Characterization of the overall actuation system of a variable stiffness actuator based on pvdf-based springs

T.C.J. (Tim) Wijlens

BSc Report

Committee:
Dr. R. Carloni
Dr. H.K. Hemmes

August 2016

039RAM2016
Robotics and Mechatronics
EE-Math-CS
University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands

Contents

1. Introduction

Mankind is constantly advancing in technology. Seeking to improve what has been done before or find new ways to improve our lives. This is also the case in the exiting field of robotics. Long gone are the days that robots were just big and heavy machinery that had very limited capability. Robots nowadays are sophisticated pieces of precision equipment that can perform tasks to a high degree of accuracy and without complaining they want a break. That is why a lot of production processes are being automated. It is also the time that robots are leaving the factories and make way into our homes. Assisting robots that can improve the lives of the elderly or people who have a disability already exist and they are here to stay. However, these robots are not yet the human-like machines that can effortlessly perform any task a human would be able to do. The robots are usually designed to do a small number of tasks. A task such as filling a glass with water is a no-brainer for a normal human. The problem that a robot faces is the fact that the weight of the glass and its content is changing. Whereas a human can anticipate how much he needs to tighten his muscle, a robot needs an additional part to help him with that. That part is a variable stiffness actuator.

1.1. Variable Stiffness Actuator

In general a Variable Stiffness Actuator is a device that uses two motors to control 1 output. One motor gives the actual input, and the other motor controls the stiffness. Schematically this is shown in Figure 1.1. A practical Variable Stiffness Actuator can have many shapes, sizes and use different techniques, but in theory they are all the same.

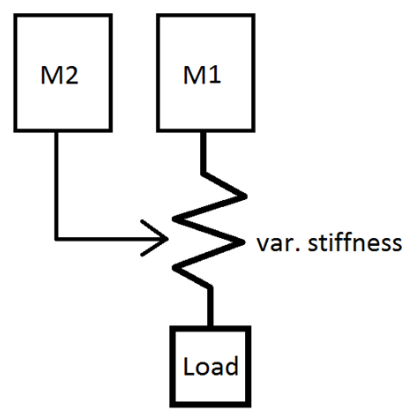

Figure 1.1: Illustration of a VSA

1.2. Piezoelectric Variable Stiffness Actuator

The next step in improving the VSA is by eliminating one of the motors from the system. Specifically, the motor controlling the stiffness. This is done by using a piezoelectric material in place of the motor + variable stiffness part of the VSA, schematically shown in Figure 1.2

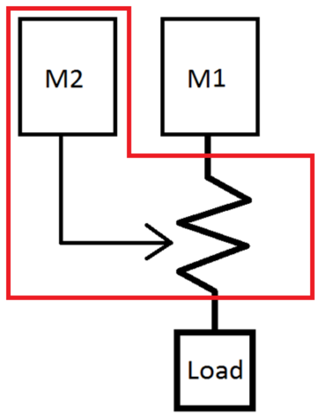

Figure 1.2: VSA with (in red) the part fulfilled by the piezoelectric material

A prototype of a VSA based on the piezoelectric material Polyvinyldifluoride (PVdF) has been built[2]. This material is found to have particular good properties needed for this application. This model consists of two rings loosely connected by multiple strips, or samples, of this piezoelectric material. Mechanical torque is given as input to the inner ring, transferred to the outer ring and outputted. When an electric field is applied to the strip of material, a mechanical strain is generated. This effect changes the behavior of the system and is what is the focus of this assignment.

The goal of this assignment is to characterize the prototype of the variable stiffness actuator. To do this a measurement setup has to be built and experiments need to be conducted.

2. The setup

The measurement setup built for the variable stiffness actuator is schematically shown in Figure 2.1. It also indicates how the devices are interacting with each other.

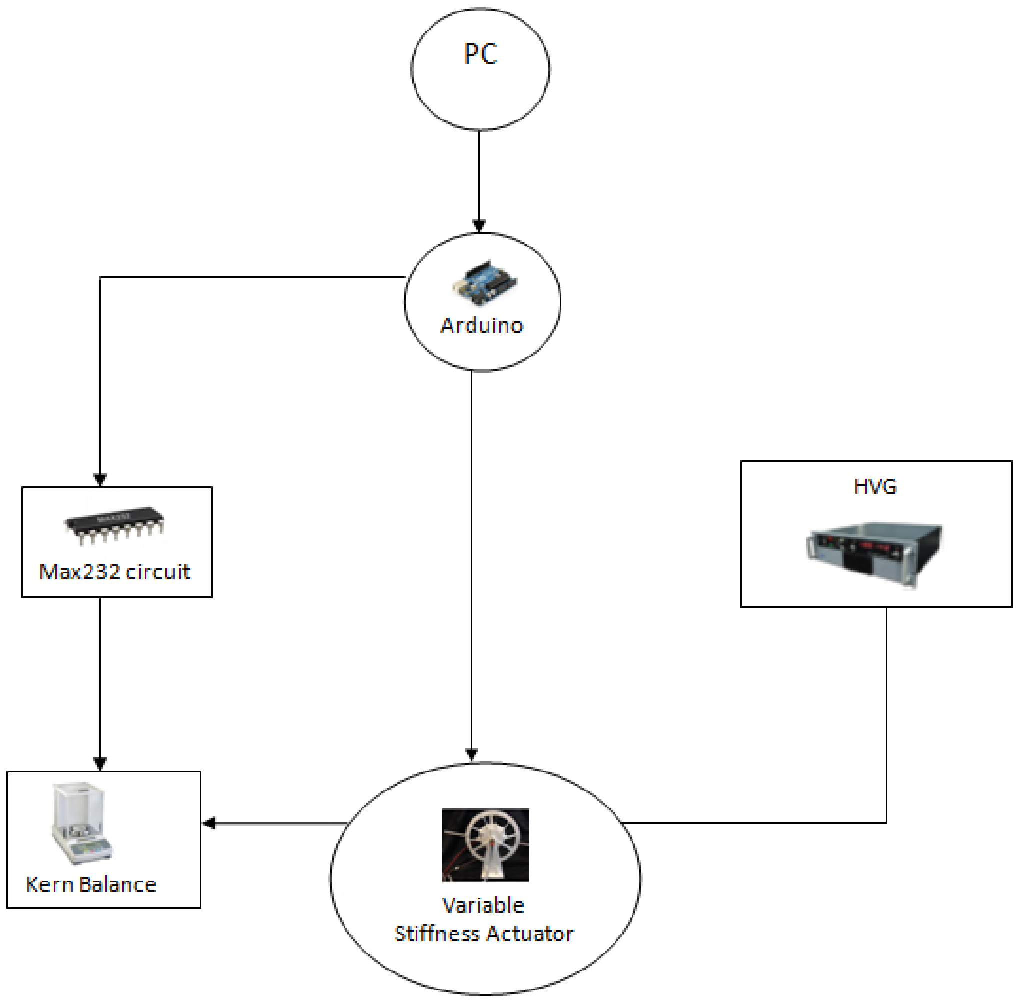

Figure 2.1: Schematic overview of the entire measurement setup

2.1. VSA Prototype

As mentioned before, a prototype for a Variable Stiffness Actuator with piezoelectric material already exists. A schematic of the prototype with all relevant parts indicated is shown in Figure 2.2. The input of the system is given by a SM-S2309S servomotor. This is a PWM (Pulse-width modulation) servomotor meaning that the position is controlled by a precise pulse whose duration (width) determines the position. The servomotor also has a feedback loop built-in that will make sure the servomotor is positioned correctly. The mechanical motion of the motor is transferred to a small gear attached to the motor shaft. This small gear is in turn coupled to a larger gear that is attached to a large sample holder.

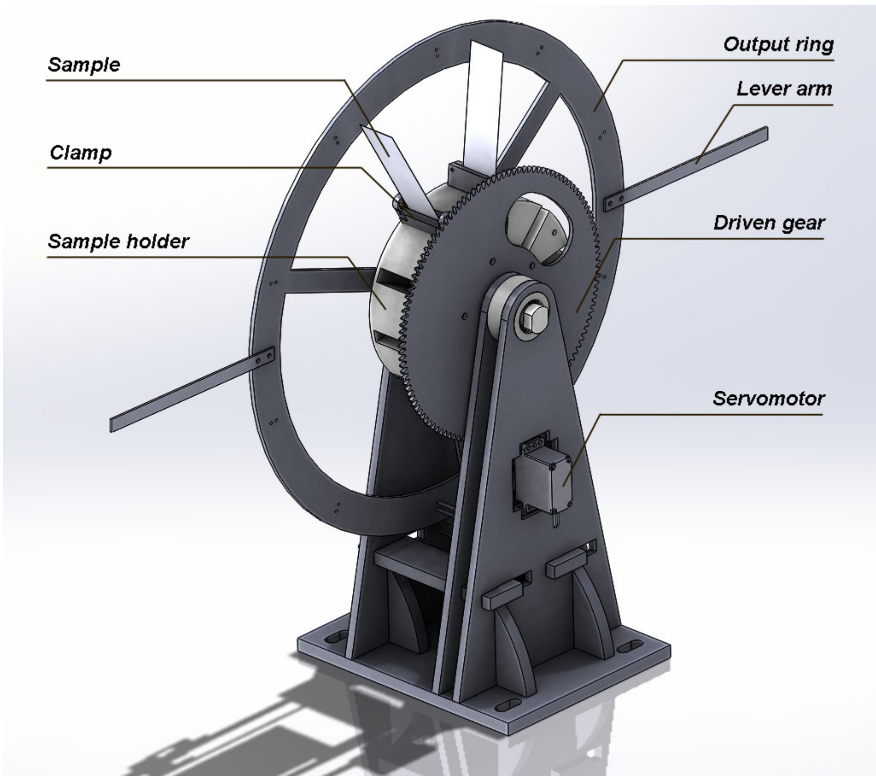

Figure 2.2: Prototype

The sample holder is where the piezoelectric samples with clamps are inserted into the prototype. In Figure 2.3 some of these samples with their clamps are shown. A total of 10 are used in the prototype evenly spread out over the holder. In order to apply a voltage over the samples, two lanes of conducting tape are taped over the outside of the sample holder and sides of the slots where the clamps are inserted. The inner and outer surfaces of the clamps themselves are also covered with two lanes of conductive tape.

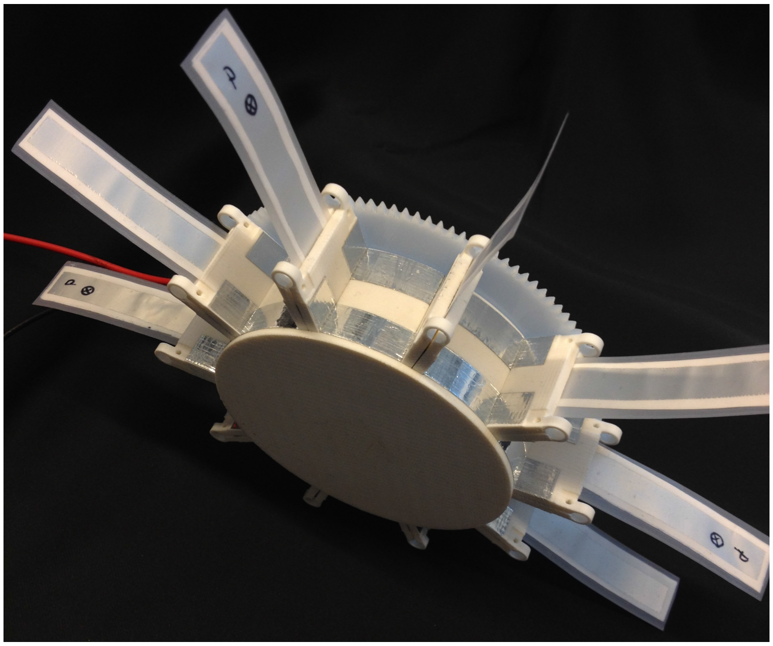

Figure 2.3: sample holder

Two legs of the samples, shown in Figure 2.4 are clamped onto these conductive lanes to form a circuit where all samples are connected in parallel.

Figure 2.4: All samples

The output of the prototype is given by the output ring. This ring, with a diameter of 22.9 cm, is where the samples are coupled. Two long plastic arms, the lever arms, attached to either side of the ring will be used to measure the output torque of the system.

2.2. Improved prototype design

2.2.1 pins and output ring

The pins that are mounted on the output ring and should prismatically couple the samples are made from a conductive material. Because there will be a 700 V potential across the samples, the pins could conduct and short the whole setup. To prevent this the pins needed to be redesigned in a non-conductive material. The material of choice is the same material the rest of the prototype is built from: Delryn. The new pins need to be stiff enough as to not flex during the measurement. This would otherwise add a additional (unknown) stiffness to the system. Therefore the physical connection of the pins to the output ring is of great importance. Before settling on a final design a number of other designs were made and constructed to evaluate the connection.

The final design of the pin is as follows: One part is a long plastic pin with a cut-out where the piezoelectric samples will be slid through. It has two hooks on the bottom where the pin will be inserted into a hole. The hooks will be snap the pin to the rest of the structure. This, in combination with a wide base area of the pin itself, ensures a good solid connection in the plane of the pin. A second part is needed to prevent movement in the perpendicular direction. This part also has a clicking mechanism and is inserted from the other side of the hole, perpendicular to the pin. The hooks snap into place and prevent the plastic pin from coming out. It also makes contact with the plastic pin and prevents movement in the plane perpendicular to that of the pin. Figure 2.5 shows a overview of both parts and the orientation in which they are to be inserted. Here the square part with the plus shaped cut-out is a test piece used to test the connection. This cut-out will be part of the output ring of the prototype.

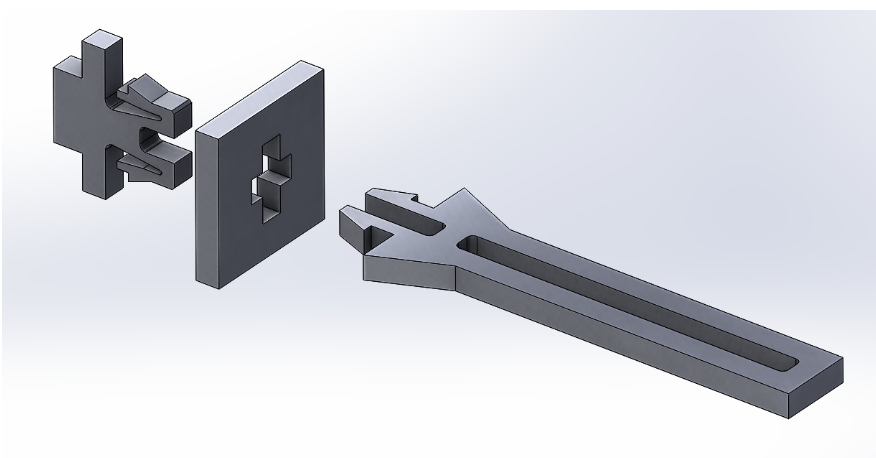

Figure 2.5: Redesigned plastic pin

Output ring

As a result of redesigning the pins, changes had to be made to the output ring as well. Originally the metal pins were inserted into small holes in the output ring. The new pins, however, are fitted using a click mechanism. Therefore an entirely new ring had to be made, keeping the same overall dimensions but with a modified pin connection. The pins are angled $+15\,\text{deg}$ and $-15\,\text{deg}$ relative to the radial. This has been done so that there is no net force applied to the output ring simply because of how the samples are held. And this also ensures the system does not perform differently when measured on either of the two arms.

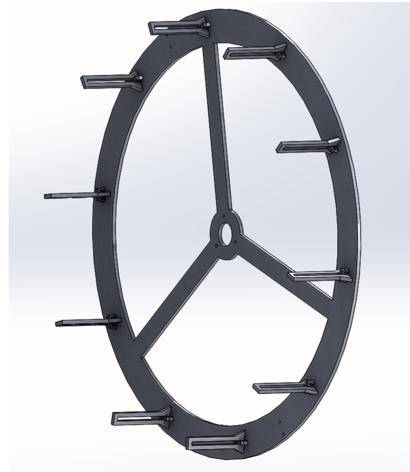

Figure 2.6: Assembly of the output ring

Balance stand

In order for the prototype to be able to transfer the forces onto the balance, a stand was designed. It needed to be as lightweight as possible because it would be placed on the weighing plate and thus count towards the 100g max weight the balance can read. The stand design is a simple pyramid shape with 2 interlocking plates. One of the plates is a little taller than the other and has a cut-out where the lever arm of the output ring will be inserted. For stability 2 more rounded parts are added. Using a click mechanism these ensure the entire stand is rigid. A design view is shown in Figure 2.7.

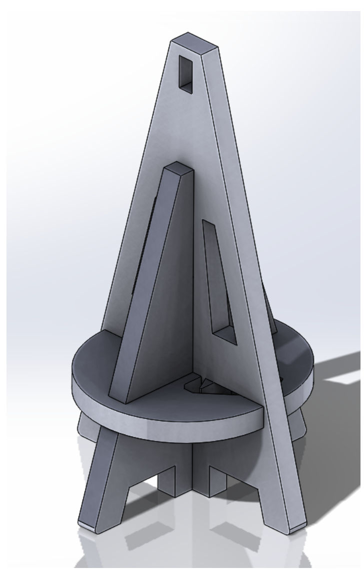

Figure 2.7: Balance stand

2.3. Analytical balance

The device used to act as a force sensor is the KERN ABT100-5m Analytical balance[5] This is a very accurate scale usually used for weighing medicine. It also has the capabilities to digitally output the weight. A connection needs to be established between the arduino and the analytical balance to retrieve these results. This is done by setting up a serial connection. All data is encoded according to the ASCII standard; Every character is first converted to a decimal value according to a ASCII chart[1] and then send as a series of bits in combination with bits that indicate the start and end of a character. The balance sends a special control character known as a carriage return when the all characters that make up the weighing result are sent. Before the balance starts outputting the weight a command needs to be sent from the arduino to the balance. The manual for the balance lists a few commands that can be used for the balance. The most important command is "D01" which will tell the balance to continuously output.

Figure 2.8: KERN ABT100-5m[5]

2.3.1 Voltage logic levels

The analytical balance uses a 25 pin RS-232c connector to communicate with the arduino. RS-232 is an electrical standard which specifies what voltage ranges correspond to a logical 1, and a logical 0. Only in these ranges are they registered as a binary 1 or 0. The logical voltage levels of the balance are different than the standard that the arduino uses, which is Transistor-Transistor Logic or TTL. The voltage levels for both the arduino and the analytical balance are shown in Table 2.1.

Logic	Arduino	Balance
0	0 V	+3 to +15 V
1	3 V	-15 to -3 V

Table 2.1: RS-232 Logic level and Voltage range

For the arduino and balance to communicate correctly, the voltage levels need to be converted. Because RS-232 is a widely used standard and thus these conversions are common, there is a single-chip solution that will scale and invert the voltage levels correctly. This is known as a MAX232 chip. Together with some additional components it forms the custom circuit needed for the balance and arduino to communicate to one another. The schematic for the circuit using the MAX232 chip is shown in Figure 2.9.

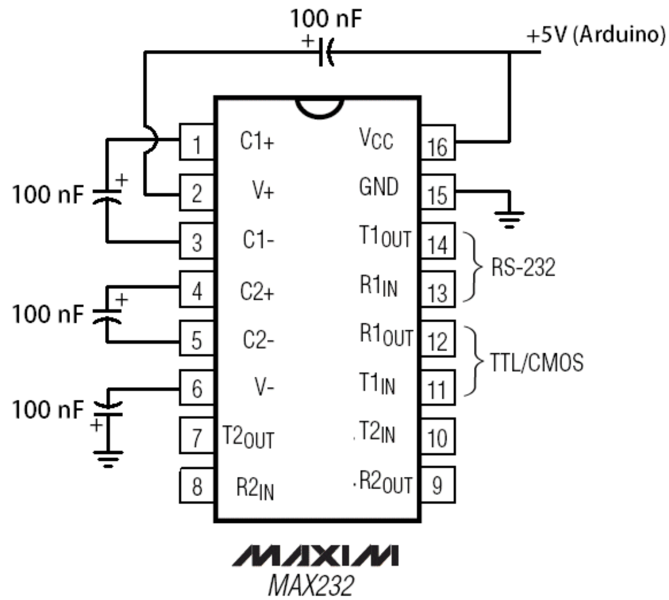

Figure 2.9: Schematic MAX232 circuit

2.3.2 Connection

The arduino and balance need to be connected. On the arduino side, there are already two allocated serial pins for communication with the PC. Because the arduino needs to independently communicate with the PC as well as with the balance these two pins cannot be used. The solution is to choose 2 other pins and configure the arduino to treat them as if they were serial pins. This can be done by using the SoftwareSerial library[3] for the arduino.

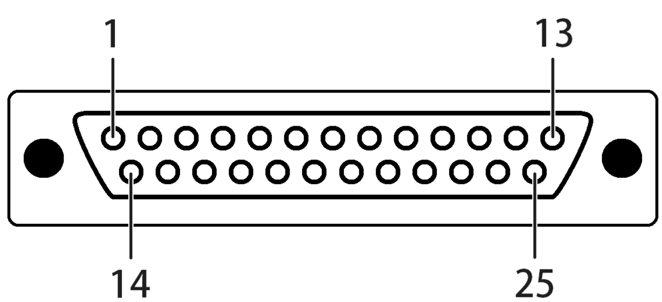

Figure 2.10: RS-232 connector pin numbering (male connector)

On the analytical balance, it uses a 25 pin RS-232c connector. A pinout for this connector is shown Figure 2.10. The functions for the pins are shown in Figure 2.11

Name	25-pin	9-pin	(DTE↔DCE)	Function (as seen by DTE)	
TD	2	3	→	transmitted data	data pair
RD	3	2	←	received data	
RTS	4	7	→	request to send (= DTE ready)	handshake pair
CTS	5	8	←	clear to send (= DCE ready)	
DTR	20	4	→	data terminal ready	handshake pair
DSR	6	6	←	data set ready	
DCD	8	1	←	data carrier detect	enable DTE input
RI	22	9	←	ring indicator	
FG	1	–		frame ground (= chassis)	
SG	7	5		signal ground	

Figure 2.11: Pin numbering and functions [4]

At this point all connections between the arduino and balance have been established. But unfortunately it is not as easy as connecting the transmitting and receiving lines to the custom circuit and hoping it all works. There is one more step to take before the two devices can start communicating wich will be treated next.

Handshaking

In all serial communications it is important that the data being sent is correctly received on the other side. Any loss of data can be disastrous in certain applications. To make sure that data is not lost if the receiving side does not have the resources to accept the data that is being sent, a safety is built in. This is known as handshaking. It uses additional pins on which the receiving and sending side can let the other know that it is ready to receive and/or send data. Only when this signal is given will any data be exchanged. In the case of the analytical balance this would mean that no data is send until such a signal is given. This would require additional pins to be wired up and, because of the differences in voltages, another circuit needs to be built. There is a way around this by tricking the balance into thinking that the receiving side is always ready to receive data, even if it is not. To do this, some extra wire are added which are looped back and soldered onto the connector. By tricking the balance into thinking the other side is always ready to receive data it eliminates the safety that prevents data from being lost. However, since it is known at all times when the balance should output data, the arduino can be programmed in such a way that it is ready to receive the data.

2.3.3 Environmental influences

For any measurement the environmental influences need to be minimized. The analytical balance is very sensitive to disturbances. examples are small vibrations of the building and movement of the air in the room. The vibrations are damped out by placing the setup on a air cushioned table. Any movement of the building or the ground it is built on will not be transferred to the measurement setup. The air movement is normally mitigated because the balance has an transparent enclosed space that prevents any movement of the air to influence the measurement. Since the prototype actuator is situated outside and has a lever arm that needs to stick in the enclosure to exert the force on the balance, this space cannot be fully closed off. A large hard plastic box was placed around the entire measurement setup.

2.3.4 High Voltage power supply

Because the change in stiffness of the samples only becomes noticeable for high voltages. a High-Voltage source is required. For the experiments, the Technix SR 300 Watt HV-Source is used. This power supply is capable of applying the required voltages of up to 700V. It is hooked up to a intermediate board that is mainly used as a safety measure if there is a short in the circuit hooked up to the HV terminals.

2.4. Arduino

Maybe the most important hardware in the measurement setup is the arduino Uno. This microcontroller controls the experiment and receives the important measurement data. The code written for the experiments with the prototype are included in appendix section 8.1. It is programmed to first step through the setup progress prompting to perform actions when they are required. When the initial setup has been completed it will automatically conduct the experiment. For every position sent to the motor (0-40 steps which translates into 0-14 degrees) the program first waits for 20 seconds. This is so the arduino will not register any weights the balance might send in the time it takes the whole setup to respond to the new position of the motor. After the wait it can be assumed that any new weights received are from the new position of the motor. The weight readouts is sent in packets of 1 character each, the code will assemble all packets into the weight as displayed on the balance. The program knows when the assembly is complete because of a delimiter character; the Carriage Return.

3. Measurement procedure

3.1. Variable Stiffness Actuator

The measurement procedure is set up as follows: Make sure all samples are correctly inserted in the sample holder and held by the plastic pins on the output ring. Also check the electrical connections of the setup. It is very important that there is no electrical connection between the legs of the sample. Start the code, shown in section 8.1, on the arduino and monitor for serial data on the PC. The arduino will first run through one cycle of the experiment, stepping through all positions of the experiment and return to the start position. After completion recheck that all samples are still inserted in their corresponding plastic pins. If all checks out cover the measurement setup with the box and apply the desired potential. Now the setup needs to rest for 1 hour after which the balance needs to be tared and the experiment can start. Each point takes around 5 minutes to complete bringing the entire experiment time to roughly 6 hours.

4. Results

4.1. 0V measurement

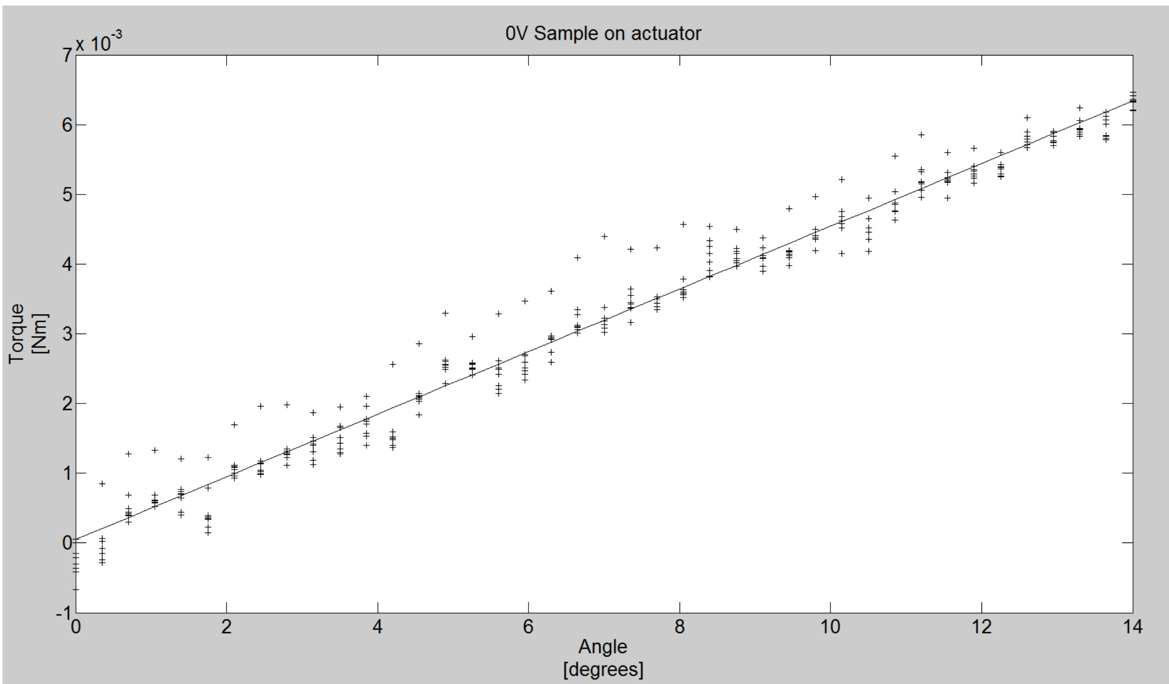

Figure 4.1: 0V measurement data from the actuator with linear fit

The experiments on the actuator are done in series of 41 measurement points with varying angles between 0 and 14 degrees. In the 0V measurement 8 of these series were carried out, all of which were done with 9 samples in the actuator. The results are shown in Figure 4.1. For the processing of the data, 8 measurement points were averaged for every angle. A linear line was fitted through these average values and is shown as a solid line in Figure 4.1. The function of this linear line is:

$$y = 4.4927 \cdot 10^{-4}x + 5.0807 \cdot 10^{-5} \tag{4.1}$$

4.2. 200V measurement

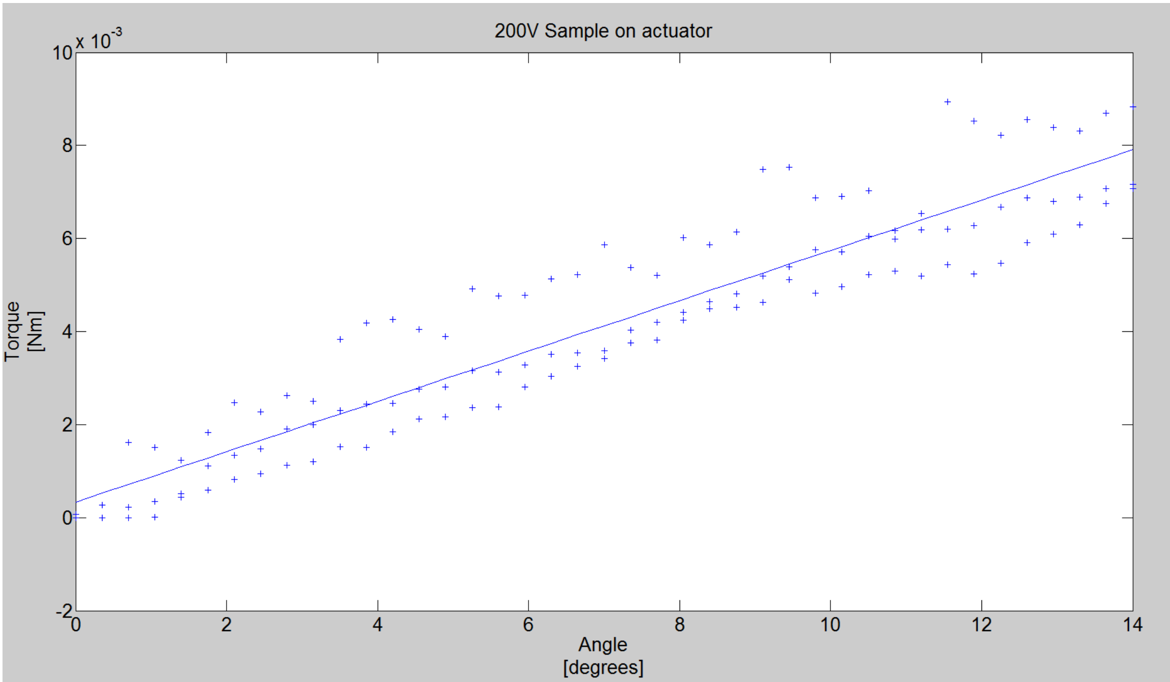

Figure 4.2: 200V measurement data from the actuator with linear fit

The series of measurements done at 200V are done with different number of samples. The first series at 200V were done with 9 samples and the remaining two were done with 7 samples, as two failed. This will be elaborated on in the discussion. While processing the data the measurement points from the last two series were scaled so they can be compared to the 9 sample series. The results of the 200V measurements are shown in Figure 4.2 The function of the linear fit is:

$$y = 5.4107 \cdot 10^{-4}x + 3.3439 \cdot 10^{-4} \quad (4.2)$$

4.3. 400V measurement

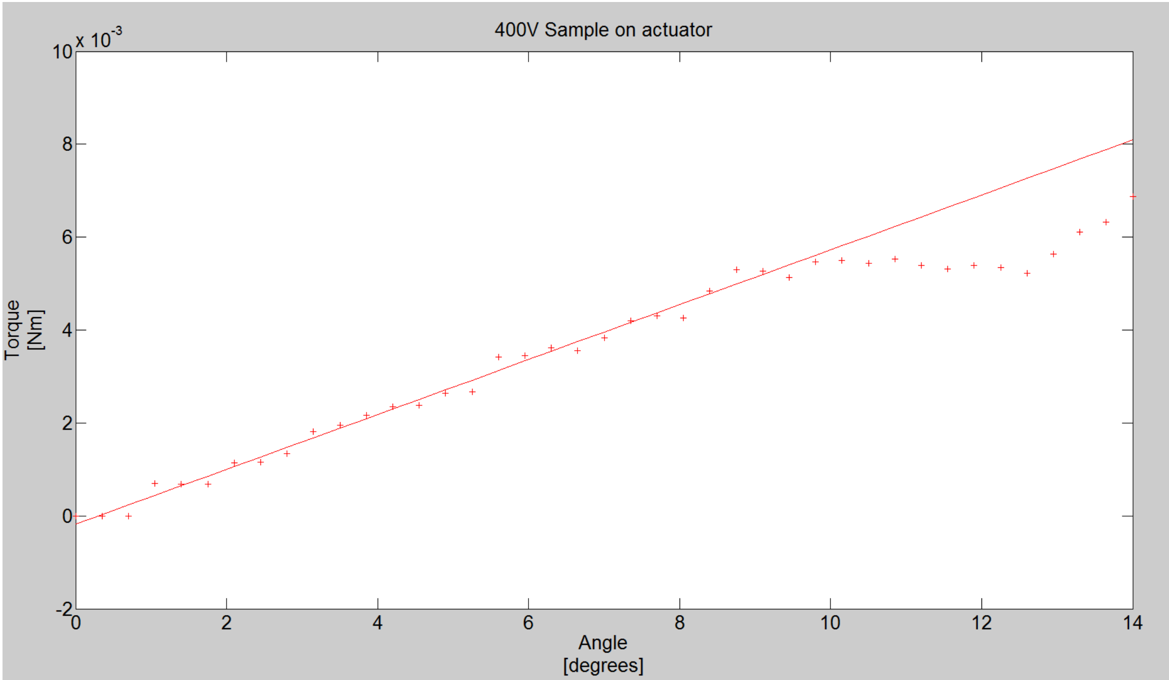

Figure 4.3: 400V measurement data from the actuator with linear fit (taken up to 9.8 degrees)

For the 400 volt measurement, due to time constraints, there is only 1 set of points. Near the end of the measurement the weighing results deviate substantially from the linear behaviour that was expected and seemed to flatten out. The linear line has been fitted on the first 28 measurement points (up to 9.8 degrees). All measurement points after this have been neglected for the linear fit. The data from the single series of measurements and the linear line that has been fitted are shown in Figure 4.3. The function for the linear fit is:

$$y = 5.9069 \cdot 10^{-4} x - 1.789 \cdot 10^{-4} \tag{4.3}$$

4.4. Individual sample test

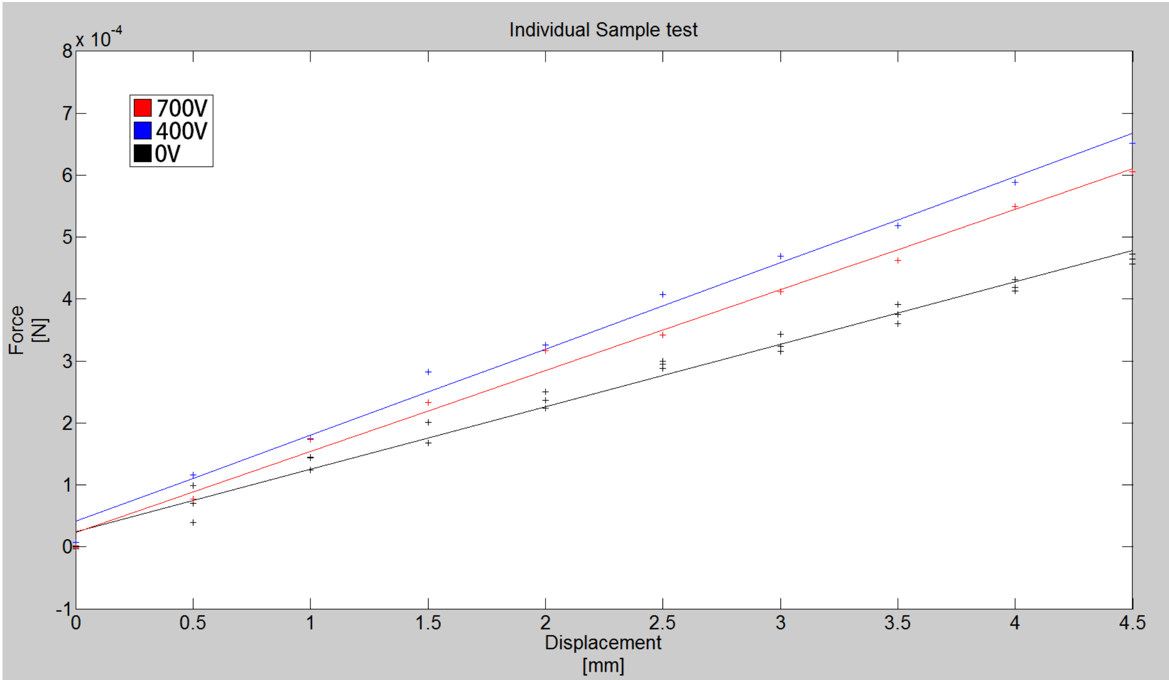

Figure 4.4: Force measurement under different voltages for 1 sample with linear fit

The individual samples that make up the Variable Stiffness Actuator were new samples. As the characteristics were not known, these needed to be retested. Figure 4.4 shows the resulting measurement data obtained from a single sample. A second sample was tested, but due various problems these results cannot be used. This will be further discussed later on.

$$0V: \qquad y = 1.0074 \cdot 10^{-4}x + 2.4942 \cdot 10^{-5} \tag{4.4}$$

$$400V: \qquad y = 1.3915 \cdot 10^{-4}x + 4.1036 \cdot 10^{-5} \tag{4.5}$$

$$700V: \qquad y = 1.3045 \cdot 10^{-4}x + 2.3508 \cdot 10^{-5} \tag{4.6}$$

5. Conclusion

This assignment aimed to characterize a prototype of a variable stiffness actuator based on the piezoelectric material Polyvinyldifluoride (PVdF). An existing actuator was modified and a measurement setup was built around it. It was able to measure the force the actuator could output for different angles between the input and output ring. Data obtained from the experiments was processed into the output torque the actuator exerts for applied voltages of 0V, 200V and 400V. Figure 5.1 shows the characteristic relation between the output torque and angle for different voltages. These relations are:

$$0V: \qquad \tau = 4.4927 \cdot 10^{-4}\theta \tag{5.1}$$

$$200V: \qquad \tau = 5.4107 \cdot 10^{-4}\theta \tag{5.2}$$

$$400V: \qquad \tau = 5.9069 \cdot 10^{-4}\theta \tag{5.3}$$

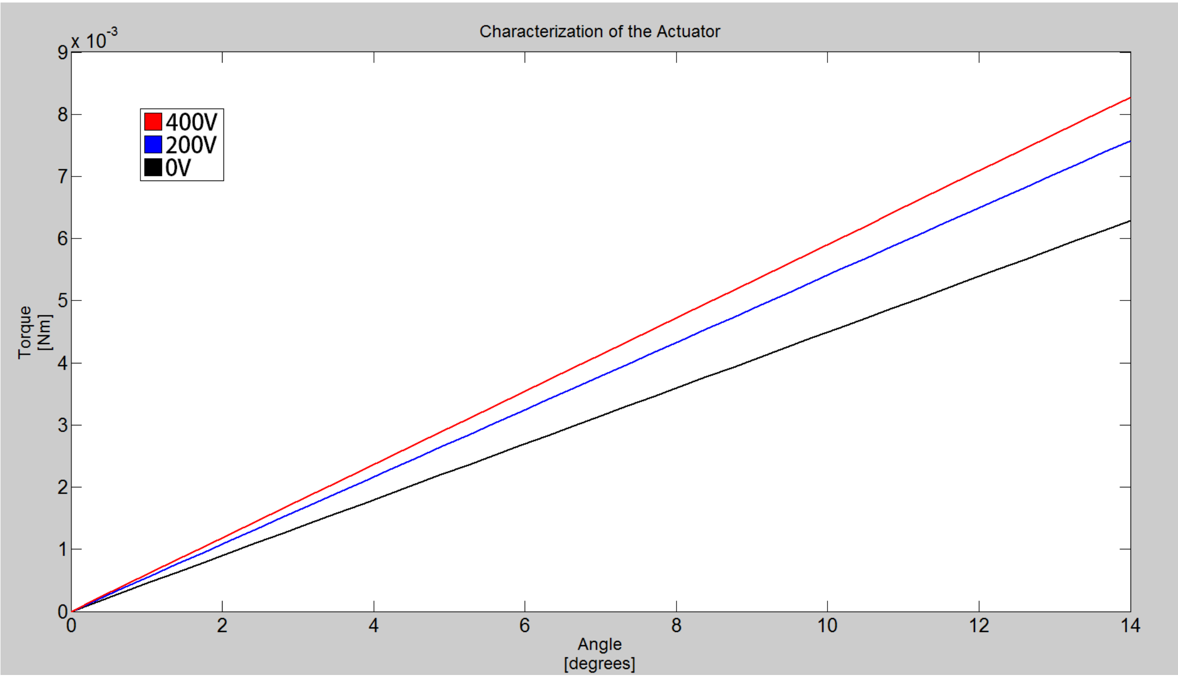

Figure 5.1: Characteristics of a variable stiffness actuator for different voltages

One of the samples that makes up the actuator was tested by itself. From the force measurement data it can be seen that the stiffness of the sample for different voltages are:

$$0V: \qquad k = 1.0074 \cdot 10^{-4} N/mm \tag{5.4}$$

$$400V: \qquad k = 1.3915 \cdot 10^{-4} N/mm \tag{5.5}$$

$$700V: \qquad k = 1.3045 \cdot 10^{-4} N/mm \tag{5.6}$$

6. Discussion

6.1. Failing samples

A large number of problems were encountered over the duration of this assignment. One of the biggest problems were the samples under investigation. Multiple piezoelectric samples were not able to withstand the potentials of up to 700V being applied to them. A total of 4 samples have failed. A detailed image of a sample, which failed around 400V, is shown in Figure 6.1. The sample was mounted correctly in the clamp making sure the conductive layer of the samples does not make contact with the conductive tape in the clamp itself. The point at which it failed thus lies outside of the clamp.

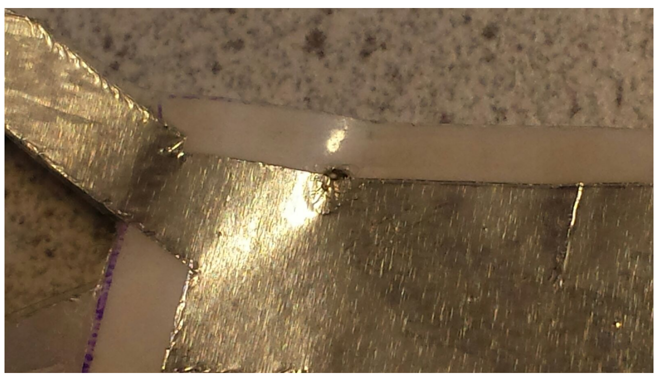

Figure 6.1: Detail of failed sample

The clear mark the short-circuit left on the sample is near the edge of the conductive layer. One explanation is that during fabrication of the sample a frayed edge might have remained. Because of the applied pressure to attach the conductive layer to the sample, it could have been possible that a small fray penetrated the isolating layer of the sample. This would now become the path of least resistance. When the applied potential is too high, it short-circuits at this point.

At a later stage when the actuator was rebuilt with the remaining samples, another sample began showing signs of failure. The faulty sample was quickly identified and isolated from the rest. While the sample was mounted in the clamp in such a way that only the legs of the sample were inserted, the voltage was increased. Around 300V the sample short-circuited. A screenshot of the moment it failed is shown in Figure 6.2.

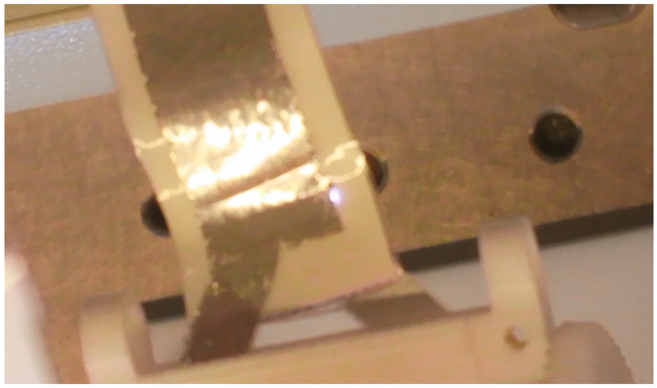

Figure 6.2: Moment a sample short-circuited

Just as with the first sample, it also short-circuited near the edge of the conductive layer. This sample however was slightly bend at the point of failure.

Another observation made was that some of the samples were falling apart. Figure 6.3 shows a side view of one of these samples. The individual layers that make up the sample were splitting. It is not known how severely this has influenced the force measurements.

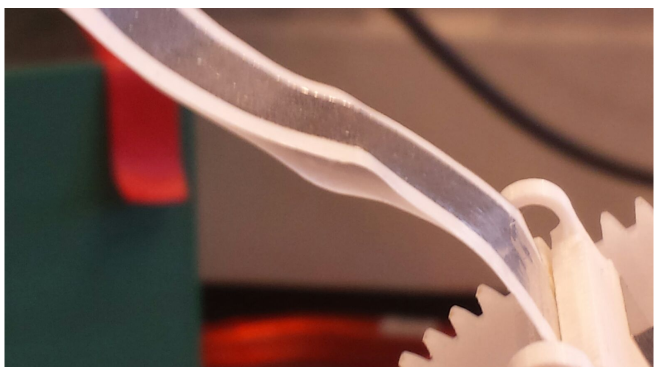

Figure 6.3: Sample splitting

6.2. 400V experiment flattening

During the single experiments where a potential of 400V was put over the samples, the force the actuator exerted seemed to flatten out above an angle of 14 degrees. The reason for this could be because there were only 5 samples remaining. Although they were evenly placed over the actuator these samples may not have been able to exert enough force.

6.3. Individual sample setup

The tests on the individual samples also required a voltage needed to be applied. The existing setup in combination with the samples would not be able to correctly apply a voltage to the sample. The conductive tape that was applied to the plates did not reach the sample, as according to the experiment it had to stick out 52 mm from the plate. Since there was a limited amount of conductive tape left, instead of replacing the whole tape with a new piece, the existing tape was extended by adding a small length of extra tape. It was only until after a number of experiments had been conducted that it was found that the tape does not conduct through its adhesive layer. Because multiple experiments had already been carried out that had a voltage applied, these experiments cannot be presented as reliable results because the voltage may in fact not have been applied across the sample. Sufficiently scratching the edge of the additional piece of tape, shown in Figure 6.4, has ensured a good enough conduction between the two separate pieces of conductive tape for further experiments.

Figure 6.4: Scratched tape

6.4. Alternative setup

During the preparations for the experiments it became clear that the analytical balance does not instantly respond to a serial command asking for a readout. The reason for this is because the manufacturer never intended the balance to be used in this way. It is usually used to precisely way medicine in pharmacies. These analytical balances can be verified on-site to proof that the weight measurement is accurate. Something which is required by law if the balance is used to weigh medicine or any high-value resource such as gold. Therefore the balance is protected so that any output it gives is considered "stable". It might have been possible to circumvent these restrictions by hacking into the balance (possibly damaging it) or load a new firmware from the manufacturer which might come with legal paperwork and large investments. Because of the uncertainties and relatively large investments in both options this was not looked into any further. Instead a new measurement setup was designed.

This alternative would allow the force to be measured at any time. The setup uses a Compression Load Cell for measuring the force and a signal conditioner to turn the analog output voltage of the load cell into a digital signal. The choice of adding the signal conditioner is because it has more bits for the Analog-to-Digital conversion: 16-bit for the arduino versus 20-bit for the signal conditioner. Another benefit is that it can also calibrate the load cell. The calibrated data can then be sent to the arduino either as a digital or a analog output. The digital output of the signal conditioner is a serial connection but, as opposed to the analytical balance, does not use the ASCII protocol for sending the data. It uses the Modbus-RTU protocol. The connector on the signal conditioner is a 3.5 mm jack.

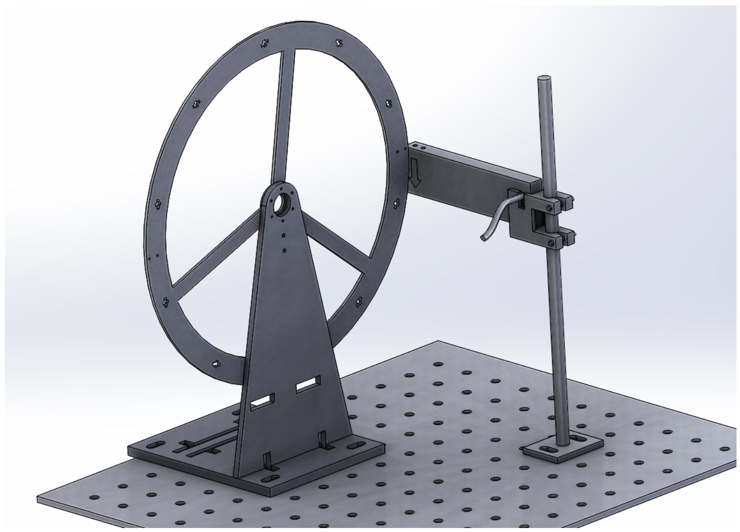

Figure 6.5: Alternative setup concept

The concept for the support is shown in Figure 6.5 The idea was to attach a bracket to the load cell and clamp that to a stiff metal rod. Some adjustment room was required in the design to be able to correctly position the load cell. Figure 6.6 shows the 3D printed bracket attached to the load cell.

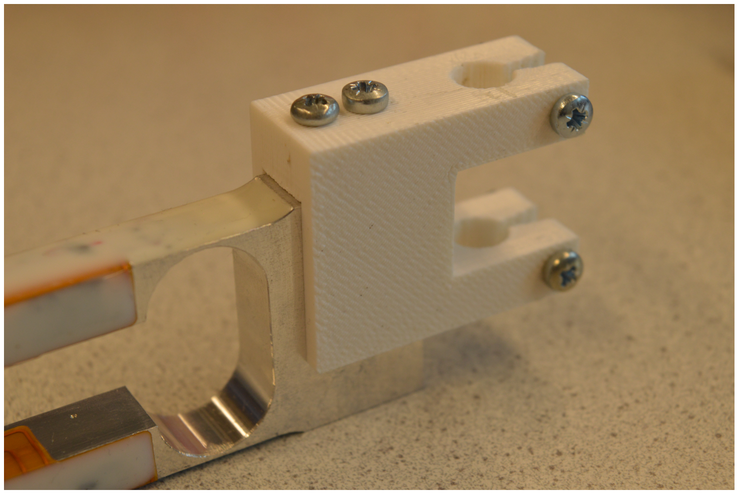

Figure 6.6: Bracket attached to the load cell

There was doubt whether or not the setup would be able to measure the small forces. In the end it was decided to return to the analytical balance and perform the measurements.

6.5. Suggestions for further research

- Further research could involve improvement of the samples, e.g. make them longer and broader. The legs of the samples could be straight and also be aligned with the lanes of conductive tape in the clamps. The legs of the samples used in the experiment had to be folded in order to fit in the clamps correctly. Every time the sample position in the clamp had to be checked, these legs would stick to the conductive tape inside the clamp and had to be carefully unstuck. By folding the already fragile legs they become even more fragile. Also, the isolation layer could be thickened so raffled edges can be avoided. Also the legs could perhaps be thickened by applying multiple layers of conductive tape, this would make them more firm.

- In order to find the exact cause of the short circuits the failed samples could be studied in more detail under a microscope.

- A more detailed investigation can be carried out regarding the servomotor. The motor is controlled by a PWM signal, whose pulse width indicates the required position. This signal is generated by the arduino. Because the arduino cannot generate a pulse with the perfect width, the servo motor might oscillate around the desired position. This oscillation carries through to the output ring and the measurement. This might be one of the reasons why the weight reading on the balance was taking a long time to stabilize.

7. Acknowledgements

This bachelor assignment is the completion of my study Advanced Technology at the University of Twente. Writing this bachelor thesis has been a challenging, yet very interesting process. Therefore I thank Dr. Raffaella Carloni for giving me the opportunity and guidance. Next to that my thanks also go to Dr. H.K. Hemmes for the in-between meetings. I thank especially the technicians of the research group Robotics and Mechatronics, Mr. H. Kuipers and Mr G. te Riet o.g. Scholten for supporting me during the problems with the experiments and for giving me very useful advice as well as other people within the Robotics and Mechatronics group. Of course, I also want to mention the support of my parents and brother. This has also been very important to me.

Bibliography

[1] Ascii code - the extended ascii table.

[2] Andrea Cremonese. Towards the realization of a variable stiffness joint with electrospun pvdf-based springs. Master's thesis, University of Twente, 2016.

[3] Mikal Hart. Softwareserial library.

[4] Paul Horowitz and Winfield Hill. *The Art of Electronics*. Cambridge University Press, second edition, 2008.

[5] KERN & Sohn GmbH. *Operating instruction Analytical balance*, version 1.1 edition, 2009.

8. Appendix

8.1. Arduino Code

```
#include <SPI.h>
#include <Servo.h>
#include "functions.h"
#include <SoftwareSerial.h>
#include <ModbusMaster.h>

#define SS 10
#define SCK 13
#define MOSI 11
#define MISO 12
SoftwareSerial mySerial(2, 3); // RX, TX
ModbusMaster node(1);

int voltageA0 = 0.0;

int pos = 0;
int serialPos = 0;

bool continue_ = false;
int incomingByte = 0;
int flushByte = 0;
unsigned int sensor_data = 0;              // complete sensor data, so position and error bits
float encoder_CPT = pow(2,14);             // encoder resolution, counts per turn (14-bit encoder)
float encoder_CPD = encoder_CPT / 360;     // encoder resolution, counts per degree of rotation
int pivot_OFFSET = 8390;
float output_POS = 0;
float output_start = 0;
float delta_POS = 0;
double zero_angle = 21;
int Theta_out = 0;
int steps = 0;
int balanceReadOut = 0;
String balString = "";
int cyclenum = 0;

Servo myservo;

void setup() {
        Serial.begin(9600);
        mySerial.begin(1200);
        Serial.print("------void setup()------");
        Serial.print('\n');
        delay(1000);

        //initialize SPI, tested to work this way with AS5048A (03-12-2013) (Retested on 7-6-2016)
        Serial.print("Initialize SPI...");
        SPI.setClockDivider(SPI_CLOCK_DIV16); //slow down the clock
        SPI.setDataMode(SPI_MODE1); //clock idle when low, rising edge data read
```

```
SPI.setBitOrder(MSBFIRST); //MSb is transferred first
SPI.begin();
Serial.print("DONE");
Serial.print('\n');
delay(1000);

Serial.print("Testing motor...");
myservo.attach(6);
for (pos = 0; pos <= 40; pos += 1)
{
        myservo.write(zero_angle + pos);
        delay(1000);
        }
                for (pos = 40; pos >= 0; pos -= 1)
        {
        myservo.write(zero_angle + pos);
        delay(1000);
}
delay(5000);
myservo.write(10);
delay(5000);
myservo.write(zero_angle);
Serial.print("DONE");
Serial.print('\n');

Serial.print("Check the samples (hit enter when done)...");
continue_ = false;
while (continue_ < 1)
{
        if (Serial.available()>0)
        {
                incomingByte = Serial.read();
                Serial.flush();
                continue_ = true;
        }
}
Serial.print("DONE");
Serial.print('\n');
delay(1000);

Serial.print("Taring the balance (input 'T' to tare, input 'G' when done)...");
continue_ = false;
while (continue_ < 1)
{
        flushByte = mySerial.read(); // since mySerial.flush() doesn't seem to work
        if (Serial.available()>0)
        {
                incomingByte = Serial.read();
                if (incomingByte == 'T')
                {
                        mySerial.print("T");
                        mySerial.print('\r');
                }
                else if (incomingByte == 'G')
                {
                        Serial.flush();
                        continue_ = true;
                }
        }
}
Serial.print("DONE");
Serial.print('\n');
delay(1000);

Serial.print("Reading initial sensor data...");
```

```
        output_start = receiveSensorData();
        Serial.print("DONE");
        Serial.print('\n');
        delay(2000);

        Serial.print("Sending command to balance to autoprint...");
        mySerial.write("D06");
        mySerial.print('\r');
        Serial.print("DONE");
        Serial.print('\n');
        delay(2000);

        Serial.print("Setup Complete.");
        Serial.print('\n');
        Serial.print('\n');

        delay(5000);
        Serial.print("------Running Experiment------");
}

void loop() {
        cyclenum += 1;
        Serial.print('\n');
        Serial.print('\n');
        Serial.print("Sample Voltage = 400V");
        Serial.print("  ,  ");
        Serial.print("Samples Remaining = 5");
        Serial.print("  ,  ");
        Serial.print("Run = ");
        Serial.print(cyclenum);
        Serial.print("  ,  ");
        Serial.print("Arm = 17.8cm");
        Serial.print('\n');

        Serial.print("*** Steps 0-to-40 ***");
        Serial.print('\n');
        Serial.print("Step");
        Serial.print('\t');
        Serial.print('\t');
        Serial.print("weight(gram)");
        Serial.print('\n');
        for (pos = 0; pos <= 40; pos += 1) {  // goes from 0 degrees to xx degrees
                myservo.write(zero_angle + pos);
                continue_ = false;
                delay(20000);
                //---------------Data of position output v. input ring---------------
                Serial.print(pos);
                Serial.print('\t');
                Serial.print('\t');

                //---------------RS-232 data readout balance---------------
                mySerial.flush();
                mySerial.print("D05");
                mySerial.print('\r');
                continue_ = false;
                while (mySerial.available()>0) //Custom flush code -> mySerial.flush doesn't work
                {
                        flushByte = mySerial.read();
                }
                while (continue_ < 1)
                {
                        if (mySerial.available() > 0)
                        {
                                balanceReadOut = mySerial.read();
                                if (balanceReadOut == '\r')
```

```
                        {
                                Serial.print(balString);
                                Serial.print('\n');
                                balString = "";
                                continue_ = true;
                        }
                        else
                        {
                                balString += char(balanceReadOut);
                        }
                }
        }
    }
    //———————————Resetting experiment for next run———————————
    delay(5000);
    myservo.write(10);
    delay(5000);
    myservo.write(zero_angle);
    delay(1800000); // wait 30 minutes
    for (int t_ = 0; t_ <= 40; t_ += 1) //send the Tare command multiple times
    {
        mySerial.print("T");
        mySerial.print('\r');
        delay(2000);
    }
}
```