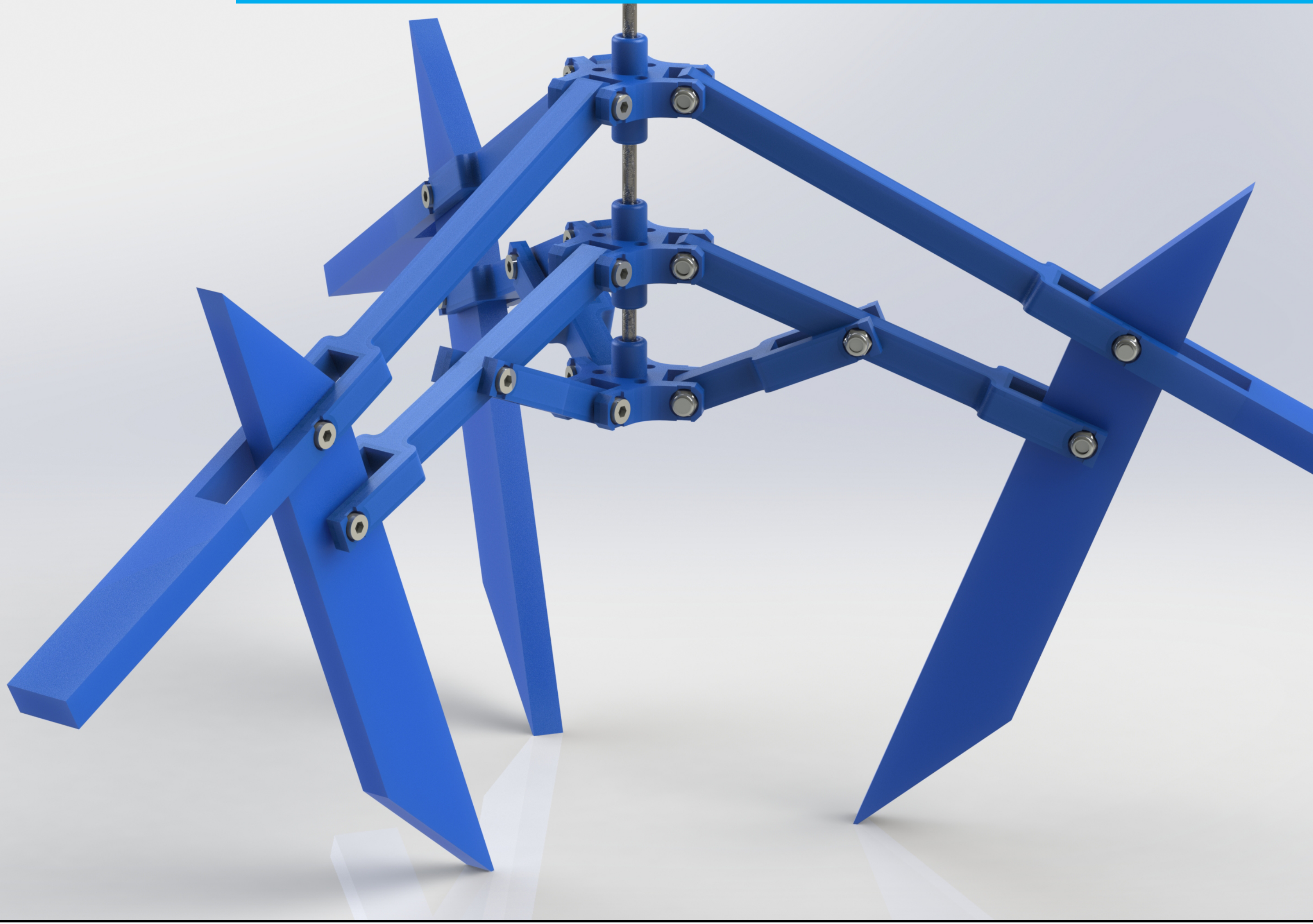
Department of Precision and Microsystems Engineering

Design of a fully actuated dynamically balanced 2-DoF gripper

H.T. Hofland

Report no	: 2024.079
Coach	: Prof. dr. Ir. J. L. Herder
Professor	: Prof. dr. Ir. J. L. Herder
Specialisation	: Mechatronic System Design (MSD)
Type of report	: MSc Thesis
Date	: 27-9-2024



Design of a fully actuated dynamically balanced 2-DoF gripper

H.T. Hofland (4446070)

September 4, 2024

Introduction

Due to the ever increasing need for automatisation there are many different types of grippers to pick and place objects of various shapes and sizes. These different grippers have been classified in multiple studies e.g. [1], [2], [3]. For this paper the most important distinction is made between mechanical and non mechanical grippers. Here mechanical grippers are considered to be made up from actuated elements that grasp the to be picked object by either frictional force or by geometric locking. The most common example of a mechanical gripper is the 2 fingered gripper with a friction based grip [4]. These are often available as off-the-shelf products where the right size finger and actuation range can be selected. On the other hand non mechanical grippers can make use of a variety of different ways to keep hold of their target object. They can use vacuum, adhesion, magnetism or bellows for instance. However, because the principles of balancing only apply to mechanical structures there will not be expanded on non-mechanical grippers any further.

Within the realm of mechanical grippers further division can be made by categorizing on gripper characteristics. First the amount of fingers a gripper uses to grasp the object can vary from 1,2 or 3 to full dexterous hands. Secondly the use of either rigid or soft elements is an important factor. Where a rigid element can usually apply more force, a soft element can conform to the object to secure it in place [5]. A third category can be based upon the way the object is held. This can be done with either friction or by geometrically constraining the degrees of freedom (DoF) of the object. The latter often leads to lower contact forces as these are used to hold the object directly instead of being converted to their friction counterpart [6]. A last distinction was made based on the amount of actuators used compared to the number of DoF which the gripper mechanism has. In case of a fully actuated gripper every degree of freedom has their own actuator hence the motion of the mechanism can be completely described from the actuator input. In the case of an underactuated gripper there is a smaller amount of actuators present than the mechanism has degrees of freedom. These grippers can generally change shape based on the contact forces with the target object hence conforming to the surface of the object which can also be used to lower contact forces [7], [8], [9]. Creating an underactuation can give a simple friction based gripper much more dexterity and with that make it useful in a wider variety of gripping cases [10], [11].

When mechanisms move they generally create reaction forces on their base. These forces are caused by the changing center of mass position of the mechanism. As stated in Newtons second law $F = ma$ [12] to accelerate the mass of an object relative to another object, a force is needed. Hence changing the center of mass of a mechanism relative to its base will generate a force between the two. Balancing a mechanism then is the act of creating or altering the mechanism in such a way that the center of mass does not change relative to the base. This in turn makes that there is no force between the base and the mechanism when the mechanism moves. There are two different stages of balance which can be applied to a mechanism. The first is when the mechanism is only balanced for reaction forces. This is called shaking force balance. In this case the rotational motion of the mechanism parts can still generate moments on the base. The second stage is called full or dynamical balance. Here the reaction moments are balanced as well as the reaction forces. Shaking force balance is a prerequisite for fully balanced systems. The advantage of a balanced system over an unbalanced mechanism is that the mechanisms motion is decoupled from the motion of the base. This means that the mechanism will not move through its own motion when the base is moving the mechanism and that in turn the base does not feel the movement when the mechanism is moved. Incidentally this also means the mechanism does not need a stable base to perform its motion and that it can still perfectly function for instance on the end of a piece of string. A useful use case of a balanced gripper to exploit this could be pick and place operations done by drones. Because balancing a pre-existing mechanism can usually only be done by adding rotating counter balance mass, balanced mechanisms are more often than not heavier than their non balanced counterparts. This means stronger joints and actuators are needed. However the control technology needed is much simpler as feed forward control is already sufficient. As a dynamically balanced system is balanced statically as well it will also carry over the characteristics that come with statically balanced mechanisms, namely that due to the balancing the mechanism does not want to move on its own as it does not have a preferred position to minimise its energy potential. Hence there are no holding forces required to keep the mechanism in place which are usually provided by feedback control. Using an inherently balanced mechanisms to build a structure can give a balanced system which does not rely on counter masses and does therefore not have the weight disadvantage of a mechanism that has been balanced after its function design was done.

Looking at these advantages and keeping in mind the use cases of high speed pick and place operations a gap seems to be present where balanced grippers can be used. However, in literature only a single example of a balanced gripper was found [13]. Moreover, this mechanism was not experimentally verified and actuated, but generated from a theoretical synthesis method and built as a proof of concept.

Because multiple design strategies and specific uses can be thought of to synthesise gripper mechanisms this paper focusses on finding a generic method to come up with these grippers. A single design is made without a predetermined use case to verify the method. This paper then presents the design and testing of a new partially inherently shaking force and shaking moment balanced 2 degree of freedom gripper. With simulations and experiments the gripper is shown to be balanced by reducing the necessary actuation forces at equal actuation speed and stoke length. Due to the higher order of freedom the gripper is able to pick and place a large range of objects with predetermined sizes. The paper starts by explaining the design methodology chosen to create the mechanism after which the resulting gripper and its characteristics are described. Next the simulations are explained which were used to verify the design is a balanced mechanism. Then the real world test setup and experiment are explained. Lastly the results of the simulations and the real world tests are shown and discussed before giving the final verdict on the balance of the mechanism and the design method chosen.

METHOD

Design of mechanism:

In order to come up with a balanced mechanism several techniques can be used. First of all the most used method is to balance a mechanism after its function has been designed. This is done by carefully adding masses in such a way that the total center of mass (CoM) of the system will be at the joint which connects the mechanism and the fixed world for every orientation of the mechanism. Because the CoM is independent of system's position through its range of motion the mechanism does not exert dynamic forces on its base [14]. However, because all this added mass does not contribute to the core function of the mechanism this often leads to heavy mechanisms which increases the internal forces in the mechanism parts and makes that joints need to be able to support much higher loads. Therefor a more optimised design can be found by creating a mechanism in which the functional elements balance each other out without the need to add mass later. This is known as inherent balance. In order to synthesize an inherently balanced mechanism tracing beams can be inserted into an open or closed kinematic chain such that they trace the CoM. This is called the method of principal vectors and was described by Otto Fisher around 1900 [15]. A specifically interesting mechanism that arises from this method is the pantograph with counter masses which can be used to balance other mechanisms as well as being balanced by itself [16].

A third way of creating balanced mechanisms is by adding known balanced mechanisms on top of each other and then offsetting the weight of the upper mechanism with a counter mass in the base mechanism which again gets the system CoM back at the fixed world joint. As there are many mechanisms for which a balanced version has been found this gives a great number of possibilities, but again resorts to adding mass at the end. However because the mechanisms which are added are already internally balanced the mass to be added is reduced which makes this an intermediate option.

All the previously named design methods are primarily concerned with the creation of a shaking force balanced system. Although this is the first requirement in order to make a system completely dynamically balanced it is not the only one. In order to cancel out all the reaction moments at the base the rotations of all parts have to be compensated as well. This can be done by making the previously described counter masses rotate by means of gearing systems such that they compensate the inertia of their counterparts completely. However in the creation of the gripper a different route was taken. Because the gripper would have multiple mechanical fingers there was chosen for an axisymmetric design. When the position of the fingers is fully determined by the input of the actuators that drive the mechanism and is not influenced by its surroundings as is the case with underactuated grippers, then the motion can be made completely symmetric around the CoM. Therefor the moments created by one finger will be compensated by the others without the additional weight penalty of rotating counter masses.

For the final design a fully actuated gripper with 2 DoF was synthesised built up from 2 pantograph mechanisms stacked on top of each other. These pantographs are connected at the tip of the long arms by a beam which will also act as the grippers finger. The CoM of the whole system was chosen at the base of the lower mechanism as that would also give a stationary base for the object to be picked up to be held against making an extra contact point. Figure 1 shows an early cardboard prototype of the mechanism.

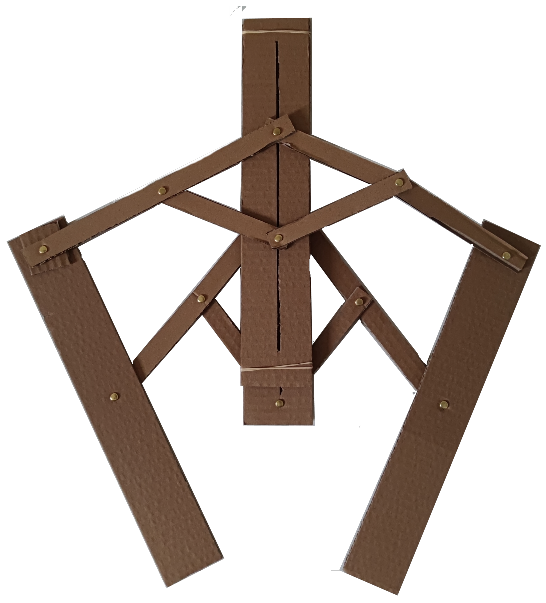


Figure 1: Cardboard model of gripper design

There was chosen for a 2 DoF gripper to get greater variety in the grippable objects as no specific use case was defined for the gripper. In analysing the motion of the mechanism it was seen that the small beams of the upper pantograph were interfering with the large beams of the lower pantograph. Fortunately the small beams of the upper pantograph can be taken out without affecting the motion of the mechanism. This lead to the final mechanism which can also be thought of as a pantograph with a two beam kinematic chain on the end, the kinematic chains together would be balancing out the weight of the slider while the other slider is balanced by taking it as a point mass on the pantograph beams. In balancing both pantographs the following formulas (Equation 1) were used as described in [14]. The slider was included to keep the mechanism axisymmetric ensuring moment balancing for the mechanism motions. The slider rail will at the same time act as the base to which other objects can be connected.

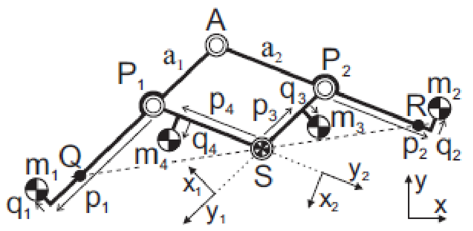


Figure 2: General 2D pantograph with masses m_1, m_2, m_3, and m_4

$$\begin{aligned} m_1 p_1 &= m_2 a_1 + m_3 p_3 \\ m_1 q_1 &= m_3 q_3 \\ m_2 p_2 &= m_1 a_2 + m_4 p_4 \\ m_2 q_2 &= m_4 q_4 \end{aligned} \tag{1}$$

Design features:

Figure 3 shows the final design of the gripper. This image shows the size version of the gripper that was used during the experimental verification of the balance. As can be seen there was chosen for a 3 fingered design. This gives the minimum number of contact points needed to constrain a gripped object fully.

Because the fingers that are put in the gripper can be of any arbitrarily chosen shape as long as the weight and CoM are in the correct place for the balance, the gripper will be able to pick up a wide size range of objects. In Figure 4 the full range as derived from the SPACAR software is shown. In Figure 5 the range is confirmed with a smaller proto type. The Balls shown in the figure vary from 17 to 67 mm. For this particular prototype the mechanism would allow for objects of 10 to 88 mm, however for objects smaller than 17 mm shorter fingers need to be made

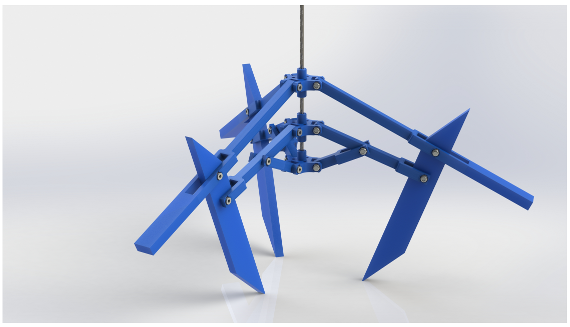


Figure 3: Final design of gripper

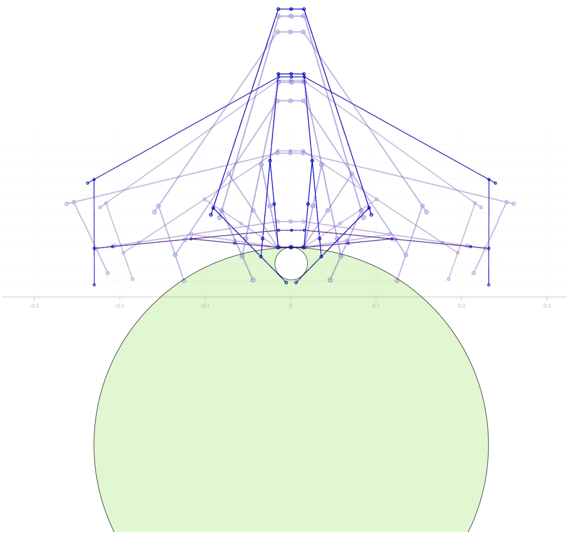


Figure 4: Range of the mechanism

to prevent self-intersection of the mechanism. The maximum size of 88 mm makes for a pinch grip where the fingers are parallel and the grip is based on friction rather than geometrical locking.

The size ratio between the long beams in the upper pantograph compared to the lower pantograph of the mechanism has in the end been chosen in such a way that the upper pantograph has a beam that is longer than the beam of the lower pantograph (see Figure 17 in Appendix B for the ratio). This ensures that even at the widest configuration of the lower pantograph a geometric lock with 60 degree angels can still be made. This geometric locking grip is preferred over a friction locked grip as the object is simply constrained in its movement instead of held by (friction) force. In the end this should lead to lower contact forces between the object and the gripper.

Simulation verification methods:

After the design of the mechanism the balance was verified with two numerical methods. First a simulation was made in which the positions of the input beams was prescribed. From these the position of all the other beams was calculated by means of their joints and lengths. For the position of the CoM of the designed beams the data from the SOLIDWORKS CAD files was used for each beam individually. With the beam position and the internal CoM of all the beams the total combined CoM of all the beams was calculated. In order to account for the hardware needed to make up the joint in a physical model extra point masses were added. By looping over different input positions and combining all the mass locations with a weighted average, the change in the position of the total CoM of the system can be computed. For a balanced system this combined CoM should not move from its location as this would impose reaction forces on the base of the gripper.

Because the first simulation does not take into account any forces, moments or inertia a second simulation was made. This simulation was run by the MATLAB extension SPACAR. This is a package specifically designed to calculate motion of rigid and flexible elements. The model is setup by an array of massless beams which are interconnected via user specified joints. The beams are described as a straight line between two specified points. These points were calculated by filling in the properties of all the beams in parametric formulas which describe the joint positions. To add mass a new beam is added between the CoM of a beam, which is gathered from CAD like before, and one of the joint points of the beam of which the mass is specified in the CoM. This mass beam is then connected to the original with a fix constraint unifying the movement of the mass beam and mass point with that of the actual beam. In the end all DoF have to be determined by either an input value, an external constraint for instance a slider or the fixed world, or a joint constraint. Dependant on the type of joint indicated is which DoF are described by it. This makes that a deterministic outcome can be calculated if the given input motion does not cause a singularity or impossible configuration. When the beam structure is fully described the simulation is run based solely on the inputs and constraints. The prescribed points are only used for the initial setup. After running the simulation the forces and motions at all the separate nodes are known which can confirm both shaking force and shaking moment balance.

In both simulations perfectly rigid beams and perfect joints were assumed. This is an inherent assumption in balanced mechanisms as an unknown flexibility will unbalance a system immediately.

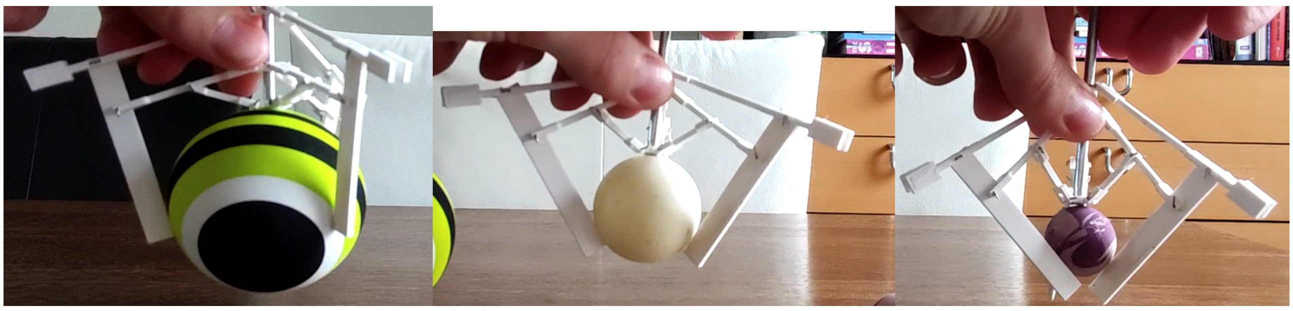


Figure 5: Real world range of the mechanism left to right 67 mm, 39 mm, 17 mm

Design of Experiment parts:

For the verification of the mechanism several different experiments can be performed. The first and easiest test is putting the mechanism in different configurations and checking if it remains in equilibrium. If this is not the case the mechanism has a lower energy state towards which it tends meaning the CoM goes down in the gravitational field. With a moving CoM the mechanism cannot be force balanced and in extension it can not be dynamically balanced. The follow up test is to agitate the mechanism by shaking it and rotating it. If this agitation leads to movement of the mechanism itself there again is a preferred state for the CoM and the mechanism is again unbalanced. When an axisymmetric design is chosen it is possible that the mechanism is moment balanced for its own motion, but when an external rotation is applied to it the mechanism has an outward tendency. This is caused by the sub mechanisms CoM. In the case of the gripper it is the CoM of one finger going outward with the rotation which is a then preferred state. Then the mechanism is not entirely dynamically decoupled from the bases' motion.

After the agitation tests have been done the mechanism will be mounted to a linear actuator. With this actuator the actuation force for a balanced version of the gripper will be compared against a non-balanced version. Because the actuator is not taken into account during the design of the gripper it is not possible to do a separate force measurement at the base of the actuator as this has to be connected to the actuator itself. Therefore forces can still travel through the actuator even if the force at the base is measured to be 0. In other mechanical balancing literature tests are done by suspending the mechanism to check if it starts moving upon actuation, however for this setup the same issue as before is encountered. Because the actuator was not taken into account there will be a change in the CoM position and therefore a force on the base of the whole system. Moreover if the actuator was kept out of suspension while the gripper was suspended an external force will be introduced moving the suspended parts. For these reasons the actuation force measurement was chosen as a verification method for the balance.

For the actuator there was chosen to use a linear induction actuator. The working of such actuators can be explained as a normal electric stepper motor straightened out. Where the magnetic stator is now a rod moving through the electromagnetic field made by the coils in the static base of the actuator. The advantage of this type of actuator is that it can reach much higher velocities than more common spindle drive linear actuators. This makes that the force the mechanism exerts on its base in case of an unbalance will be much greater. Another common high-speed linear actuator is a pneumatic actuator, but this does not have any positional or force feedback. The reached position in this actuator will be determined by the equilibrium between the force of the mechanism or the end stop and the gas pressure on the piston.

The actuator has a positioning repeatability of $\pm$0,150 mm and an accuracy of $\pm$0,5 mm. With the control electronics from the supplier the current can be measured and with an online user interface the speed, stroke length and end positions can be set. As an input signal a binary 5 V signal is used where the switch from off to on indicates extension to the set length in the UI. This signal is made with an Arduino UNO which calculates the timing of the switch based on the input extension frequency (how many times per minute should the actuator go back and forth). The Arduino code can be seen in Appendix K.

In order to gain positional data a 24 step rotary encoder was attached to the mechanism with a gearing system. A gearing rack was bolted to the magnetic rod of the actuator by means of two friction based clamps. The rotary encoder was connected to the rack via a single gear. One full rotation of the encoder leads to 42 mm (linear)travel of the gear. Hence this gave a measurement precision of ±0.875 mm. Despite this being larger than the positional precision of the actuator no gearing reduction was introduced because this would force higher rotational speeds than the specifications of the encoder allow for.

To measure the force exerted by the actuator an amperage measurement was done between the power supply and the actuator. Combining this with the set supply voltage gives an electrical power. For the measurement a ACS712 20 A range current sensor was used. This sensor gives an analogue output voltage of 0 to 5 V for a range of -20 to +20 A. The Arduino analogue input has 10-bit range for 0 to 5 V hence a current difference of ±0.02 A can be measured. From the position and electrical power the actuation force can be calculated assuming a lossless system.

$$F = \frac{U\ I\ \Delta t}{\Delta x} \tag{2}$$

In the end the electric circuit is build up from three parts: 1) power circuit for the actuator, 2) position measurement circuit with the rotary encoder, 3) amperage measurement circuit.

The full electric schematic can be seen in Appendix A

The test setup is shown in Figure 6

Experiment description:

Shaking test:

The first test is to hold the base of the mechanism in different orientations and see if it moves due to the influence of gravity. This would indicate a preferred position (lower energy solution) which means the CoM moves with the movement of the mechanism. When, after holding the mechanism in any orientation of the base, the mechanism does not move on its own it can be adjusted to any different mechanism position by hand while holding the base in the same orientation. Doing this for different positions for both sliders and different base orientations gives a good indication for balance behaviour. For this experiment the base will be held at a base orientation of approximately 0 ,30 ,45 ,60 ,90 ,and 180 ° and both sliders will be put in 3 different positions each testing at least 6*9 = 54 different orientation position combinations.

Secondly after putting the mechanism in the different orientations and positions as described above the base will be shaken. The forces transferred through the base should not affect the position of the mechanism itself. Only the position of the complete mechanism in space should change. If the position of the mechanism parts does change due to the shaking force that means the mechanism is not force decoupled from its base which is an inherent property of shaking force balanced mechanisms.

As a last unactuated test the base will be rotated at the earlier specified base orientation angles. Because the mechanism is made axisymmetric it can be that even though it is dynamically balanced for the slider movements, the CoM of the individual fingers goes towards and away from the base. This means higher rotational velocity leads to centripetal forces on these CoM's which in turn change the position of the mechanism.

Actuated test:

For the test with the actuator five different cases will be tested. 1) actuator and measurement system separately, 2) top slider actuated while holding the bottom slider in place with a (assumed) balanced gripper configuration, 3) bottom slider actuated while holding the top slider in place with a (assumed) balanced gripper configuration, 4) top slider actuated while holding the bottom slider in place with a unbalanced gripper configuration, 5) bottom slider actuated while holding the top slider in place with a unbalanced gripper configuration. 1) will give a baseline of power usage for the actuator and measurement system which can be compared to the other cases. The other four cases test the movements of the top and bottom pantograph separately in such a way that the other pantograph cannot move and hence disturb the balance. The unbalance will be made by installing a different unbalanced finger beam in the mechanism in only 1 of the arms making the design asymmetric and the sub mechanism unbalanced. Per case at least 100 back and forth movement cycles will be made and logged in a text format data file. The data included will be: time elapsed from start of running code, position measured by rotary encoder, input from Arduino controller in binary, text version of the input (high/low for easier interpretation operator during test), Amperage measurement value in 10-bit number. This data is logged by a opensource software called CoolTerm that reads the Arduino serial monitor.

Results

Simulation results:

Position simulation results

In Figure 7 the graphical representation of the positional simulation is shown. The code computes the

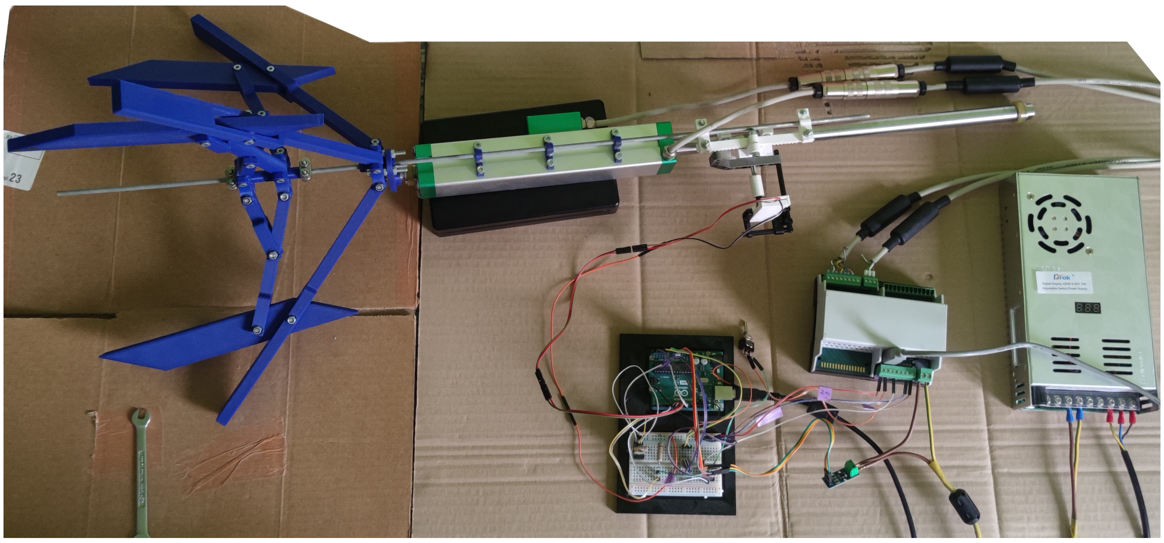


Figure 6: Actuated test setup

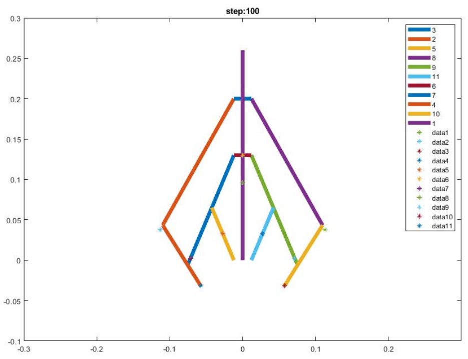
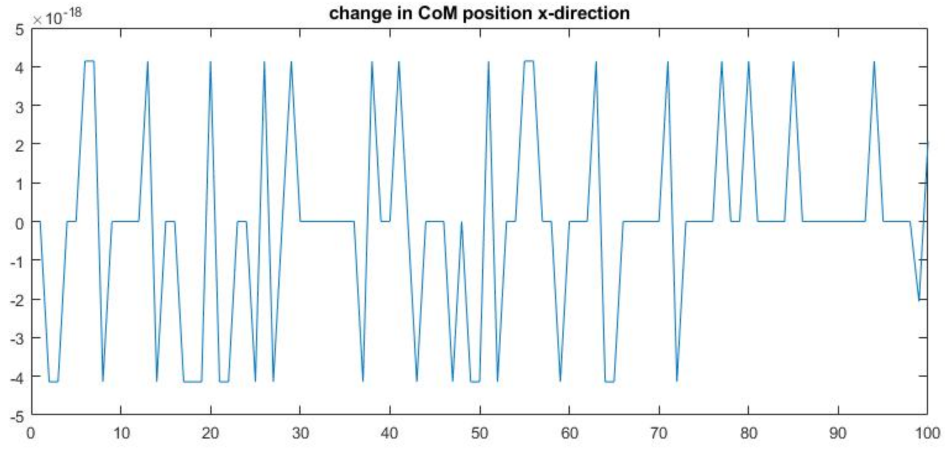


Figure 7: Position simulation graphic

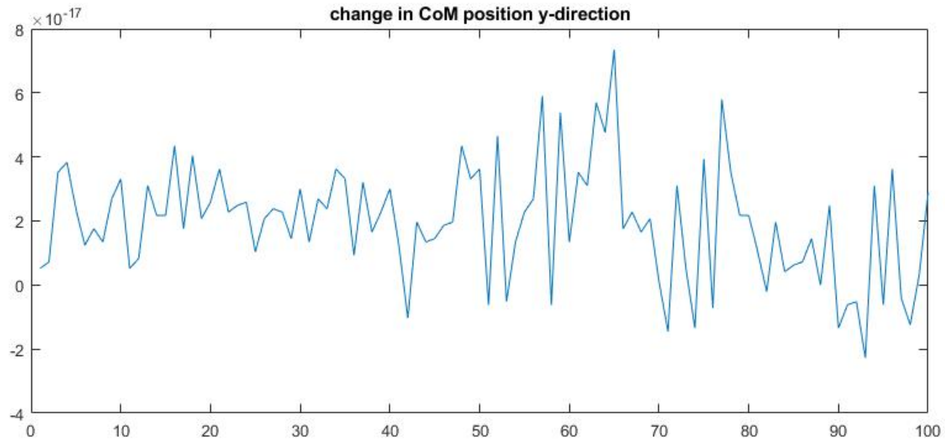
position of every beam based on the input of the two sliders. Afterwards it calculates the combined CoM of all the beam CoMs and masses. In Figure 8a and 8b the CoM positions in x and y respectively are shown. The x-axis of the plots are the step number in the simulation where the sliders are moved linearly from 10 to 170 mm for slider 1 (lower slider) and from 70 to 200 mm for slider 2 (upper slider) in 100 steps. It can be seen that for this simulation the CoM is computed to change in the order of 10^{-18} m in the x-direction and 10^{-17} m in the y-direction. These fluctuations can be explained by the numerical error of the numbers used in the calculation.

SPACAR simulation results

The SPACAR rigid body simulation results for 4 distinct cases have been calculated as can be seen in Appendix C. The cases are 1) both sliders move from roughly the middle of their range of motion away from each other, 2) the lower slider moves from top to bottom of its range while upper slider is held still 3) the upper slider moves from the bottom to the top of its range while the lower slider stays still, and 4) Both sliders move from top to bottom together. Figure 9 shows the graphical representation of the simulated movement of the first case. Here the red dotted lines are the initial position and the blue lines are the current position of the mechanism at the time selected on the slider underneath. The circles represent the nodes which are used to create the beams that make up the mechanism, represent the mass, and are used as hinge, slider, or rotational joints. In Figure 10 the change in the CoM position can be seen for case 1. Similar to the positional simulation we see numbers in the order of 10^{-17} m which is at the numerical precision limit of the numbers used in MATLAB. Figure 12 shows the corresponding reaction forces on both the sliders separate (yellow and orange), the sum of the forces on both sliders combined (purple) and the reaction force at the base (blue). Note that these forces are internal in the mechanism and in the case where the mechanism is balanced these forces should be equal and opposite (blue = -purple). This is because the mechanism will not add to the forces acting on the base and hence the external actuator forces are the same as the base forces. The reaction forces calculated at the base minus the actuation forces for



(a) CoM-x



(b) CoM-y

Figure 8: Simulated Change in CoM position

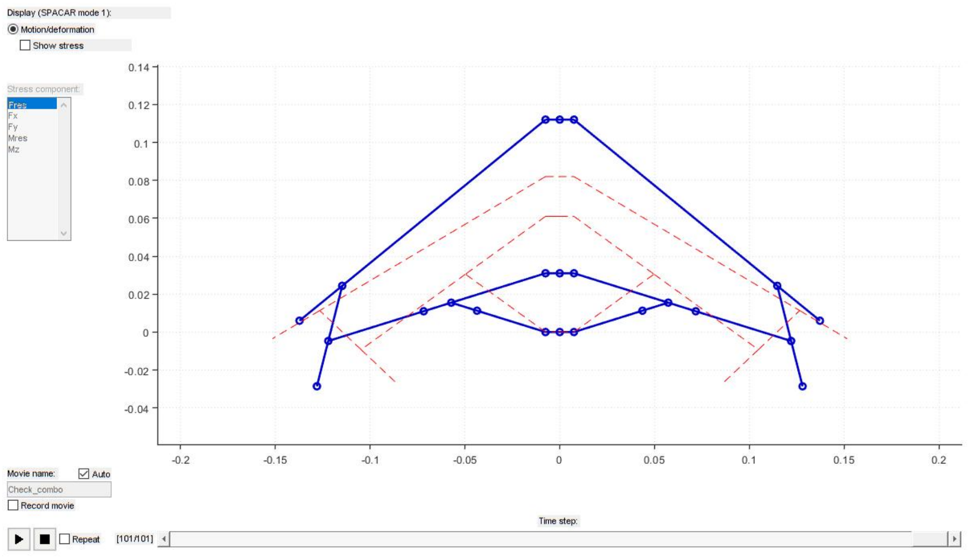


Figure 9: SPACAR simulated movement with red initial and blue current state of mechanism

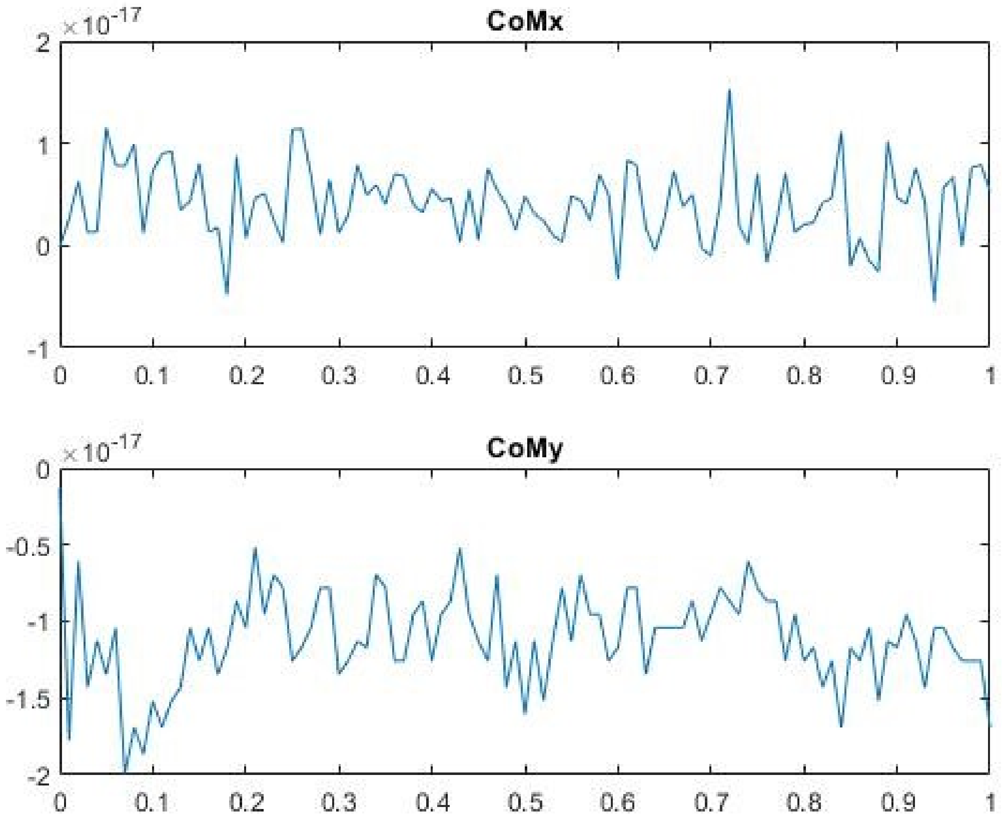


Figure 10: SPACAR simulation CoM change for case 1

case 1 can be seen in Figure 11. The same graphs for all four cases are shown in Appendix C. In all four cases it is seen that the forces are in the order 10^{-16} or 10^{-17} N, which again is the numerical precision limit for the simulation. From this it is concluded that the simulated mechanism is balanced. As the separate and the combined motions do not generate any force.

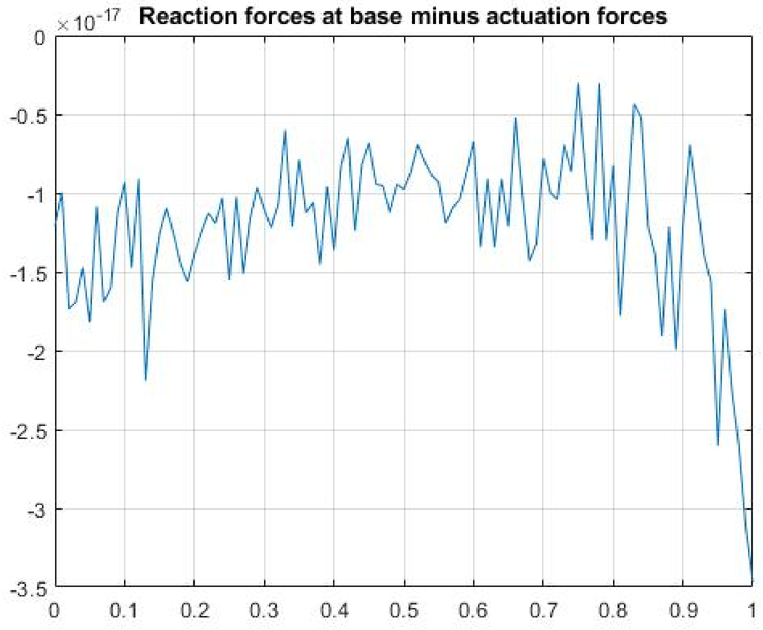


Figure 11: Case 1: reaction forces when both sliders go from middle away from each other

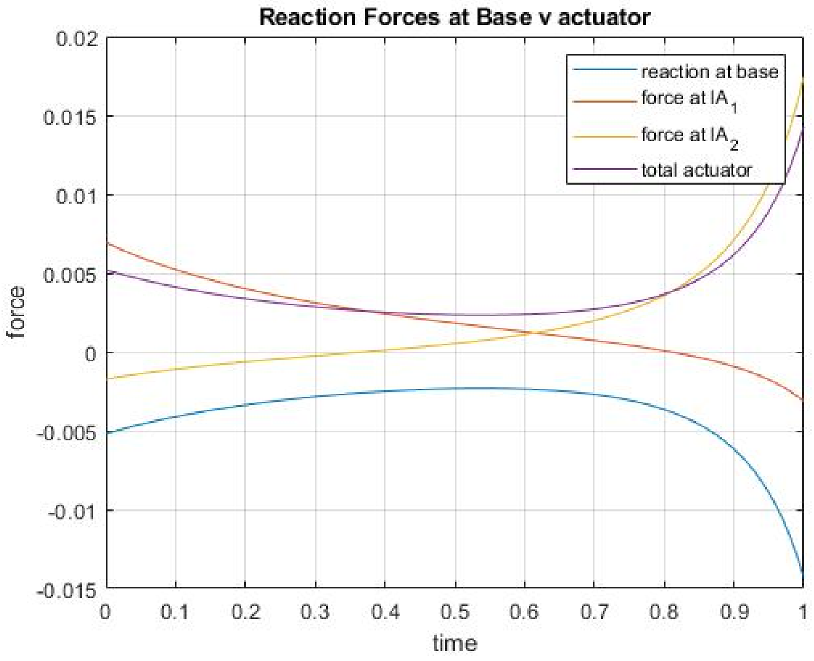


Figure 12: SPACAR simulation sum of actuation forces and base reaction forces are equal and opposite

Experimental results:

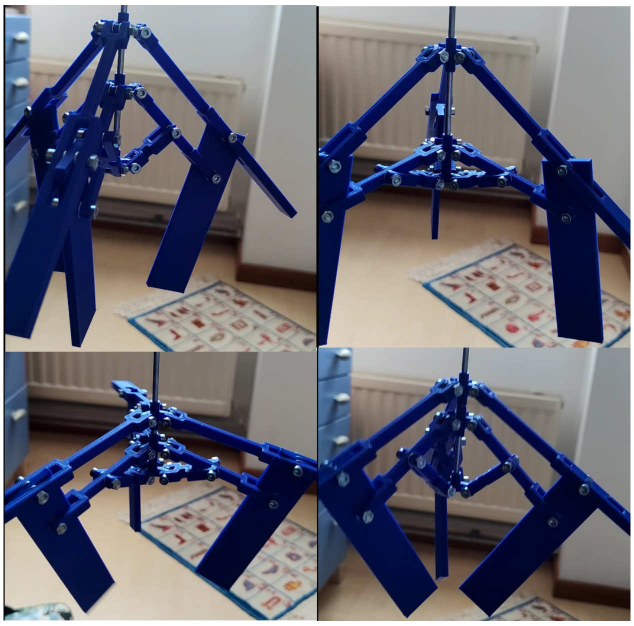


Figure 13: Different slider positions shaking test

Shaking test results

When the mechanism was assembled and suspended from the base it was seen that it did not move when being picked up or carried around. This was a good first sign because before the joint point masses were implemented in the SPACAR model the hardware immediately moved up and down the base when the base was lifted until it reached the preferred state. With this initial stability in free suspension the mechanism was put into different positions as previously described and shown in Figure 13 and in none of these positions did the mechanism show any movement or instability. Also, when shaking the base there was no internal movement seen in any of the different positions only a translation of the mechanism as a whole. All of this combined gives a good indication that the mechanism is shaking force balanced. Lastly when the mechanism was spun around the axis it was observed that there was movement. This shows that the mechanism is not decoupled from a rotation of the base. This means the mechanism seems to be balanced for the movement of the mechanism itself, but in the end is not completely dynamically balanced.

Actuated test results

Figure 14 shows the acquired mean force data from the different load cases plotted on top of each other for comparison. The graph with the complete standard deviation can be seen in Appendix D. First it was noticed that the standard deviation (shaded area) was very large compared to the mean value where often the peaks in the standard deviation were a factor 20 larger than the mean. Looking in the raw data it was found that there was drift present in the position data during the measurement. This makes that when the position goes up more during one cycle than on other cycles it a higher actuation force is calculated. Because of the high number of cycles per load case it is assumed that this random drift noise does not influence the mean as the random drift noise cancels out when averaged. Where Figure 14 is the zoomed in version on the means Figure 19a shows the complete data including the full standard deviations for all the load cases. In Figure 14 it can be seen that the force required for the actuator and measurement system alone is a dominant factor in the force required for the total system with gripper included. In Figure 15a the mean force difference between the load case with the unbalanced upper pantograph gripper and the free actuator is shown. This was done by subtracting the mean of the actuator and measurement system alone from the mean of the unbalanced upper pantograph. The same comparative graphs for the other four test cases are shown in Appendix E. Figure 15b is a similar comparison, but now between the case of the unbalanced upper pantograph and the balanced gripper. From this we can determine the influence of the balancing on the required actuation force. The force profile between balanced and unbalanced load case compared to the free actuator is seen to be very similar for both the top and bottom slider. In both cases the peak forces are higher in the unbalanced case compared to the balanced case. This is what can be expected for balanced mechanisms as these do not push the actuator in a particular direction as their movement does not generate force on the base. In Figures 15b and 20f the comparison is made between the balanced and unbalanced versions of the same slider. Over the whole motion the reduction in force from balancing the mechanism seems about 0,5 N where the unbalanced gripper needs around 1 N and the balanced gripper 0,5 N. This is a reduction of 50%. In the peaks the difference is more in the range of 0,2 N where the peak for the top slider in the unbalanced case is 2,38 N compared to 2,17 N for the balanced case. For the bottom slider the peak is 1,26 N for the unbalanced gripper and 1,09 N for the balanced gripper. This means a force reduction of 9% for the top slider and 14% in the bottom slider.

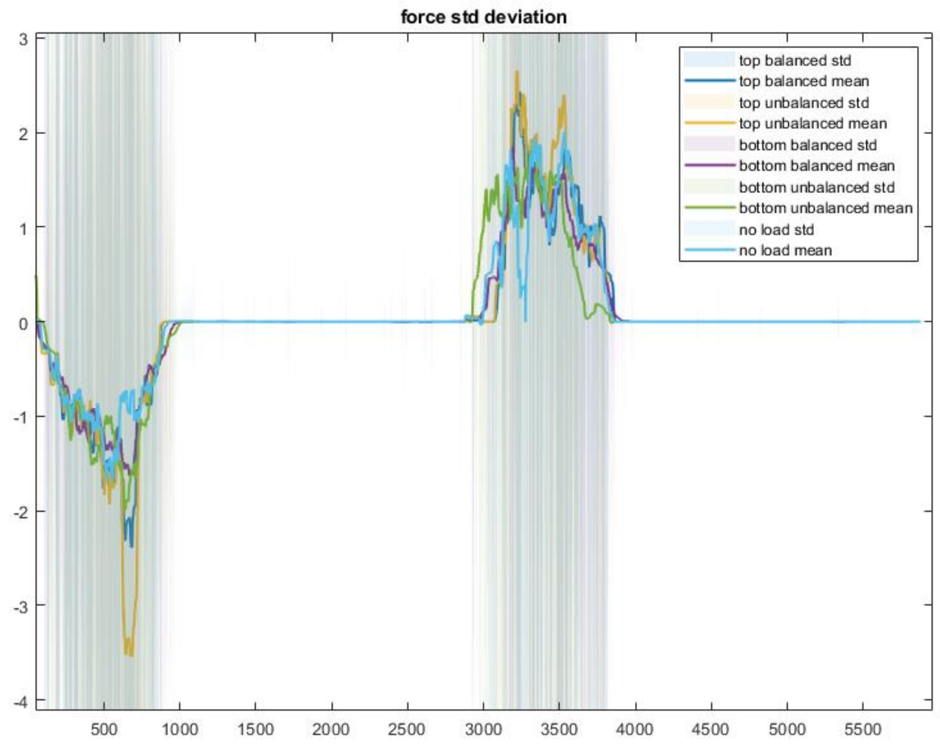


Figure 14: Measured forces actuated test 5 different load cases zoomed in on mean values

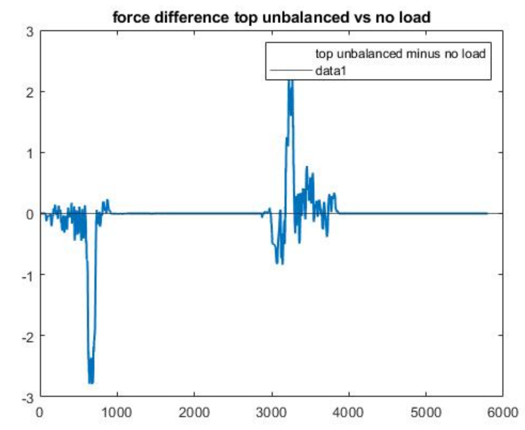
Discussion

While the proposed mechanically balanced gripper shows the chosen design method works for generating balanced mechanisms and functions well as a gripper in all tests and simulations, some notes have to be made.

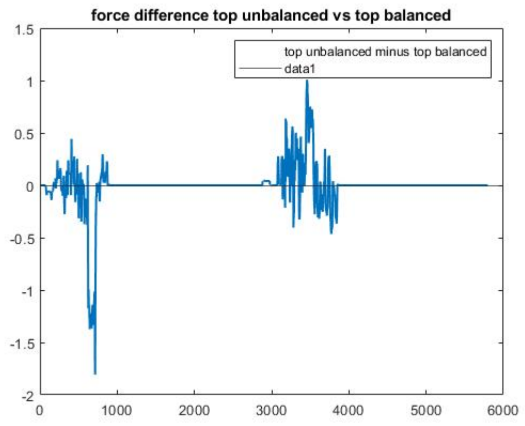
First of all the method of design and verification chosen did not allow for the shaking forces to be measured directly. Hence the indirect method of going by the actuation forces had to be used. This makes interpreting the data gathered from the actuated test more difficult. The shaking forces are combined with other influences such as friction in sliders and joints and possible inconsistency in the electric power used to make the actuator move masking the sought after shaking force values. As the results from the three other tests indicate the design is balanced and a reduction in the actuation forces is seen in the actuated test the balance seems to be verified. If this method will be used in future work it is critical to back up the actuated test by the simulations. Also the addition of the non-actuated test in the real world can already reveal unbalanced behaviour even if the simulations give a ”balanced” result as some simple components like joints may have been overlooked in initial simulations.

For the actuated test the force required for the actuator without any additional load was very high compared to the load cases with the gripper. Therefore the actuator behaviour obscures the force differences between the different gripping scenarios. In order to solve this it would be better to make the gripper out of a more dense material to make it heavier. This will also make the difference between the balanced and unbalanced cases more visible. However due to manufacturing constraints and mechanism complexity there was chosen for 3D printed plastic which compared to most common machining metals is very light weight.

According to the manufacturer the 3D printer itself has a printing precision of ±0,1 mm. Together with tolerances of the other hardware i.e. nuts, bolts, and washers a perfect balance can never be reached unless precision measurement and subsequent piece-by-piece tailoring of the counter mass parts can be done. So while the simulations show that the intended gripper is perfectly balanced this will not be the case for the real world prototype. However as it is observed that the mechanism sliders move with very little friction and because there was an unbalance



(a) Force top slider in unbalanced gripper minus force non loaded actuator



(b) Force top slider in unbalanced gripper minus force top slider in balanced gripper

Figure 15: Mean actuation force comparison for different load cases

seen in an earlier version of the gripper it is known that the friction forces are low compared to what the unbalance forces would be if the model was not sufficiently balanced. Therefore it is assumed that the observed balance is not a result of the friction forces holding the mechanism in place.

The chosen method of adding inherent balance elements introduces additional weight to compensate for the addition, potentially negating some of the benefits of using inherently balanced elements to begin with. In order to take full advantage of the inherent balance the mass tracing beam method might be a better option, however this leads to much more complex designs and gives less freedom to design the function of a mechanism. In the end it was calculated that around 60% of the weight of the mechanism was there for balancing purposes. A more precise number can not be given as the gripper fingers are at the same time used as a balancing beam. The rest of the purely balancing mass is located at the finger side on the upper pantograph.

As was shown in the shaking test the gripper is not fully decoupled from the rotational motion of its base. While all the internal motions will not apply a force on the base of the gripper, the base will apply some force to the gripper when in motion. For high-speed operations where the gripper is placed as an end effector on an actuating arm this might lead to the need for additional actuation force on the mechanism. In many applications however there will not be constant and or high rotational velocity, here the shaking force balance and internal shaking moment balance will still be beneficial for control purposes and vibration reduction.

Furthermore, because the actuator is not incorporated in the gripper, the system as a whole will generate a force on its base due to the shifting CoM of the actuator. This negates some of the advantages of choosing a balanced gripper design. Because the gripper itself has a stationary CoM it does not contribute to this force.

In general the method chosen as a whole seems to be a valid way of generating a mechanically balanced mechanism. As the design presented is a planar mechanism multiplied 3 fold around an axis this method will also work for any other multiplication larger than 1, most notably the subclass of (fully planar) 2D mechanisms. The lower limit on the multiplication for this mechanism is found in the axisymmetric choice of design which is not present with only a single finger. For the case of non-planar mechanisms it is assumed that adding already balanced mechanisms and balancing out the remaining forces will also work, however the mathematics for the simulations will become much more complex.

Conclusion

In this paper a partially balanced 2 DoF gripper was designed and verified. In two different simulations the design was shown to be completely balanced for all internal movements after which a prototype was made. In the real world actuated test it provided a 50% force reduction compared to the imbalanced version in most of the motion and around 10% force reduction in the parts where the forces peak. The gripper is shown to be able to pick up a large range of objects from about 11% up to 100% of its own width. For different applications this gripper can be scaled to desired dimensions and the finger length and shape can be adjusted to need. In the end it is concluded that the chosen method of adding (inherently) balanced elements to create new balanced mechanisms is a valid approach.

References

[1] G. J. Monkman, S. Hesse, R. Steinmann, and H. Schunk, *Robot Grippers*. John Wiley & Sons, Ltd, 2006.

[2] K. Tai, A.-R. El-Sayed, M. Shahriari, M. Biglarbegian, and S. Mahmud, "State of the Art Robotic Grippers and Applications," *Robotics*, vol. 5, p. 11, June 2016. Number: 2 Publisher: Multidisciplinary Digital Publishing Institute.

[3] Z. Samadikhoshkho, K. Zareinia, and F. Janabi-Sharifi, "A Brief Review on Robotic Grippers Classifications," in *2019 IEEE Canadian Conference of Electrical and Computer Engineering (CCECE)*, pp. 1–4, May 2019. ISSN: 2576-7046.

[4] L. Birglen and T. Schlicht, "A statistical review of industrial robotic grippers," *Robotics and Computer-Integrated Manufacturing*, vol. 49, pp. 88–97, Feb. 2018.

[5] M. Ceccarelli, G. Figliolini, E. Ottaviano, A. Mata, and E. Criado, "Designing a robotic gripper for harvesting horticulture products," *Robotica*, vol. 18, Jan. 2000.

[6] A. Bicchi and V. Kumar, "Robotic grasping and contact: a review," in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No.00CH37065)*, vol. 1, pp. 348–353 vol.1, Apr. 2000. ISSN: 1050-4729.

[7] T. Laliberté, L. Birglen, and C. Gosselin, "'Underactuation in robotic grasping hands'," *Japanese Journal of Machine Intelligence and Robotic Control*, vol. 4, pp. 77–87, Jan. 2002.

[8] G. Kragten, "Underactuated Robotic Hands for Grasping in Warehouses," in *Automation in Warehouse Development*, pp. 117–131, London: Springer London, 2012.

[9] H. Heidari, M. J. Pouria, S. Sharifi, and M. Karami, "Design and fabrication of robotic gripper for grasping in minimizing contact force," *Advances in Space Research*, vol. 61, pp. 1359–1370, Mar. 2018.

[10] R. R. Ma, A. Spiers, and A. M. Dollar, "M2 Gripper: Extending the Dexterity of a Simple, Underactuated Gripper," in *Advances in Reconfigurable Mechanisms and Robots II* (X. Ding, X. Kong, and J. S. Dai, eds.), Mechanisms and Machine Science, (Cham), pp. 795–805, Springer International Publishing, 2016.

[11] P. N. Koustoumpardis, K. X. Nastos, and N. A. Aspragathos, "Underactuated 3-finger robotic gripper for grasping fabrics," in *2014 23rd International Conference on Robotics in Alpe-Adria-Danube Region (RAAD)*, pp. 1–8, Sept. 2014.

[12] I. Newton, *Philosophiae naturalis principia mathematica*. Jussu Societatis Regiae ac Typis Josephi Streater. Prostat Venales apud Sam. Smith ad insigna Principis Walliae in Coemiterio D. Pauli, aliosq, nonnullos Bibliopolas, 1687.

[13] v. d. Wijk, Volkert, *Methodology for analysis and synthesis of inherently force and moment-balanced mechanisms*. University of Twente, 2014. OCLC: 6893348902.

[14] V. van der Wijk, *Methodology for analysis and synthesis of inherently force and moment-balanced mechanisms: theory and applications*. PhD, University of Twente, Enschede, The Netherlands, Apr. 2014. ISBN: 9789036536301.

[15] V. van der Wijk and J. L. Herder, "Synthesis method for linkages with center of mass at invariant link point — Pantograph based mechanisms," *Mechanism and Machine Theory*, p. S0094114X11001868, Oct. 2011.

[16] V. H. Arakelian and M. R. Smith, "Shaking Force and Shaking Moment Balancing of Mechanisms: A Historical Review With New Examples," *Journal of Mechanical Design*, vol. 127, pp. 334–339, Mar. 2005.

Appendix A
Electrical Diagram

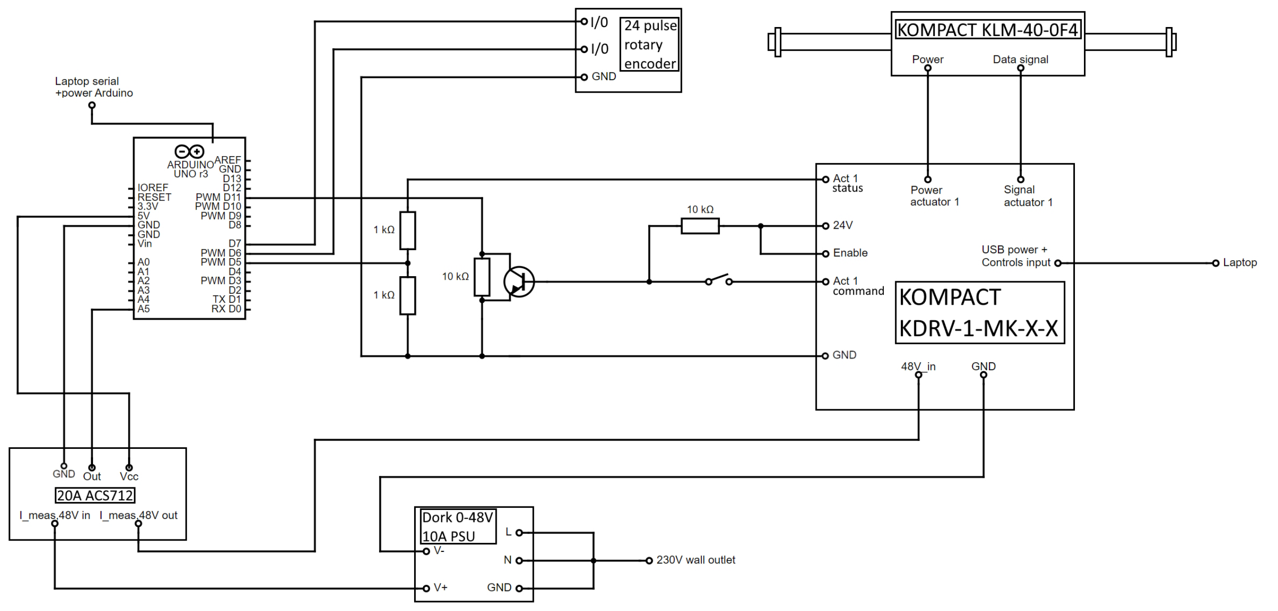


Figure 16: Setup actuated test circuit

Appendix B
Beam sizing

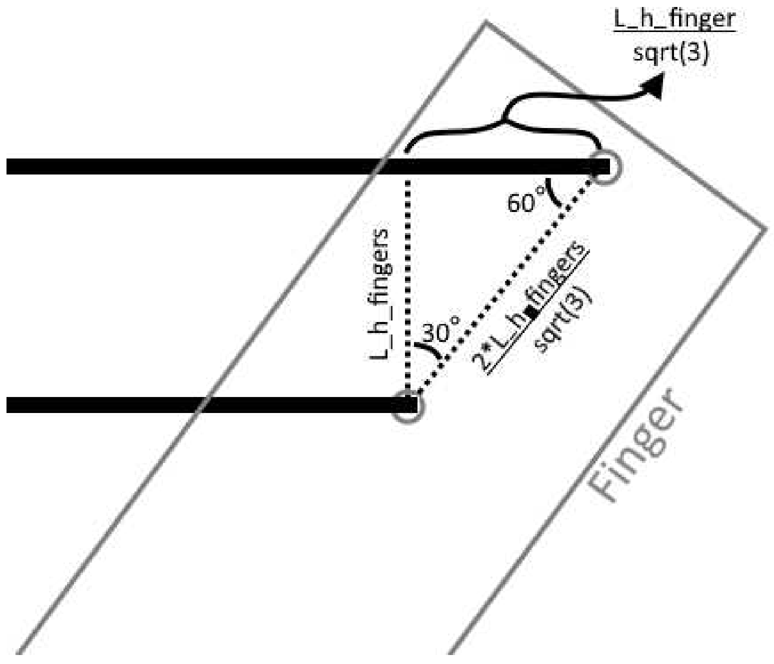
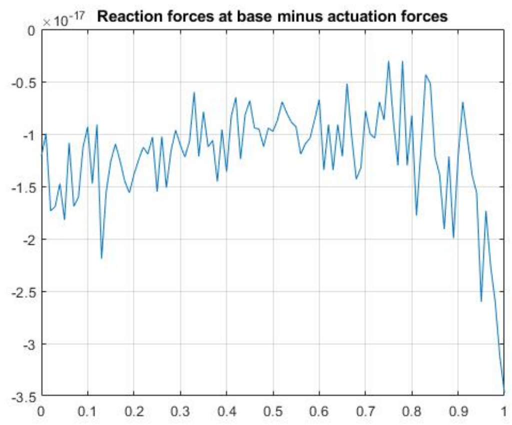
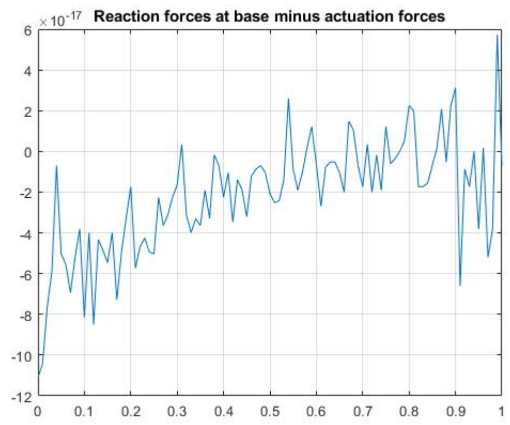


Figure 17: Ratio between finger height and length

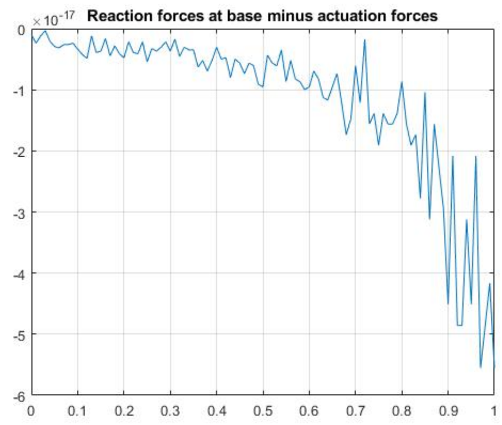
Appendix C
SPACAR simulation all cases



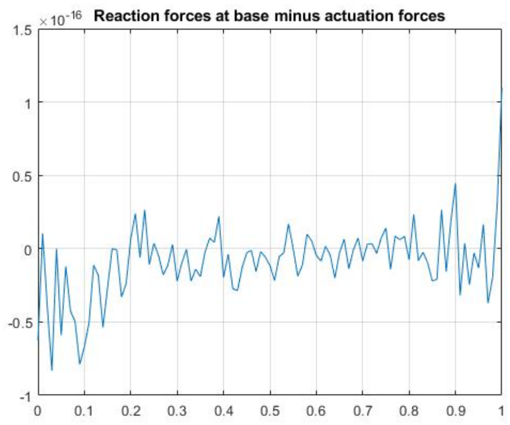
(a) Case 1: reaction forces when both sliders go from middle away from each other



(b) Case 2: reaction forces when lower slider goes from top to bottom



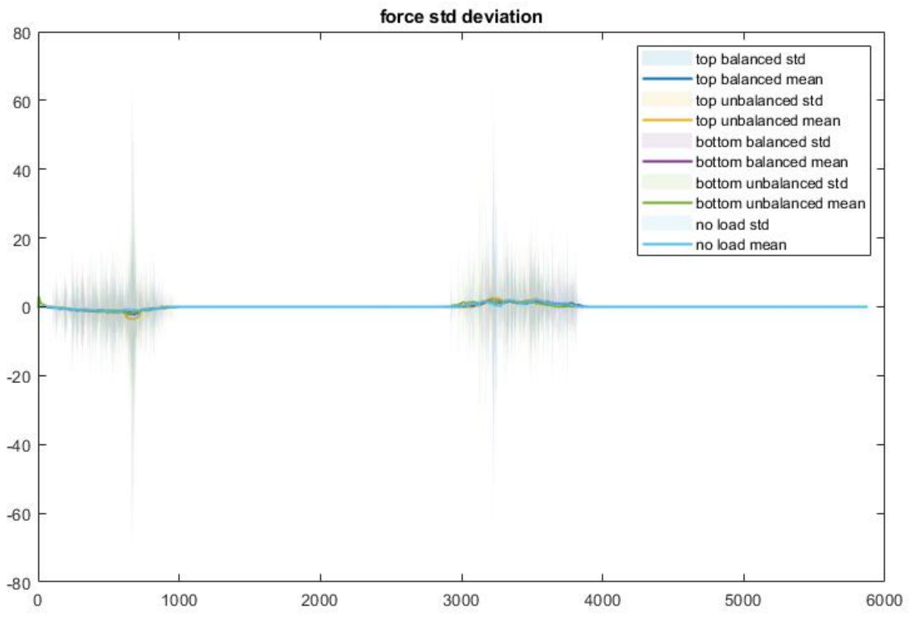
(c) Case 3: reaction forces when upper slider goes from bottom to top of range



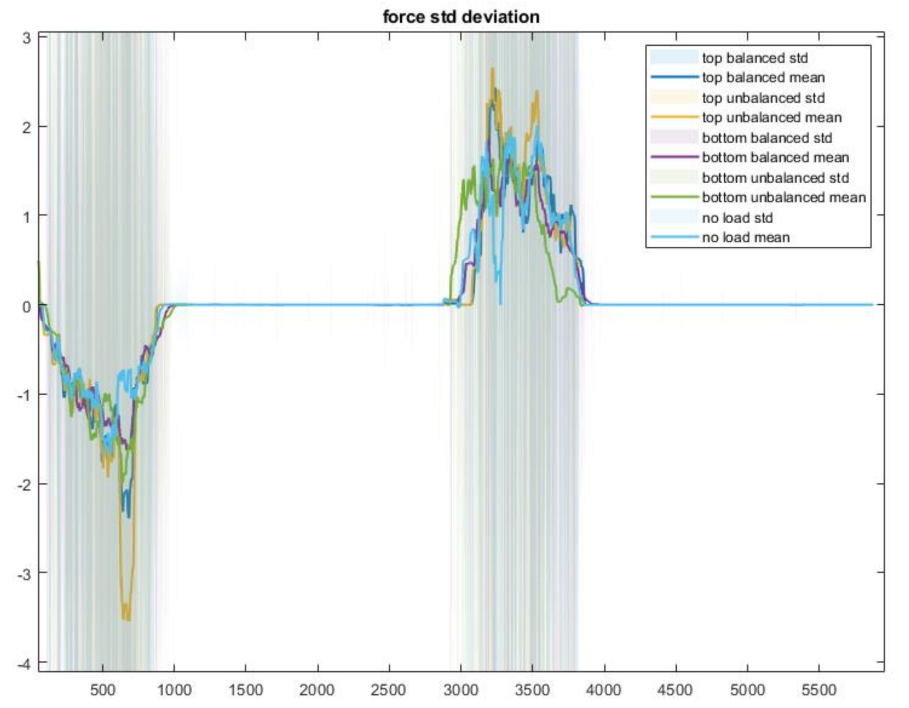
(d) Case 4: reaction forces when both sliders move from top to bottom together

Figure 18: SPACAR Reasction forces in different load cases

Appendix D
Actuated test All cases



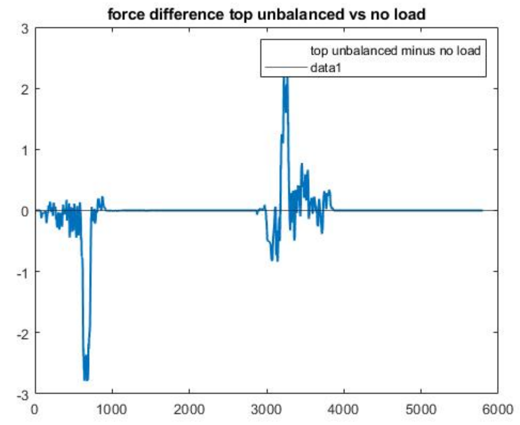
(a) Full measurements



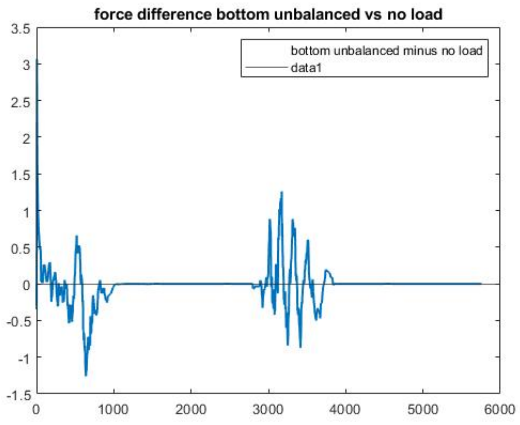
(b) Zoomed on to mean

Figure 19: Measured forces actuated test 5 different load cases

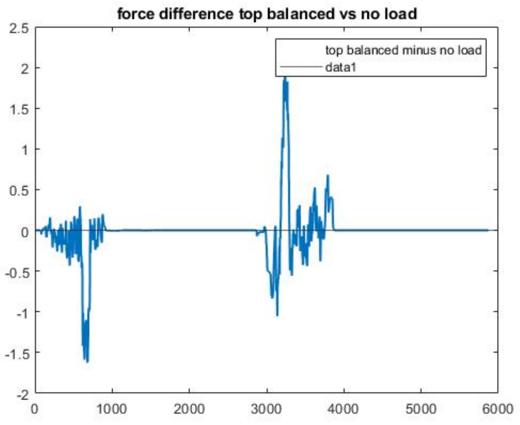
Appendix E
Actuated test force comparison



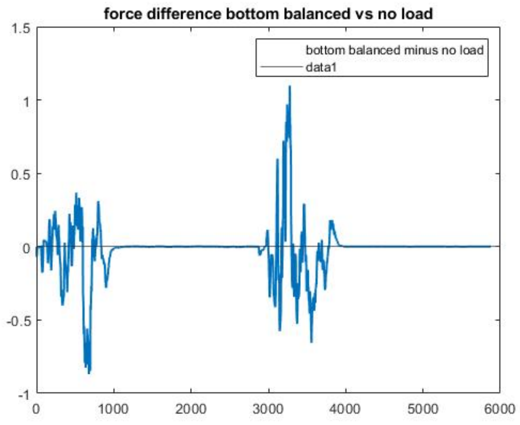
(a) Force top slider in unbalanced gripper minus force non loaded actuator



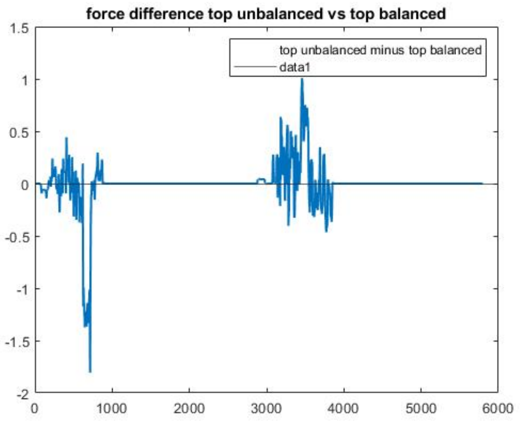
(b) Force bottom slider in unbalanced gripper minus force non loaded actuator



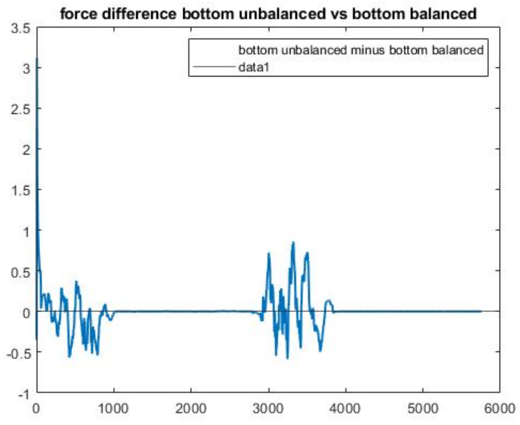
(c) Force top slider in balanced gripper minus force non loaded actuator



(d) Force bottom slider in balanced gripper minus force non loaded actuator



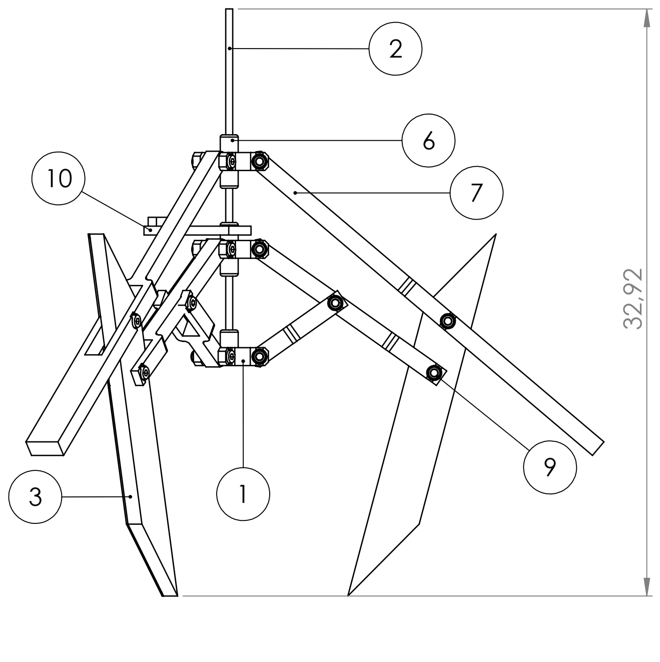
(e) Force top slider in unbalanced gripper minus force top slider in balanced gripper

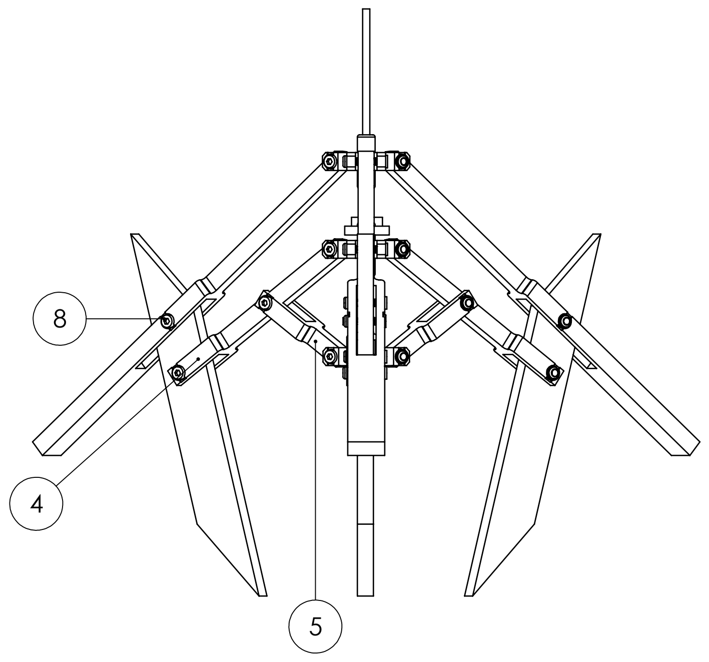


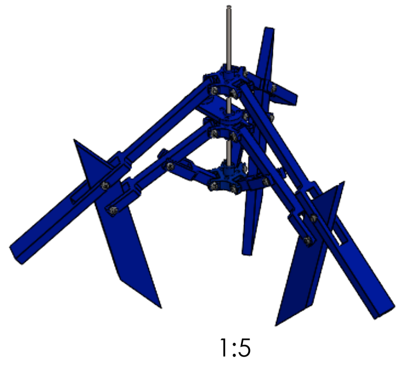
(f) Force bottom slider in unbalanced gripper minus force bottom slider in balanced gripper

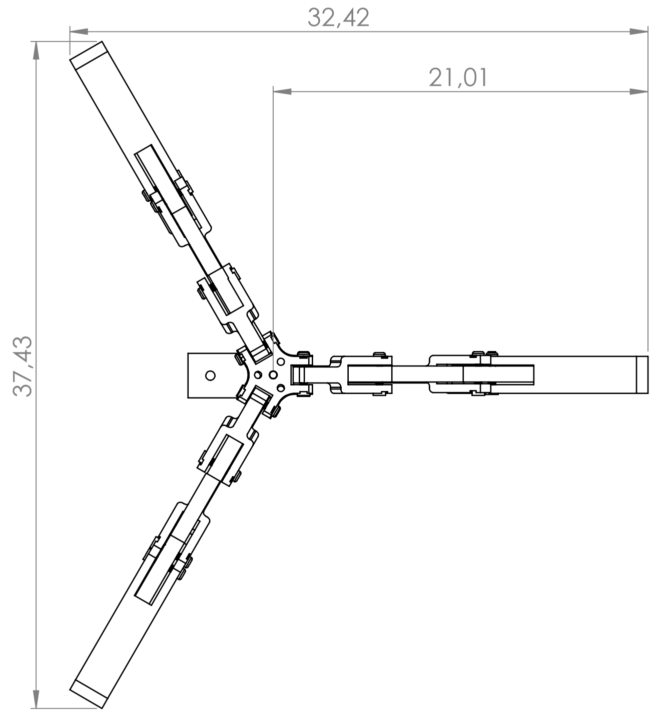
Figure 20: Mean actuation force comparison for different load cases

Appendix F
Technical drawings









ITEM NO.	PART NAME	QTY.
1	M4_wide_base	1
2	4 mm round steel bar	1
3	M4_wide_finger	3
4	M4_wide_Low_panto_L	3
5	M4_wide_Low_panto_s	3
6	M4_wide_slider	2
7	M4_wide_Upp_panto	3
8	M4x20 HEX CAP BOLT	18
9	M4 LOCK NUT	18
10	Act-slid_bracket	1

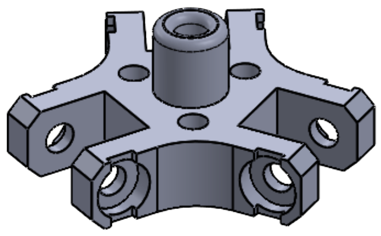
units mm	scale 1:3	quantity 1	date 21-7-2024	remark This is a generic overview drawing of the assembly with BOM, indicated measurements are there to inform general size of total assembly as these can change with motion of the assembly
material			mass gr	
author Rik Hofland - 4446070			group ME MSC HTE MSD	

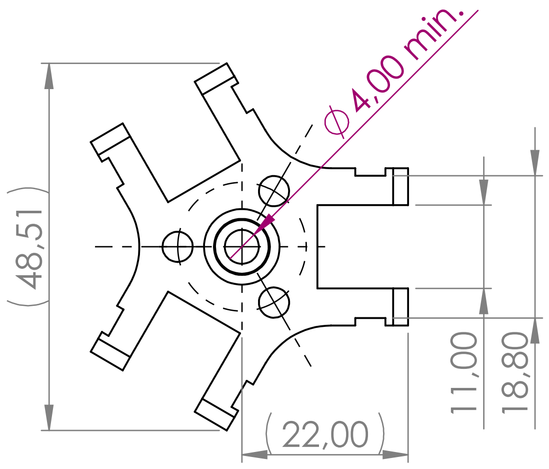
name ASSY_M4_wide_joint_balanced_2DoF

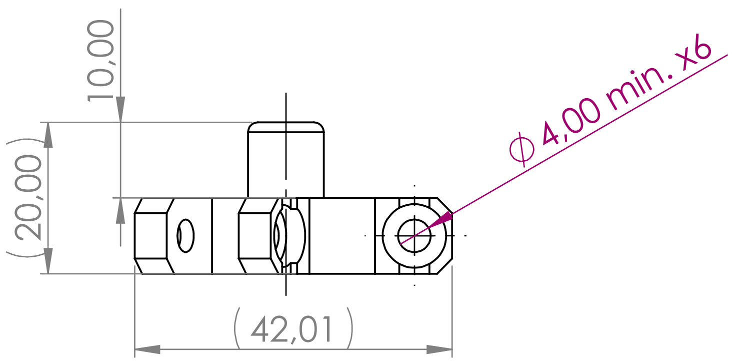
D:\Documenten\TU_Delft\Year_2\Solidworks\M4_size_wide_joint_balanced_2DoF\Drawings\

TU Delft Delft University of Technology

format	drawing no.
A3	1





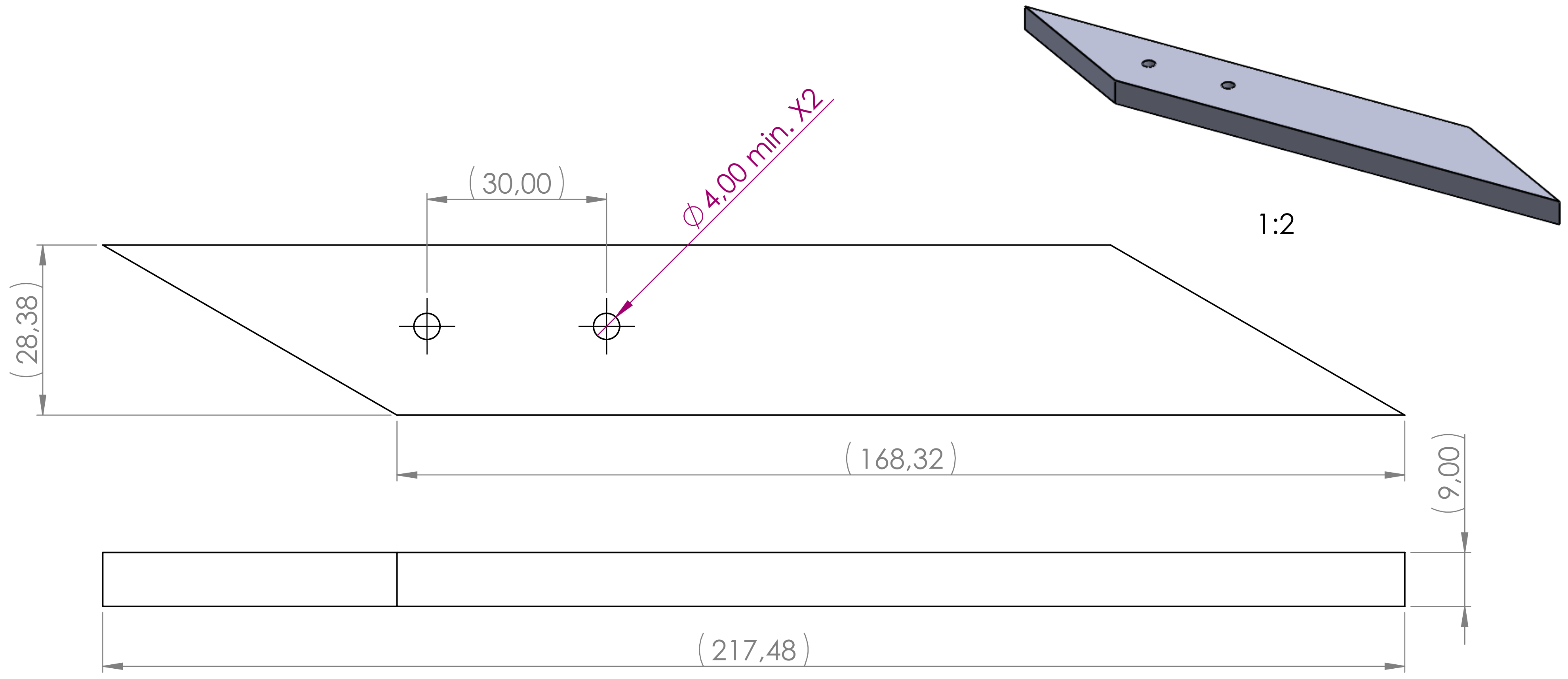


NOTE: Part is designed to be 3D printed. General tolerance is based on available machine capability

⌓	0.2	General Tolerance on the part unless otherwise specified

	units mm	scale 1:1	quantity 1	date 21-7-2024	remark
material				mass gr	
author	Rik Hofland - 4446070			group ME MSC HTE MSD	TU Delft Delft University of Technology
name	M4_wide_base		format A4	drawing no. 1	

D:\Documenten\TU_Delft\Year_2\Solidworks\M4_size_wide_joint_balanced_2DoF\Drawings\

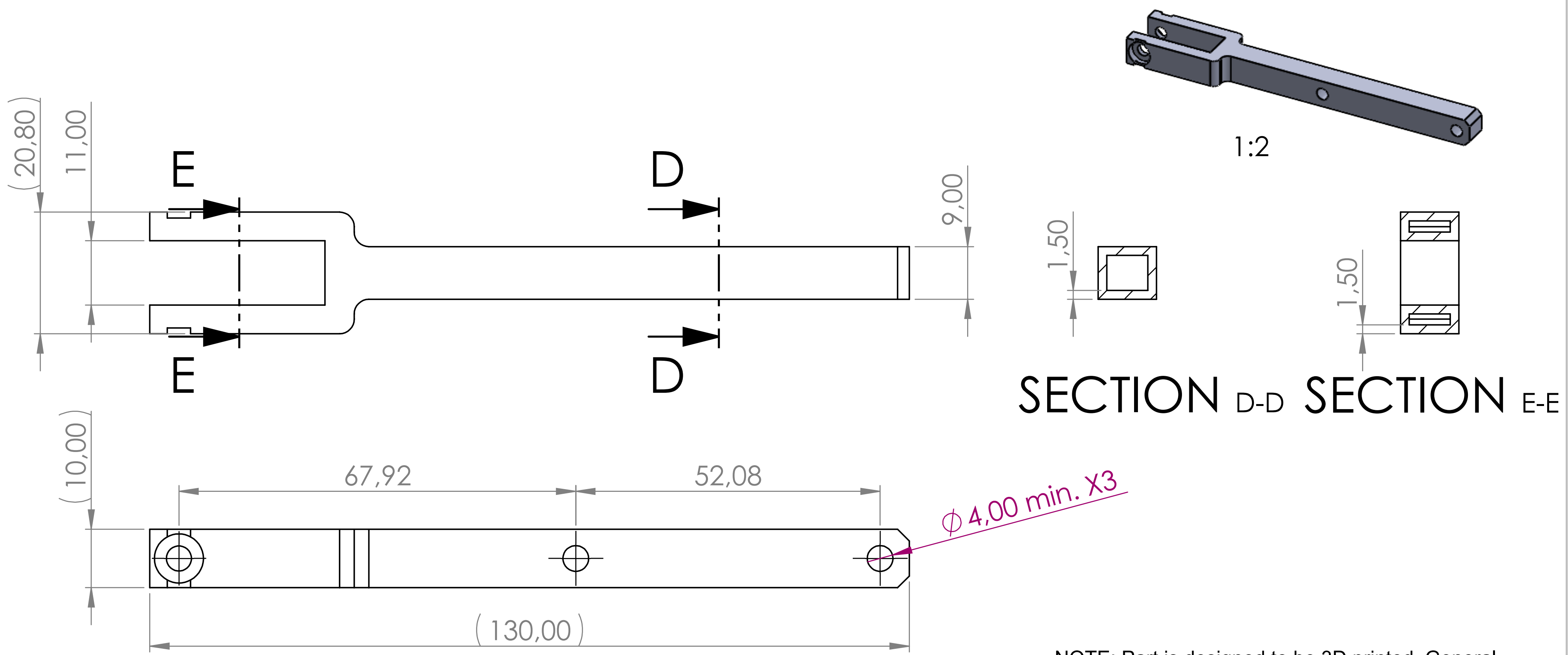


NOTE: Part is designed to be 3D printed. General tolerance is based on available machine capability

⌓	0.2

General Tolerance on the part unless otherwise specified

TU Delft Delft University of Technology	name M4_wide_finger D:\Documenten\TU_Delft\Year_2\Solidworks\M4_size_wide_joint_balanced_2DoF\Drawings\	units mm	scale 1:1	quantity 3	date 21-7-2024	remark <<remarks>>
		material			mass gr	
		author Rik Hofland - 4446070		group ME MSC HTE MSD	format A4	drawing no. 1

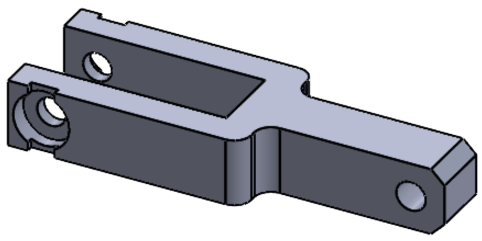


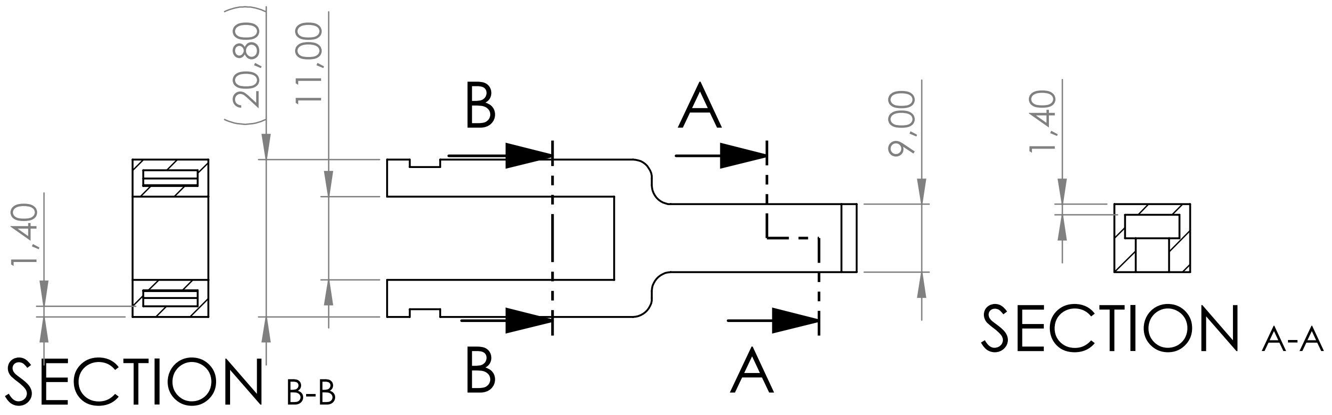
NOTE: Part is designed to be 3D printed. General tolerance is based on available machine capability

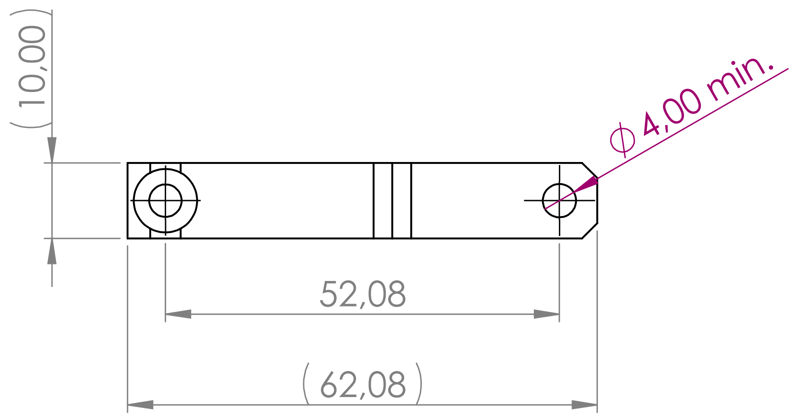
⌓	0.2

General Tolerance on the part unless otherwise specified

Delft University of Technology	name M4_wide_Low_panto_L		units mm	scale 1:1	quantity 1	date 21-7-2024	remark <<remarks>>
	D:\Documenten\TU_Delft\Year_2\Solidworks \M4_size_wide_joint_balanced_2DoF\Drawings\	material				mass gr	
		author Rik Hofland - 4446070			group ME MSC HTE MSD	format A4	drawing no. 1





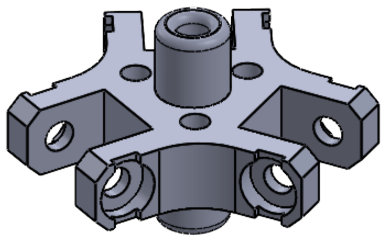


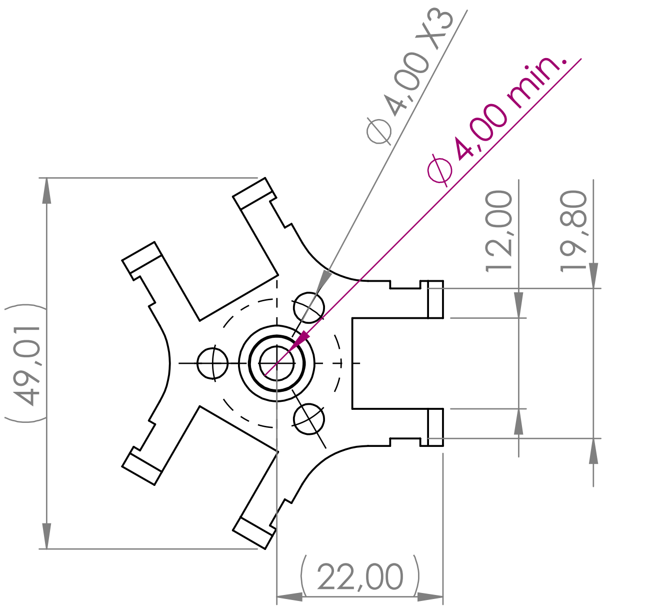
NOTE: Part is designed to be 3D printed. General tolerance is based on available machine capability

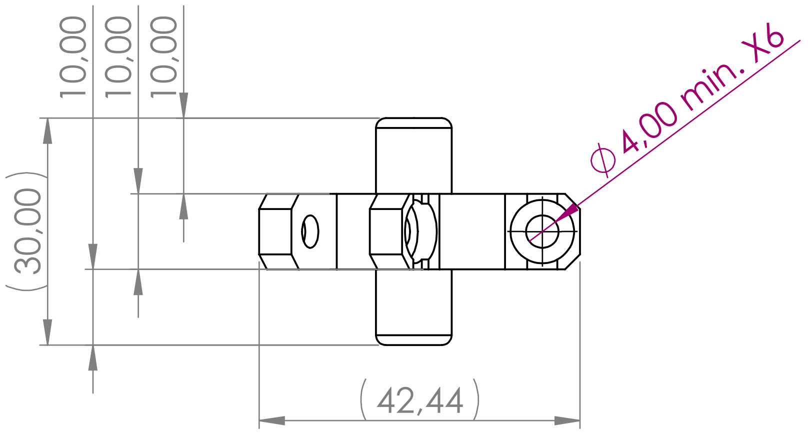
⌓	0.2	General Tolerance on the part unless otherwise specified

	units mm	scale 1:1	quantity 1	date 21-7-2024
remark <<remarks>>				
material			mass gr	
author Rik Hofland - 4446070			group ME MSC HTE MSD	TU Delft Delft University of Technology
name M4_wide_Low_panto_s			format A4	drawing no. 1

D:\Documenten\TU_Delft\Year_2\Solidworks\M4_size_wide_joint_balanced_2DoF\Drawings\



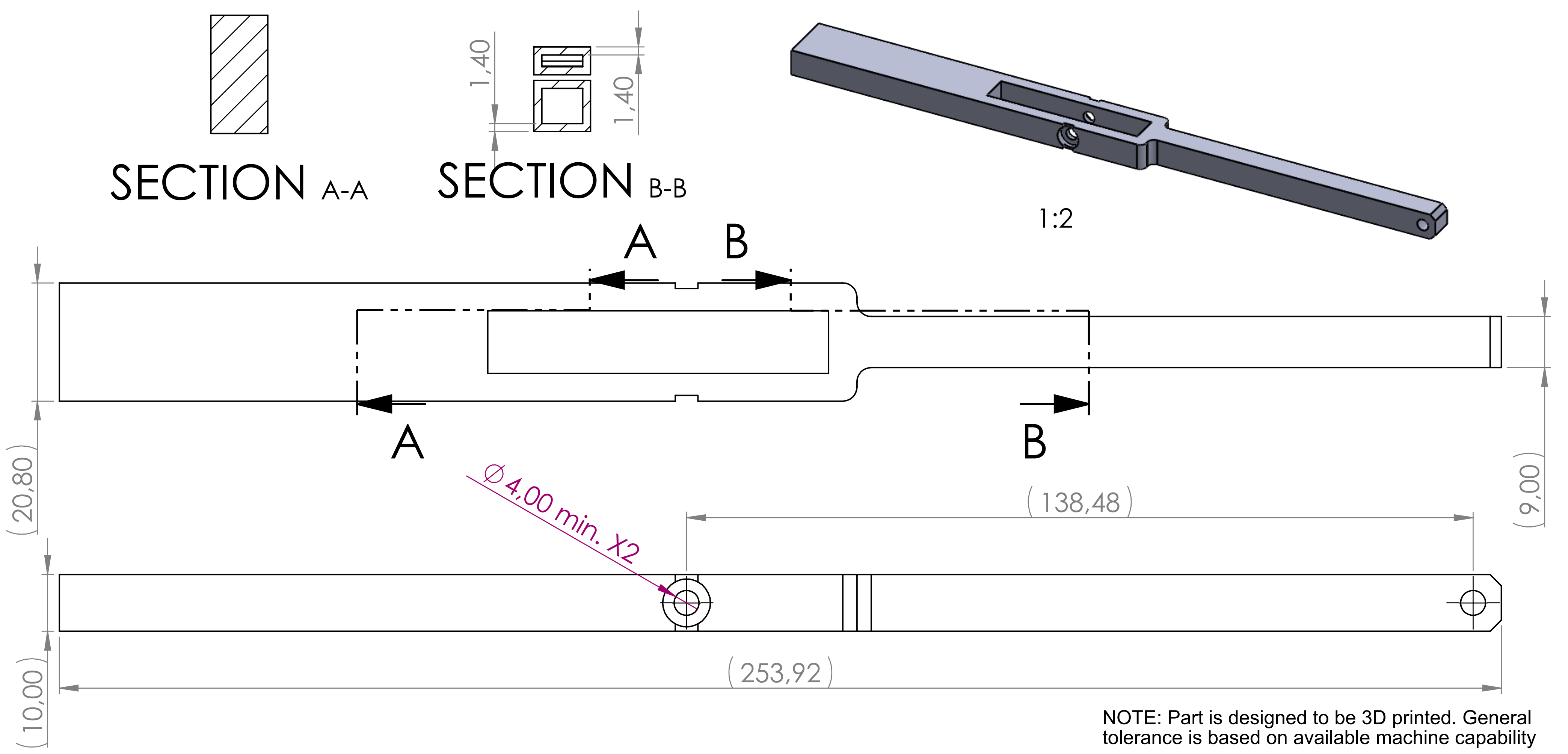




NOTE: Part is designed to be 3D printed. General tolerance is based on available machine capability

⌓	0.2	General Tolerance on the part unless otherwise specified

	units mm	scale 1:1	quantity 1	date 21-7-2024	remark <<remarks>>
material				mass gr	TU Delft Delft University of Technology
author Rik Hofland - 4446070				group ME MSC HTE MSD	
name M4_wide_slider				format A4	drawing no. 1



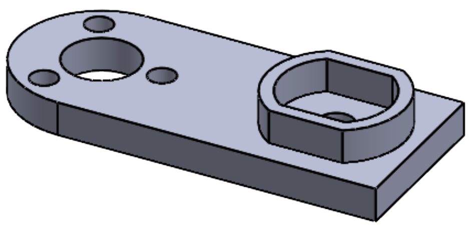
NOTE: Part is designed to be 3D printed. General tolerance is based on available machine capability

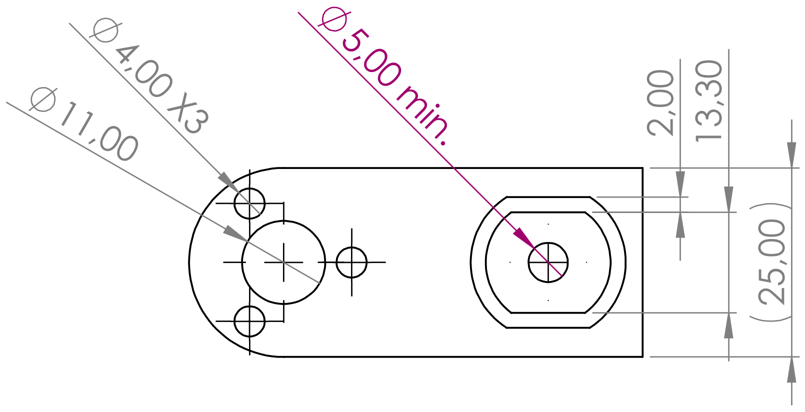
| ⌓ | 0.2 |

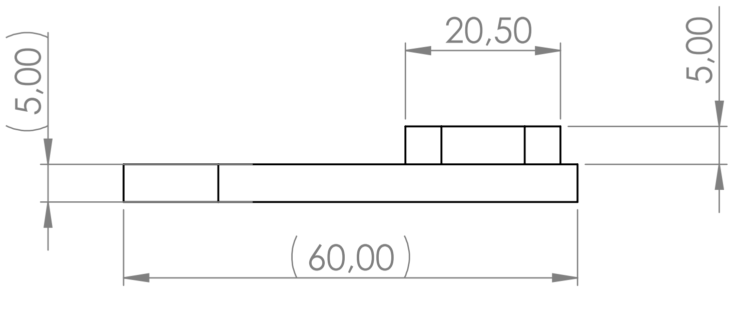
General Tolerance on the part unless otherwise specified

Delft University of Technology

name: M4_wide_Upp_panto	units: mm	scale: 1:1	quantity: 1	date: 21-7-2024	remark: <<remarks>>
D:\Documenten\TU_Delft\Year_2\Solidworks\M4_size_wide_joint_balanced_2DoF\Drawings\	material:			mass: gr	
	author: Rik Hofland - 4446070		group: ME MSC HTE MSD	format: A4	drawing no. 1







NOTE: Part is designed to be 3D printed. General tolerance is based on available machine capability

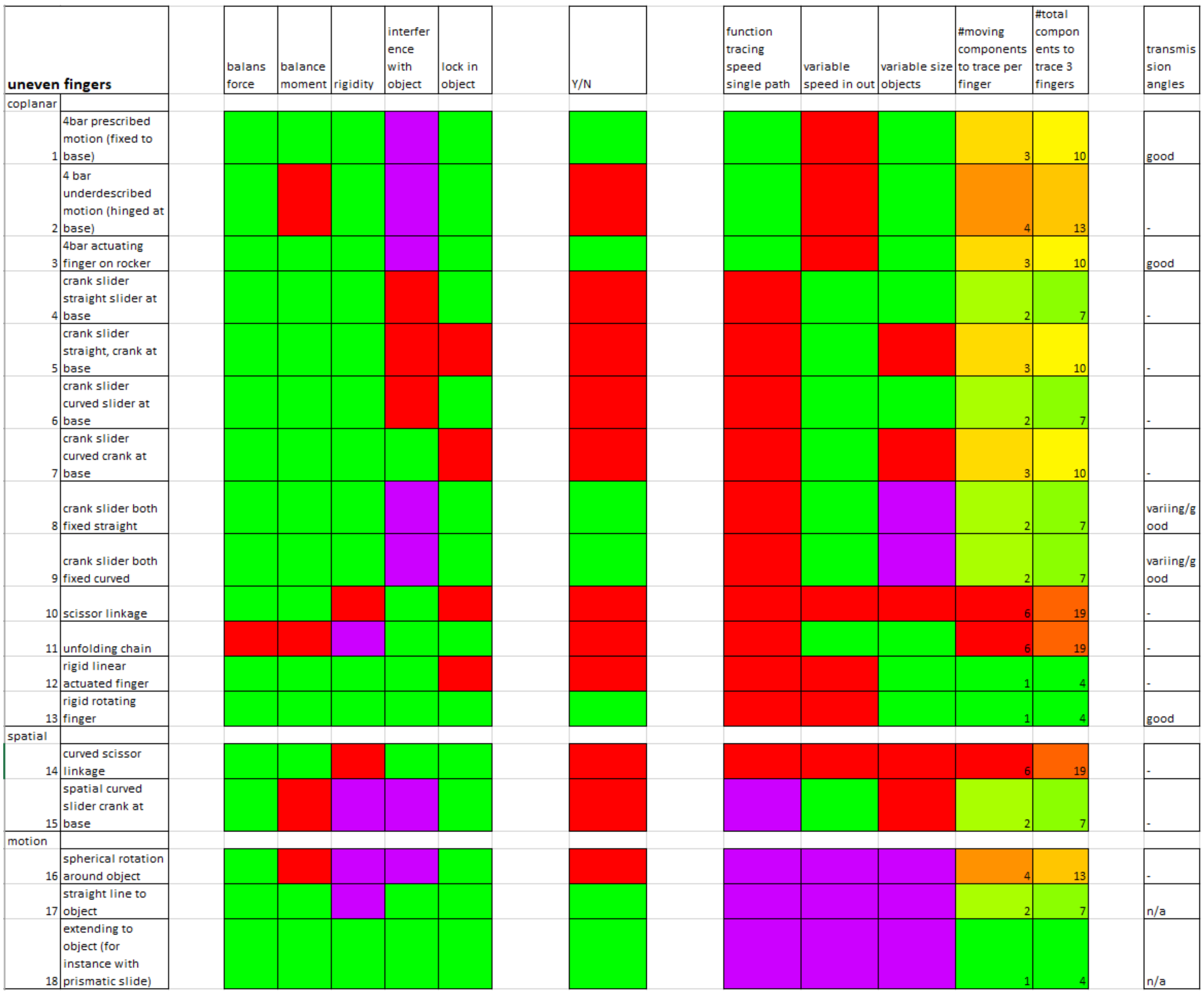
⌓	0.2	General Tolerance on the part unless otherwise specified

	units mm	scale 1:1	quantity 1	date 21-7-2024	remark <<remarks>>
material				mass gr	
author Rik Hofland - 4446070				group ME MSC HTE MSD	TU Delft Delft University of Technology
name Act-slid_bracket				format A4	drawing no. 1

D:\Documenten\TU_Delft\Year_2\Solidworks\M4_size_wide_joint_balanced_2DoF\Drawings\

Appendix G
Design Decisions

In order to come to a design for the gripper at first a comparison table was made to select suitable mechanisms for a balanced gripper design (Figure 21). In this table green is judged good, purple is depending on design, red is not possible or bad characteristic. The characteristics for both force and moment balance reflect if it would be possible to balance the mechanism. The criterion of rigidity is added because a gripper has to carry a load as well as its own mass hence a configuration that leads to a low rigidity when gripping an object is discarded. The column of interference with object refers to the motion the mechanism will make during the gripping movement. The griper can not move through the to be gripped object. The lock in object column describes if the mechanism will be able to create a geometric locking design or if it will give a design depending on friction only. These columns together already make a hard selection in the proposed solutions as can be seen in the Yes/No column.

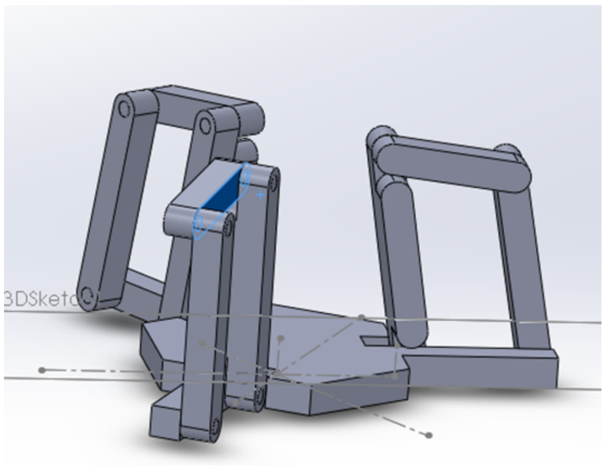


	uneven fingers	balans force	balance moment	rigidity	interference with object	lock in object	Y/N	function tracing speed single path	variable speed in out	variable size objects	#moving components to trace per finger	#total components to trace 3 fingers	transmission angles
	coplanar												
1	4bar prescribed motion (fixed to base)	🟩	🟩	🟩	🟪	🟩	🟩	🟩	🟥	🟩	3	10	good
2	4 bar underdescribed motion (hinged at base)	🟩	🟥	🟩	🟪	🟩	🟥	🟩	🟥	🟩	4	13	-
3	4bar actuating finger on rocker	🟩	🟩	🟩	🟪	🟩	🟩	🟩	🟥	🟩	3	10	good
4	crank slider straight slider at base	🟩	🟩	🟩	🟥	🟩	🟥	🟥	🟩	🟩	2	7	-
5	crank slider straight, crank at base	🟩	🟩	🟩	🟥	🟥	🟥	🟥	🟩	🟥	3	10	-
6	crank slider curved slider at base	🟩	🟩	🟩	🟥	🟩	🟥	🟥	🟩	🟩	2	7	-
7	crank slider curved crank at base	🟩	🟩	🟩	🟩	🟥	🟥	🟥	🟩	🟥	3	10	-
8	crank slider both fixed straight	🟩	🟩	🟩	🟪	🟩	🟩	🟥	🟩	🟪	2	7	variing/good
9	crank slider both fixed curved	🟩	🟩	🟩	🟪	🟩	🟩	🟥	🟩	🟪	2	7	variing/good
10	scissor linkage	🟩	🟩	🟥	🟩	🟥	🟥	🟥	🟥	🟥	6	19	-
11	unfolding chain	🟥	🟥	🟪	🟩	🟩	🟥	🟥	🟩	🟩	6	19	-
12	rigid linear actuated finger	🟩	🟩	🟩	🟩	🟥	🟥	🟥	🟥	🟩	1	4	-
13	rigid rotating finger	🟩	🟩	🟩	🟩	🟩	🟩	🟥	🟥	🟩	1	4	good
	spatial												
14	curved scissor linkage	🟩	🟩	🟥	🟩	🟩	🟥	🟥	🟥	🟥	6	19	-
15	spatial curved slider crank at base	🟩	🟥	🟪	🟪	🟩	🟥	🟪	🟩	🟥	2	7	-
	motion												
16	spherical rotation around object	🟩	🟥	🟪	🟪	🟩	🟥	🟪	🟪	🟪	4	13	-
17	straight line to object	🟩	🟩	🟪	🟩	🟩	🟩	🟪	🟪	🟪	2	7	n/a
18	extending to object (for instance with prismatic slide)	🟩	🟩	🟩	🟩	🟩	🟩	🟪	🟪	🟪	1	4	n/a

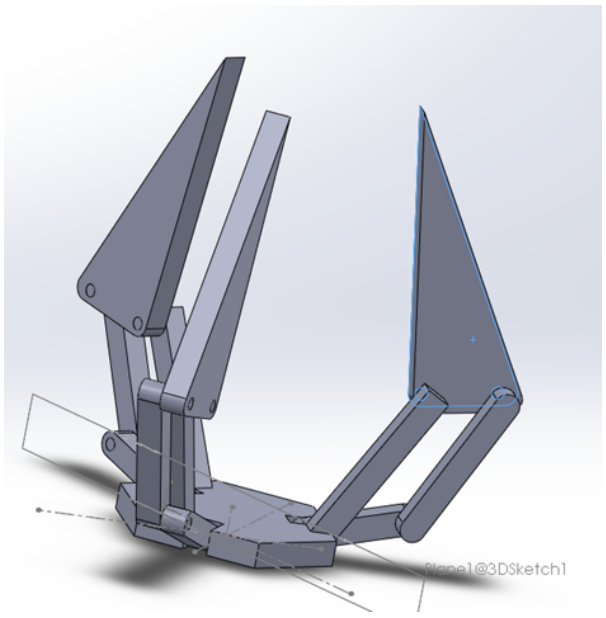
Figure 21: Evaluation of different mechanisms to use as base for gripper, in the table green is judged good, purple is depending on design, red is not possible or bad characteristic

In the second batch of columns a selection of other characteristics is evaluated per design with regards to extra features or complexities that are or can be added in a design with this type of mechanism. An example can be if the mechanism has a different movement velocity with the same actuations stroke in different parts of the stroke. This might be used to create a soft touch to the objects i.e. slowing down when closing in on the object and doing the fast bulk movement away from the object. The complexity of the design is gauged by the amount of components needed to make the mechanism work and balance it out.

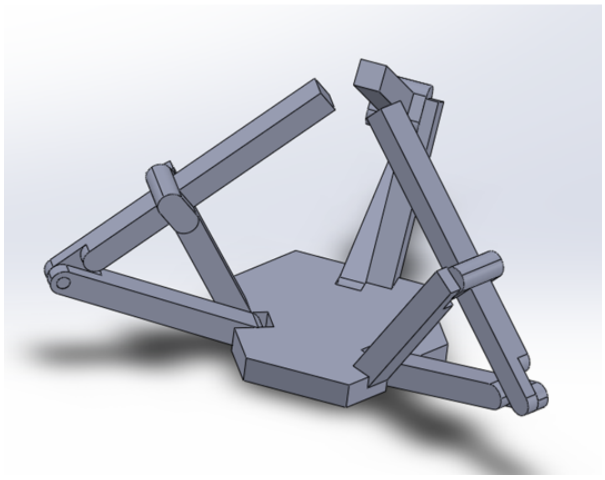
From this table four different mechanisms were chosen to work out further in CAD. The 4 bar fixed at base fully actuated Figure 22a, the 4 bar actuating finger on rocker Figure 22b, the crank slider both fixed relative to base Figure 22c, and the rigid rotating finger Figure 22d.



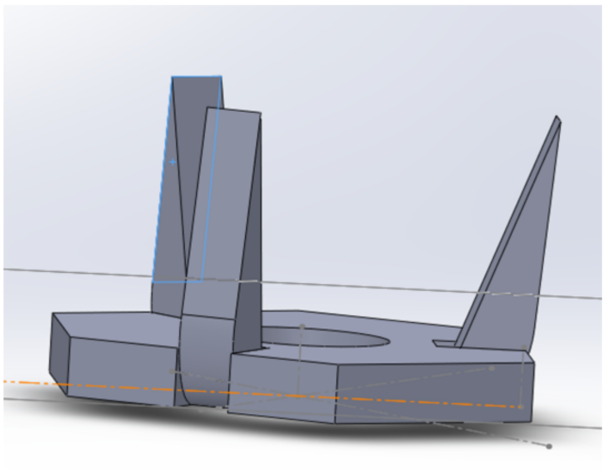
(a) 4 bar fixed at base fully actuated



(b) 4 bar actuating finger on rocker



(c) Crank slider both fixed relative to base

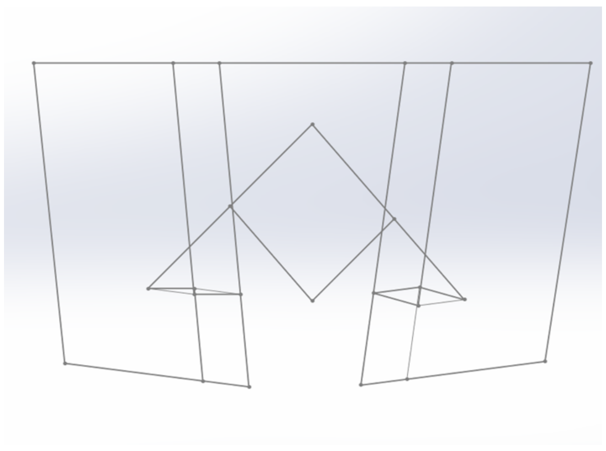


(d) Rigid rotating finger

Figure 22: 4 designs chosen from mechanism table

After the mechanisms were draw in CAD there was looked at adding mass tracing elements to the design to find the CoM of the mechanism where the base would have to be connected. In doing this it was clear that this design approach would lead to a very complex mechanism with a lot of elements (Figure 23).

A different approach was chosen where there was looked at the motion of the fingers. They were then connected to the base with inherent balanced mechanisms. In this phase again 4 different mechanism designs were generated where the fingers rotate around a fixed point Figure 24a, the finger only translates in horizontal direction by means of 2 linked pantographs Figure 24b, the fingers both rotate and translate attached to a single pantograph Figure 24c, and the fingers both rotate and translate attached to 2 independent pantographs Figure 24d.


(a) 4 bar fixed at base fully actuated with mass tracing elements 2D

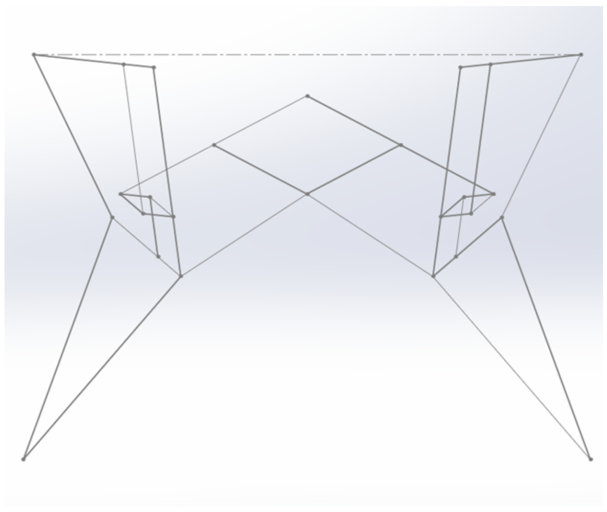
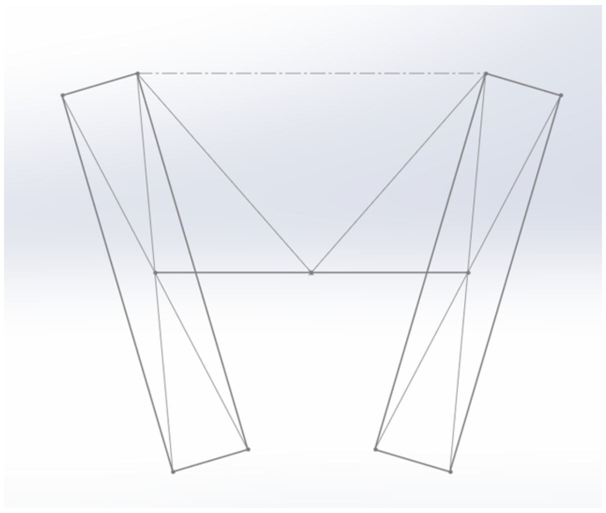
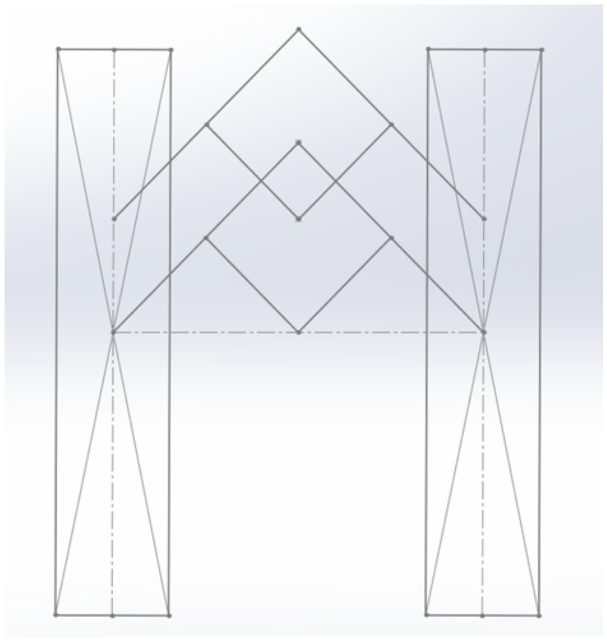
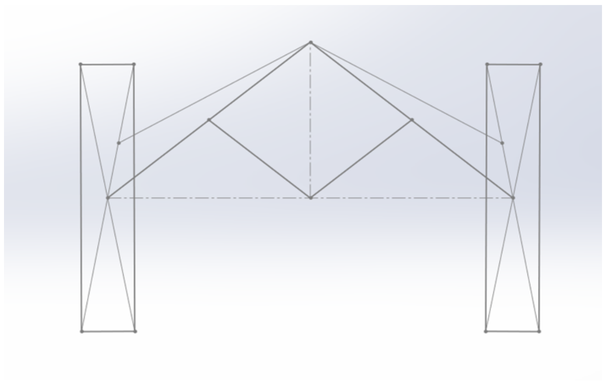

(b) 4 bar actuating finger on rocker with mass tracing elements 2D

Figure 23: Mass tracing elements in both selected 4 bar mechanisms


(a) 4 bar fixed at base fully actuated


(b) 4 bar actuating finger on rocker


(c) Crank slider both fixed relative to base

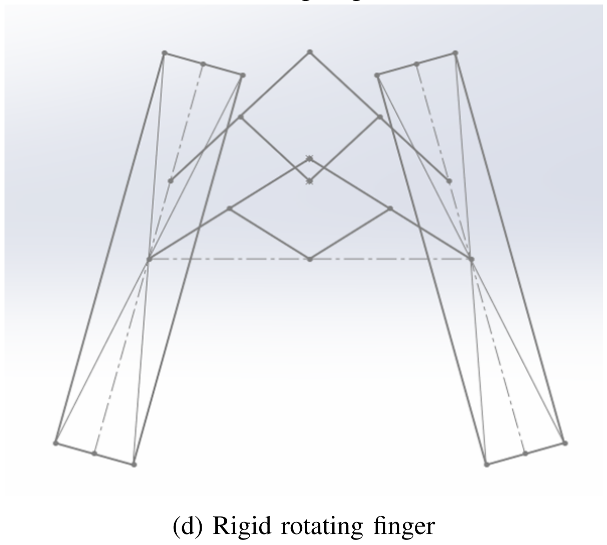

(d) Rigid rotating finger

Figure 24: 4 designs chosen from finger motion attached with inherent balanced elements

In comparing the four different designs it was seen that the gripper with only rotating fingers could have an ideal grip for only one size objects and in extending this concept to multi-fingered 3D designs it would get complex gearing systems to not have to rely on equal rotation from actuator control which is not mechanical balance. The translating gripper can grip different sized objects, but it can only do so by a pinch/friction grip and will never geometrically constrain the objects. The grippers combining two motions can both achieve the geometrical locking. However, the version with the single pantograph only has an ideal grip for 1 size objects where as the double pantograph version has a range of objects it can pick up with the ideal geometric grip. The counter advantage of the single pantograph gripper is its simplicity. In the end there was chosen to go for the size range rather than the simpler mechanism. This would lead to a larger array of possible use cases which was preferred because no specific case was specified and hence the gripper could not be optimised for such a case.

APPENDIX H
MATLAB CODE POSITION SIMULATION

```
clear all
close all
%{

phi1 =        0;
phi2 =    2.0349;
phi3 =    1.6873;
phi4 =    3.4481;
phi5 =    1.4543;
phi6 =        0;
phi7 =        0;
phi8 =   -2.0349;
phi9 =   -1.6873;
phi10 =  -3.4481;
phi11 =  -1.4543;
phid1 =       0;
phid2 =       0;
phid3 =       0;
phid4 =       0;
phid5 =       0;
phid6 =       0;
phid7 =       0;
phid8 =       0;
phid9 =       0;
phid10 =      0;
phid11 =      0;


%CoM's alle verschillende bodies
Cx1  = x1;
Cy1  = y1 - com1;
Cx2  = x2 -(l6/2) -com2*cos(phi2);
Cy2  = y2 -com2*sin(phi2) -y7;
Cx3  = x3 -(l6/2) -com3*cos(phi3);
Cy3  = y3 -com3*sin(phi3) -y6;
Cx4  = x4 -com4x*cos(phi4) -l3*cos(phi3) -(l6/2);
Cy4  = y4 -com4y*sin(phi4) -l3*sin(phi3) -y6;
Cx5  = x5 -(l5/2)*cos(phi5) -(l6/2);
Cy5  = y5 -(l5/2)*sin(phi5) - 5*10-3;
Cx6  = x6;
Cy6  = y6 -l_act1;
Cx7  = x7;
Cy7  = y7 -l_act2;
Cx8  = x8 -(l6/2) -com2*cos(phi8);
Cy8  = y8 -com2*sin(phi8) -y7;
Cx9  = x9 -(l6/2) -com3*cos(phi9);
Cy9  = y9 -com3*sin(phi9) -y6;
Cx10 = x10 -com4x*cos(phi10) -l9*cos(phi9) -(l6/2);
Cy10 = y10 -com4y*sin(phi10) -l9*sin(phi9) -y6;
Cx11 = x11 -(l11/2)*cos(phi11) -(l6/2);
Cy11 = y11 -(l11/2)*sin(phi11) - 5*10-3;

% EoM_Balanced_2DoF(phi2,phi3,phi4,phi5,phi8,phi9,phi10,phi11,phid2,phid3
    ,phid4,phid5,phid8,phid9,phid10,phid11)
%}
clc
%%
%masses kg
m1  = 12.70604*10^-3;
m2  = 51.75832*10^-3;
m3  = 34.19995*10^-3;
m4  = 101.17311*10^-3;
m5  = 18.74735*10^-3;
m6  = 3.63381*10^-3;
m7  = m6;
m8  = m2;
m9  = m3;
m10 = m4;
m11 = m5;

%inertias kg*m^2
I1  = 460636.56*10^-9;
I2  = 177590.40*10^-9;
I3  = 156700.05*10^-9;
I4  = 300239.04*10^-9;
I5  = 11764.35*10^-9;
I6  = 3371.28*10^-9;
I7  = I6;
I8  = I2;
I9  = I3;
I10 = I4;
I11 = I5;

%lengths m
l1 = 260*10^-3;
l2 = 2*92.32*10^-3;
l3 = 2*75*10^-3;
%l4 = 240*10^-3;
l4 = 2*30*10^-3;
l6 = 24*10^-3;
l7 = l6;
l8 = l2;
l9 = l3;
l10 = l4;

%specific values for balance calculations
m2_link = 14.030*10^-3; %weight of link 2 without counterweight
Com2_link = 127.5*10^-3; %CoM position of link 2 measured from pivot with
    4
CoM3_link = 107.5*10^-3;

```

```matlab

%%
%caltculate the length of counterweight at part 2
%x = ;
y_counter = 30*10^-3;
z_counter = 10*10^-3;
btwn_length = 40*10^-3;
square_stock = 10*10^-3;

M_26 = (m2_link/1000)*Com2_link+l2*(m6/1000)/2;%the devide by 1000 is to
    be able to compare volumes and not masses, other option is to multiply
     volumes by density for different densities
%side2 = 2*10*10*tussen*tussen/2+ 2*x*y*z*(x/2+tussen);
xcounter1 = -btwn_length +sqrt(btwn_length^2-4*0.5*-((M_26-(2*
    square_stock*square_stock*btwn_length*btwn_length/2))/(2*y_counter*
    z_counter)));
xcounter2 = -btwn_length -sqrt(btwn_length^2-4*0.5*-((M_26-(2*
    square_stock*square_stock*btwn_length*btwn_length/2))/(2*y_counter*
    z_counter)));
xcounter = max(xcounter1,xcounter2)
%bal = side1-side2;

%calculate the length of the finger
offset1 = 80*10^-3;
%x = 248.9602;
y_finger = 40*10^-3;
z_finger = 10*10^-3;

Upp_panto = ((m2/1000)+(m6/1000)/2)*offset1;%side1
%side2 = x*y*z*(x/2-5-offset1);

x_finger1 = (5*10^-3+offset1)+sqrt((5*10^-3+offset1)^2-4*0.5*-((Upp_panto
    )/(y_finger*z_finger)));
x_finger2 = (5*10^-3+offset1)-sqrt((5*10^-3+offset1)^2-4*0.5*-((Upp_panto
    )/(y_finger*z_finger)));
x_finger = max(x_finger1,x_finger2)
%bal = side1-side2

%%
%CoM specific lengths
l_com1 = 95.56*10^-3;
l_com2 = l2+(m7/2*l2)/m2;%141.64*10^-3;
l_com3 = (0*(m7/2)+CoM3_link*m3+(m2+m6/2+m4)*l3)/(m2+m6/2+m4+m3+m7/2);
% l_com4x = 0;%0.17*10^-3;
% l_com4y = 0;%0.02*10^-3;
l_com4 = (l4*(m2+m6/2))/m4;%from CoM of m2+m6/2 and m4 which needs to be
    at the foot of the pantograph
%dx4hole = 75.15*10^-3;
%dy4hole = 9.02*10^-3;
l_com8 = l_com2;%141.64*10^-3;
l_com9 = l_com3;

```

```
%calculate length of small pantograph arm for balance
l5 = (0*(m7/2)+l_com3*m3+(m2+m6/2+m4)*l3)/(m2+m4+m3+m8+m6+m10+m9+m7+m11/2)%107.5*10^-3;
l11 = l5;

% %m2+m6/2+m4+m3+m7/2
%
% lCoM3 = (0*(m7/2)+l_com3*m3+(m2+m6/2+m4)*l3)/(m2+m6/2+m4+m3+m7/2);
% a1 = (0*(m7/2)+l_com3*m3+(m2+m6/2+m4)*l3)/(m2+m4+m3+m8+m6+m10+m9+m7+m11/2);

%%
l_A1 = linspace(10*10^-3, 130*10^-3);
l_A2 = linspace(70*10^-3, 200*10^-3);

x = zeros(length(l_A1),11);
y = zeros(length(l_A1),11);
phi = zeros(length(l_A1),11);

for i = 1:length(l_A1)

    x(i,1)     = 0;
    y(i,1)     = l_com1;
    phi(i,1)   = 0;



    x(i,3)     = -l6/2-l_com3*sin(pi -acos(l_A1(i)/(2*l5)));
    y(i,3)     = l_A1(i) +l_com3*cos(pi-acos(l_A1(i)/(2*l5)));
    phi(i,3)   = pi-acos(l_A1(i)/(2*l5));

    li         = sqrt(l3^2+(l_A2(i)-l_A1(i))^2-2*l3*(l_A2(i)-l_A1(i))*cos(phi(i,3)));
    phi3li     = acos((-((l_A2(i)-l_A1(i))^2) + li^2 + l3^2) /(2*li*l3));
    phili4     = acos((-(l2^2) + li^2 + l4^2)/(2*li*l4));

    phi(i,4)   = phi(i,3)-pi + phi3li + phili4;
    x(i,4)     = -l6/2-l3*sin(phi(i,3))+l_com4*sin(phi(i,4));
    y(i,4)     = l_A1(i)+l3*cos(phi(i,3))-l_com4*cos(phi(i,4));
                                         %balance term
    phi(i,2)   = pi - atan(-((l6/2)+x(i,4)-(l_com4+l4)*sin(phi(i,4)))/(l_A2(i) - (y(i,4)+(l_com4+l4)*cos(phi(i,4)))));
    x(i,2)     = -l6/2-l_com2*sin(phi(i,2));
    y(i,2)     = l_A2(i)+l_com2*cos(phi(i,2));

    x(i,5)     = -l6/2 - (l5/2)*sin(acos(l_A1(i)/(2*l5)));
    y(i,5)     = l_A1(i)/4;
    phi(i,5)   = acos(l_A1(i)/(2*l5));
    x(i,6)     = 0;
    y(i,6)     = l_A1(i);
    phi(i,6)   = 0;
    x(i,7)     = 0;
    y(i,7)     = l_A2(i);
    phi(i,7)   = 0;

    x(i,9)     = l6/2-l_com9*sin(-pi + acos(l_A1(i)/(2*l11)));
    y(i,9)     = l_A1(i) +l_com9*cos(-pi+acos(l_A1(i)/(2*l11)));
    phi(i,9)   = -pi + acos(l_A1(i)/(2*l11));


    li2        = sqrt(l9^2+(l_A2(i)-l_A1(i))^2-2*l9*(l_A2(i)-l_A1(i))*cos(
       phi(i,9)));
    phi9li     = acos((-((l_A2(i)-l_A1(i))^2) + li2^2 + l9^2) /(2*li2*l9))
       ;
    phili10    = acos((-(l8^2) + li2^2 + l10^2)/(2*li2*l10));

    phi(i,10) = phi(i,9)+pi - phi9li -phili10;
    x(i,10)    = l6/2-l9*sin((phi(i,9)))+l_com4*sin(phi(i,10));
    y(i,10)    = l_A1(i)+l9*cos(phi(i,9))-l_com4*cos(phi(i,10));

    phi(i,8)   = -pi+atan(((-l6/2)+x(i,10)-(l10+l_com4)*sin(phi(i,10)))/(
       l_A2(i) - (y(i,10)+(l10+l_com4)*cos(phi(i,10)))));
    x(i,8)     = l6/2-l_com8*sin(phi(i,8));
    y(i,8)     = l_A2(i)+l_com8*cos(phi(i,8));

    x(i,11)    = l6/2 + (l11/2)*sin(acos(l_A1(i)/(2*l11)));
    y(i,11)    = l_A1(i)/4;
    phi(i,11) = -acos(l_A1(i)/(2*l11));
end

%
% dl = zeros(100,1);
% for i = 1:100
% dl(i) = sqrt((x(i,10)+l10*sin(phi(i,10))-l6/2)^2+(l_A2(i) - (y(i,10)+
   l10*cos(phi(i,10))))^2);
% end
%



%%
% close all
% plot([x(1,3)+l3/2*sin(phi(1,3))    x(1,3)-l3/2*sin(phi(1,3))]   ,[y
    (1,3)-l3/2*cos(phi(1,3))    y(1,3)+l3/2*cos(phi(1,3))],'LineWidth',5)
% hold on
% plot([x(1,2)+l2/2*sin(phi(1,2))    x(1,3)-l2/2*sin(phi(1,2))]   ,[y
    (1,2)+l2/2*cos(phi(1,2))    y(1,2)-l2/2*cos(phi(1,2))],'LineWidth',5)
% plot([x(1,5)+l5/2*sin(phi(1,5))    x(1,5)-l5/2*sin(phi(1,5))]   ,[y
    (1,5)-l5/2*cos(phi(1,5))    y(1,5)+l5/2*cos(phi(1,5))],'LineWidth',5)
% plot([x(1,8)+l8/2*sin(phi(1,8))    x(1,8)-l8/2*sin(phi(1,8))]   ,[y
    (1,8)+l8/2*cos(phi(1,8))    y(1,8)-l8/2*cos(phi(1,8))],'LineWidth',5)
% plot([x(1,9)+l9/2*sin(phi(1,9))    x(1,9)-l9/2*sin(phi(1,9))]   ,[y
    (1,9)-l9/2*cos(phi(1,9))    y(1,9)+l9/2*cos(phi(1,9))],'LineWidth',5)
% plot([x(1,11)+l11/2*sin(phi(1,11)) x(1,11)-l11/2*sin(phi(1,11))],[y
    (1,11)-l11/2*cos(phi(1,11)) y(1,11)+l11/2*cos(phi(1,11))],'LineWidth
',5)
% plot([x(1,6)+l6/2                    x(1,6)-l6/2]             ,[y(1,6)                   y(1,6)],'LineWidth',5)
% plot([x(1,7)+l7/2                    x(1,7)-l7/2]             ,[y(1,7)                   y(1,7)],'LineWidth',5)
% plot([x(1,4)                         x(1,4)+l4*sin(phi(1,4))]     ,[y(1,4)                   y(1,4)+l4*cos(phi(1,4))],'LineWidth',5)
% plot([x(1,10)                        x(1,10)+l10*sin(phi(1,10))]  ,[y(1,10)                  y(1,10)+l10*cos(phi(1,10))],'LineWidth',5)
% plot([x(1,1)                         x(1,1)]                  ,[y(1,1)-l_com1             y(1,1)+l1-l_com1],'LineWidth',5)
% legend('3','2','5','8','9','11','6','7','4','10','1')
% for i = 1:11
% plot(x(1,i), y(1,i),'*')
% end

figure(1)

for j = 5:5:length(l_A1)
        plot([x(j,3)+l_com3*sin(phi(j,3))            x(j,3)-(l3-l_com3)*sin(phi(j,3))]   ,[y(j,3)-l_com3*cos(phi(j,3))           y(j,3)+(l3-l_com3)*cos(phi(j,3))],'LineWidth',5)
        hold on
        title(['step:',num2str(j)])
        plot([x(j,2)+l_com2*sin(phi(j,2))      x(j,2)-(l2-l_com2)*sin(phi(j,2))] ,[y(j,2)-l_com2*cos(phi(j,2))     y(j,2)+(l2-l_com2)*cos(phi(j,2))],'LineWidth',5)
        plot([x(j,5)+l5/2*sin(phi(j,5))            x(j,5)-l5/2*sin(phi(j,5))]    ,[y(j,5)-l5/2*cos(phi(j,5))           y(j,5)+l5/2*cos(phi(j,5))],'LineWidth',5)
        plot([x(j,8)+l_com8*sin(phi(j,8))      x(j,8)-(l8-l_com8)*sin(phi(j,8))] ,[y(j,8)-l_com8*cos(phi(j,8))     y(j,8)+(l8-l_com8)*cos(phi(j,8))],'LineWidth',5)
        plot([x(j,9)+l_com9*sin(phi(j,9))            x(j,9)-(l9-l_com9)*sin(phi(j,9))]   ,[y(j,9)-l_com9*cos(phi(j,9))           y(j,9)+(l9-l_com9)*cos(phi(j,9))],'LineWidth',5)
        plot([x(j,11)+l11/2*sin(phi(j,11))         x(j,11)-l11/2*sin(phi(j,11))],[y(j,11)-l11/2*cos(phi(j,11))         y(j,11)+l11/2*cos(phi(j,11))],'LineWidth',5)
        plot([x(j,6)+l6/2                          x(j,6)-l6/2]                         ,[y(j,6)                               y(j,6)],'LineWidth',5)
        plot([x(j,7)+l7/2                          x(j,7)-l7/2]                         ,[y(j,7)                               y(j,7)],'LineWidth',5)
        plot([x(j,4)                               x(j,4)-(l4+l_com4)*sin(phi(j,4))] ,[y(j,4)                               y(j,4)+(l4+l_com4)*cos(phi(j,4))],'LineWidth',5)
        plot([x(j,10)                              x(j,10)-(l10+l_com4)*sin(phi(j,10))]  ,[y(j,10)                              y(j,10)+(l10+l_com4)*cos(phi(j,10))],'LineWidth',5)
        plot([x(j,1)                                    x(j,1)]                          ,[y(j,1)-l_com1                          y(j,1)+l1-l_com1],'LineWidth',5)
        legend('3','2','5','8','9','11','6','7','4','10','1')
        xlim([-300,300]*10^-3)
        ylim([-100,300]*10^-3)
        set(gcf,'Position',[100,100,1000,700])
        pause(0.000001)
        hold off
end
for i = 1:11
    hold on
    plot(x(length(l_A1),i), y(length(l_A1),i),'*')
end

%%
M = [m2 m3 m4 m5 m6 m7 m8 m9 m10 m11].';%[0 0 0 m4 0 0 0 0 0 m10 0].';%%1 weg gelaten want die beweegt niet
CoM_x = zeros(length(l_A1),1);
CoM_y = zeros(length(l_A1),1);

for j = 1:length(l_A1)
CoM_x(j) = x(j,2:11)*M/sum(M);
CoM_y(j) = y(j,2:11)*M/sum(M);
end
figure
plot(1:length(l_A1),CoM_x);
title('change in CoM position x-direction')


figure
plot(1:length(l_A1),CoM_y);
title('change in CoM position y-direction')
% ylim([-10,10]*10^-3)
%%
%
% t = 20*10^-2;
% dt = t/length(l_A1);
%
% v_x = zeros(length(l_A1),11);
% a_x = zeros(length(l_A1),11);
% v_y = zeros(length(l_A1),11);
% a_y = zeros(length(l_A1),11);
% v_phi = zeros(length(l_A1),11);
% a_phi = zeros(length(l_A1),11);
%
%
% for n = 2:100
%     v_x(n,:) = (x(n,:)-x(n-1,:))/dt;
%     v_y(n,:) = (y(n,:)-y(n-1,:))/dt;
%     v_phi(n,:) = (phi(n,:)-phi(n-1,:))/dt;
% end
% for n = 3:100
%     a_x(n,:)  = (x(n,:)-2*x(n-1,:)+x(n-2,:))/dt^2;
%     a_y(n,:)  = (y(n,:)-2*y(n-1,:)+y(n-2,:))/dt^2;
%     a_phi(n,:) = (phi(n,:)-2*phi(n-1,:)+phi(n-2,:))/dt^2;
% end
%
% I = [I2 I3 I4 I5 I6 I7 I8 I9 I10 I11];%1 weg gelaten want die beweegt
   niet
%
% F_x = a_x.*M.';
% F_y = a_y.*M.';
% F_phi = a_phi.*I;
%
% figure
% subplot(3,1,1)
% plot(1:length(l_AB),v_x)
% title('x velocities of bodies')
% subplot(3,1,2)
% plot(1:length(l_AB),v_y)
% title('y velocities of bodies')
% subplot(3,1,3)
% plot(1:length(l_AB),v_phi)
% title('moments velocities of bodies')
% legend('1','2','3','4','5','6','7','8','9','10','11')
%
% figure
% subplot(3,1,1)
% plot(1:length(l_A1),F_x)
% title('x Forces on bodies')
% subplot(3,1,2)
% plot(1:length(l_A1),F_y)
% title('y Forces on bodies')
% subplot(3,1,3)
% plot(1:length(l_A1),F_phi)
% title('moments on bodies')
% legend('1','2','3','4','5','6','7','8','9','10','11')
%
% figure
% subplot(3,1,1)
% plot(1:length(l_A1),a_x)
% title('x accelerations of bodies')
% subplot(3,1,2)
% plot(1:length(l_A1),a_y)
% title('y accelerations of bodies')
% subplot(3,1,3)
% plot(1:length(l_A1),a_phi)
% title('rotation accelerations of bodies')
% legend('1','2','3','4','5','6','7','8','9','10','11')
%
```

Appendix I
Matlab code SPACAR simulation

```
%% 2 DOF basic pantograph
clear all
close all

addpath('D:\Documenten\TU_Delft\Year_2\Spacar')

%% known parameters
%masses kg
m1   =  1*10^-3;
%m2   =  20.14787030*10^-3;
m3   =  9.28964478*10^-3;
% m4   = 20.01308796967819*10^-3;%.92010620*10^-3;
m5   =  4.90762524*10^-3;
m6   =  9.36522987*10^-3;
m7   = m6;
%m8   = m2;
m9   = m3;
% m10 = m4;
m11 = m5;
m_joint = 3.778*10^-3;

%lengths m
l1 = 200*10^-3;
l2 = 138.48*10^-3;
l3 = 120*10^-3;
%l4 = 240*10^-3;
l4 = 30*10^-3;
l6 = 15*10^-3;
l7 = l6;
l8 = l2;
l9 = l3;
l10 = l4;
l_j2e = 5*10^-3;

%specific values for balance calculations
m2_link = 9.76369795*10^-3; %weight of link 2 without counterweight
CoM2_link = l2-78.95074887*10^-3-l_j2e; %CoM position of link 2 measured
   from pivot with 4
CoM3_link = l3-69.13293431*10^-3-l_j2e;
CoM5_link = 31.85564927*10^-3-l_j2e;


%material
density = 1240;

n_arms = 3; %number of arms/fingers

%% Conversion from without to with joints
m5j = m5+m_joint;
```

```matlab
m11j = m5j;

m6j = m6+n_arms*m_joint;
m7j = m6j;

CoM3_link_j = (m3*CoM3_link+l3*m_joint)/(m3+m_joint);
CoM9_link_j = CoM3_link_j;
m3j = m3+m_joint;
m9j = m3j;


%% kinematics
%caltculate the length of counterweight at part 2
%x = ;
y_counter = 20.8*10^-3;
z_counter = 10*10^-3;
btwn_length = 30*10^-3;
square_stock_1 = 10*10^-3;
square_stock_2 = 4.9*10^-3;

M_26 = (m2_link/density)*CoM2_link+l2*((m6j)/density)/n_arms;%the devide
   by 1000 is to be able to compare volumes and not masses, other option
   is to multiply volumes by density for different densities
%side2 = 2*10*10*tussen*tussen/2+ 2*x*y*z*(x/2+tussen);
x_counter1 = -(l_j2e+btwn_length)+sqrt((l_j2e+btwn_length)^2-4*(1/2)*-((
   l2*m7j/n_arms + m2_link*CoM2_link - 2*btwn_length*square_stock_1*
   square_stock_2*density*(l_j2e+btwn_length/2))/(y_counter*z_counter*
   density)));
x_counter2 = -(l_j2e+btwn_length)-sqrt((l_j2e+btwn_length)^2-4*(1/2)*-((
   l2*m7j/n_arms + m2_link*CoM2_link - 2*btwn_length*square_stock_1*
   square_stock_2*density*(l_j2e+btwn_length/2))/(y_counter*z_counter*
   density)));
x_counter = max(x_counter1,x_counter2);
%bal = side1-side2;
check_upp_bal = m2_link*CoM2_link+m7j/n_arms*l2-(l_j2e+btwn_length/2)*2*
   btwn_length*square_stock_1*square_stock_2*density-(l_j2e+btwn_length+
   x_counter/2)*x_counter*y_counter*z_counter*density;

m2 = m2_link+2*btwn_length*square_stock_1*square_stock_2*density+
   x_counter*y_counter*z_counter*density;
m8 = m2;
%calculate the length of the finger
y_finger = 20*10^-3;
z_finger = 9*10^-3;

% Upp_panto = ((m2/density)+(m7j/2/density)+(m_joint/density))*l4;%side1
%side2 = x*y*z*(x/2-5-l4);


x_finger1 = (l4+l_j2e)+sqrt((l4+l_j2e)^2-4*(1/2)*-(((m2+m7j/n_arms+
   m_joint)*l4)/(density*y_finger*z_finger)));
```

```matlab
x_finger2 = (l4+l_j2e)-sqrt((l4+l_j2e)^2-4*(1/2)*-(((m2+m7j/n_arms+
    m_joint)*l4)/(density*y_finger*z_finger)));


% x_finger1 = (l_j2e+l4)+sqrt((l_j2e+l4)^2-4*0.5*-((Upp_panto)/(y_finger*
    z_finger)));
% x_finger2 = (l_j2e+l4)-sqrt((l_j2e+l4)^2-4*0.5*-((Upp_panto)/(y_finger*
    z_finger)));
x_finger = max(x_finger1,x_finger2);
%bal = side1-side2
check_fing_bal = (x_finger/2-l_j2e-l4)*x_finger*y_finger*z_finger*density
    -(m7j/n_arms+m2+m_joint)*l4;
m4 = x_finger*y_finger*z_finger*density;
m10 = m4;

%manuel overwrite other shape finger
m4 = 0.053036163187271;%0.055575132029231;%0.058229805666138;
m10 = m4;
l_com4 = 0.024631157877283;%0.025370942195569;%60.56534182*10^-3-l_j2e-l4
    ;%57.8373162259695*10^-3-l_j2e-l4%57.99699317*10^-3-l_j2e-l4;

%% Calculations for balanced gripper
%CoM specific lengths
l_com1 = 95.56*10^-3;
l_com2 = l2+(-CoM2_link*m2_link + 2*(l_j2e+btwn_length/2)*btwn_length*
    square_stock_1*square_stock_2*density + (l_j2e+btwn_length+x_counter
    /2)*x_counter*y_counter*z_counter*density)/m2;%l2+(m7j/2*l2)/m2
    ;%141.64*10^-3;
l_com3 = (0*(m6j/n_arms)+CoM3_link_j*m3j+(m2+m7j/n_arms+m4+m_joint)*l3)/(
    m2+m_joint+m6j/n_arms+m4+m3j+m7j/n_arms);
%l_com4 = x_finger/2-l4-l_j2e;%(l4*(m2+m7j/2+m_joint))/m4;%from CoM of m2
    +m7/2 and m4 which needs to be at the foot of the lower pantograph
l_com8 = l_com2;%141.64*10^-3;
l_com9 = l_com3;
l_com10 = l_com4;


check_m2_bal = (l_com2-l2)*m2-l2*m7j/n_arms;

%calculate length of small pantograph arm for balance
l11 = ((m7j/n_arms+m2+m_joint+m4+m3j+m6j/n_arms)*(m_joint+m5)*l_com3 -
    m5j*CoM5_link*m5)/(2*(m7j/n_arms+m2+m_joint+m4+m3j+m6j/n_arms)*(
    m_joint+m5)+m5j*m_joint);
l5 = l11;%+.0271575458043;

l_com5 = (m5*CoM5_link+m_joint*l5)/(m5+m_joint);
l_com11 = l_com5;


%Check pantograph
% m1 p1 = m2 a1 + m3 p3
% m1 q1 = m3 q3
% m2 p2 = m1 a2 + m4 p4
% m2 q2 = m4 q4
M1 = m7j/n_arms+m2+m_joint+m4+m3j+m6j/n_arms;
M2 = m7j/n_arms+m8+m_joint+m10+m9j+m6j/n_arms;
M3 = m11j;
M4 = m5j;
A1 = l11;
A2 = l5;
P1 = l_com3;
P2 = l_com9;
P3 = l_com11;
P4 = l_com5;

Check_panth_l = M1*P1-M2*A2-M3*P3;
Check_panth_r = M2*P2-M1*A2-M4*P4;



% %m2+m6/2+m4+m3+m7/2
%
% lCoM3 = (0*(m7/2)+l_com3*m3+(m2+m6/2+m4)*l3)/(m2+m6/2+m4+m3+m7/2);
% a1 = (0*(m7/2)+l_com3*m3+(m2+m6/2+m4)*l3)/(m2+m4+m3+m8+m6+m10+m9+m7+m11
    /2);

m_tot = m1+m2+m3+m4+m5+m6+m7+m8+m9+m10+m11;

%% Starting positions CoM's of elements
l_A1 = 90*10^-3;
l_A2 = 110*10^-3;

% x1    = 0;
% y1    = l_com1;
% phi1  = 0;

x3    = -l6/2-CoM3_link_j*sin(pi -acos(l_A1/(2*l5)));
y3    = l_A1 +CoM3_link_j*cos(pi-acos(l_A1/(2*l5)));
phi3  = pi-acos(l_A1/(2*l5));

li         = sqrt(l3^2+(l_A2-l_A1)^2-2*l3*(l_A2-l_A1)*cos(phi3));
phi3li     = acos((-((l_A2-l_A1)^2) + li^2 + l3^2) /(2*li*l3));
phili4     = acos((-(l2^2) + li^2 + l4^2)/(2*li*l4));

phi4  = phi3-pi + phi3li + phili4;
x4    = -l6/2-l3*sin(phi3)+l_com4*sin(phi4);
y4    = l_A1+l3*cos(phi3)-l_com4*cos(phi4);
                                    %balance term
phi2  = pi - atan(-((l6/2)+x4-(l_com4+l4)*sin(phi4))/(l_A2 - (y4+(l_com4+
    l4)*cos(phi4))));
x2    = -l6/2-l_com2*sin(phi2);
y2    = l_A2+l_com2*cos(phi2);

x5    = -l6/2 - l_com5*sin(acos(l_A1/(2*l5)));
```

```matlab
y5     = l_com5*cos(acos(l_A1/(2*l5)));
phi5   = acos(l_A1/(2*l5));
x6     = 0;
y6     = l_A1;
phi6   = 0;
x7     = 0;
y7     = l_A2;
phi7   = 0;

x9     = l6/2-CoM9_link_j*sin(-pi + acos(l_A1/(2*l11)));
y9     = l_A1 +CoM9_link_j*cos(-pi+acos(l_A1/(2*l11)));
phi9   = -pi + acos(l_A1/(2*l11));


li2        = sqrt(l9^2+(l_A2-l_A1)^2-2*l9*(l_A2-l_A1)*cos(phi9));
phi9li     = acos((-((l_A2-l_A1)^2) + li2^2 + l9^2) /(2*li2*l9));
phili10    = acos((-(l8^2) + li2^2 + l10^2)/(2*li2*l10));

phi10 = phi9+pi - phi9li -phili10;
x10   = l6/2-l9*sin((phi9))+l_com10*sin(phi10);
y10   = l_A1+l9*cos(phi9)-l_com10*cos(phi10);

phi8  = -pi+atan(((-l6/2)+x10-(l10+l_com10)*sin(phi10))/(l_A2 - (y10+(l10
   +l_com10)*cos(phi10))));
x8    = l6/2-l_com8*sin(phi8);
y8    = l_A2+l_com8*cos(phi8);

x11   = l6/2 + l_com5*sin(acos(l_A1/(2*l11)));
y11   = l_com5*cos(acos(l_A1/(2*l11)));
phi11 = -acos(l_A1/(2*l11));

%% model parameters

% linkage positions
X = [-l7/2, l_A2;                                                        %
   point1
          0, l_A2;                                   %point2
      -l6/2, l_A1;                                   %point3
          0, l_A1;                                   %point4
      -l6/2, 0;                                      %point5
      -l6/2-l5*sin(phi5), l5*cos(phi5);              %point6
      -l7/2-l2*sin(phi2), l_A2+l2*cos(phi2);         %point7
      -l6/2-l3*sin(phi3), l_A1+l3*cos(phi3);         %point8
       l7/2, l_A2;                                   %point9
       l6/2, l_A1;                                   %point10
       l6/2, 0;                                      %point11
       l6/2-l11*sin(phi11), l11*cos(phi11);          %point12
       l7/2-l8*sin(phi8), l_A2+l8*cos(phi8);         %point13
       l6/2-l9*sin(phi9), l_A1+l9*cos(phi9);         %point14
          0, 0];                                     %point15/origin
```

```
%mass positions
 %x1 ,y1 ;  %m1  point hoeft niet want beweegt niet

XM = [ x2 ,y2 ;  %m2  point
           x3 ,y3 ;  %m3  point
           x4 ,y4 ;  %m4  point
           x5 ,y5 ;  %m5  point
           x6 ,y6 ;  %m6  point
           x7 ,y7 ;  %m7  point
           x8 ,y8 ;  %m8  point
           x9 ,y9 ;  %m9  point
           x10,y10;  %m10 point
           x11,y11;];%m11 point
 %%
 init_build('2D')


bm1_1 = add_plbeam(X(15,:),X(5,:), 1);
bm1_2 = add_plbeam(X(15,:),X(11,:),1,{'transrot_p',bm1_1,'p'});
bm5   = add_plbeam(X(5,:),X(6,:),  1,{'trans_p'   ,bm1_1,'q'});
bm3_1 = add_plbeam(X(3,:),X(6,:),  1,{'trans_q'   ,bm5  ,'q'});
bm3_2 = add_plbeam(X(6,:),X(8,:),  1,{'transrot_p',bm3_1,'q'});
bm6_1 = add_plbeam(X(4,:),X(3,:),  1,{'trans_q'   ,bm3_1,'p'});
bm6_2 = add_plbeam(X(4,:),X(10,:), 1,{'transrot_p',bm6_1,'p'});
bm11  = add_plbeam(X(11,:),X(12,:),1,{'trans_p'   ,bm1_2,'q'});
bm9_1 = add_plbeam(X(10,:),X(12,:),1,{{'trans_p'  ,bm6_2,'q'},{'trans_q',
   bm11,'q'}});
bm9_2 = add_plbeam(X(12,:),X(14,:),1,{'transrot_p',bm9_1,'q'});
bm4   = add_plbeam(X(8,:),X(7,:)  ,1,{'trans_p',bm3_2, 'q'});
bm10  = add_plbeam(X(14,:),X(13,:),1,{'trans_p',bm9_2,'q'});
bm2   = add_plbeam(X(1,:), X(7,:) ,1,{'trans_q',bm4, 'q'});
bm8   = add_plbeam(X(9,:),X(13,:), 1,{'trans_q',bm10,'q'});
bm7 = add_plbeam(X(1,:),X(9,:)    , 1,{{'trans_p',bm2,'p'},{'trans_q',bm8,
   'p'}});


bmm3   = add_plbeam(X(6,:),XM(2,:)    ,1,{'transrot_p',bm3_1,'q'});
bmm5   = add_plbeam(X(5,:),XM(4,:)    ,1,{'transrot_p',bm5, 'p'});
bmm6   = add_plbeam(X(3,:),XM(5,:)    ,1,{'transrot_p',bm6_1,'q'});
bmm9   = add_plbeam(X(12,:),XM(8,:)   ,1,{'transrot_p',bm9_1,'q'});
bmm11  = add_plbeam(X(11,:),XM(10,:)  ,1,{'transrot_p',bm11,'p'});
bmm4   = add_plbeam(X(8,:),XM(3,:)    ,1,{'transrot_p',bm4, 'p'});
bmm10  = add_plbeam(X(14,:),XM(9,:)   ,1,{'transrot_p',bm10,'p'});
bmm2   = add_plbeam(X(7,:),XM(1,:)    ,1,{'transrot_p',bm2, 'q'});
bmm8   = add_plbeam(X(13,:),XM(7,:)   ,1,{'transrot_p',bm8, 'q'});
bmm7   = add_plbeam(X(1,:),XM(6,:)   ,1,{'transrot_p',bm7, 'p'});


add_nodalloads('xm',XM(2,:), {'trans_p',bmm3, 'q'},m3j);
add_nodalloads('xm',XM(4,:), {'trans_p',bmm5, 'q'},m5j);
add_nodalloads('xm',XM(5,:), {'trans_p',bmm6, 'q'},m6j);
add_nodalloads('xm',XM(8,:), {'trans_p',bmm9, 'q'},m9j);
```

```matlab
add_nodalloads('xm',XM(10,:),{'trans_p',bmm11,'q'},m11j);
add_nodalloads('xm',XM(3,:), {'trans_p',bmm4, 'q'},m4);
add_nodalloads('xm',X(7,:), {'trans_p',bm4, 'q'},m_joint);
add_nodalloads('xm',XM(9,:), {'trans_p',bmm10, 'q'},m10);
add_nodalloads('xm',X(13,:), {'trans_p',bm10, 'q'},m_joint);
add_nodalloads('xm',XM(1,:), {'trans_p',bmm2, 'q'},m2);
add_nodalloads('xm',XM(6,:), {'trans_p',bmm7, 'q'},m7j);
add_nodalloads('xm',XM(7,:), {'trans_p',bmm8, 'q'},m8);


add_constrdofs('fix',X(15,:),{'trans',bm1_1,'p'},[1,2]);
add_constrdofs('fix',X(15,:),{'rot'  ,bm1_1,'p'},[1]);


add_constrdofs('fix',X(4,:),{'trans',bm6_1,'p'},[1]);
add_constrdofs('fix',X(4,:),{'rot',bm6_1,'p'},[1]);
add_constrdofs('fix',X(1,:),{'trans',bm7,'p'},[1]);
add_constrdofs('fix',X(1,:),{'rot',bm7,'p'},[1]);

    add_inputs('inputx',X(1,:),{'trans',bm7,'p'},[2],[l_A2,-0.07,0]);
add_inputs('inputx',X(4,:),{'trans',bm6_1,'p'},[2],[l_A1,-0.07,0]);



build_model('Check_combo',{'timestep 1 100'})

%% run model
spacar(-1,'Check_combo');
spavisual('Check_combo',1)

%%

mass  = [m3j m5j m6j m9j m11j m4 m_joint m10 m_joint m2 m7j m8].';
nodes = [41 43 45 47 49 51 28 53 31 55 59 57];

CoMx = (x(:,lnp(nodes,1))*mass)/sum(mass);
CoMy = (x(:,lnp(nodes,2))*mass)/sum(mass);

figure
title('Center of mass')
subplot(2,1,1)
plot(time,CoMx)
title('CoMx')
subplot(2,1,2)
plot(time,CoMy-3.733580507763*10^-3)
title('CoMy')

%%
% plot base reaction forces x,y
figure
plot(time,fxtot(:,lnp(1,1:2)))
title('Reaction Forces at Base')
```

```
xlabel('time')
ylabel('force')
legend('fx','fy')
% ylim([-1 1]*10^-25)

figure
plot(time,fxtot(:,lnp(1,2)))
hold on
plot(time,fxtot(:,lnp(15,2)))
plot(time,fxtot(:,lnp(33,2)))
plot(time,fxtot(:,lnp(15,2))+fxtot(:,lnp(33,2)))
title('Reaction Forces at Base v actuator')
xlabel('time')
ylabel('force')
legend('reaction at base','force at lA_1','force at lA_2','total actuator
   ')
grid on

figure
plot(time,fxtot(:,lnp(1,2))+fxtot(:,lnp(15,2))+fxtot(:,lnp(33,2)))
title('Reaction forces at base minus actuation forces')
grid on

% figure
% plot(x(:,lnp(nodes,1)),x(:,lnp(nodes,2)))
% legend('2','3','4','5','6','7','8','9','10','11')
% title('Motion of the CoM''s')
```

APPENDIX J
MATLAB CODE ACTUATED TEST

```
clear all
close all

% tb = top balanced
% tu = top unbalanced
% bb = bottom balanced
% bu = bottom unbalanced
% n  = no load

%% plots on/of

norm_pos                 = 0;
raw_amp                  = 0;
all_pos_cycles_separate = 0;
std_pos                  = 0;
all_amp_cycles_separate = 0;
raw_F                    = 0;
std_amp                  = 0;
std_F                    = 0;
comp_amp                 = 0;
comp_F                   = 1;

%% input data from .txt
[timetb, postb, input_texttb, input_bintb, Act_stattb, Amp_metertb] =
   readvars('2023-06-25 19-22-13 bal top 20 cpm 40mm 1 sec.txt');
[timetu, postu, input_texttu, input_bintu, Act_stattu, Amp_metertu] =
   readvars('2023-06-25 18-12-23 unbal top 20cpm 1 sec trans 40 mm.txt');
[timebb, posbb, input_textbb, input_binbb, Act_statbb, Amp_meterbb] =
   readvars('2023-06-17 16-47-41 act low 20 CPM 40 mm 1 sec tran A pos.
   txt');
[timebu, posbu, input_textbu, input_binbu, Act_statbu, Amp_meterbu] =
   readvars('2023-06-18 16-04-35 low 20 CPM 40 mm 1sec trans unbal.txt');
[timen , posn , input_textn , input_binn , Act_statn , Amp_metern ] =
   readvars('2023-06-25 17-56-19 no load.txt');


gear_circ = 3.6*10^-2; %omtrek metend tandwiel in m
encoder_steps = 24;     %number of steps of rotary encoder per full turn

%% raw data and normalised plot
if norm_pos == 1
    figure
    plot(timetb-timetb(2),(postb-min(postb))/(max(postb)-min(postb)),'
       LineWidth', 2)
    hold on
    plot(timetb-timetb(2),input_bintb,'LineWidth', 2)
    plot(timetu-timetu(2),(postu-min(postu))/(max(postu)-min(postu)),'
       LineWidth', 2)
    plot(timebb-timebb(2),(posbb-min(posbb))/(max(posbb)-min(posbb)),'LineWidth', 2)
    plot(timebu-timebu(2),(posbu-min(posbu))/(max(posbu)-min(posbu)),'LineWidth', 2)
    plot(timen -timen(2) ,(posn -min(posn ))/(max(posn )-min(posn )),'LineWidth', 2)
    title('normalised position data')
    legend('top balanced','input','top unbalanced','bottom balanced','bottom unbalanced','no load')
end

amptb = (Amp_metertb-511) * (20/512); % calculating from 10 bit array to amps measured 2.5V is 0 Amp 100 mV per step
amptu = (Amp_metertu-511) * (20/512); % calculating from 10 bit array to amps measured 2.5V is 0 Amp 100 mV per step
ampbb = (Amp_meterbb-511) * (20/512); % calculating from 10 bit array to amps measured 2.5V is 0 Amp 100 mV per step
ampbu = (Amp_meterbu-511) * (20/512); % calculating from 10 bit array to amps measured 2.5V is 0 Amp 100 mV per step
ampn  = (Amp_metern -511) * (20/512); % calculating from 10 bit array to amps measured 2.5V is 0 Amp 100 mV per step

if raw_amp == 1
    figure
    plot(timetb, amptb)
    hold on
    plot(timetu, amptu)
    plot(timebb, ampbb)
    plot(timebu, ampbu)
    plot(timen , ampn )
    title('amparage raw data')
    legend('top balanced','top unbalanced','bottom balanced','bottom unbalanced','no load')
end

Ftb = zeros(length(timetb),1);
for i=2:length(timetb)
    if postb(i)-postb(i-1) == 0
        Ftb(i) = 0;
    else
        Ftb(i) = 48*amptb(i)*((timetb(i)-timetb(i-1))/1000)/((postb(i)-postb(i-1))*gear_circ/encoder_steps);
    end
end

Ftu = zeros(length(timetu),1);
for i=2:length(timetu)
    if postu(i)-postu(i-1) == 0
        Ftu(i) = 0;
    else
        Ftu(i) = 48*amptu(i)*((timetu(i)-timetu(i-1))/1000)/((postu(i)-postu(i-1))*gear_circ/encoder_steps);
```

```matlab
    end
end

Fbb = zeros(length(timebb),1);
for i=2:length(timebb)
    if posbb(i)-posbb(i-1) == 0
        Fbb(i) = 0;
    else
        Fbb(i) = 48*ampbb(i)*((timebb(i)-timebb(i-1))/1000)/((posbb(i)-
           posbb(i-1))*gear_circ/encoder_steps);
    end
end

Fbu = zeros(length(timebu),1);
for i=2:length(timebu)
    if posbu(i)-posbu(i-1) == 0
        Fbu(i) = 0;
    else
        Fbu(i) = 48*ampbu(i)*((timebu(i)-timebu(i-1))/1000)/((posbu(i)-
           posbu(i-1))*gear_circ/encoder_steps);
    end
end

Fn = zeros(length(timen),1);
for i=2:length(timen)
    if posn(i)-posn(i-1) == 0
        Fn(i) = 0;
    else
        Fn(i) = 48*ampn(i)*((timen(i)-timen(i-1))/1000)/((posn(i)-posn(i
           -1))*gear_circ/encoder_steps);
    end
end

if raw_F == 1
    figure
    plot(timetb,Ftb)
    hold on
    title('Force')
    plot(timetu,Ftu)
    plot(timebb,Fbb)
    plot(timebu,Fbu)
    plot(timen,Fn)
    title('Force raw data')
    legend('top balanced','top unbalanced','bottom balanced','bottom
       unbalanced','no load')
end

%% longer time span averaging
%
% n = 10;
%
% F_avg = zeros(length(time),1);
% amp_avg = zeros(length(time),1);
%
% for i=n:n:length(time)
%     if pos(i)-pos(i-1) == 0
%         Ftb(i) = 0;
%     else
%         amp_sum = 0;
%         for j = 0:n-1
%             amp_sum = amp_sum + amp(i-j);
%         end
%         amp_avg(i) = amp_sum/n;
%         F_avg(i) = 48*amp_avg(i)*((time(i)-time(i-n+1))/1000)/((pos(i)-
    pos(i-1))*gear_circ/encoder_steps);
%     end
% end
%
%
%
% figure
% plot(time,F_avg)
% title('Force average')

%% Splice data top balanced

cyclestb = 0;
for i = 2:length(timetb)-2
  if input_bintb(i)-input_bintb(i-1) == -1
    cyclestb = cyclestb+1;
  end
end

j = 0;
n = 1;
timetb_s  = zeros(cyclestb,1);
postb_s   = [0];
amptb_s   = [0];
inputtb_s = [0];
Ftb_s     = [0];
% Ftb_avg_s = [0];
% amptb_avg_s = [0];

for i = 2:length(timetb)-2
  if input_bintb(i)-input_bintb(i-1) == -1
    j = j+1;
    n = 1;
  end

  if j == 0

  else
    timetb_s(j,n)  = timetb(i);
    postb_s(j,n)   = postb(i);
    amptb_s(j,n)    = amptb(i);
    inputtb_s(j,n) = input_bintb(i);
    Ftb_s(j,n)      = Ftb(i);
%     Ftb_avg_s(j,n) = Ftb_avg(i);
%     amptb_avg_s(j,n) = amptb_avg(i);
    n = n+1;
  end
end

%% Splice data top unbalanced

cyclestu = 0;
for i = 2:length(timetu)-2
  if input_bintu(i)-input_bintu(i-1) == -1
    cyclestu = cyclestu+1;
  end
end

j = 0;
n = 1;
timetu_s   = zeros(cyclestu,1);
postu_s    = [0];
amptu_s    = [0];
inputtu_s = [0];
Ftu_s      = [0];
% Ftu_avg_s = [0];
% amptu_avg_s = [0];

for i = 2:length(timetu)-2
  if input_bintu(i)-input_bintu(i-1) == -1
    j = j+1;
    n = 1;
  end

  if j == 0

  else
    timetu_s(j,n)  = timetu(i);
    postu_s(j,n)    = postu(i);
    amptu_s(j,n)    = amptu(i);
    inputtu_s(j,n) = input_bintu(i);
    Ftu_s(j,n)      = Ftu(i);
%     Ftu_avg_s(j,n) = Ftu_avg(i);
%     amptu_avg_s(j,n) = amptu_avg(i);
    n = n+1;
  end
end

%% Splice data bottom balanced

cyclesbb = 0;
for i = 2:length(timebb)-2
```

```
  if input_binbb(i)-input_binbb(i-1) == -1
    cyclesbb = cyclesbb+1;
  end
end

j = 0;
n = 1;
timebb_s   = zeros(cyclestb,1);
posbb_s    = [0];
ampbb_s    = [0];
inputbb_s = [0];
Fbb_s      = [0];
% Fbb_avg_s = [0];
% ampbb_avg_s = [0];

for i = 2:length(timebb)-2
  if input_binbb(i)-input_binbb(i-1) == -1
    j = j+1;
    n = 1;
  end

  if j == 0

  else
    timebb_s(j,n)   = timebb(i);
    posbb_s(j,n)    = posbb(i);
    ampbb_s(j,n)    = ampbb(i);
    inputbb_s(j,n) = input_binbb(i);
    Fbb_s(j,n)      = Fbb(i);
%     Fbb_avg_s(j,n)  = Fbb_avg(i);
%     ampbb_avg_s(j,n)  = ampbb_avg(i);
    n = n+1;
  end
end

%% Splice data bottom unbalanced

cyclesbu = 0;
for i = 2:length(timebu)-2
  if input_binbu(i)-input_binbu(i-1) == -1
    cyclesbu = cyclesbu+1;
  end
end

j = 0;
n = 1;
timebu_s   = zeros(cyclesbu,1);
posbu_s    = [0];
ampbu_s    = [0];
inputbu_s = [0];
Fbu_s      = [0];
% Fbu_avg_s = [0];
% ampbu_avg_s = [0];

for i = 2:length(timebu)-2
  if input_binbu(i)-input_binbu(i-1) == -1
    j = j+1;
    n = 1;
  end

  if j == 0

  else
    timebu_s(j,n)   = timebu(i);
    posbu_s(j,n)    = posbu(i);
    ampbu_s(j,n)    = ampbu(i);
    inputbu_s(j,n) = input_binbu(i);
    Fbu_s(j,n)      = Fbu(i);
%     Fbu_avg_s(j,n) = Fbu_avg(i);
%     ampbu_avg_s(j,n) = ampbu_avg(i);
    n = n+1;
  end
end

%% Splice data no load

cyclesn = 0;
for i = 2:length(timen)-2
  if input_binn(i)-input_binn(i-1) == -1
    cyclesn = cyclesn+1;
  end
end

j = 0;
n = 1;
timen_s   = zeros(cyclesn,1);
posn_s    = [0];
ampn_s    = [0];
inputn_s = [0];
Fn_s      = [0];
% Fn_avg_s = [0];
% ampn_avg_s = [0];

for i = 2:length(timen)-2
  if input_binn(i)-input_binn(i-1) == -1
    j = j+1;
    n = 1;
  end

  if j == 0

  else
    timen_s(j,n)   = timen(i);
    posn_s(j,n)    = posn(i);
    ampn_s(j,n)    = ampn(i);
    inputn_s(j,n) = input_binn(i);
    Fn_s(j,n)      = Fn(i);
%     Fn_avg_s(j,n) = Fn_avg(i);
%     ampn_avg_s(j,n) = ampn_avg(i);
    n = n+1;
  end
end

%% plot sliced data
if all_pos_cycles_separate ==1
    figure
    hold on
    for i= 1:cycles-1
        plot(time_s(i,:)-time_s(i,1), (pos_s(i,:)-min(pos_s(i,:)))/(max(pos_s(i,:))-min(pos_s(i,:))))
    end
    title('position all cycles')
end

if std_pos ==1
    figure
    stdshade(((postb_s-min(postb_s(i,:)))/(max(postb_s(i,:))-min(postb_s(i,:)))),.1,'b',timetb_s(1,:)-timetb_s(1,1))
    hold on
    % plot(time_s(1,:)-time_s(1,1),input_s(1,:),'LineWidth', 2, 'Color','k')
%     stdshade(((postu_s-min(postu_s(i,:)))/(max(postu_s(i,:))-min(postu_s(i,:)))),.1,timetu_s(1,:)-timetu_s(1,1))
%     stdshade(((posbb_s-min(posbb_s(i,:)))/(max(posbb_s(i,:))-min(posbb_s(i,:)))),.1,timebb_s(1,:)-timebb_s(1,1))
%     stdshade(((posbu_s-min(posbu_s(i,:)))/(max(posbu_s(i,:))-min(posbu_s(i,:)))),.1,timebu_s(1,:)-timebu_s(1,1))
%     stdshade(((posn_s -min(posn_s(i,:))) /(max(posn_s(i,:)) -min(posn_s(i,:)))) ,.1,timen_s(1,:) -timen_s(1,1))
    title('normalised position')
%     legend('top balanced','top unbalanced','bottom balanced','bottom unbalanced','no load')
end

if all_amp_cycles_separate == 1
    figure
    hold on
    for i= 1:cyclestb-1
        plot(timetb_s(i,:)-timetb_s(i,1), amptb_s(i,:))
    end
    for i= 1:cyclestu-1
        plot(timetu_s(i,:)-timetu_s(i,1), amptu_s(i,:))
    end
    for i= 1:cyclesbb-1
        plot(timebb_s(i,:)-timebb_s(i,1), ampbb_s(i,:))
    end
```

```matlab
    for i= 1:cyclesbu-1
        plot(timebu_s(i,:)-timebu_s(i,1), ampbu_s(i,:))
    end
    for i= 1:cyclesn-1
        plot(timen_s(i,:)-timen_s(i,1), ampn_s(i,:))
    end
    title('amp all cycles')
    legend('top balanced','top unbalanced','bottom balanced','bottom
       unbalanced','no load')
end

if std_amp ==1
%Plot mean and std deviation of cycle
    figure
    stdshade(amptb_s,.1,[0      0.4470 0.7410],timetb_s(1,:)-timetb_s
       (1,1),50);
    hold on
    stdshade(amptu_s,.1,[0.9290 0.6940 0.1250],timetu_s(1,:)-timetu_s
       (1,1),50);
    stdshade(ampbb_s,.1,[0.4940 0.1840 0.5560],timebb_s(1,:)-timebb_s
       (1,1),50);
    stdshade(ampbu_s,.1,[0.4660 0.6740 0.1880],timebu_s(1,:)-timebu_s
       (1,1),50);
    stdshade(ampn_s, .1,[0.3010 0.7450 0.9330],timen_s(1,:) -timen_s(1,1)
       ,50);
    title('amp std deviation')
    legend('top balanced mean','top balanced std','top unbalanced mean','
       top unbalanced std','bottom balanced mean','bottom balanced std','
       bottom unbalanced mean','bottom unbalanced std','no load mean','no
        load std')
    % ylim([-.5 2])
end

% P_mech is F*s/t = P_elec = U I
% F*s/t = U*I
% F = U*I*t/s

if std_F == 1
    figure
    stdshade(Ftb_s,.1,[0      0.4470 0.7410],timetb_s(1,:)-timetb_s(1,1)
       ,50);
    hold on
    stdshade(Ftu_s,.1,[0.9290 0.6940 0.1250],timetu_s(1,:)-timetu_s(1,1)
       ,50);
    stdshade(Fbb_s,.1,[0.4940 0.1840 0.5560],timebb_s(1,:)-timebb_s(1,1)
       ,50);
    stdshade(Fbu_s,.1,[0.4660 0.6740 0.1880],timebu_s(1,:)-timebu_s(1,1)
       ,50);
    stdshade(Fn_s ,.1,[0.3010 0.7450 0.9330],timen_s(1,:) -timen_s(1,1)
       ,50);
    title('force std deviation')
    legend('top balanced std','top balanced mean','top unbalanced std','
       top unbalanced mean','bottom balanced std','bottom balanced mean',
       'bottom unbalanced std','bottom unbalanced mean','no load std','no
        load mean')
%     ylim([-15 15])
end

% figure
% stdshade(F_avg_s,.1,'b',time_s(1,:)-time_s(1,1))
% title('Force average std deviation')

amptu_s_avg = zeros(1,length(amptu_s));
amptb_s_avg = zeros(1,length(amptb_s));
ampbu_s_avg = zeros(1,length(ampbu_s));
ampbb_s_avg = zeros(1,length(ampbb_s));
ampn_s_avg  = zeros(1,length(ampn_s));

for i = 1:length(amptu_s)
    amptu_s_avg(i) = sum(amptu_s(:,i))/cyclestu;
    amptb_s_avg(i) = sum(amptb_s(:,i))/cyclestb;
    ampbu_s_avg(i) = sum(ampbu_s(:,i))/cyclesbu;
    ampbb_s_avg(i) = sum(ampbb_s(:,i))/cyclesbb;
    ampn_s_avg(i)  = sum(ampn_s(:,i))/cyclesn;
end

Ftu_s_avg = zeros(1,length(Ftu_s));
Ftb_s_avg = zeros(1,length(Ftb_s));
Fbu_s_avg = zeros(1,length(Fbu_s));
Fbb_s_avg = zeros(1,length(Fbb_s));
Fn_s_avg  = zeros(1,length(Fn_s));

for i = 1:length(Ftu_s)
    Ftu_s_avg(i) = sum(Ftu_s(:,i))/cyclestu;
    Ftb_s_avg(i) = sum(Ftb_s(:,i))/cyclestb;
    Fbu_s_avg(i) = sum(Fbu_s(:,i))/cyclesbu;
    Fbb_s_avg(i) = sum(Fbb_s(:,i))/cyclesbb;
    Fn_s_avg(i)  = sum(Fn_s(:,i))/cyclesn;
end

%% plot differences balanced vs unbalanced
% figure
% title('difference top balanced vs unbalanced avg')
% hold on
% plot(timetu_s(1,:)-timetu_s(1,1),amptu_s_avg-amptb_s_avg)
% legend('unbalanced minus balanced')

if comp_amp == 1
    figure
    title('amperage difference top balanced vs unbalanced')
    hold on
    stdshade(1*(amptu_s_avg-amptb_s_avg-0.0),0,[0      0.4470 0.7410],
       timetu_s(1,:)-timetu_s(1,1),50);
    legend('unbalanced minus balanced')
    yline(0)

    % figure
    % title('difference bottom balanced vs unbalanced')
    % hold on
    % plot(timebu_s(1,:)-timetu_s(1,1),amptu_s_avg-amptb_s_avg)
    % legend('unbalanced minus balanced')

    figure
    title('amperage difference bottom balanced vs unbalanced')
    hold on
    stdshade(1*(ampbu_s_avg - ampbb_s_avg+0.0),0,[0      0.4470 0.7410],
       timebu_s(1,:)-timebu_s(1,1),50);
    legend('unbalanced minus balanced')
    yline(0)

    figure
    title('amperage difference top balanced vs no load')
    hold on
    stdshade(1*(amptb_s_avg - ampn_s_avg),0.5,[0      0.4470 0.7410],
       timetb_s(1,:)-timetb_s(1,1),50);
    legend('amperage unbalanced minus balanced')
    yline(0)

    figure
    title('amperage difference bottom balanced vs no load')
    hold on
    stdshade(1*(ampbb_s_avg-ampn_s_avg+0.01),0.5,[0      0.4470 0.7410],
       timebb_s(1,:)-timebb_s(1,1),50);
    legend('unbalanced minus balanced')
    yline(0)

    figure
    title('amperage difference top unbalanced vs no load')
    hold on
    stdshade(1*(amptu_s_avg-ampn_s_avg-0.022),0.5,[0      0.4470 0.7410],
       timetu_s(1,:)-timetu_s(1,1),50);
    legend('unbalanced minus balanced')
    yline(0)

    figure
    title('amperage difference bottom unbalanced vs no load')
    hold on
    stdshade(1*(ampbu_s_avg-ampn_s_avg+0.025),0.5,[0      0.4470 0.7410],
       timebu_s(1,:)-timebu_s(1,1),50);
    legend('unbalanced minus balanced')
    yline(0)
end

if comp_F == 1
    figure
    stdshade(Fbu_s_avg-Fn_s_avg,.0,[0      0.4470 0.7410],timebu_s(1,:)-
       timebu_s(1,1),50);
    hold on
    title('force difference bottom unbalanced vs no load')
    legend('bottom unbalanced minus no load')
    yline(0)
%     ylim([-15 15])

    figure
    stdshade(Fbb_s_avg-Fn_s_avg,.0,[0      0.4470 0.7410],timebb_s(1,:)-
       timebb_s(1,1),50);
    hold on
    title('force difference bottom balanced vs no load')
    legend('bottom balanced minus no load')
    yline(0)
%     ylim([-15 15])

    figure
    stdshade(Ftu_s_avg-Fn_s_avg,.0,[0      0.4470 0.7410],timetu_s(1,:)-
       timetu_s(1,1),50);
    hold on
    title('force difference top unbalanced vs no load')
    legend('top unbalanced minus no load')
    yline(0)
%     ylim([-15 15])

    figure
    stdshade(Ftb_s_avg-Fn_s_avg,.0,[0      0.4470 0.7410],timetb_s(1,:)-
       timetb_s(1,1),50);
    hold on
    title('force difference top balanced vs no load')
    legend('top balanced minus no load')
    yline(0)
%     ylim([-15 15])

    figure
    stdshade(Fbu_s_avg-Fbb_s_avg,.0,[0      0.4470 0.7410],timebu_s(1,:)-
       timebu_s(1,1),50);
    hold on
    title('force difference bottom unbalanced vs bottom balanced')
    legend('bottom unbalanced minus bottom balanced')
    yline(0)
%     ylim([-15 15])

    figure
    stdshade(Ftu_s_avg-Ftb_s_avg,.0,[0      0.4470 0.7410],timetu_s(1,:)-
       timetu_s(1,1),50);
    hold on
    title('force difference top unbalanced vs top balanced')
    legend('top unbalanced minus top balanced')
    yline(0)
%     ylim([-15 15])
```

```

end

%     figure
%     stdshade(Ftb_s,.7,[0      0.4470 0.7410],timetb_s(1,:)-timetb_s
    (1,1),50);
%     hold on
%     title('force top balanced')
%
%     figure
%     stdshade(Ftu_s,.7,[0.9290 0.6940 0.1250],timetu_s(1,:)-timetu_s
    (1,1),50);
%     hold on
%     title('force top unbalanced')
%
%     figure
%     stdshade(Fbb_s,.7,[0.4940 0.1840 0.5560],timebb_s(1,:)-timebb_s
    (1,1),50);
%     hold on
%     title('force bottom balanced')
%
%     figure
%     stdshade(Fbu_s,.7,[0.4660 0.6740 0.1880],timebu_s(1,:)-timebu_s
    (1,1),50);
%     hold on
%     title('force bottom unbalanced')
%
%
%     figure
%     stdshade(Fn_s ,.7,[0.3010 0.7450 0.9330],timen_s(1,:) -timen_s(1,1)
    ,50);
%     hold on
%     title('force no load')
```

APPENDIX K
ARDUINO CODE ACTUATED TEST

```
// In this version of the PLC the cycling rate is fixed
// The cycling rate is written in cycles per minutes

//=====SETUP=====
//---VARIABLES INITIALIZATION---
int   delay_time = 0;//Delay time sets amount of data per cycle by delaying next measurement
float cycle_time = 0;//Cycle duration in ms calculated depending on the cycling rate

bool input_actuator = 0;
float cycling_rate = 0; //Cycling rate

int encoderPos=0;
int encoderStepsPerRevolution= 24;

bool encoderALast= LOW; //remembers the previous state of the encoder pin A

void setup()
{
//---PIN SETUP---
pinMode(11, OUTPUT); // Set the digital pin D1 as output for the signal, this output is vizualised by the L13 led
pinMode(A0, INPUT); // Set the analog pin A0 as rotary button output with multiplier 1 for calculation
pinMode(A1, INPUT); // Set the analog pin A1 as rotary button output with multiplier 2 for calculation
pinMode(A2, INPUT); // Set the analog pin A2 as rotary button output with multiplier 4 for calculation
pinMode(A3, INPUT); // Set the analog pin A3 as rotary button output with multiplier 8 for calculation
pinMode(A5, INPUT); // Set the analog pin A5 as Amperage meter input 2.5 V = 0 100 mV/A max 20 A
pinMode( 3, INPUT); // Set to check if output 11 is correct
pinMode( 5, INPUT); // Komp-act status actuator controller output

//--Encoder setup
pinMode(7, INPUT); //Encoder pin A
pinMode(6, INPUT); //Encoder pin B
digitalWrite(7, HIGH); //Set Encoder pin A to HIGH
digitalWrite(6, HIGH); //Set Encoder pin B to HIGH


Serial.begin(115200); // open the serial port at 115200 bps:
}

//=====LOOP CODE=====
void loop()
```

```
{
cycling_rate = 20;
cycle_time = (60.0*1000.0)/(2*cycling_rate);  //---CACLULATE cycle---
delay_time = 1;// ms delay inbetween


//---CYCLE---
for(int i=0; i<(cycle_time/delay_time)+1; i++) {
boolean encoderA= digitalRead(7);

if ((encoderALast == HIGH) && (encoderA == LOW)){
if (digitalRead(6) == LOW){
encoderPos--;
}
else{
encoderPos++;
}
}
Serial.print(millis());
Serial.print(,);
Serial.print(encoderPos);
Serial.print(,);
encoderALast= encoderA;

digitalWrite(11, LOW); // Home
Serial.print(Low,);
Serial.print(digitalRead(11));
Serial.print(,);
Serial.print(digitalRead(5));
Serial.print(,);
Serial.println(analogRead(A5));
delay(delay_time);
}
for(int i=0; i<(cycle_time/delay_time)+1; i++) {
bool encoderA= digitalRead(7);
if ((encoderALast == HIGH) && (encoderA == LOW)){
if (digitalRead(6) == LOW){
encoderPos--;
}
else{
encoderPos++;
}
}
Serial.print(millis());
Serial.print(,);
Serial.print(encoderPos);
Serial.print(,);
encoderALast= encoderA;

digitalWrite(11, HIGH); //Extended
Serial.print(High,);
Serial.print(digitalRead(11));
Serial.print(,);
Serial.print(digitalRead(5));
Serial.print(,);
Serial.println(analogRead(A5));
delay(delay_time);
}
}
```