The improvement of the design and control of a soft robotic phantom that simulates the respiratory motion in the human liver

C.R. (Charlotte) Overwijk

BSc Report

Committee:
Dr.ir. M. Abayazid
H. Naghibi Beidokhti, MSc
Prof.dr.ir. G.J.M. Krijnen
Dr. E.H.F. van Asseldonk

August 2018

026RAM2018
Robotics and Mechatronics
EE-Math-CS
University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands

Contents

Abstract

The focus of this thesis is the improvement of the design and control of a soft robotic phantom that simulates the respiratory motion in the human liver. The robotic phantom is meant to be used for testing and perfecting the needle insertion procedures. In the first section of this paper background information is given, about the movement of the liver during the respiratory cycle. An MR-compatible simulator was previously developed in RaM which needed to be improved. This results in the research question and the aim of this report. The research question is: how can the current phantom be improved to represent the liver more realistically during the respiratory motion?

The aim of this study was to improve the soft robotic phantom to be a realist representation of the patient's liver and to better simulate respiratory motion of the liver. A control system needs to be developed for the simulator, enabling different respiratory motion simulation.

The improvement in the design has been achieved by adding the liver, a skin, a diaphragm and new actuators. While the improvement of the movement and control has been achieved by using an electronic pressure regulator and controlling the pressure with programming an arduino unit. A new actuator was utilized to achieve more realistic respiratory displacement. With the improvements made in this study, a more realistic respiratory motion with a better representation of patient's liver is achieved

1 Introduction

More than 700.000 people worldwide are diagnosed with some form of liver cancer every year. It is a leading cause in cancer deaths worldwide as well and it accounts for more than 600.000 deaths each year[1]. When the doctor wants to investigate a liver tumor an MRI scan can be very helpful. Sometimes they can tell a benign tumor from a malignant one by just looking at the MRI scan.[2]. If this is not the case then it would be useful to be able to take a biopsy while the patient is still in the MRI since the MRI can help finding and targeting the tumor.
One of the biggest hurdles with this idea is that the biopsy needle has to be inserted while being in the MRI and while the liver is moving due to respiration. Developing a robotic system that simulates the respiratory motion in the human liver can help testing and improving the surgical robots for a more accurate functioning, it can replace animal testing and it can help to train clinicians. Since this has to happen within an MRI system it has to be MRI-compatible.

1.1 Liver movement

When breathing, the pressure in the lungs and the volume of the lungs change by the motion of the diaphragm and the ribs. When inhaling the diaphragm contracts and goes down. This makes the organs of the abdomen move in the inferior direction. At the same time the external intercostal muscles (the muscles between the ribs) contract and make the liver move in the anterior direction. This movement is indicated in Figure 1. The reference direction of the human body can be seen in Figure 2. According to research done on 12 subjects using MRI, the liver moves 9.0±3.5 mm in the anterior-posterior direction (AP) and 24.4±16.4 mm in the superior-inferior direction (SI) during one breathing cycle [3].

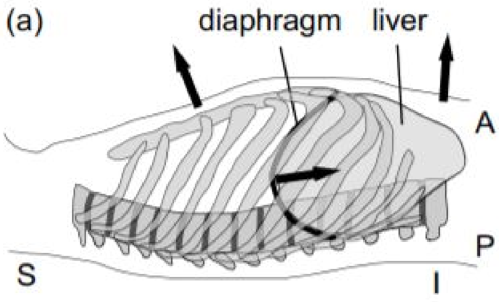

Figure 1: Maximum exhalation position, the arrow show the direction of the movement of the ribs and the diaphragm during inhalation[4].

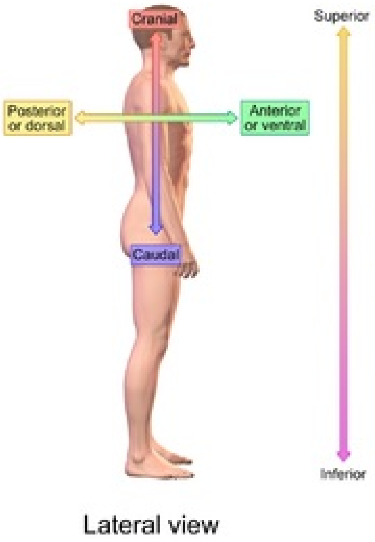

Figure 2: The directional references of the body [5].

To be able to represent the human displacement of the liver during the breathing cycle in the phantom, the actual displacement of the liver is analyzed first. In two previous researches data has been gathered on different patients[4], [6]. The measurement data from these researches are shown in Figures 3 and 4. Figure 3 shows the movement of the liver and the kidney during breathing and has a breathing cycle that takes around 3.5 seconds and the displacement is between 11 and 13 mm, this data will be further referred to as dataset 1. Figure 4 shows the superior-inferior motion of the diaphragm in two subjects. Figure 4 a) has a breathing cycle of 5.4 seconds and the displacement is between 15 and 17 mm Figure 4 b) has a very irregular breathing pattern,the mean breathing time is 3.4 seconds and the displacement is between 7 and 15 mm, these will be further referred to as dataset 2a and 2b. These breathing times and displacements will be used later in the report for the analysis.

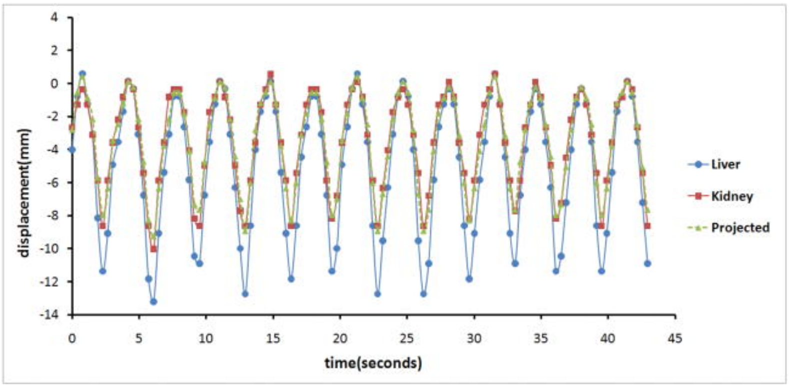

Figure 3: Movement patterns of the liver (blue line) and kidney (red line) along with the projected displacement (green line) of the kidney based on the liver motion in a healthy human during normal breathing[6].

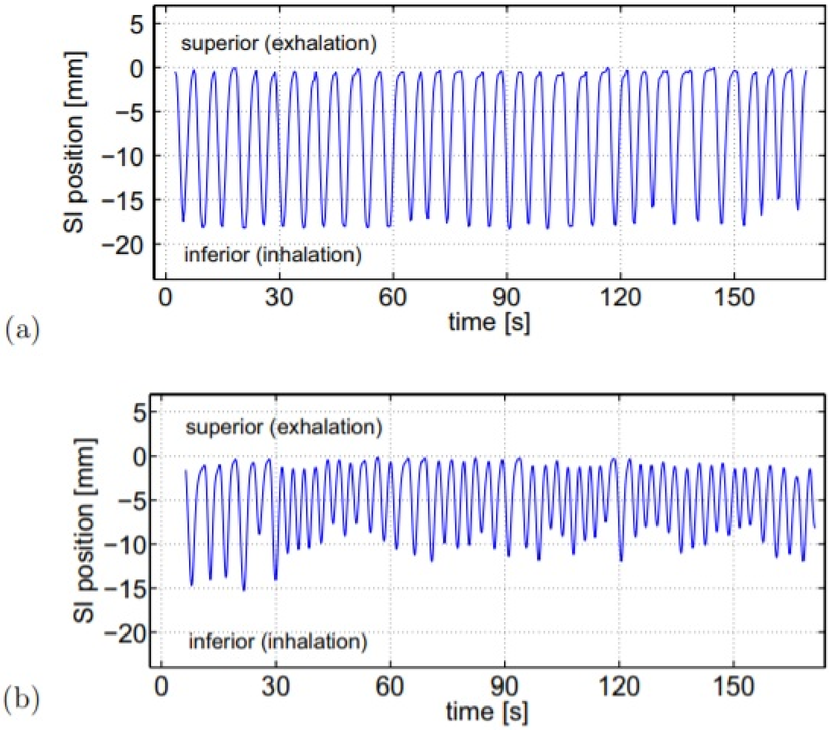

Figure 4: Superior-inferior motion of the diaphragm in two subjects showing large differences in the regularity of breathing. (a) Example of regular breathing. (b) Example of irregular breathing.[4]

1.2 MRI compatibility

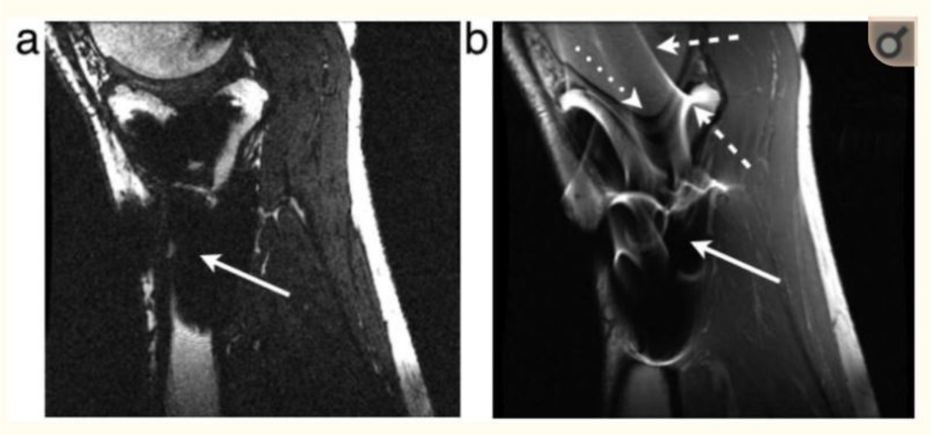

Figure 5: Examples of artifacts in the knee due to the presence of stainless steel screws in (a) gradient echo image with ±62.5 kHz receive bandwith and (b) spin echo image with ±16 kHz receive bandwidth. Solid arrows show signal loss. The dotted arrow shows geometric distortion. Dashed arrows show signal pile-up.[10]

According to the Magnetic Resonance Safety Testing Services, a device shall be considered "MR compatible" if it is MR safe and the device, when used in the MRI environment, has been demonstrated to neither significantly affect the quality of the diagnostic information nor have its operations affected by the MR system. A system is MR safe when the device, when used in the MRI environment, has been demonstrated to present no additional risk to the patient or other individual, but may affect the quality of the diagnostic information [9]. Since MRI works with magnetic fields only a few parts of metal or alloys can be used, since this already results in artifacts in the MRI image. Metals have high intrinsic magnetic susceptibilities, a high susceptibility indicates that a material is attracted into a magnetic field. This causes significant local field disturbances in the resonance frequency. Due to the altering resonance frequencies the image pixels shift away from their true positions leading to significant geometric distortions including signal voids (black areas) and signal pile-ups (bright areas) as can be seen in Figure 5.

Therefore there can't be any electrical parts in the phantom. As a result pneumatic actuators are used in the current phantom.

1.3 Previous works

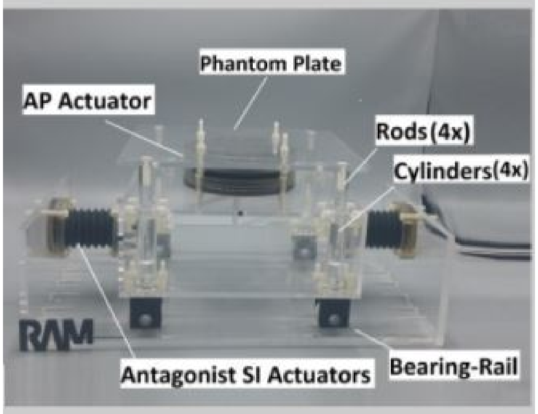

Figure 6: The current phantom. AP: Anterior-Posterior, SI: Superior-Inferior.

Previously, only a few dynamic simulators for liver respiratory motions have been developed. One of them used two artificial lungs to push a phantom in the SI direction, this phantom was for the CT scanner[11]. There was an ultrasound compatible liver phantom as well, this phantom uses a servomotor actuated simulator to replicate the liver respiratory motion[12]. Both designs could only simulate the respiratory motion in superior-inferior directions, whereas, the liver movement is both in superior-inferior and anterior-posterior directions. The phantom that is developed in the Robotics and Mechatronics Lab Utwente and that is MRI compatible[13] and can move in both the superion-inferior an anterior-posterior direction. As shown in Figure 6 the phantom system has 6 different parts:

1. An AP actuator
2. A phantom plate
3. Two antagonist SI actuators
4. A bearing rail
5. Four cylinders
6. Four rods

The AP actuator is a soft pneumatic actuator which is meant to provide anterior posterior motion. The phantom plate is a plate where the liver phantom will be positioned on. For the superior and inferior motion two antagonist pneumatic soft actuators are used as well as a bearing rail for guidance. The four vertical cylinders and rods are used as constraining for the perturbation in the transverse plain of the AP actuator. In the current set-up, manual analogue regulators were used to adjust the feeding air pressure. The pressure for each actuator, therefore, needed to be estimated prior to the test, and during the simulation the pressure was kept unchanged. In the current state of the phantom, there were improvements required, mainly the following:

1. The analogue regulators can be replaced with digital controllable regulators so the feeding air pressure is easier to control.
2. The feeding air pressure can be adjusted during the simulation, to have smoother respiratory motion.
3. The phantom could be improved to be a more realistic representation of human liver in terms of incorporating more tissues which can influence the simulation and in terms of the geometrical and elastic properties of the liver.

1.4 Research question and aim

This thesis is focused on the following research question: How can the current phantom be improved to represent the liver more realistically during the respiratory motion?
This is achievable on many levels so the aim is to stick to the following subsections.

The design should be restricted like it is in the human body (diaphragm / skin).

There should be a phantom that represents the liver.

The movement should be more realistic in respect of the displacement and the velocity.

A control system needs to be developed for the simulator enabling different respiratory motion simulation.

In the next sections of this report, The liver model and the improvements made to the design are described first in section 2. The motion control is discussed in section 3. After this the experimental setup will be further explained in section 4. Section 5 describes the results, followed by the discussion in section 5 and 6,respectively. which is the last chapter before the results, discussion and conclusion.

2 Phantom design

2.1 Liver

2.1.1 Shape and geometry

The liver is the largest and most massive internal organ of the human body. The mass of a healthy human liver is 1.2–1.5 kg. By ultrasound, a normal liver is less than 16 cm in the midclavicular line[7]. These are two requirements for the liver phantom since it has to represent a realistic liver.

2.1.2 Mechanical properties

The density of a human liver is around 1.05 g/cm^3[8]. To represent the liver a gelatin liver was made. The liver is made of gelatin since gelatin is a substance that can be made with comparable density to human soft tissue. When using 1178 ml of water and 118 grams of gelatin powder, the density of the gelatin will be ($\rho = mass/volume$=1256/1178) 1.07 g/cm^3.

2.1.3 Fabrication

To fabricate the liver phantom, first a liver mold needs to be designed. A liver model was already available within Solidworks. A mold has been made around this model and exists out of three parts so it's easy to get the liver out of the mold. See Figure 7 for the design of the liver mold and Figure 8 for a picture of the resulting gelatin liver. There are two holes within the design, one for the filling and one for the air to get out. A detailed description of how the liver was made is given in Appendix A.

The gelatin liver will have a "tumor" in it, this is made from PVC. An electromagnetic sensor (EM-sensor) is attached to the tumor, more information about this sensor will be later in the report. This is necessary for precisely measuring the tumor motion for later analysis.

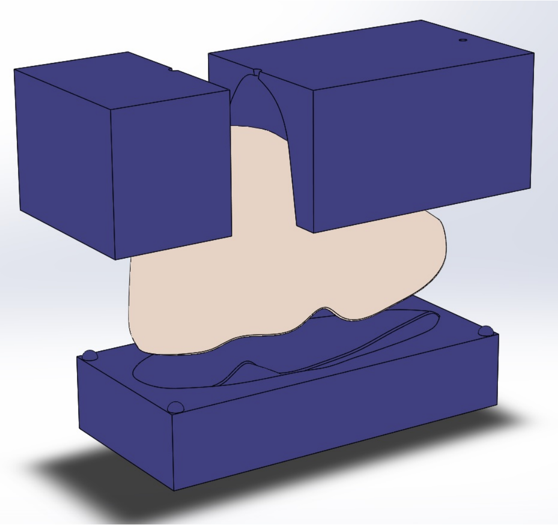

Figure 7: The liver mold.

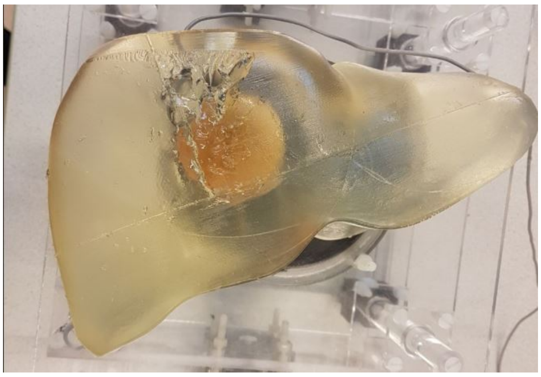

Figure 8: A gelatin liver with a PVC-tumor

2.2 The skin and diaphragm

In order to make the phantom more realistic a skin and a diaphragm have to be added. These are surrounding tissues of the liver and they both contribute to restrictions when breathing. They are both simulated using latex. The diaphragm has resistance in the superior-inferior direction and the skin has resistance in both the superior-inferior and anterior-posterior direction. The diaphragm can be seen in Figure 9 and the phantom covered in skin can be seen in Figure 10, the red dots are skin markers. Adding the skin and diaphragm can also offer the features required for the studies that estimate the motion of tumor based on the motion of diaphragm and skin markers, the skin markers are used in a different research. The latex thickness is 0.38 mm this is approximately the thickness of the skin over the abdomen (0.4 mm - 0.5 mm) but it's not comparable with the thickness of the diaphragm (10 mm - 20 mm).

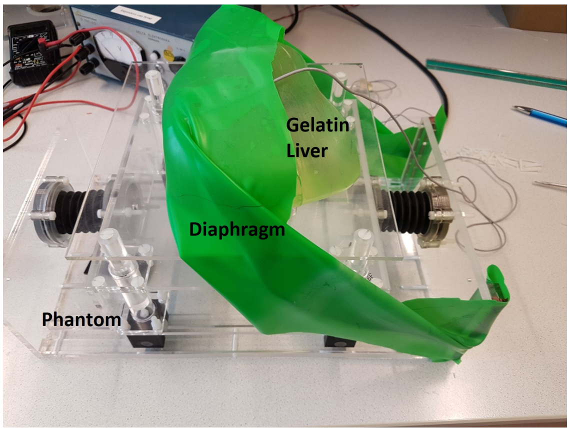

Figure 9: The liver with diaphragm.

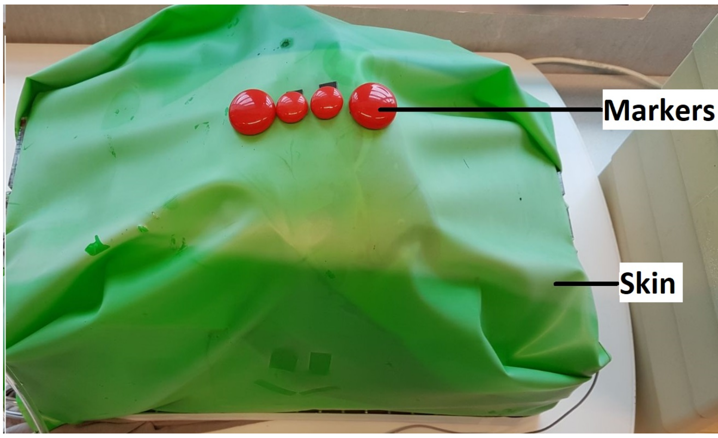

Figure 10: The phantom with skin.

2.3 Actuator

During this research three different actuator materials and two different designs have been used. There is the previous actuator, this is an actuator that's fabricated using a high spatial resolution rapid manufacturing 3D printer (Objet260, Stratasys, MN, USA). Furthermore there are the new actuators, which are molded instead of printed. Moreover, the two other actuators were molded with two softer materials (Ecoflex 10 and Ecoflex 50). See Figure 11 for the different actuators. The difference between Ecoflex 10 and 50 is the elasticity of the material. Ecoflex 50 is more rigid than Ecoflex 10. Since the actuators need an extra contradiction force, the phantom is tilted 15 degrees so the actuators will be compressed more.

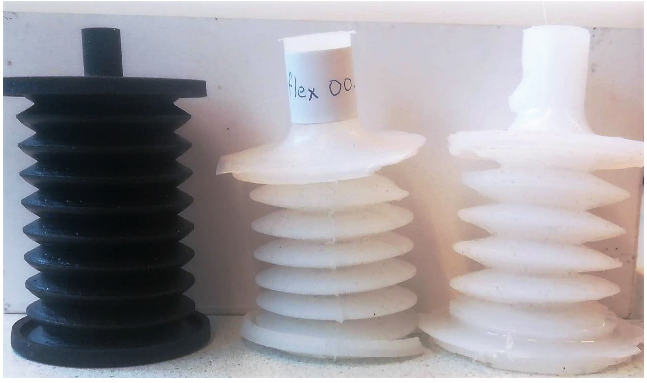

Figure 11: The three different actuators. On the left is the 3D printed actuator, in the middle is the Ecoflex 10 actuator and on the right is the Ecoflex 50 actuator

3 Motion control

The desired output of the phantom is the displacement. There is only one pressure regulator and the superior-inferior and the anterior-posterior movement can't be controlled by the same regulator, so the focus is on the superior-inferior movement. To reach a certain displacement the pneumatic actuators has to fill with air, this is achieved by controlling the air pressure with voltage.
The control of the actuator is implemented through an automatic script based on cases and for-loops. Case 1 is 'breathing in', this is a for-loop which increases the voltage gradually. Case 2 is 'breathing out', where the for loop decreases the voltage gradually. Within these for-loops there are two kinds of delay. The first delay is the 'step' delay, this delay is very small and not noticeable. The step delay will be referred to as breathing delay. The second delay is the delay when breathing in is done and the breath is being hold for a few seconds, this will be further referred to as holding delay . Both delays can be seen as an example in Figure 12. To determine the value of both the delays you take the breathing cycle time of a dataset(minus the two 'holding delays') and then divide it by two (so you have the breath in/ breath out time). After the first division it has to be divided by the volt increments. (The volt increments will be explained in the next part.) The output signal is a voltage.
The script starts with an on/off button which will determine if the phantom will go into potmeter mode (where the phantom can be controlled using a potentiometer) or an automatic script. The script can be seen in Appendix B. The potmeter script is based on a threshold value. If the potmetervalue is higher than the threshold, then the input voltage of the pressure regulator will increase. If the potmetervalue is lower than the threshold, then the input voltage of the pressure regulator will decrease.
The voltage is PWM driven, to make the PWM signal continuous a capacitor and a resistor have been added, more about this can be seen in Appendix D. The PWM signal is made by an arduino UNO and is controlled by a script. Figure 13 shows how the arduino and every part is connected, the point where the four colored wires connect is the 'pressure regulator'.

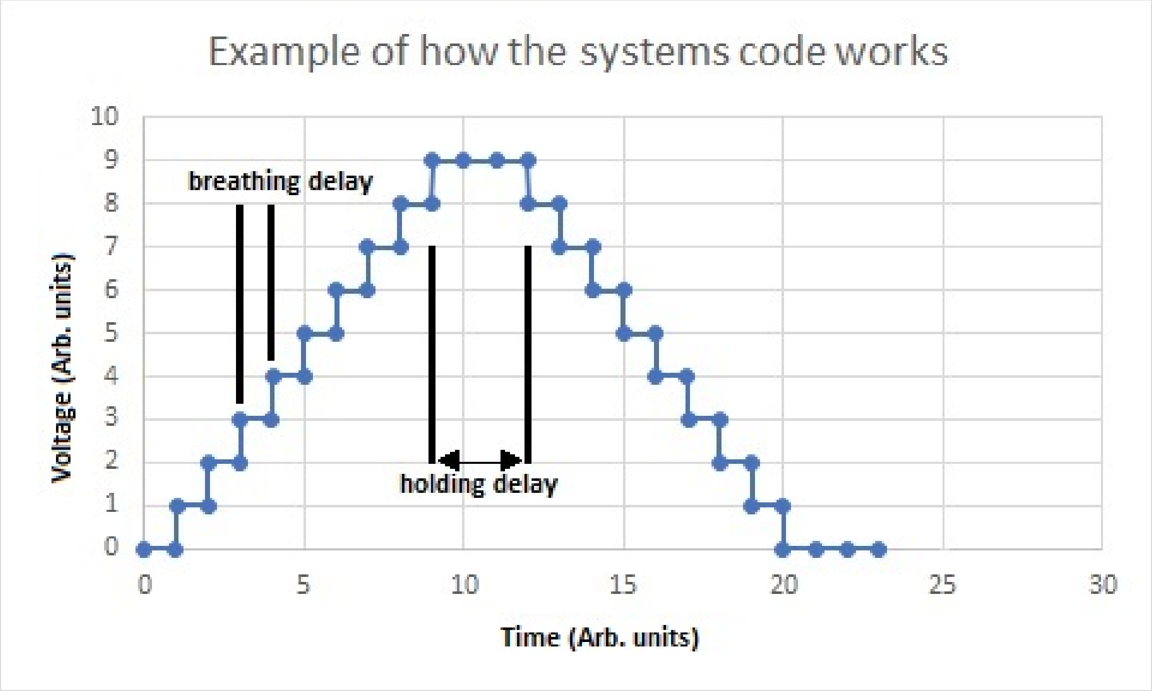

Figure 12: An example of how the delays work

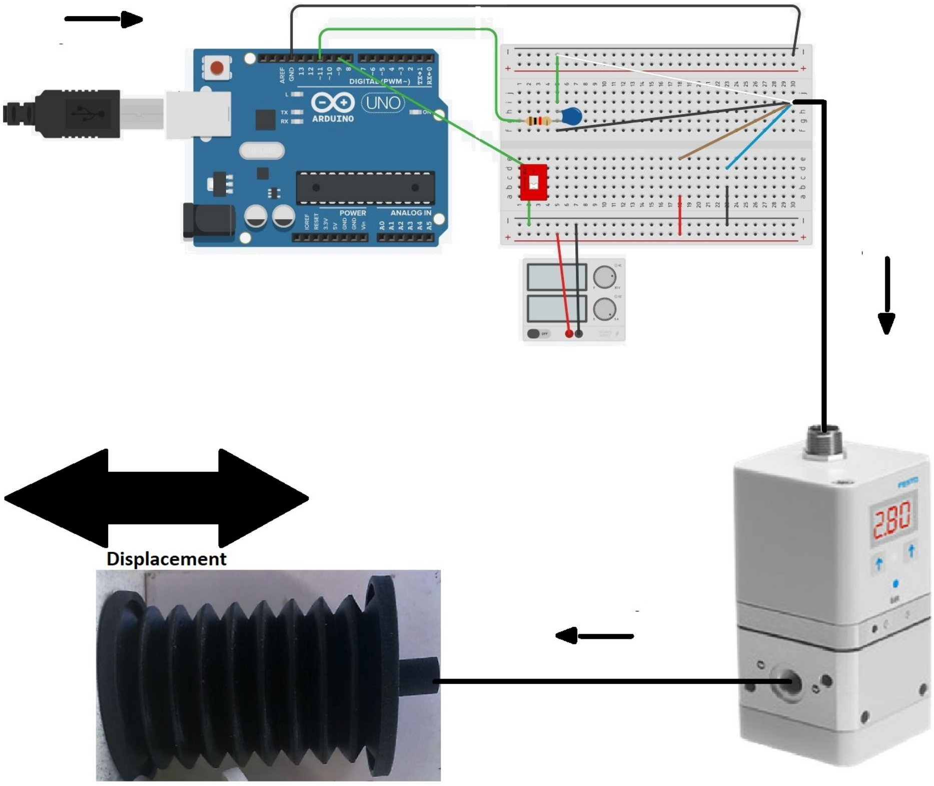

Figure 13: The set-up of the system

4 Experiments

4.1 Materials

4.1.1 The pressure regulator

For controlling the liver phantom originally a manual pressure regulator was used. This was done so the actuator never got more pressure than it could handle. But when it was replaced by a digital one, an extra function is added. This function is that the movement can be more smooth since the pressure can be increased gradually. Therefore was the manual regulator replaced with a digitally controlled regulator(festo, vppe-3-1-1/8-2-010-e1.)
The input pressure of the actuators can be controlled by changing the voltage of the digital regulator.The input voltage of the regulator works in a range of 0-10 V. Setpoint signals that have a value less than 0.1 V will be seen as 0 V. Within this range the air pressure can change from 0 bar to 2 bar. So with a change of 0.1 V the pressure will change 0.02 bar.(This has been tested and will be discussed in section 5.1.2) The arduino is able to produce 5 V. This happens in 255 increments, meaning 0.02 V at each increment. Since 1 bar is more then enough for the actuators, there is no need for adding an amplifier.

4.1.2 EM-sensor

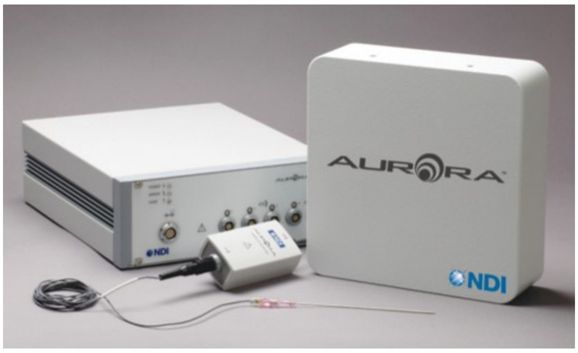

Figure 14: The aurora 6 DOF system.

The EM-sensor that's used is the Aurora 6DOF Reference, 25 mm Disc, this system can be seen in Figure 14. Electromagnetic spatial measurement systems determine the location of objects that are attached to sensor coils. Voltages are induced when the object is placed inside a controlled, altering magnetic field. These voltages are used by the measurement system to calculate the position and orientation of the object[16]. See Appendix A for a manual of the EM-sensor.

4.1.3 The solenoid

In the original phantom a compatible pneumatics PV3211-24VDC-1/8 solenoid was used. This solenoid switches the pressure from the superior-inferior actuators and is only able to switch between the valves when the pressure is higher then 1.5 bar. The actuators start to break around 0.2 bar. This wasn't an ideal situation since there is only one pressure regulator and that would be in front of the solenoid. So the decision was made to switch to the Festo MHE2-M1H-3/2G-QS-4. This solenoid starts to work at -0.9 bar. Since this solenoid works with 4 mm tubes and the rest of the phantom with 6 mm tubes a few coupling pieces were needed. In the beginning of the research the solenoid was used to switch between the two actuators in the superior-inferior direction and to turn the anterior-posterior direction on and off. In the current design, since there is only one actuator left and the pressure regulator can be set to zero as well, this isn't necessary anymore. For the AP direction without a second digital pressure regulator the solenoid is still useful, but if there is a second regulator for this then all the solenoids can be removed and everything can be code controlled.

4.2 Data processing

For gathering and processing the data Matlab 2018a was used. The latest version of Matlab was needed since a function is used that's only available in 2018a (smoothdata). Four scripts are being used, two for starting and gathering the data: $Start - NDI - Session, NDI - read - igtl$ and two processing scripts: ProcessEMstep and ProcessEMsmooth. All the scripts can be found in Appendix C.

4.3 Evaluation experiment

An effective way to analyze the liver phantom is to see how much it moves in the superior- inferior direction. This is since there is only one digital pressure regulator and the superior-inferior movement is more important than the anterior-posterior movement. The motion of the phantom can be tracked by using a digital camera that films the movements a marker attached to the phantom or by using the EM-sensor that's attached to the tumor and tracks the movement of the tumor. Since the digital camera will follow the movement of the "skin" that covers the liver, the EM-sensor is the better choice. To see if the results are representative they will be compared to Figure 3 (Dataset 1) and Figure 4 (Dataset 2). There will be 5 different experiments. The first experiment will be with the binary code and the 3D printed actuator. The second experiment is with the automatic script and the 3D printed actuator, the third is with the automatic script and an Ecoflex 10 actuator. The fourth and fifth are both done with the automatic script and Ecoflex 50, but one is based on dataset 1 and the other is based on dataset 2. The values in Table 1 are the values used within the automatic script.

Table 1: An overview of the experiments and the used values

Code	**Holding delay(ms)**	**Breathing delay(ms)**	**Max pressure(bar)**
Binary code, 3D printed actuator	-	-	0.19
Automatic script, 3D printed actuator	140	35	0.19
Automatic script, Ecoflex 10	100	73	0.9
Automatic script, Ecoflex 50 dataset 1	100	50	0.10
Automatic script, Ecoflex 50 dataset 2a	150	73	0.10

5 Results

A few typical results of the measurements are shown in this section. There are 3 different actuators being used and 2 different motion control methods. All results have been reproduced and the extra figures are shown in Appendix E.

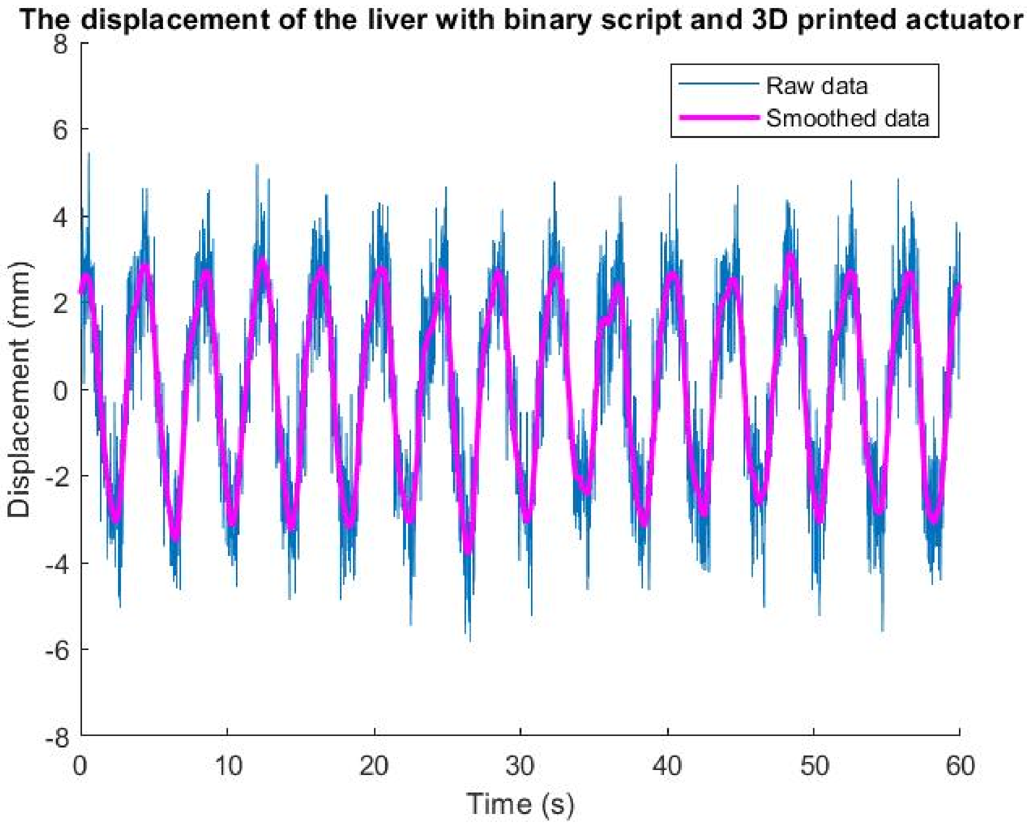

Figure 15: Measurements of the displacement in the tumor measured with the binary code and 3D printed actuator

When using the binary code, which can be seen in Appendix B, the displacement of the tumor over time is periodically and the maximum displacement of the tumor is not constant while in the code it was constant as can be seen in Figure 15. The maximum displacement is between 4 and 6 mm and a breathing cycle takes around 4 seconds. There appears to be around 2 mm of noise on the data points.

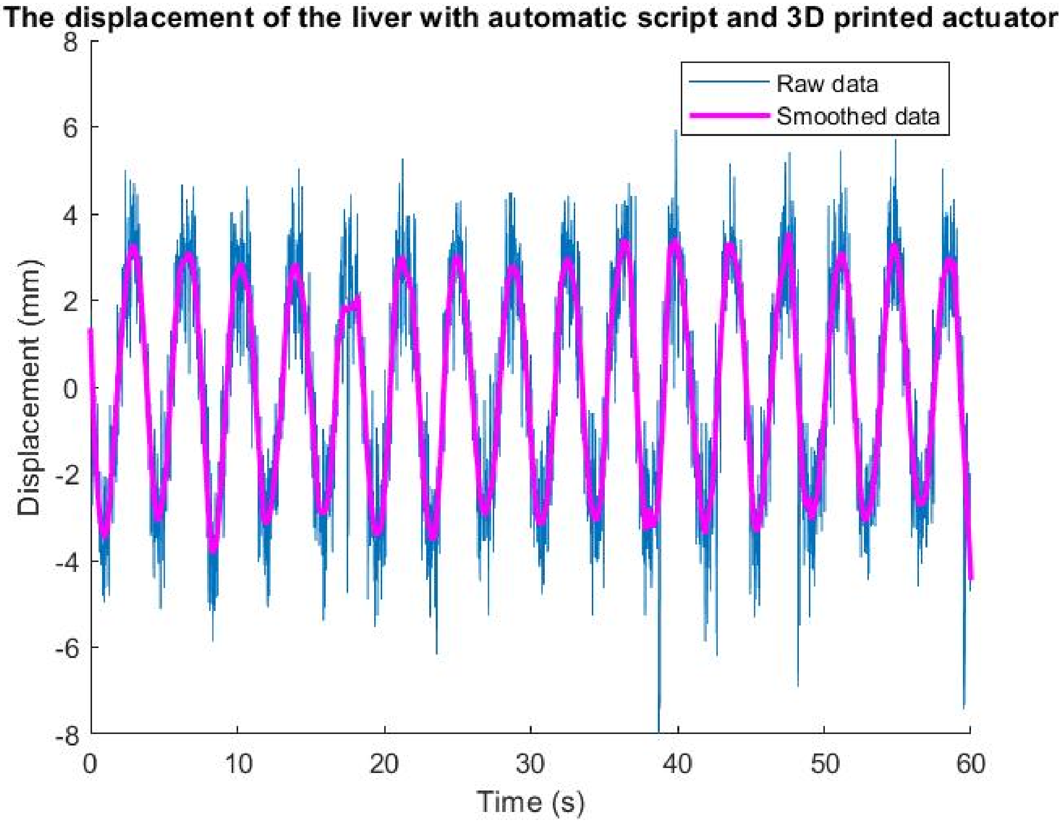

Figure 16: Measurements of the displacement in the tumor measured with the automatic script and 3D printed actuator

Figure 16 shows the measuring data of the automatic script and the 3D printed actuator. The maximum displacement is again not constant but the signal is periodically. The maximum displacement is between 5 and 7 mm and a breathing cycle takes around 3.3 seconds.

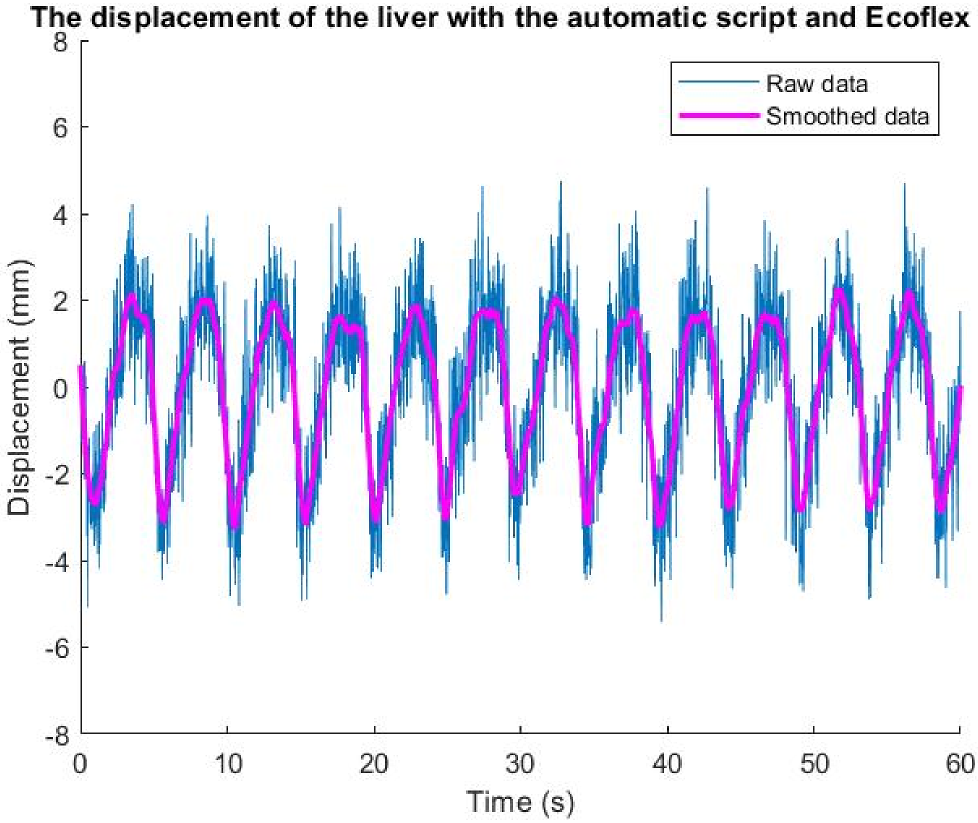

Figure 17: Measurements of the displacement in the tumor measured with with the automatic script and Ecoflex 10 actuator

In Figure 17 the automatic script has been used with the Ecoflex 10 actuator. The maximum displacement is around 4 mm and a breathing cycle takes around 5 seconds.

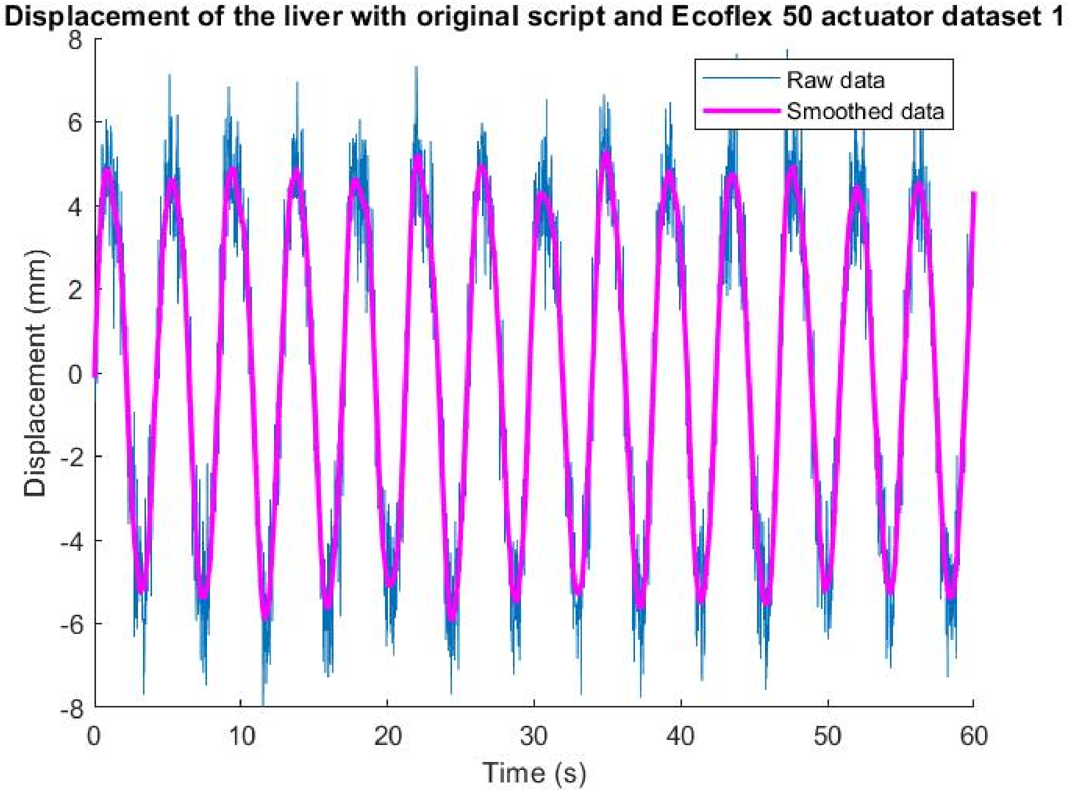

Figure 18: Measurements of the displacement in the tumor measured with the automatic script and Ecoflex 50 actuator dataset 1

Figure 18 has been made with the automatic script and the Ecoflex 50 actuator. The signal is periodically and has a constant displacement. The maximum displacement is between 1 and 1.2 cm and has a breathing cycle of around 4 seconds. This result has been controlled such that it should be compared with Figure 3.

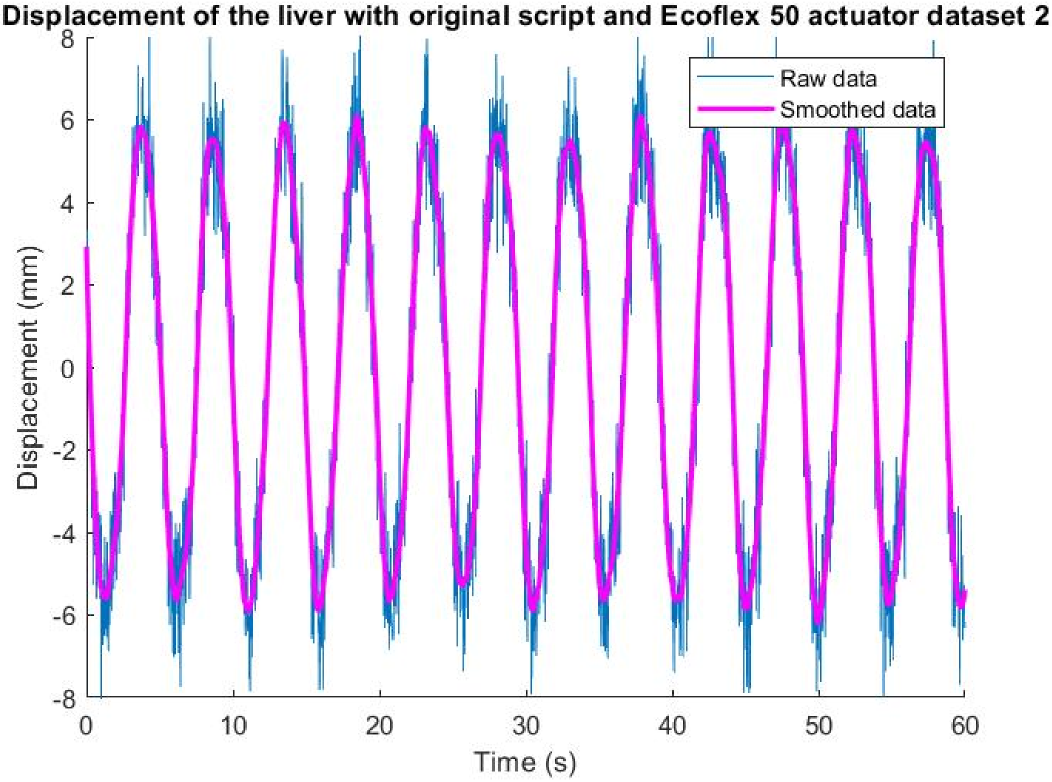

Figure 19: Measurements of the displacement in the tumor measured with the binary script and the Ecoflex 50 actuator using dataset 2

To have a larger displacement and a more consistent output the Ecoflex 10 actuator was replaced by an Ecoflex 50 actuator. The signal is periodically and almost has a constant maximum displacement. The maximum displacement is between 1.0 and 1.3 cm and has a breathing cycle of around 5 seconds as can be seen in Figure 19. This result has been controlled such that it should be compared with Figure 4 for the first four experiments.

Table 2: An overview of different displacements and breathing cycles

Code	Displacement (mm)	Breathing cycle (s)
Binary code, 3D printed actuator	4-6	4
Automatic script, 3D printed actuator	5-7	3.5
Automatic script, Ecoflex 10	4	5
Automatic script, Ecoflex 50 dataset 1	10-12	4
Automatic script, Ecoflex 50 dataset 2a	12-13	5
Dataset 1	11-13	3.5
Dataset 2a	15-17	5.4

Table 2 shows an overview of different displacements and breathing cycles.

6 Discussion

6.1 Design

When comparing the previous design of the phantom with the new design of the phantom there are a lot of improvements but there are things that can still be improved.

The gelatin liver is a good representation of the human liver. The density is almost the same, 1.05 g/cm^3 for the human liver and 1.07 g/cm^3 for the gelatin liver and the clavicular length of the gelatin liver was 12.5 cm while the human liver has a maximum length of 16 cm.

Both the diaphragm and the skin are represented by using latex. The latex has been chosen for the thickness of the skin and not for the characteristics of the diaphragm. The latex is not thick enough to really represent the diaphragm, it would be recommended to look into this more.

During the measurement the phantom is tilted, the reason for this is that the actuator has a more compressed starting positioin which results in a bigger displacement. When comparing the phantom with a real patient, it would result in the patient laying in the MRI under an angle, but the common position in the MRI is horizontal. The phantom can be horizontal when the actuators have are improved and have a realistic enough displacement horizontally.

The 3D printed actuator was very brittle and there was an air leakage. The Ecoflex actuator was not brittle and no air leakage and able to achieve a larger displacement. The Ecoflex actuators need improvements since it was observed that the actuators are both buckling and ballooning and this results in a smaller displacement. Buckling is bending as a result of force/weakness in our case and ballooning is the excessive radial expansion in stead of horizontal expansion, but it either needs a more rigid material or ringlets. Ecoflex 50 is already better than Ecoflex 10 since it's a more rigid material and therefore revealed less buckling and ballooning. The buckling and ballooning results in the difference of maximum displacements in the results, since it's not constant the same rate of buckling and ballooning.

6.2 Control

When looking at these results it can be observed that the 3D printed actuator wasn't able to achieve a big enough displacement to be realistic. The automatic script, Ecoflex 50 dataset 1 and 2a both have realistic values. They are comparable to dataset 1 and 2. The breathing cycle could be more precise, but now the breathing cycle of the datasets has been read out of a figure. The breathing cycle could be more precise if datapoints were analyzed. Dataset 2 was based on the movement of the diaphragm, in this research it is assumed that it is similar to the movement of the liver.

Even though the breathing cycle isn't exactly the same as the datasets, it's still good result that does represent the motion of the human liver during respiration.

When looking at the results it shows that every respiratory curve has a noise of around 2 mm, this is due to the Aurora NDI system since it has a accuracy of 2 mm with a frequency of 40 Hz. This results in the high frequency noise. Another possibility is that some of the noise is caused by the vibration of the gelatin liver, this won't be high frequency, but it will contribute to the noise.

At this moment only a repetitive code has been used. The irregular part of the code has been silenced since the regular code needs validation first. Since the regular part of the code can be compared with human breathing, it's possible to add a small randomness to the delays later. This results in a non predictable breathing pattern, but still within the ranges of normal breathing. The recommendation for the breathing delay for the Ecoflex 50 with the automatic script would be 50-73 ms and for the holding delay it would be 100-150 ms. This would make the breathing more realistic which would help surgeons and surgical robots to learn how to react on small changes within breathing.

For further research it would be recommended to start with changing the script from open-looped to close-looped. This will ensure that the phantom always has a certain displacement.This can be achieved with the feedback that the pressure regulator provides. This is only an option when there is a certain and clear correlation between the air pressure and the displacement. This is not possible yet since the the behavior of the actuators varies due to different buckling and ballooning possibilities.

7 Conclusion

When starting this thesis there was a research question and an aim in mind.
The research question was: how can the current phantom be improved to represent the liver more realistically during the respiratory motion?
And the aim was:

The design should be restricted like it is in the human body (diaphragm / skin).
There should be a phantom that represents the liver.
The movement should be more realistic in respect of the displacement and the velocity.
A control system needs to be developed for the simulator, enabling different respiratory motion simulation

Most of the aiming points have been achieved and are a direct answer to the research question. The design represents the physical environment more realistically. The gelatin liver almost has the same density and dimensions as a human liver. The density of the gelatin liver was 1.07 g/cm^3 while the density of a human liver is $1.05 g/cm^3$, the mass of the gelatin liver is 1256 g, this is within the range of a human liver (1200-1500 g). The midclavicular length of the gelatin liver is 12.5 cm while the human liver has a maximum of 16 cm. The latex skin has approximately the same thickness as the skin over the abdomen, the latex is 0.38 mm while the skin over the abdomen is 0.4-0.5 mm. A latex diaphragm has been added but there is room for improvement here since the thickness isn't comparable to the thickness of the human diaphragm.

The movement is more realistic, it is a smoother movement with better control and a system has been developed for simulating the different respiratory motions. The 3D printed actuator has been replaced by an Ecoflex actuator. The Ecoflex 50 actuator has better results than the Ecoflex 10 actuator since it's more rigid and has less buckling and ballooning. The results of the Ecoflex 50 actuator with the automatic script code are realistic and can be compared with the liver displacement during the respiration.
The phantom still has the potential to become even better, but it's a good start and the current phantom is a realistic representation of the respiratory motion in the human liver.

References

[1] 'Key Statistics About Liver Cancer', American Cancer Society, $www.cancer.org/cancer/liver-cancer/about/what-is-key-statistics.html$

[2] 'Tests for Liver Cancer', American Cancer Society, $www.cancer.org/cancer/liver-cancer/detection-diagnosis-staging/how-diagnosed.html$

[3] Bussels B, Goethals L, Feron M, et al, 'Respiration-induced movement of the upper abdominal organs: A pitfall for the three-dimensional conformal radiation treatment of pancreatic cancer.', Radiotherapy and Oncology, page 69–74, 2003, seen 24-04-2018, DOI: 10.1016/S0167-8140(03)00133-6

[4] Martin Von Siebenthal, 'Analysis and Modelling of Respiratory Liver Motion using 4DMRI', ETH Zurich, page 1-2, 2008, seen 24-04-2018, DOI:10.1.1.428.3001

[5] 'Anatomical terms of location', $www.en.wikipedia.org/wiki/Anatomical_terms_of_location$

[6] Ruitian Song, PhD, Aaryani Tipirneni, BS, Perry Johnson, MS, Ralf B. Loeffler, PhD, and Claudia M. Hillenbrand, PhD, 'Evaluation of respiratory liver and kidney movements for MRI navigator gating', J Magn Reson Imaging. 2011 Jan; 33(1): 143–148. seen 31-07-2018 doi: 10.1002/jmri.22418

[7] 'Hepatomegaly: Differential diagnosis and evaluation', $https://www.uptodate.com/contents/hepatomegaly-differential-diagnosis-and-evaluation$

[8] International commission on radiological protection, Adult reference computational phantoms, ICRP Publication 110, Ann. ICRP 39 (2), 2009 pp.48-51 Table A.1 seen 5-8-2018

[9] $http://www.magneticresonancesafetytesting.com/genpg.asp?pgname=Terminology$

[10] Brian Hargreaves, Pauline W. Worters, Kim Butts Pauly, John M. Pauly, Kevin M. Koch, and Garry E. Gold 'Metal Induced Artifacts in MRI' AJR Am J Roentgenol. 2011 Sep; 197(3): 547–555. doi: 10.2214/AJR.11.7364

[11] L. Maier-Hein et al., "Respiratory motion compensation for CT- guided interventions in the liver," Comput. Aided Surg., vol. 13, no. 3, pp. 125–138, 2008. Seen 5-8-2018

[12] C. Li, S. P. Ang, J. Liu, and H. Yu, "A Dynamic Liver Phantom for Ultrasound Image Guided Biopsy," in The 15th International Conference on Biomedical Engineering, 2014, vol. 43, pp. 152–155. Seen 5-8-2018

[13] Hamid Naghibi, Paulo A. C. Costa, Momen Abayazid, 'A soft robotic phantom to simulate the dynamic respiratory motion of human liver.

[14] $https://www.researchgate.net/figure/Basic-block-diagram-of-PID-controller_fig1_317686313$

[15] Arduino,'PWM', $https://www.arduino.cc/en/Tutorial/PWM$

[16] NDI, 'Aurora', $https://www.ndigital.com/medical/products/aurora/y$

8 Appendix

8.1 A: Manuals

8.1.1 Manual gelatin liver.

1. Measure 1200 milliliters of hot tapwater.
2. Measure 120 grams of gelatin powder.
3. Apply a layer of Mann Ease Release 200.
4. Put the gelatin powder into the water and let it soak until it dissolves.
5. Apply another layer of Mann Ease Release 200.
6. Put the 'tumor' in the mold and duct tape all the 'seams'.
7. Let the substance cool down until room temperature and fill the mold.
8. Put the mold with the substance in the fridge until it's solid.

8.1.2 Manual EM sensor

The setup consists of: a field generator, a power supply and a probe. Your computer has to have the NDI MAtlab-software.

1. Turn on the PC and NDI system.
2. Open Matlab and execute $start - NDI - session$.
3. A pop-up window will appear, push the launch server button.
4. Read the IPaddress in the workspace window.
5. Put the following command into the command window $[T, Q, EA, F, err] = ndi_read_igtl(IPaddress, 18944, 20,' Monitor');$, The IPaddress should be the IPaddress of step 4.
6. Push stop, when the measurement is done.
7. Save the workspace data.

8.2 B: Arduino scripts

8.2.1 Binary code

```
int solenoidPin = 13;                    //This is the output pin on the Arduino

void setup()
{
  pinMode(solenoidPin, OUTPUT);          //Sets that pin as an output
}

void loop()
{
  digitalWrite(solenoidPin, HIGH);      //Switch Solenoid ON
  delay(2500);                          //Wait 5 Second
  digitalWrite(solenoidPin, LOW);       //Switch Solenoid OFF
  delay(2500);                          //Wait 5 Second
}
```

8.2.2 Final code

```
int OutputPin = 11;
int PotPin = 2;
int button = 9;
double i;
double Voltage;
enum States {SelectInput, Auto, Potmeter, Rest};
States State;
enum States2{StepForward1, StepBackward1};
States2 State2;
long randomdelay1;
long randomdelay2;

  //Initiate Serial communication.
void setup() {
  Serial.begin(9600);      // open the serial port at 9600 bps:
  pinMode(OutputPin, OUTPUT);
}

void PotmeterControl()  {                    // Get position from Potmeter
    double PotmeterValue = analogRead(PotPin);

    if (PotmeterValue>0.6) {
        i+=2;
    }
    else if (PotmeterValue<0.5) {
        i-=2;
    }
    else {
    }
        i=Voltage;
        analogWrite(OutputPin,Voltage);
}
void loop() {
    State=SelectInput;
   Serial.print("State machine starts\n");
   switch(State){

      case SelectInput:
      {
          if (button==1)
         {
            State=Potmeter;
          }
          else                     //button==0
          {
           State=Auto;
          }
      }
        break;

      case Potmeter:
        Serial.print("The input is a potmeter\n");
        PotmeterControl();
        break;

      case Auto:
        Serial.print("The input is an automatic script\n");
        do{
         switch (State2)
    {
      State2=StepForward1;
      case StepForward1:
         randomdelay1=random(50, 73);
        randomdelay2=random(100, 150);
        for (int i = 0; i <= 77; i++) {
          analogWrite(OutputPin, i);
          delay(randomdelay1);} // the random delay can be replaced with a set value.
        delay(randomdelay2);
        State2 = StepBackward1;
        break;

      case StepBackward1:

        for (int a = 77; a >= 0; a--) {
          analogWrite(OutputPin, a);
          delay(randomdelay1);}// 5 V output--> 1 bar
        delay(randomdelay2);
        State2 = StepForward1;
        break;
          }
      }while (button ==1);
      State=Rest;
      break;

      case Rest:
        Serial.print("The phantom is in rest mode\n");
          do{
            i=i;
            i=Voltage;
            analogWrite(OutputPin,Voltage);
            } while (button ==0);
            State=Auto;
      break;
   }  }
```

8.3 C: Matlab scripts

8.3.1 Start NDI Session

start ndi session - F. van der Heijden

```
clear all
close all
clc

% addpath(’D:\NDI_Toolbox’)
igtlLaunchServer
IPaddress = SearchIP;
```

8.3.2 read data script

```
function [T, Q, EA, t, err] = ndi_read_igtl(host_ip,port,thresh,option)
% function [T, Q, EA, t, err] reads data from the NDI aurora system.
% Inputs:   ipadd:      The IP address of the computer that is running  e.g. ‘130.89.204.55’
%           port:       The port number of the Plusserver (it is 18944).
%           thresh:     The number of track points or number of seconds (depending on option).
%           option:     Set to ’Count’ to obtain thresh number of samples.
%                       Set to ’Second’ to obtain samples for thresh number of seconds.
% Set to ’Monitor’ to display real-time measurement values for thresh number of seconds.
% Outputs:  T:          The positions Tx, Ty, Tz stored in a 3 x N array.
%           Q:          The quaternions q0, qx, qy, qz stored in a 4 x N array.
%           EA:         The Euler Angles in zyx order stored in a 3 x N array.
%           t:          Relative time stamp of the measurements stored in a 1 x N array.
%           err:        The tracked data are only valid if err == 0.

    % Receiving position and orientation from an OpenIGTLink server
    % Connect to the OpenIGTLink server
        igtlConnection = igtlConnect(host_ip,port);

    % Init
    displayReceivedTransforms = 0;
    addNoise = 1;

    %Estimate matrix sizes
    if strcmp(option,’Count’)  || strcmp(option,’count’)
        N = thresh;
    elseif strcmp(option,’Second’) || strcmp(option,’second’)
        N = thresh*50; %Size unknown, so loop longer than roughly 40 samples per second per tool
elseif strcmp(option,’Monitor’) || strcmp(option,’monitor’)
        N = thresh*50; %Size unknown, so loop longer than roughly 40 samples per second per tool
    else
        error(’Unknown option’);
    end

    T = zeros(3,N);
    Q = zeros(4,N);
    EA = zeros(3,N);
    t = zeros(1,N);
    err = zeros(1,N);
    index = 1;
    startTime = -1;
    while true
        % Get a transform
        transform = igtlReceivePosition(igtlConnection);

        % Process the received transform
        if ~isempty(transform.t) && ~isempty(transform.o)
            % Init monitor
            if startTime < 0 && (strcmp(option,'Monitor')  || strcmp(option,'monitor'))
s = {'dummydummydummydummydummydummydummydummydummydummy','b','c'};
h = msgbox(s,'NDI EM tracker');
foo = get(h,'children');
                if ~isempty(get(foo(1),'Children'))
                    hoo = get(foo(1),'children');
                else
                    hoo = get(foo(2),'children');
                end
set(hoo,'fontname','courier new');
set(hoo,'fontsize',12)
            end

            % Compute relative timestamp (from the start of the function)
            if startTime < 0
                startTime = transform.timestamp;
                relativeTime = 0;
            else
                relativeTime = transform.timestamp - startTime;
            end

% Fill outputs
T(:,index) = transform.t;
Q(:,index) = transform.o;
EA(:,index)=quaternions2eulerangles([transform.o(2) transform.o(3) transform.o(4)transform.o(1)]);
t(index) = relativeTime;
  if sum([transform.t == [0; 0; 0]; transform.o == [1; 0; 0; 0]]) == 7
            err(index) = 1;
        else
            err(index) = 0;
        end

% Monitor
            if strcmp(option,'Monitor')  || strcmp(option,'monitor')
                S{1} = sprintf(' T = %#6.4g   %#6.4g   %#6.4g',T(:,index));
                S{2} = sprintf('EA = %#6.4g   %#6.4g   %#6.4g',EA(:,index));
                S{3} = sprintf(' t = %#6.4g    error = %1d',t(:,index),err(:,index));
                if ishandle(h)
                    set(hoo,'string',S);
                    if err(:,index)
                        set(hoo,'color',[1 0 0]);
                        beep;
                    else
                        set(hoo,'color',[0 0 0]);
                    end
                    drawnow;
                else
                    break; % Break while loop loop when Ok button is clicked
                end
            end

            % Print data
            if displayReceivedTransforms
```

```
              disp(['----------- ',transform.name,' (',num2str(relativeTime),') -----------']);
              disp('---Translation---');
              disp(transform.t);
              disp('---Orientation---');
              disp(transform.o);
            end

            if err(index); beep; end % Make sound on error

            % Stop loop
            if strcmp(option,'Count')  || strcmp(option,'count')
                if index >= thresh % Break while loop when count is reached
                    break;
                end
            else % Break while loop when time in seconds has passed
                if t(index) >= thresh
                    break;
                end
            end
            index = index + 1;
        else
            % No transforms are received, probably timeout
            disp('No transforms are available');
        end
    end
    % Cleanup trailing zeros
    T(:,index+1:end) = [];
    Q(:,index+1:end) = [];
    EA(:,index+1:end) = [];
    t(index+1:end) = [];
    err(index+1:end) = [];

    % Add noise to position (for didactical purposes)
    if addNoise
        noise = randn(size(T));
        T = T + noise;
    end

    % Disconnect from the OpenIGTLink server
    igtlDisconnect(igtlConnection);
end

function EA = quaternions2eulerangles(Q)
% quarternions2eulerangles: conversion of quarternions to Euler angles
% OBSOLETE: use quat2eul from the robotics toolbox
%
%   Function use:
%     EA = quarternions2eulerangles(q)
%
%   Function input:
%     q: vector of quaternions: 4x1 vector or 1x4 vector
%     (or 4xN if N orientations are to be transformed)
%     q(1) is the scalar
%
%   Function output:
%     EA: vector of Euler angles (in degrees, 3x1 or 3xN array) ZYX convention
%       EA(1) around z-axis
%       EA(2) around y-axis
%       EA(3) around x-axis
%
%   Algorithm from "Guidance and Control of Ocean Vehicles" by Thor I. Fossen
%  (page 15-16)


[n, m] = size(Q);
if n==1 && m==4,  Q=Q'; end
[n, ~] = size(Q);
if n~=4, error('input dimension invalid'); end
Q = circshift(Q,[-1 0]);    % Fossen used the [qx qy qz q0] convention
                            % Matlab and NDI uses [q0 qx qy qz]

E31 = 2*(Q(1,:).*Q(3,:)-Q(2,:).*Q(4,:));
E32 = 2*(Q(2,:).*Q(3,:)+Q(1,:).*Q(4,:));
E33 = 1 - 2*(Q(1,:).*Q(1,:)+Q(2,:).*Q(2,:));
E21 = 2*(Q(1,:).*Q(2,:)+Q(3,:).*Q(4,:));
E11 = 1 - 2*(Q(2,:).*Q(2,:)+Q(3,:).*Q(3,:));

theta = -asind(E31);
phi   = atan2(E32,E33)*180/pi;
psi   = atan2(E21,E11)*180/pi;

EA = [psi; theta; phi];
end
```

8.3.3 Processdata

```
clear all;close all;
load('yourdataname.mat')
Ty=T(2,:);
Tz=T(3,:);
a=0*pi/180
tumor_SI=Ty*cos(a)+Tz*sin(a); %correction for the angle the phantom was
A=mean(tumor_SI)
SDy=smoothdata(tumor_SI,'sgolay',60);
B=tumor_SI-A;
C=SDy-A;

figure
hold on
plot(t,B)
plot(t,C,'m','Linewidth',2)
axis([0 60 -8 8])
title('The displacement of the liver')
xlabel('Time (s)')
ylabel('Displacement (mm)')
hold off
```

8.4 D: PWM

8.4.1 PWM

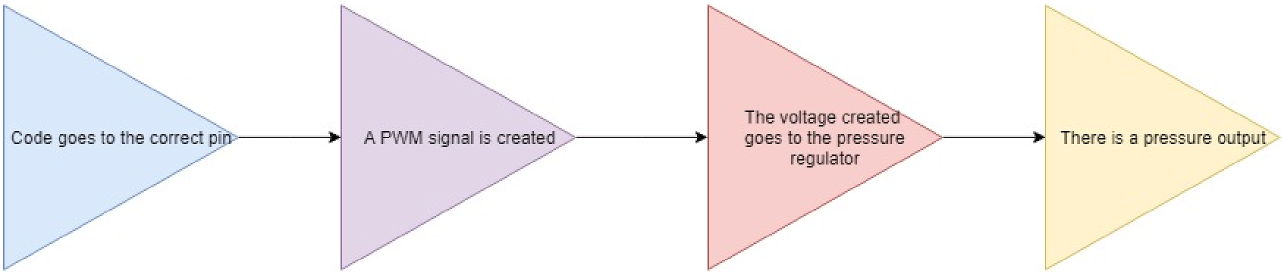
Code goes to the correct pin
A PWM signal is created
The voltage created goes to the pressure regulator
There is a pressure output

Figure 20: The control system of the pressure regulator

PWM is a technique for getting analog results with digital means. A square wave is created by using digital control, this is a signal which switches between on and off. Due to the on/off pattern and the frequency of this, there can be voltages between 5 Volts and 0 Volts. The duration of "on time" is called the pulse width. Since the switching happens very fast the result is as if the signal is a steady voltage between 0 and 5v. The frequency of the PWM signal on most pins is approximately 490 Hz. Since the voltage signal is a crucial step within the control of the pressure regulator as can be seen in Figure 20 this should be resolved. This is achievable by adding a resistor and a capacitor or just a capacitor[15].

8.4.2 Resistor and capacitor

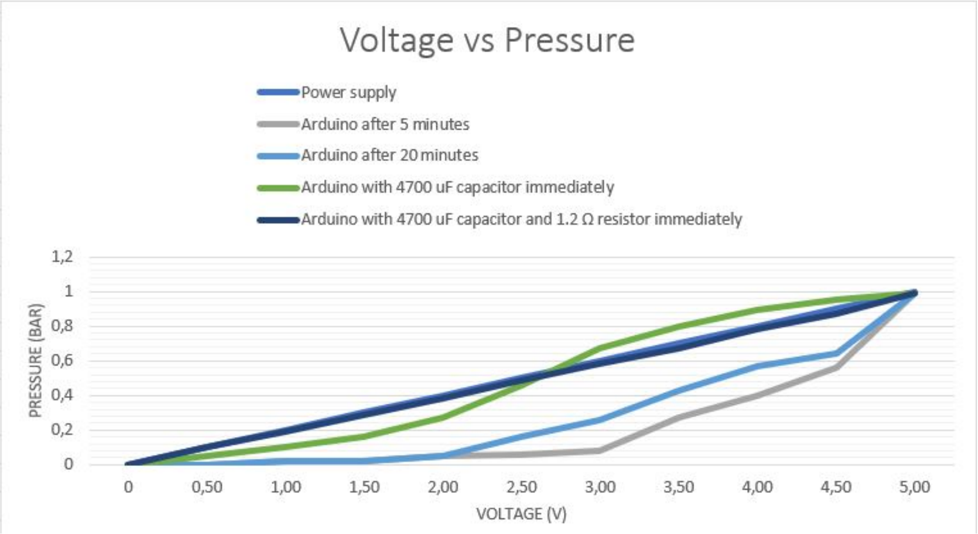

Figure 21: The voltage plotted versus the pressure.

When using a resistor and a capacitor an RC filter is made. The RC filter is a simple circuit used to convert a pulsed signal to a continuous signal. The capacitor shunts the varying voltage to ground, producing a constant DC voltage. The transfer function of an RC filter is:
$H_{RC}(w) = \frac{1}{RCw+1}$
It would be ideal for the transfer function to be equal to one, since the input voltage will be equal to the output voltage. When testing what setup worked best, there was a 4700μF capacitor used and a 1.0 Ω and 1.2 Ω resistor or no resistor. In Figure 21 the different voltages versus pressure can be seen under different circumstances. As can be seen the arduino with the 4700 μF although already better still does not show the required linear dependency between voltage and pressure. When a resistor of 1.2 Ω is added, it does have a near linear dependency between voltage and

pressure. When using the formula for the cutoff-frequency it shows that this value (177.3 Hz) is indeed underneath the 490 Hz for the PWM and still well above the frequency corresponding to the breathing delay of around 20 Hz, required for the experiments.
$\omega_c = \frac{1}{RC} = \frac{1}{1.2*4700*10^{-6}} = 177,3Hz$

8.5 E: Extra Data

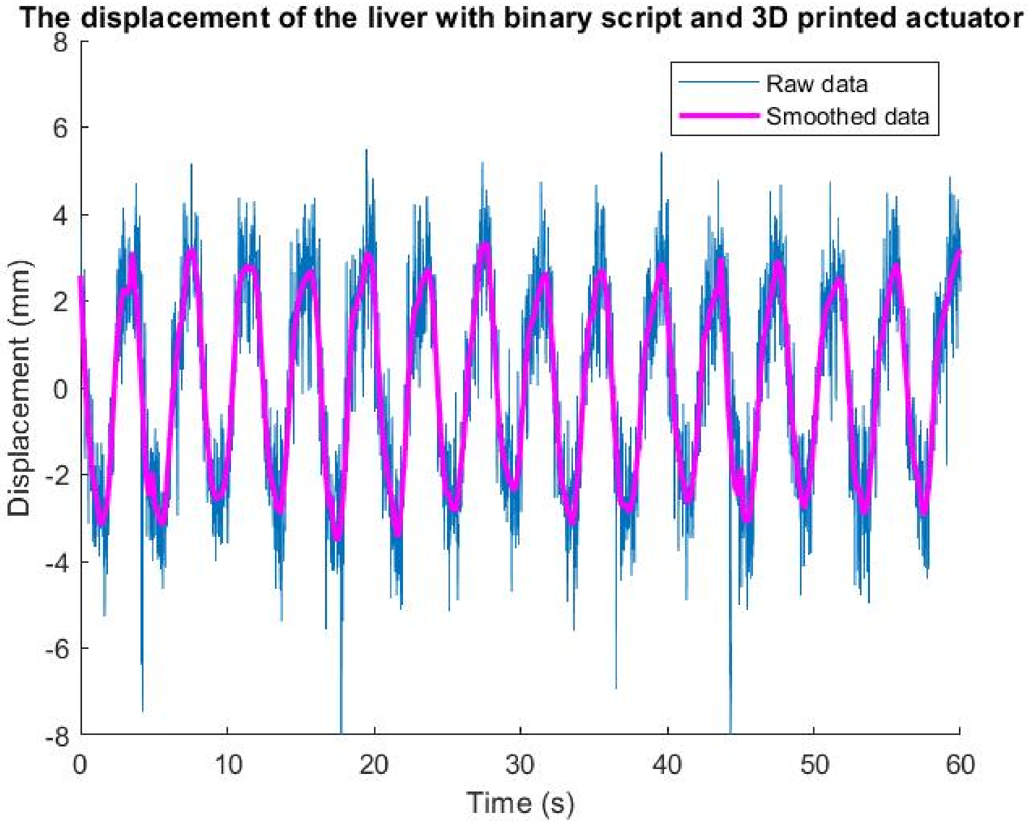

Figure 22: Results with the binary code and 3D printed actuator second time

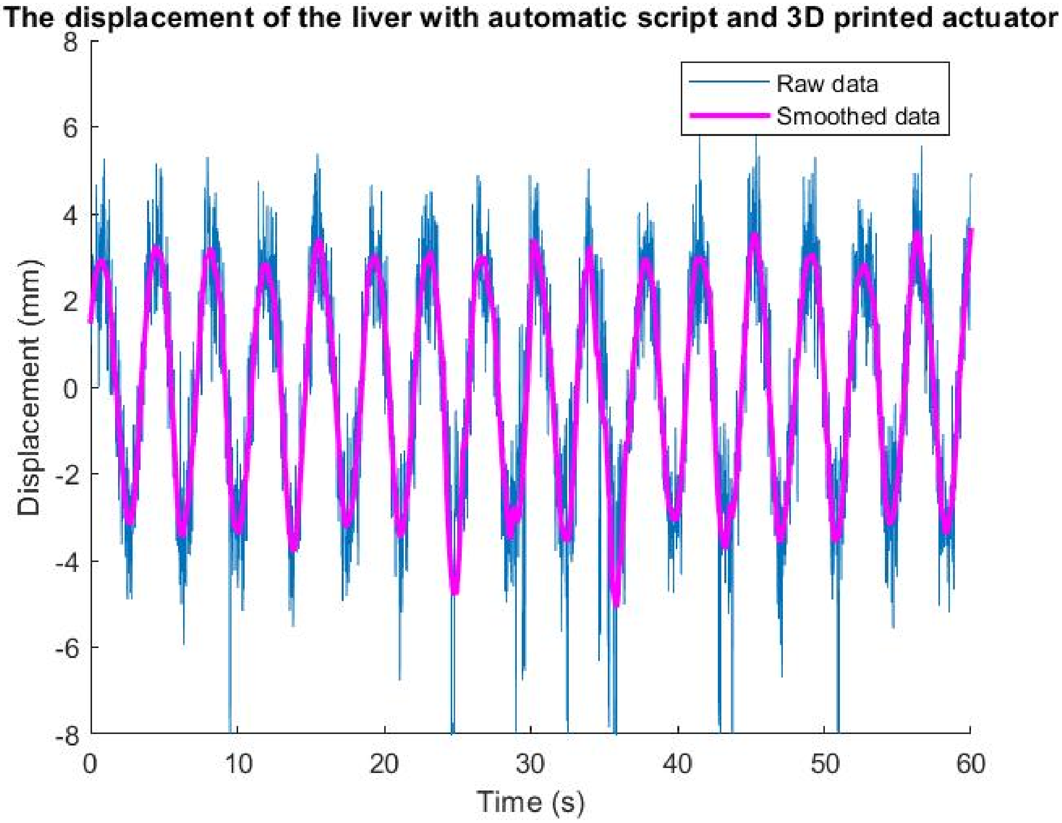

Figure 23: Results with the automatic script and 3D printed actuator second time

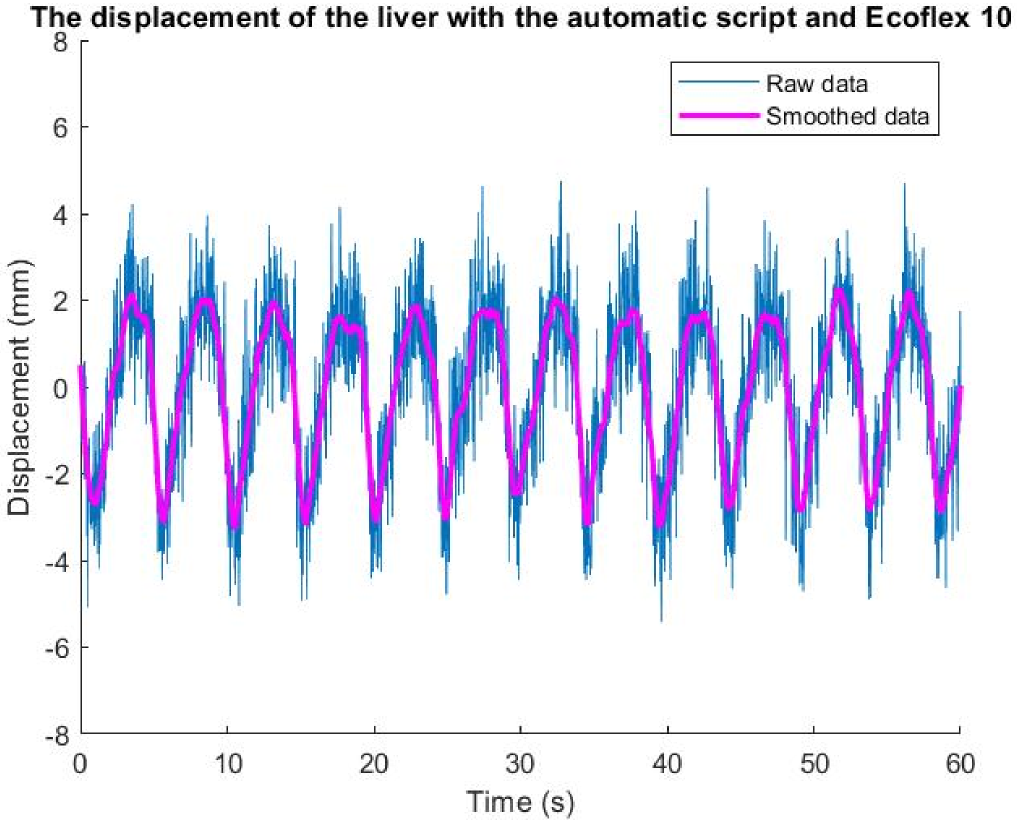

Figure 24: Results with the automatic script and Ecoflex 10 actuator second time

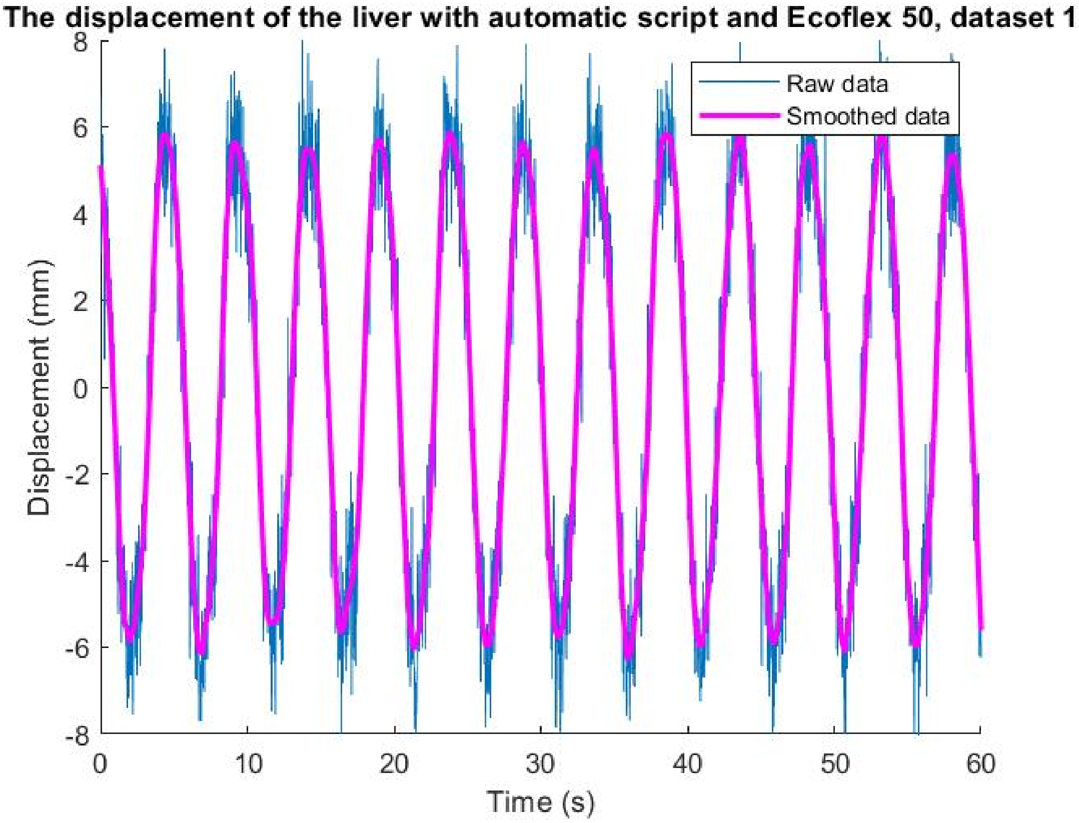

Figure 25: Results with the automatic script and Ecoflex 50 actuator, dataset 1 second time

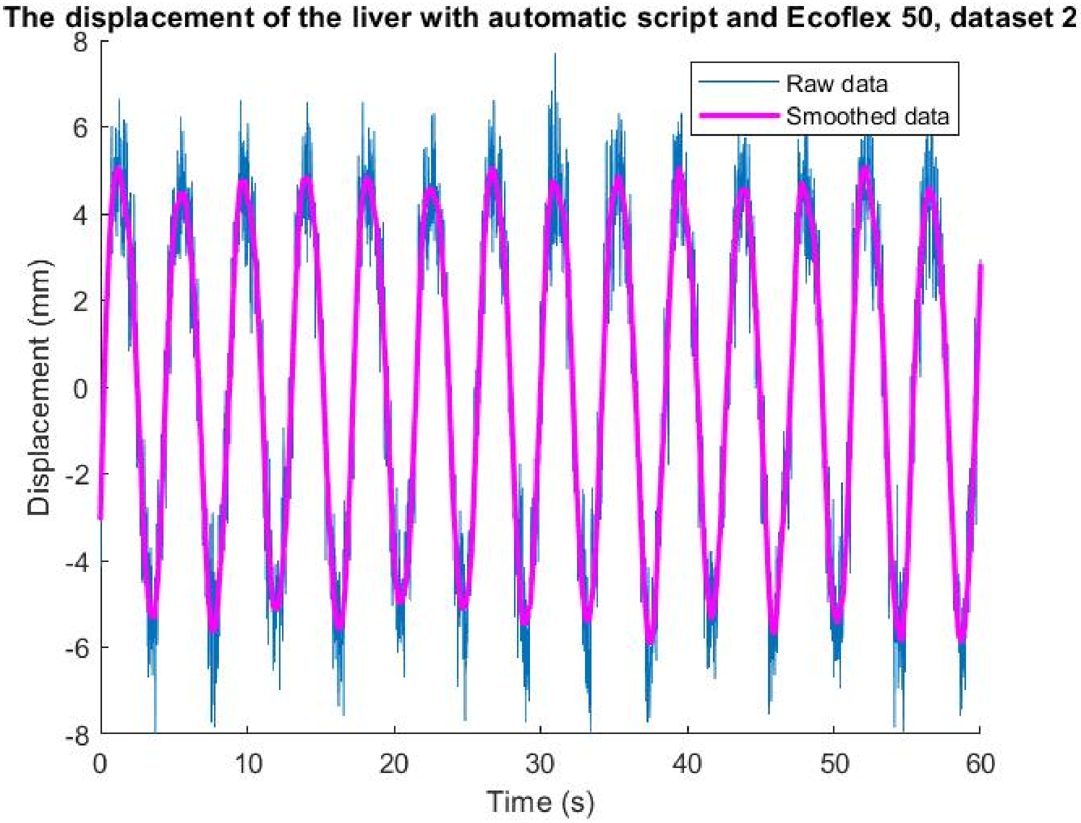

Figure 26: Results with the automatic script and Ecoflex 50 actuator dataset 2 second time