spinFilament(int spd){
  //spins a filament strung between two stepper motors

  rightStepper.setSpeed(spd);
  leftStepper.setSpeed(-spd);
  rightStepper.runSpeed();
  leftStepper.runSpeed();
}

/*Not currently being used, but may provide insight if you choose to program the LCD
void testdrawtext(char *text, uint16_t color) {
  tft.setCursor(0, 0);
  tft.setTextColor(color);
  tft.setTextWrap(true);
  tft.print(text);
}
*/
```

5.1 Appendix B: Twisting Rig Code

```
/*
 * William Donaldson 2018
```

```
 * Code for a rig that twists nylon fishing lines to create artificial muscles
 */


#include "HX711.h"
#include <LiquidCrystal.h>

int calibration_factor=210; //Found experimentally by comparing to known weights
#define CLK  13
#define DOUT 12
HX711 scale(DOUT, CLK);

const int rs = 22, en = 24, d4 = 26, d5 = 28, d6 = 30, d7 = 32;
LiquidCrystal lcd(rs, en, d4, d5, d6, d7);

const int button1=20, button2=19;
const int potPin=A0;

const int B1A=23, B1B=25;

bool spin=false;
int spd;
bool Exit=false;

const int IRdetector=2;
volatile byte irCount=0;
int revCount=0;
float rpm=0.0;
int prevTime;
int NumberOfBlades=45;

void setup() {
  Serial.begin(9600);

  lcd.begin(16, 2);

  scale.set_scale(calibration_factor);
  scale.tare(); //Assuming there is no weight on the scale at start up, reset the scal

  pinMode(button1, INPUT);
  pinMode(button2, INPUT);

  pinMode(potPin, INPUT);

  pinMode(B1A, OUTPUT);
  pinMode(B1B, OUTPUT);

  pinMode(IRdetector, INPUT);
  attachInterrupt(digitalPinToInterrupt(IRdetector), counter, CHANGE);
}

void loop() {
  lcd.setCursor(0, 0);
  lcd.print("1. Measure Force");
  lcd.setCursor(0, 1);
  lcd.print("2. Insert Twist");

  if(digitalRead(button1)==HIGH){
```

```
    force();
  }

  if(digitalRead(button2)==HIGH){
    twist();
  }
  delay(10);
}

void twist(){
  lcd.clear();
  lcd.print("1. Start/pause");
  lcd.setCursor(0,1);
  lcd.print("2. Back");
  delay(500);
  while(digitalRead(button1)==LOW &&digitalRead(button2)==LOW){ //wait for instruction
    delay(5);
  }
  if(digitalRead(button1)==HIGH){
      spin=true;
      revCount=0;  //reset from previous run
      delay(500);
      lcd.clear();
      lcd.print("Twisting...");
      lcd.setCursor(0,1);
      lcd.print("press 1 to pause");
  }

  if(digitalRead(button2)==HIGH){
      spin=false;
      delay(300);
  }
  if(spin==true){
    while(spin==true){
        //spd=map(analogRead(potPin), 0, 1023, 0, 255);
        spd=160;  //since line above doesn't vary speed much
        analogWrite(B1A, 0);
        analogWrite(B1B, spd);
        if(irCount>(NumberOfBlades*2)){
          revCount+=1;
          irCount=0;
          Serial.println(revCount);
        }

        if(digitalRead(button1)==HIGH){ //pause
          delay(300);
          analogWrite(B1A, 0);
          analogWrite(B1B, 0);
          lcd.clear();
          lcd.print("Paused");
          lcd.setCursor(0,1);
          lcd.print("1=restart 2=stop");
          while(digitalRead(button1)==LOW &&digitalRead(button2)==LOW){ //wait for ins
            delay(20);
          }
          if(digitalRead(button1)==HIGH){  //resume twisting
            delay(300);
            lcd.clear();
```

```
            lcd.print("Twisting...");
            lcd.setCursor(0,1);
            lcd.print("press 1 to pause");
            //do nothing, return to twisting
          }
          if(digitalRead(button2)==HIGH){    //stop twisting completely
            delay(300);
            spin=false; //stop twisting
          }
        }
      }
    lcd.clear();
    lcd.print("# of revs:");
    lcd.print(revCount);
    delay(1000);
    lcd.setCursor(0,1);
    lcd.print("Press 2 to exit");
    while(digitalRead(button2)==LOW){ //wait for instructions
      delay(20);
    }
    delay(300);
  }
}

void counter(){
  irCount++;
}

void force(){
  lcd.clear();
  lcd.print("1. Calibrate");
  lcd.setCursor(0, 1);
  lcd.print("2. Measure Force");
  delay(300);
  while(digitalRead(button1)==LOW &&digitalRead(button2)==LOW){ //wait for instruction
    delay(5);
  }

  if(digitalRead(button1)==HIGH){
    lcd.clear();
    lcd.print("Hang known mass"); //hang known weight as a counterweight and find cali
    lcd.setCursor(0,1);
    lcd.print("Tune with pot"); //change calibation factor with potentiometer
    delay(3000);
    while(digitalRead(button2)==LOW){    //Need to press button to exit loop
      lcd.setCursor(0,0);
      lcd.print("Measured: ");
      lcd.print((scale.get_units()/1000.0)*9.81, 0);
      lcd.print("g");
      lcd.setCursor(0,1);
      lcd.print("Offset: ");
      calibration_factor=analogRead(potPin);
      lcd.print(calibration_factor);
      scale.set_scale(calibration_factor);
      delay(100);
      lcd.clear();
    }
    delay(500);
```

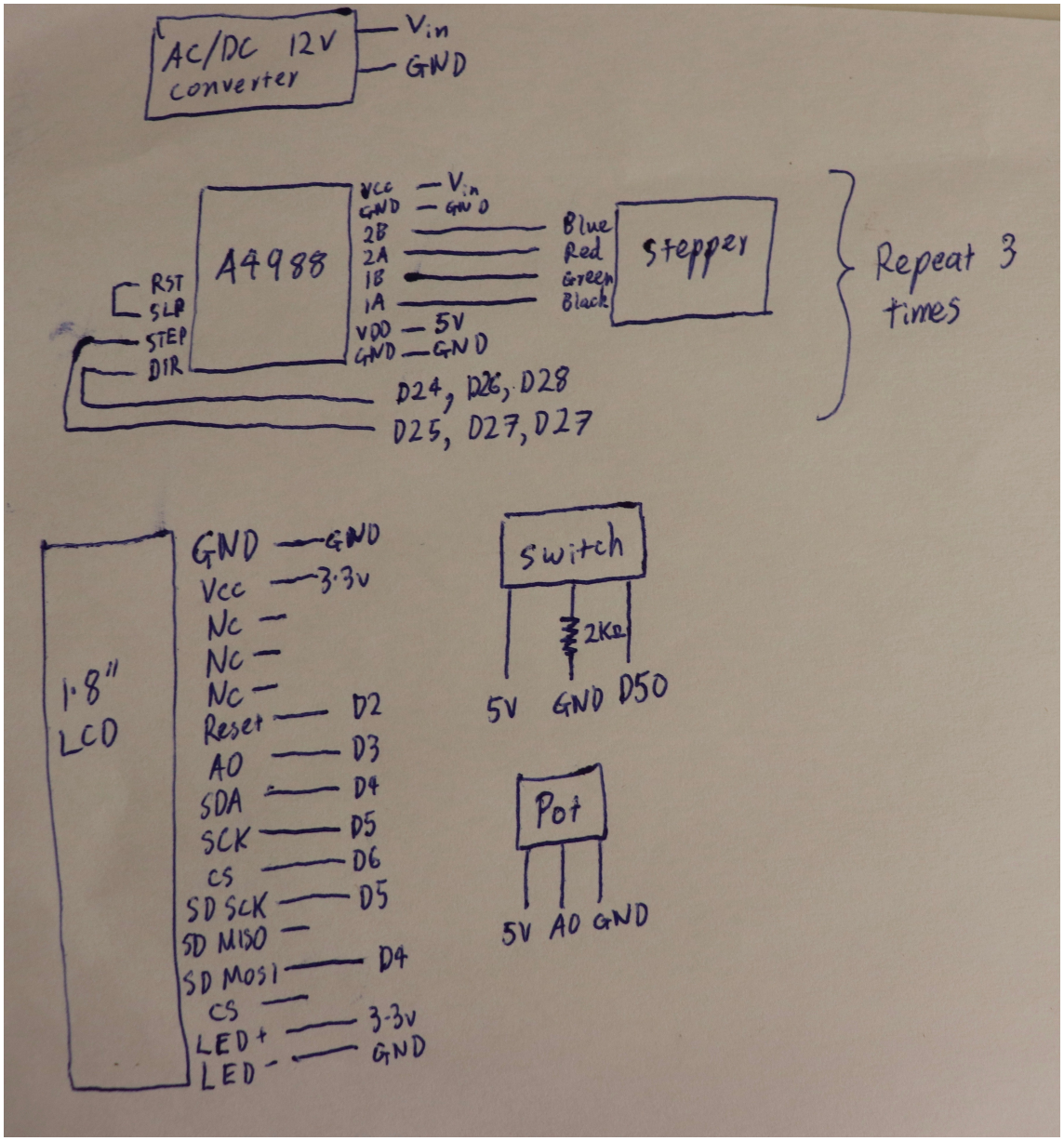
```
  }

  if(digitalRead(button2)==HIGH){
    lcd.clear();
    delay(200);
    while(digitalRead(button2)==LOW){ //wait for instructions
      lcd.setCursor(0,0);
      lcd.print("Force: ");
      lcd.print((scale.get_units()/100000.0)*9.81, 3);
      lcd.print("N");
      lcd.setCursor(0,1);
      lcd.print("Offset: ");
      lcd.print(calibration_factor);
      delay(100);
    }
    delay(300);
  }

}
```

6 Appendix C: Command Structure Twisting Rig

You can toggle through these commands using button one and two. While using the force calibration function, use the potentiometer on the left side to adjust the offset value.

- Press button 1 to measure force with the load cell
  - Press button 1 to recalibrate the load cell
    - Hang a known mass (maximum 780grams) on the load cell, use the potentiometer to change the offset until the measured mass equals the known mass. Return to the main menu by pressing button 2.
  - Press button 2 to measure the force on the load cell
    - Connect counterweight to the load cell. The force in Newtons will be displayed on the LCD. Note: if the offset value is wrong your measured force will be incorrect. Recalibrate the load cell if needed. If you turn off the power to the rig the offset value will return to the default value of 210. Press button 2 to return to the main menu.
- Press button 2 to insert twist into nylon fishing line
  - Press button 1 to begin spinning DC motor
    - Press button 1 to pause the DC motor
      - Press button 1 to restart the DC motor spinning
      - Press button 2 to stop the DC motor completely. The number of revolutions of the motor will be displayed. Press button 2 again to return to the main menu.
  - Press button 2 to return to the main menu

7 Appendix D: Circuit for Nichrome Winding Rig



8 Appendix E: Circuit for Nylon Twisting Rig