Functional Mechanical Design Group Project

Group 5

Oskar Ståhlbom Hüseyin Yasin Öztürk Lirim Shaqiri

Cem Güngör Farnood Arab

Objective

The goal of the project is to design a high-speed machine capable of embossing small details onto surfaces such as sheets of paper. The embossing process involves the use of two metal matrices (punches): a relief matrix (cliché) and a hollow matrix (counter matrix). When compressed together, these matrices create a relief effect on the surface.

In the designed mechanism, the counter matrix remains stationary, while the cliché moves during each cycle with a displacement of 50 mm.

The mechanism is supposed to have a production rate of 180 pieces every minute. With respect to the machine master angle (α), the cycle-time is splitted as follows in Table 1:

	Rise	Dwell	Return	Dwell
$\boldsymbol{\alpha}$	$0° - 60°$	$60° - 140°$	$140° - 200°$	$200° - 360°$
$\boldsymbol{x}$	$0 - 50\ mm$	$50\ mm$	$50 - 0\ mm$	$0\ mm$

Table 1: Motion Curve of Embossing End

During the first dwell phase, the cliché exerts a force of 600N on the counter matrix.

Mechanism

To achieve the desired operation, various mechanism alternatives are explored. Within the context of the course, we focused on three primary types: four-bar linkages, slider-crank mechanisms, and cam mechanisms. Each type offers distinct advantages:

- Cams: Ideal for generating precise motion curves, but they are not well-suited for handling high forces. Additionally, achieving a 50 mm displacement using only a cam would require a large cam, increasing both cost and unnecessary rotational momentum.
- Four-Bar Linkages: Excellent for altering leverage ratios, making them beneficial in our design to optimize force transmission.
- Slider-Crank Mechanisms: Necessary for achieving linear motion at the end of the embossing cycle.

To capitalize on the strengths of each mechanism, all are integrated to the final design, combining their unique advantages to create an efficient and effective embossing machine.

After deciding on general type of the mechanism, the link diagram can be represented as shown in Figure 1:

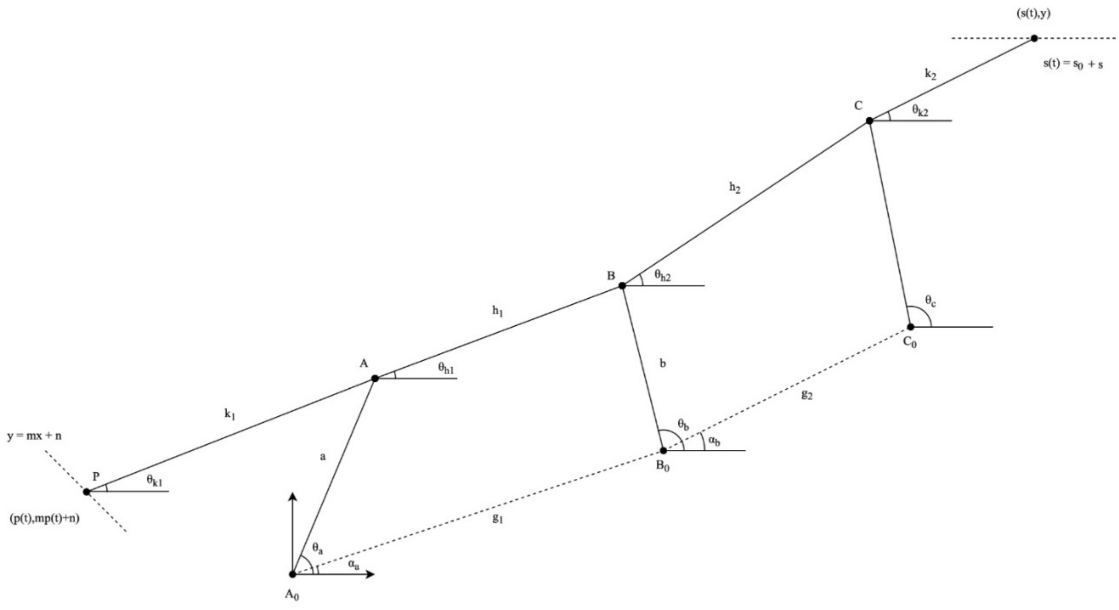


Figure 1: Diagram of the mechanism

As shown in Figure 1, each link and angle are assigned a parameter. The motion of the follower (left end) and the embossing end (right end) is constrained along a linear path.

The aim is to utilize the variable transmission ratio property of the four-bar mechanism. The transmission ratio can be defined as:

$$\tau = \frac{\Delta S}{\Delta P} \tag{1}$$

Where:

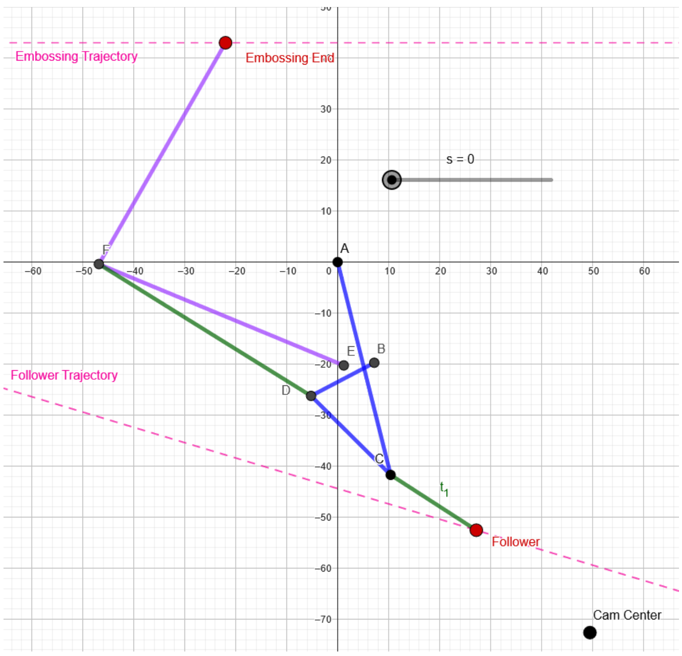
τ: Transmission ratio

ΔS: Embossing slider end displacement stepwise difference

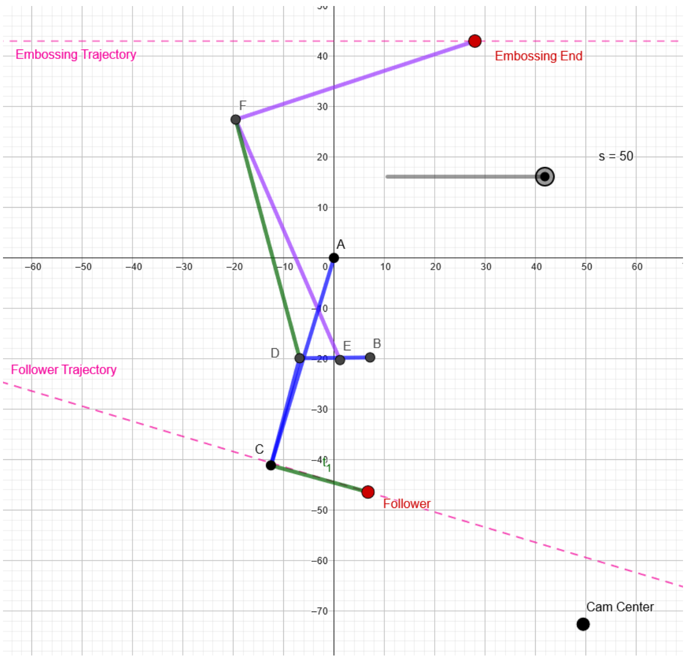
ΔP: Camside slider end displacement stepwise difference

The equation approximates the velocity ratio between the output and the input.

During the rise and return phases, as indicated in Table 1, it is desirable to maintain a low transmission ratio to keep the cam size small. During the first dwell phase, to reduce the force exerted on the cam during embossing, a high transmission ratio is more desirable. Additionally, the mechanism must be designed to accommodate the required 50 mm displacement without being excessively large. By applying these conditions, the parameters related with the link lengths and their orientations shown in Figure 1 are optimized. After several design iterations with the aid of the optimization algorithm developed in MATLAB (see section parameter optimization), the final mechanism is illustrated in Figure 2.



(a) Initial Position



(b) Embossing Position

Figure 2: Mechanism in two different stages. Four-bar is in dark blue, Slider Crank is in purple, connectors are in green, two ends are in red, and trajectory of ends are in pink.

A stroke in each rotation of the cam is implemented, requiring a cam rotation speed of 180 rpm. At this speed, each cycle described in Table 1 must be completed in 1/3 second. Consequently, we can define the motion curve in terms of time as shown in Table 2.

	Rise	Dwell	Return	Dwell
$\boldsymbol{\alpha}$	$0° - 60°$	$60° - 140°$	$140° - 200°$	$200° - 360°$
$\boldsymbol{t}$	$0 - 0.056\ s$	$0.056 - 0.130\ s$	$0.130 - 0.185\ s$	$0.185 - 0.333\ s$
$\boldsymbol{x}$	$0 - 50\ mm$	$50\ mm$	$50 - 0\ mm$	$0\ mm$

Table 2: Motion curve of Embossing End with respect to time and angle

Parameter optimization

The parameters related with the definition of the mechanism for the project includes:

The link lengths: $a, b, c, h1, h2, g1, g2, k1, k2$

The ground link orientation: α_a, α_b

The input slider guide position and orientation: m, n

The output slider position: s_0, Y

A design is represented as a single point χ within the parameter space

$$\chi = \{a, b, c, g_1, g_2, h_1, h_2, \alpha_a, \alpha_b, m, n, s_0, Y\}$$

Any random point chosen from the parameter space may or may not define a unique mechanism based on the kinematic constraints (see chapter kinematics).

Since the primary concern is to satisfy a given motion curve at the output side, the relation between the input and output slider displacements can be expressed as a function of the parameter set χ and the motion curve $s(\psi)$:

$$p(\chi, s(\psi)) \qquad (2)$$

Where:

ψ: Master angle

$s(\psi)$: Output slider displacement

$p(\psi)$: Input slider displacement

Considering the points that do define a unique mechanism, the objective is to quantitatively assess the performance of the resulting mechanism at any given point in parameter space.

During the rise and return periods of the motion curve, no external force is applied at the tip of the output slider, therefore high transmission ratio is preferrable since it allows for a larger displacement at the output slider compared to the input displacement. At the point where embossing takes place, it is preferrable to have a force advantage to lower the force applied on the cam. By defining an objective transmission ratio curve $\tau_o(\psi)$ and evaluating the standard deviation of the resultant transmission ratio curve $\tau(\chi, \psi)$ for a given parameter set χ yields a single scalar value, where the parameters could be optimized. In the case of the optimization study, the cost function is decided to be the standard deviation between two curves and the parameters are adjusted by gradient descend method to lower the cost function.

The transmission ratio is given by the following relation

$$\tau(\chi, \psi) = \frac{ds(\psi)}{d\psi} \cdot \frac{d\psi}{dp(\psi)} \approx \frac{\Delta s(\psi)}{\Delta p(\psi)} = \frac{s(\psi_{i+1}) - s(\psi_i)}{p(\psi_{i+1}) - p(\psi_i)} \tag{3}$$

The cost function $C(\chi)$:

$$C(\chi) = \sqrt{\frac{1}{n} \sum_{i=1}^{n} \left(\tau_o(\psi_i) - \tau(\psi_i) \right)} \tag{4}$$

Applying the gradient descend algorithm to adjust the design parameters

$$\chi_{i+1} = \chi_i - \gamma \nabla C(\chi) \tag{5}$$

Where:

γ: Learning rate

The kinematic relations makes it difficult to take the gradient of the cost function continuously, therefore, the term is approximated numerically by the finite difference method. The stopping criteria is defined based on a threshold for the gradient norm as following

$$|\nabla C(\chi)| < \varepsilon$$

Initial parameters are chosen randomly, within a reasonable range. The set of final parameters that yielded the minimum cost function after a number of trials are chosen as the design parameters for the project.

Cam Design

The feasibility of the resulting cam profile from the kinematic analysis on MATLAB is assessed based on many quantitative and qualitative metrics related with the size and performance. Most notable factors are determined as the pressure angle and maximum resultant contact pressure. After determining the pitch curve, the cam profile is generated considering the follower body. A pitch curve is defined with respect to the cam center that is chosen. Within the context of the kinematics, it is useful to define parameters of angular offset and base radius of the cam instead of testing arbitrary cam center points, therefore, the cam center is described by three parameters alone:

1. Angular offset (φ_c): The angle between the line along the slider direction and the line connecting the slider end and the cam center at master angle $= 0$
2. Cam-base radius(R_b): The radius of the cam during the zero-dwell phase
3. Follower radius(R_c): The radius of the follower.

By preliminary analyses conducted based on the performance of the cam, said parameters are selected as:

$$\varphi_c = 24.3°$$

$$R_b = 30mm$$

$$R_c = 5mm$$

The resulting cam profile is shown in figure 3 below:

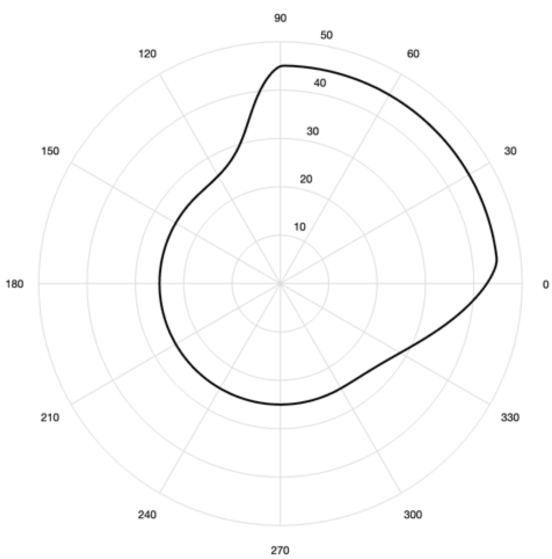


Figure 3: The profile of the resulting cam

The pressure angle at any given master angle can be calculated considering the slider direction as stationary and accounting the change in the normal vector of the contact point. The pressure angle in this case can be displayed with respect to the master angle as shown in figure 4:

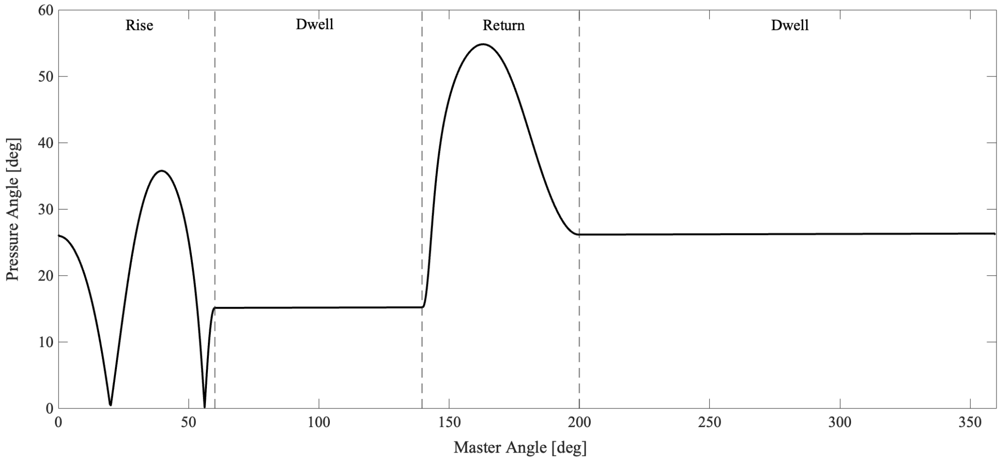


Figure 4: Pressure angle with respect to the master angle

From the Hertz relationship, the contact pressure of the cam can be estimated assuming a constant load of $F = F_{max} \approx 170N$ according to the max load experienced by the cam. To apply the relationship, the radius of curvature of the contact point given the master angle must be solved. The following figure shows the relationship between the curvature of the contact and the master angle as:

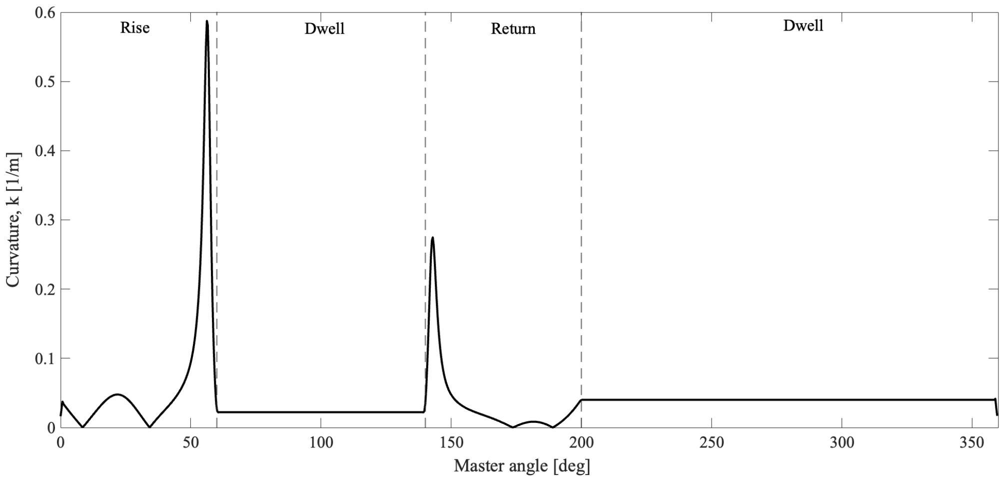


Figure 5: Curvature with respect to the master angle

Applying the relationship:

$$p = \sqrt{0.175\,\frac{SE}{bR_r}\left(1 + \frac{R_r}{\rho}\right)} \qquad (6)$$

Where:

θ_p: Pressure angle (see fig. 4)

$\rho = \frac{1}{k}$: Radius of curvature (see fig. 5)

$S = \frac{F}{\cos\theta_p}$: Force between the cam and the follower

$E \cong 200GPa$: Modulus of elasticity of steel

$b = 0.02\ m$: The thickness of the cam

$R_r = 0.005\ m$: The radius of the follower

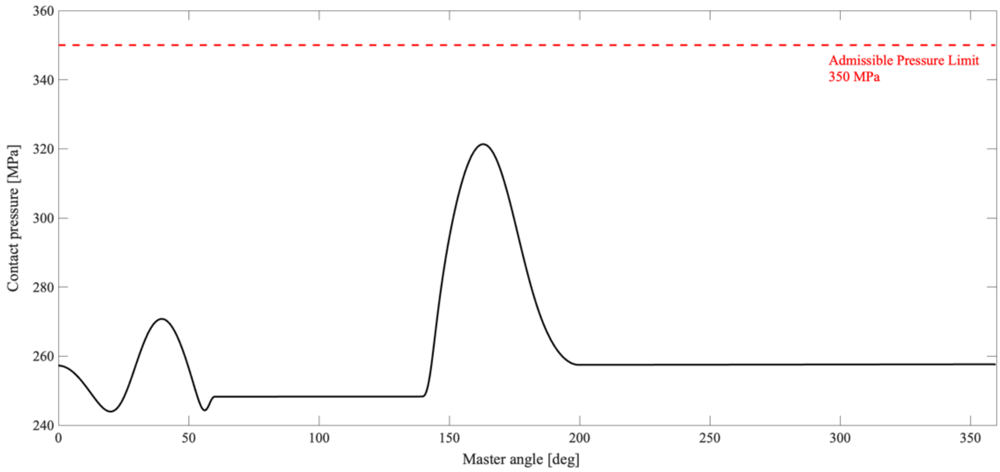


Figure 6: Contact pressure with respect to the master angle

Therefore, it is clear that a thickness of 2cm for the cam is suitable for the operation of the mechanism even in the highest contact force case.

Kinematic Model

To create the kinematic model in MSC Adams, the initial positions of the points shown in Figure 2 were used. Using these points, we created seven links, a circle, and a cylinder to represent the embossing die.

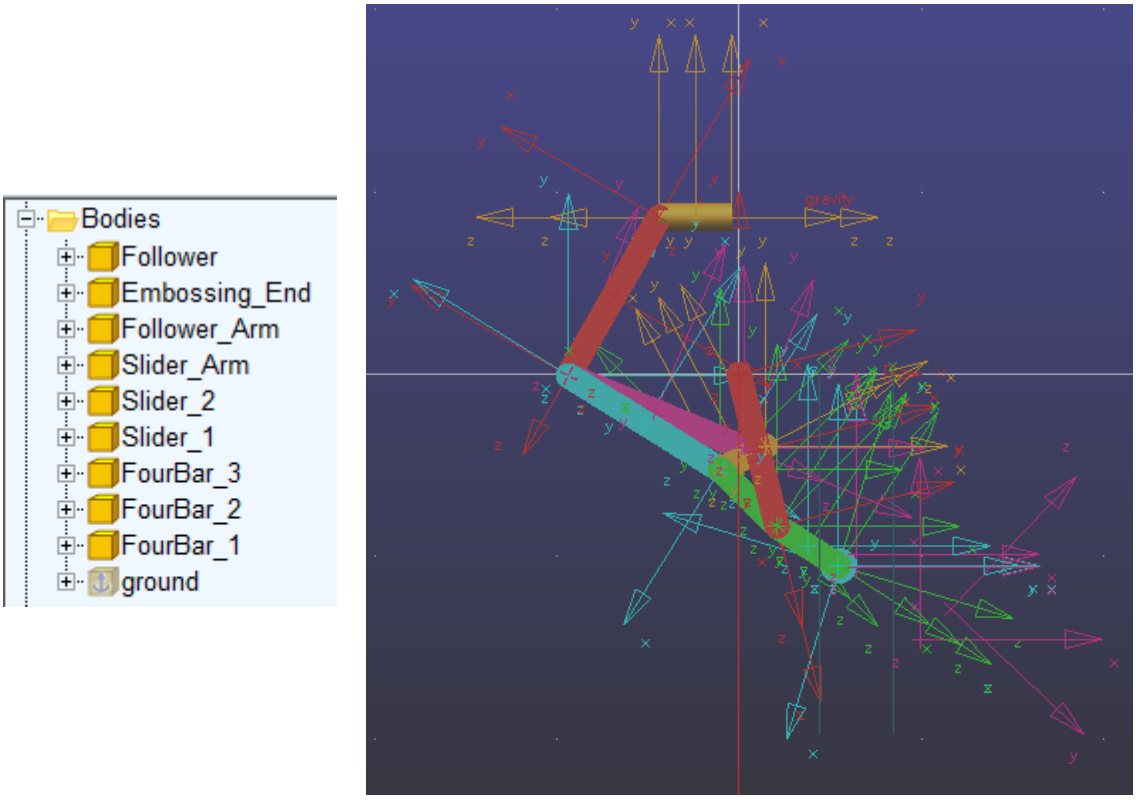


Figure 7: Bodies in Adams View

The connections between the bodies are as follows:

- **Revolute Joints**: Three revolute joints are used to connect the ground to the links and the cam (two for the four-bar mechanism and one for the slider-crank mechanism).
- **Translational Joints**: For kinematic purposes, the follower circle and embossing end are connected to the ground with translational joints.
- **Hooke Joints**: Four Hooke joints are used:
  - Between the two four-bar links
  - Between the four-bar and the slider-crank
  - Between the slider arm and the embossing end
  - Between the follower circle and the follower arm
- **Spherical Joints**: Four spherical joints are used:
  - Between the two four-bar links
  - Between the two slider links
  - Between the slider link and the slider arm
  - Between the four-bar and the follower arm

By applying the required motion curve given in Table 2 to the translational joint on the embossing end the model is verified prior of addition the cam. Model does not have any degrees of freedom left. Also, there are no redundant constraint equations.

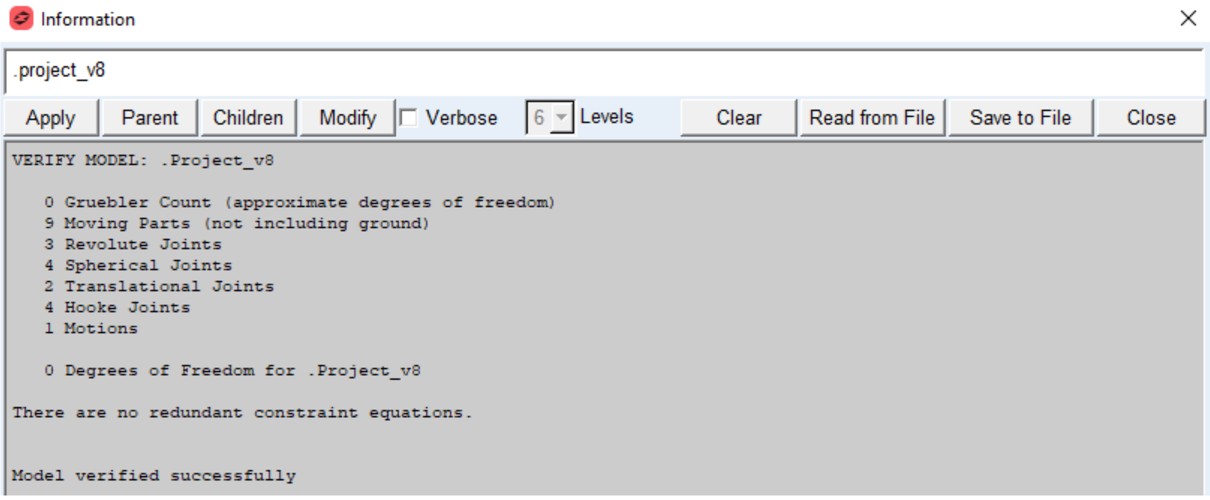


Figure 8: Information window in Adams View

Creating the Cam:

A dummy body was created for the cam and connected to the ground with a revolute joint. An angular motion of 180 rpm was defined for this revolute joint. Using the trace tool in MSC Adams, the outer surface curve was then created on the cam body. By applying a curve-to-curve constraint between the cam curve and the follower circle, the kinematic model is completed successfully. After disabling the motion on the embossing end, the mechanism can be driven by the cam rotation. For visual purposes (for kinematic case), cam curve is extruded, and cylinder is added to follower body.

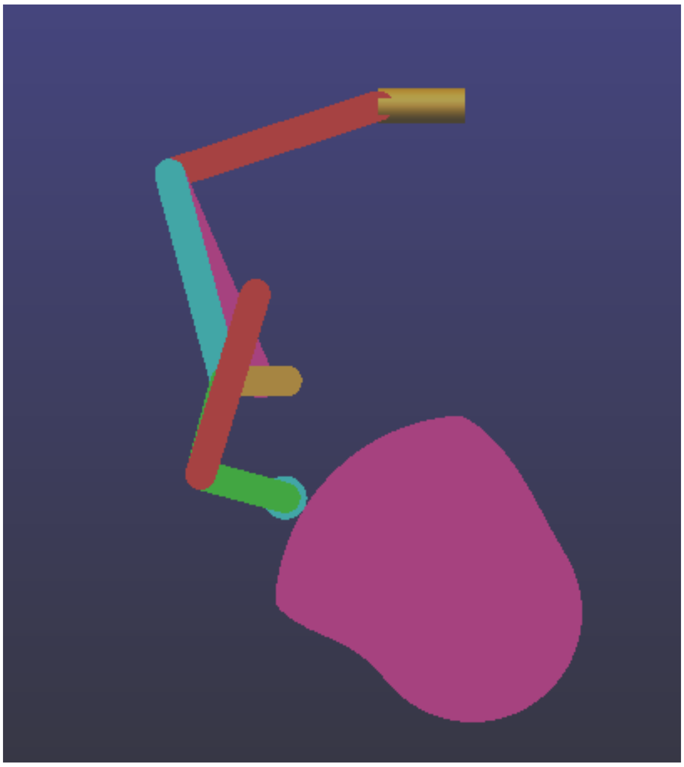


Figure 9: Kinematic model in Adams View

However, due to the curve-to-curve constraint, Z-X rotation of the cam becomes redundant. It is solved by removing the revolute joint on the cam and changing it with spherical and perpendicular primitive (X and Z axis selected). Motion is given to the cam by a general point motion. With the addition to the previous joints, model verification can be seen as:

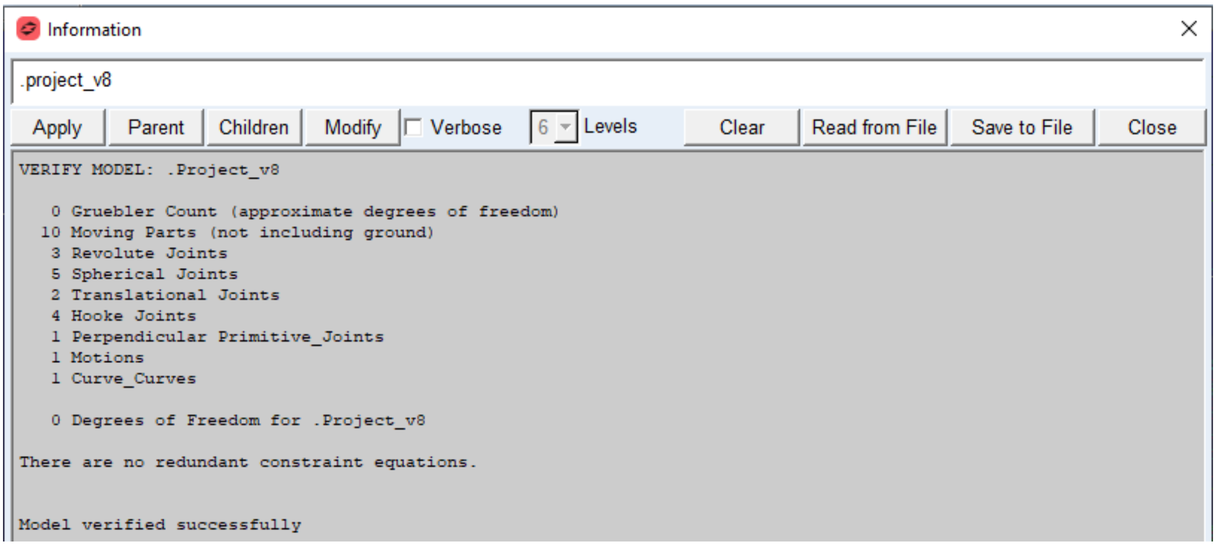


Figure 10: Information window in Adams View

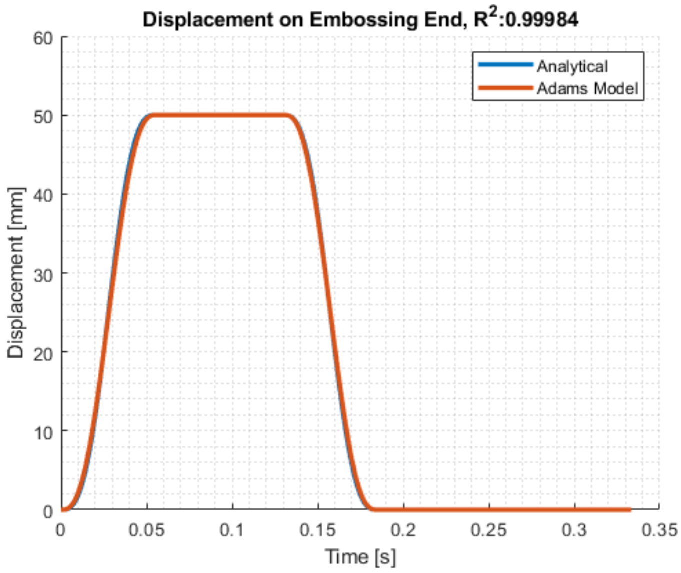
Results

After building the kinematic model, the results could be checked and compared with the analytical requirements.

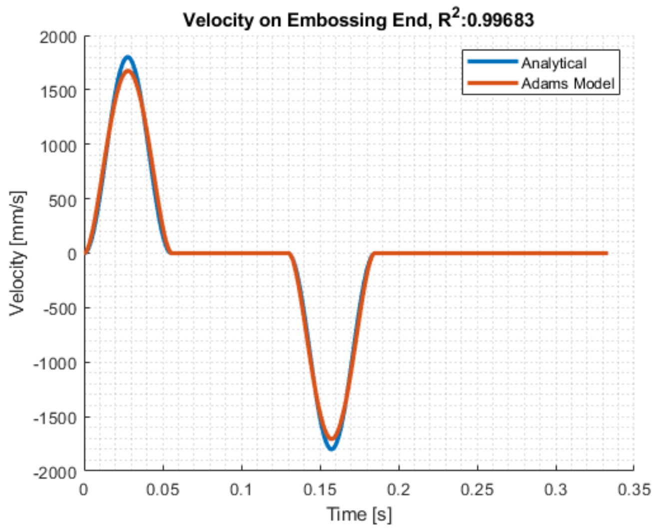
Analytical solution of the motion curve in Table 2 is modeled with a cycloidal motion curve because it does not have any discontinuities. Acceleration, velocity and displacement is analytically modeled with following equations:

$$\begin{gathered} y = h\left[\frac{1}{t_a} - \frac{1}{2\pi}\sin\left(\frac{2\pi t}{t_a}\right)\right] \\ \dot{y} = \frac{h}{t_a}[1 - \cos\left(\frac{2\pi t}{t_a}\right) \\ \ddot{y} = c_a\frac{h}{t_a^2}\sin\left(\frac{2\pi t}{t_a}\right) \end{gathered} \tag{2}$$

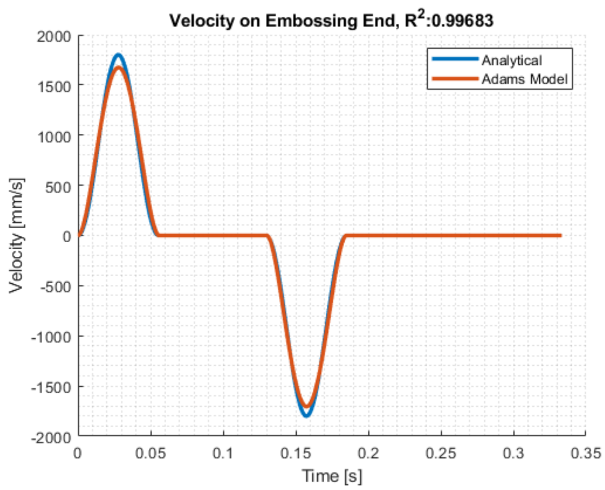
The Adams results are compared to the MATLAB as shown below:



(a) Displacement



(b) Velocity



(c) Acceleration

Figure 11: Comparison of analytical and kinematic model

As seen in Figure 11, two results are close with very high (>0.99) coefficient of determination (R^2). Most of the differences are around maximum and minimum values. So, we can check the max-min values:

KINEMATIC

	Minimum	Maximum	Minimum Error (%)	Maximum Error (%)
Analytical Displacement [mm]	0.00	50.00	0.00	0.00
Adams Displacement [mm]	0.00	50.00		
Analytical Velocity [mm/s]	-1799.97	1799.93	5.31	7.00
Adams Velocity [mm/s]	-1704.47	1673.91		
Analytical Acceleration [mm/s²]	-101787.60	101786.71	6.26	6.26
Adams Acceleration [mm/s²]	-95415.90	95412.18		

Table 3: Comparison of maximum and minimum values between analytical and kinematic model

As seen from the results displayed no significant difference is presentbetween the two models.

Dynamic Model

Cylinder and cam body

Constraints on the follower changed

Constraints on the cam changed

Spring added

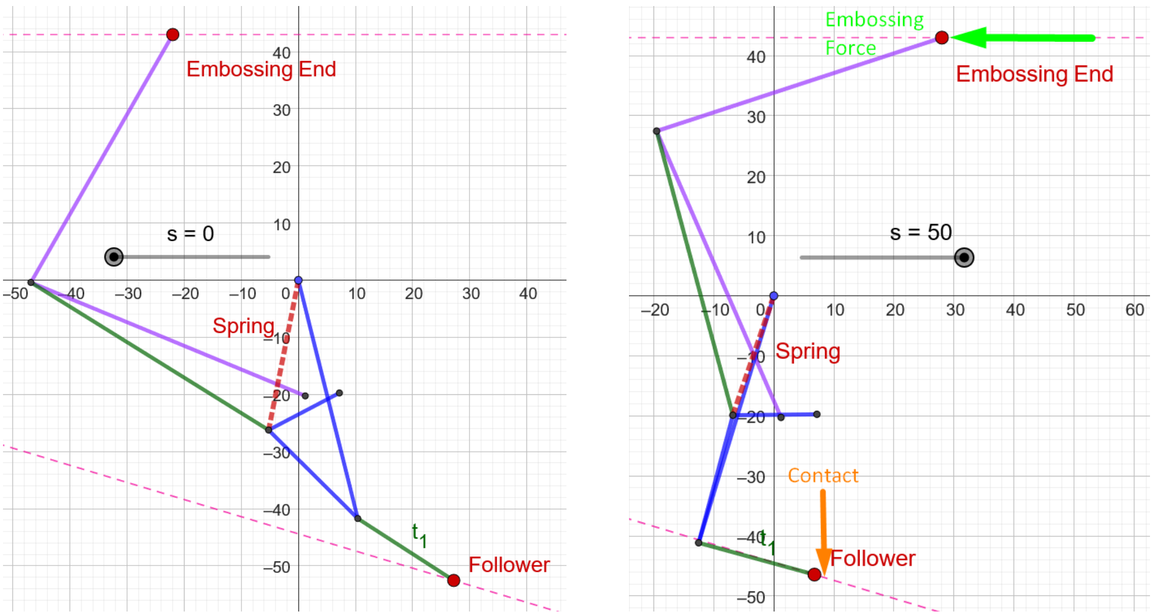
To create the dynamic model, in addition to the kinematic model, three forces are added. Also two connectors are changed while disabling the curve-to-curve constraints.

Forces can be listed as:

- An embossing force is applied to the embossing end in the opposite direction of displacement. Step functions are defined to activate this force 5 ms before the embossing end reaches 50 mm, with the force reaching its maximum value over 15 ms. For the release of the force, the deactivation begins 10 ms before the embossing end returns from the 50 mm displacement, reducing to zero over 15 ms. The maximum force applied is 600 N, as specified in the objective.

- Contact between the cam and follower bodies is added. The follower is designed as a roller, allowing it to rotate freely. To model this rotation, Coulomb friction is activated in this contact. This contact enables the cam to drive the system. With the change from a curve-to-curve constraint to contact, the degree of freedom of the cam changes, necessitating the use of different joints. For the contact solution, it is more effective to use sphere-extrude contact to utilize the analytical solution of a circle, rather than using extrude-extrude contact. Therefore, the follower is modeled with the sphere command in Adams, not with the extrude command.
- With this type of contact, the force closed cam in the mechanism will only be able to push the follower. To maintain continuous contact, a spring should be added to the mechanism. We decided to use a linear spring positioned between two joints on the four-bar linkage. With this configuration, the spring can provide contact pressure during the rise, return, and second dwell phases. During the first dwell phase (embossing phase), the spring force is approximately perpendicular to the follower arm, so it does not significantly increase the pressure on the cam surface. As a result, we do not expect a major increase in force on the cam surface during the embossing phase due to the spring.

These forces can be seen Figure 12 below:



(a) Initial Position *(b) Embossing Position*

Figure 12: Forces added for dynamic model

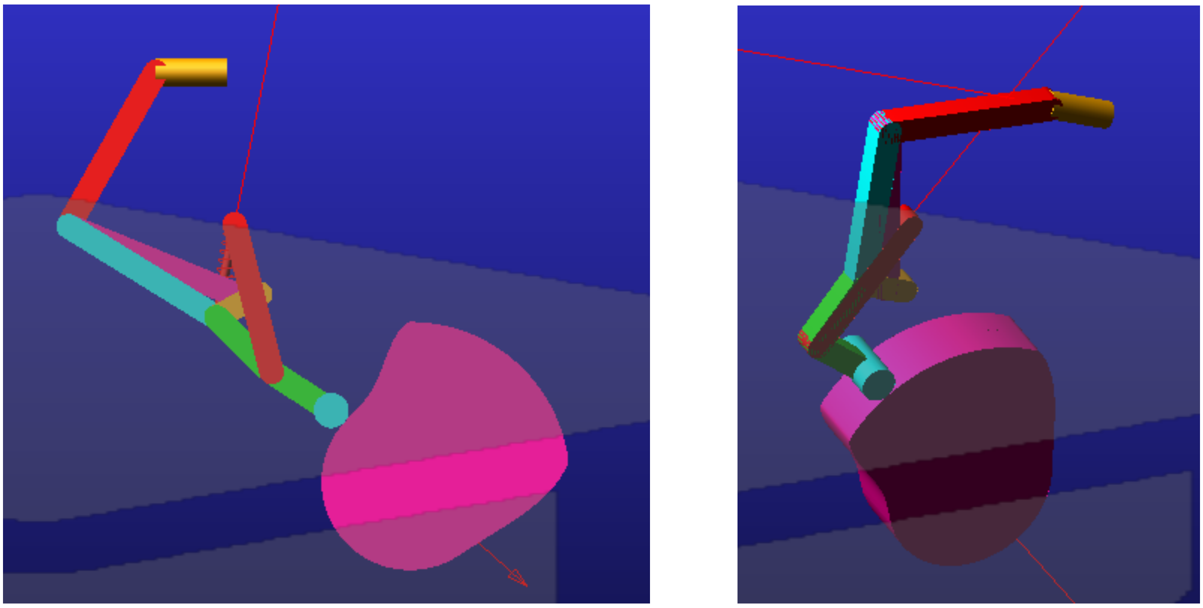
After defining the contact force, thickness of the cam and follower is decided to be 20 mm by using the following contact pressure equation (see section cam design)

Cam and follower chosen as a steel with properties $p_{amm} = 350\ MPa$ and $E = 207\ GPa$.

Modified joints can be listed as:

- In the dynamic model, the follower must be able to rotate. Therefore, the translational joint should be modified. The follower should have two degrees of freedom: it should be able to move along a line and rotate around an axis perpendicular to that line. Thus, the translational joint on the follower is replaced with inline and parallel axis primitives.
- Because of the removal of the curve-to-curve connector, spherical joint and perpendicular primitive is replaced with a revolute joint on cam.

The model can visualized such as:



(a) Initial Position *(b) Embossing Position*

Figure 13: Dynamic model in Adams View

The dynamic model is verified as shown:

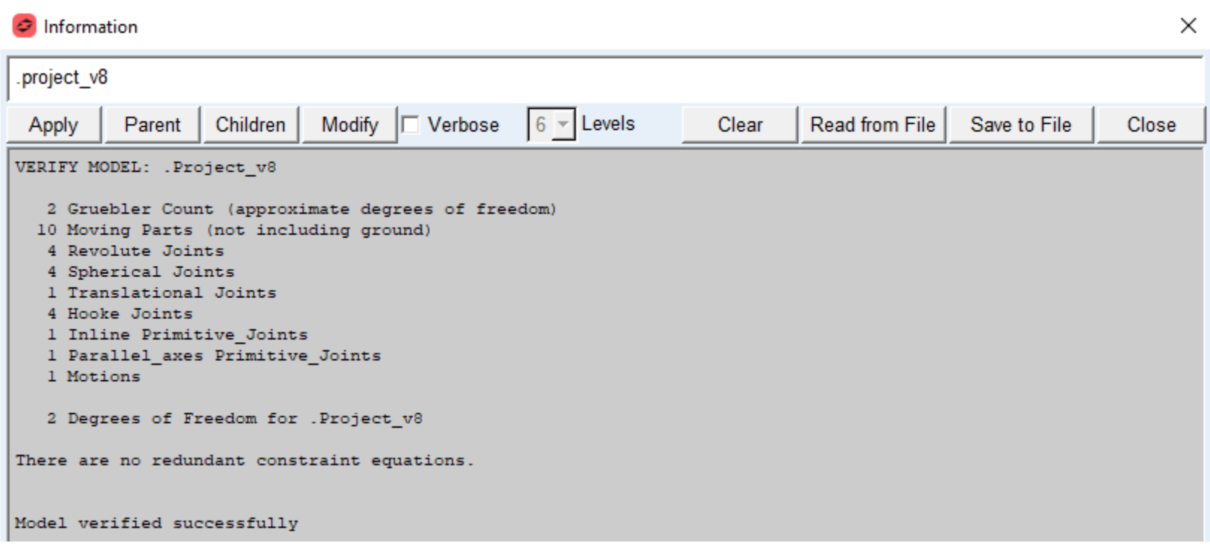


Figure 14: Information window in Adams View

These two degrees of freedom in dynamic model are:

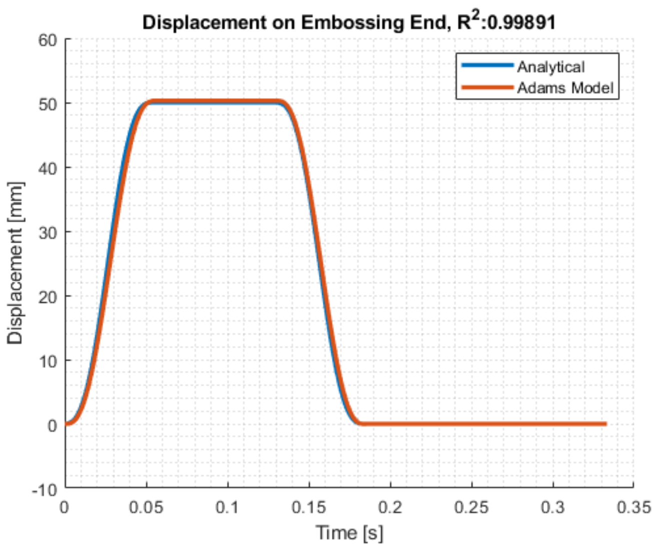
- Movement of the embossing end (Mechanism)
- Rotation of the follower

There are no redundant constraints.

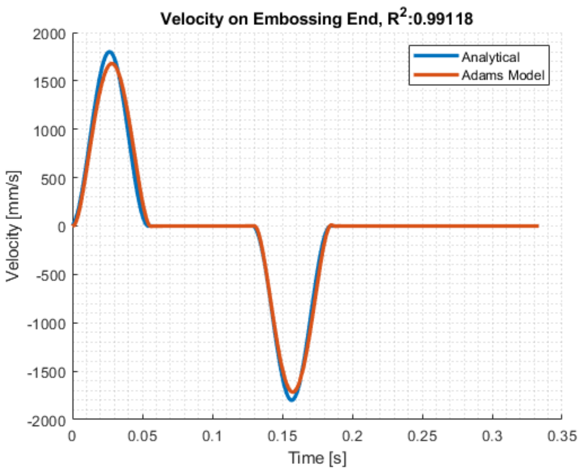
Results:

After building the kinematic model, we can check the results and compare them with the analytical requirements. We can use the same analytical model defined in kinematic part with equation (2)

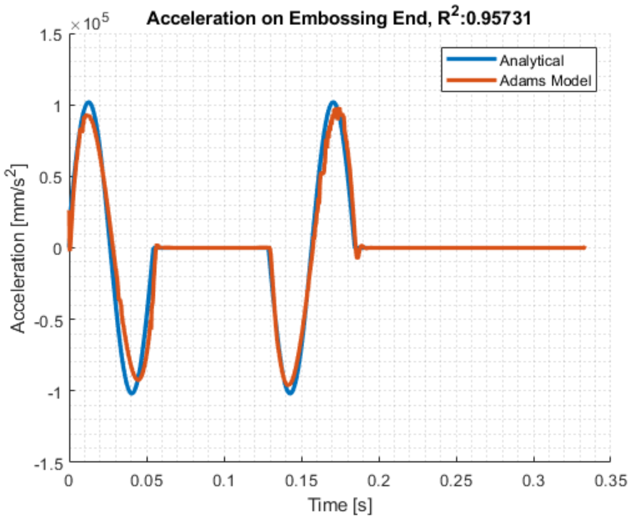
After we imported the displacement, velocity, acceleration, contact force and torque on cams to MATLAB, we can compare the first three with analytical model and analyze other two.



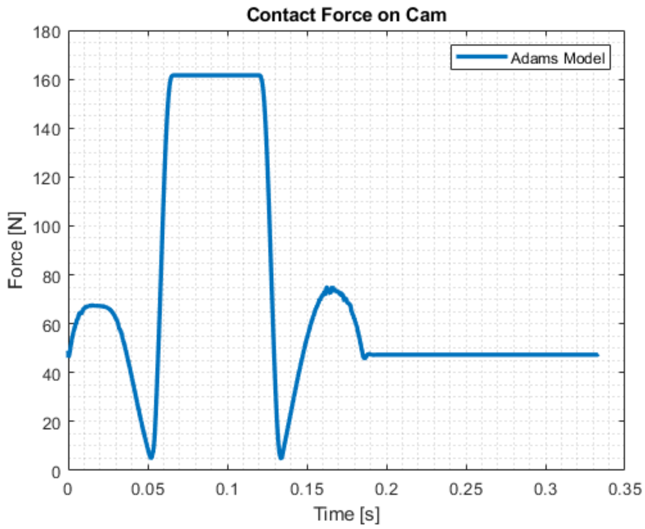
(a) Displacement



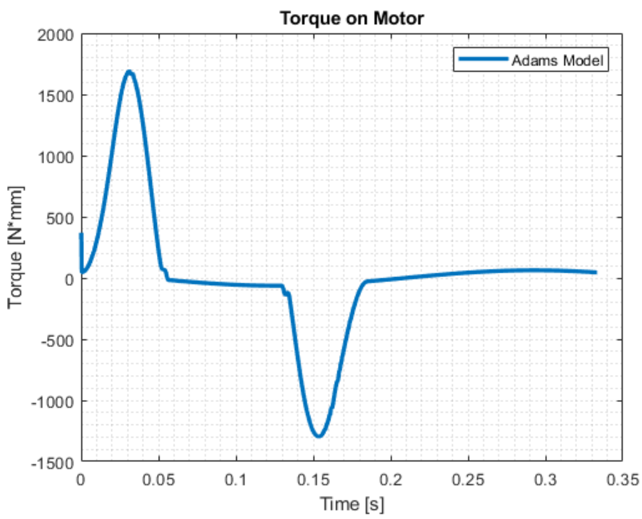
(b) Velocity



(c) Accelaration



(d) Contact force on cam



(e) Torque on motor

Figure 15: Comparison and results of dynamic model

As seen in Figure 15, the results of the two models are closely aligned in terms of displacement and velocity, with a very high coefficient of determination ($R^2 > 0.99$). However, for acceleration, the R^2 is slightly lower, around 0.96. This discrepancy is due to the introduction of contact into the model, which is more complex to utilize than a curve-to-curve connection. Achieving perfect results requires finer calibration of the spring parameters, position, and cam shape.

Contact force reaches its maximum value in embossing phase as expected. It reaches almost null values just before and after the embossing phase.

Most of the differences occur around the maximum and minimum values. These maximum values can be used in further calculations. Therefore, its possible to examine the max-min values:

DYNAMIC

	Minimum	**Maximum**	**Error on Minimum**	**Error on Maximum**
Analytical Displacement	0.00	50.00	0	0.01
Adams Displacement	-0.02	50.27		
Analytical Velocity	-1799.99	1799.93	0.05	0.07
Adams Velocity	-1715.50	1678.47		
Analytical Acceleration	-101782.07	101787.38	0.05	0.04
Adams Acceleration	-96443.51	98204.72		
Contact Force	4.94	161.59		
Torque on Motor	-1300.44	1691.58		

Conclusion

Based on given criteria, the development of a mechanism that is capable of high-speed embossing is developed from synthesis to dynamical model. The tools developed to accomplish the given task can be utilized for development of similar mechanisms utilizing different kinds of planar mechanisms and driven by a cam, as explained in the prior sections. The feasibility of manufacturing and assembling such machine requires further analysis to determine the remaining parameters that were beyond the scope of the report (e.g. the thickness of the links). Ultimately, the linkage mechanism can be thought of as an intermediary, whose task is to manipulate the loads exerted by the output, in such a manner that results in a better input characteristic, in this case lets the cam to be relatively smaller compared to the direct implementation.

Appendix:

Analytical kinematic relations

When the close loop is chosen to have two unknown angles, it is possible to eliminate either one of the unknowns by summing the squares of real and imaginary parts. This arrangment yields an equation in the form of the equation below,

$$w_1 \sin\alpha + w_2 \cos\alpha = C$$

Where w_1, w_2 and c can be expressed as other known variables such as link lengths and angles corresponding to the angular position of the cam.

$$\sin\alpha = \left(2\tan\frac{\alpha}{2}\right)\left(1 + 2\tan^2\frac{\alpha}{2}\right)^{-1}$$

$$\cos\alpha = \left(1 - \tan^2\frac{\alpha}{2}\right)\left(1 + \tan^2\frac{\alpha}{2}\right)^{-1}$$

By applying the double angle formulae below, it is possible to rearrange terms to obtain a 2nd order polynomial as below.

$$(C + w_2)\tan^2\frac{\alpha}{2} - 2w_1\tan\frac{\alpha}{2} + (C - w_2) = 0$$

$$\therefore \alpha = 2\left(\arctan\left(\frac{w_1 \pm \sqrt{w_1^2 + w_2^2 - C^2}}{w_2 + C}\right) + \pi n\right), \;\; n \epsilon N^+$$

The two solutions of the angle α where $n = 0$ represents the open and cross configuration of the mechanism. In literature, it is common to distinguish those configurations as shown in equation below.

$$\alpha = 2\left(\arctan\left(\frac{w_1 + \sigma\sqrt{w_1^2 + w_2^2 - C^2}}{w_2 + C}\right) + \pi n\right), \;\; n \epsilon N^+$$

open configuration: $\sigma = 1$

cross configuration: $\sigma = -1$

If there are no real solutions for the angle α there is no angle that satisfies the position of the mechanism.

By applying the equation, it is possible to obtain every angle of the mechanism as a function of the link lengths and the cam position. Since the output slider position is provided, analytical solution is obtained starting from the angle corresponding to the link that is connected to the output slider:

1. $A_0 B_0 C_0 C S A_0$

$$g_1 e^{i\alpha_a} + g_2 e^{i\alpha_b} + c e^{i\theta_c} + k_2 e^{i\theta_{k_2}} - s_0 - s(\psi) - iY = 0$$

Solving for θ_{k_2}:

$$\begin{Bmatrix} Re \\ Im \end{Bmatrix} = \begin{Bmatrix} c\cos\theta_c = \big(s_0 + s(\psi) - (g_1\cos\alpha_a + g_2\cos\alpha_b)\big) - k_2\cos\theta_{k_2} \\ c\sin\theta_c = \big(Y - (g_1\sin\alpha_a + g_2\sin\alpha_b)\big) - k_2\sin\theta_{k_2} \end{Bmatrix}$$

$$c^2 = \Big(\big(s_0 + s(\psi) - (g_1\cos\alpha_a + g_2\cos\alpha_b)\big) - k_2\cos\theta_{k_2}\Big)^2 + \Big(\big(Y - (g_1\sin\alpha_a + g_2\sin\alpha_b)\big) - k_2\sin\theta_{k_2}\Big)^2$$

$$\theta_{k_2} = 2\arctan\left(\frac{w_1 - \sqrt{w_1^2 + w_2^2 - C^2}}{w_2 + c}\right)$$

Where:

$w_1 = -2k_2(Y - g_1 \sin\alpha_a - g_2 \sin\alpha_b)$

$w_2 = -2k_2\big(s_0 + s(\psi) - (g_1 \cos\alpha_a + g_2 \cos\alpha_b)\big)$

$C = c^2 - k_2^2 - \big(Y - (g_1 \sin\alpha_a + g_2 \sin\alpha_b)\big)^2 - \big(s_0 + s(\psi) - (g_1 \cos\alpha_a + g_2 \cos\alpha_b)\big)^2$

Solving for θ_c:

$$\begin{Bmatrix} Re \\ Im \end{Bmatrix} = \begin{Bmatrix} k_2 \cos\theta_{k_2} = \big(s_0 + s(\psi) - (g_1 \cos\alpha_a + g_2 \cos\alpha_b)\big) - c\cos\theta_c \\ k_2 \sin\theta_{k_2} = \big(Y - (g_1 \sin\alpha_a + g_2 \sin\alpha_b)\big) - c\sin\theta_c \end{Bmatrix}$$

$$k_2^2 = \Big(\big(s_0 + s(\psi) - (g_1 \cos\alpha_a + g_2 \cos\alpha_b)\big) - c\cos\theta_c\Big)^2 + \Big(\big(Y - (g_1 \sin\alpha_a + g_2 \sin\alpha_b)\big) - c\sin\theta_c\Big)^2$$

$$\theta_c = 2\arctan\left(\frac{w_1 + \sqrt{w_1^2 + w_2^2 - C^2}}{w_2 + c}\right)$$

Where:

$w_1 = -2c(Y - g_1 \sin\alpha_a - g_2 \sin\alpha_b)$

$w_2 = -2c\big(s_0 + s(\psi) - (g_1 \cos\alpha_a + g_2 \cos\alpha_b)\big)$

$C = k_2^2 - c^2 - \big(Y - (g_1 \sin\alpha_a + g_2 \sin\alpha_b)\big)^2 - \big(s_0 + s(\psi) - (g_1 \cos\alpha_a + g_2 \cos\alpha_b)\big)^2$

2. $B_0C_0CBB_0$

$$g_2 e^{i\alpha_b} + c e^{i\theta_c} - h_2 e^{i\theta_{h_2}} - b e^{i\theta_b} = 0$$

Solving for θ_b:

$$\begin{Bmatrix} Re \\ Im \end{Bmatrix} = \begin{Bmatrix} h_2 \cos\theta_{h_2} = (g_2 \cos\alpha_b + c\cos\theta_c) - b\cos\theta_b \\ h_2 \sin\theta_{h_2} = (g_2 \sin\alpha_b + c\sin\theta_c) - b\sin\theta_b \end{Bmatrix}$$

$$h_2^2 = \big((g_2 \cos\alpha_b + c\cos\theta_c) - b\cos\theta_b\big)^2 + \big((g_2 \sin\alpha_b + c\sin\theta_c) - b\sin\theta_b\big)^2$$

$$\theta_b = 2\arctan\left(\frac{w_1 + \sqrt{w_1^2 + w_2^2 - C^2}}{w_2 + c}\right)$$

Where:

$w_1 = -2b(g_2 \sin\alpha_b + c\sin\theta_c)$

$w_2 = -2b(g_2 \cos\alpha_b + c\cos\theta_c)$

$C = h_2^2 - b^2 - (g_2 \sin\alpha_b + c\sin\theta_c)^2 - (g_2 \cos\alpha_b + c\cos\theta_c)^2$

Solving for θ_{h_2}:

$$\begin{Bmatrix} Re \\ Im \end{Bmatrix} = \begin{Bmatrix} b\cos\theta_b = (g_2\cos\alpha_b + c\cos\theta_c) - h_2\cos\theta_{h_2} \\ b\sin\theta_b = (g_2\sin\alpha_b + c\sin\theta_c) - h_2\sin\theta_{h_2} \end{Bmatrix}$$

$$b^2 = \left((g_2\cos\alpha_b + c\cos\theta_c) - h_2\cos\theta_{h_2}\right)^2 + \left((g_2\sin\alpha_b + c\sin\theta_c) - h_2\sin\theta_{h_2}\right)^2$$

$$\theta_c = 2\arctan\left(\frac{w_1 - \sqrt{w_1^2 + w_2^2 - C^2}}{w_2 + c}\right)$$

Where:

$w_1 = -2h_2(g_2\sin\alpha_b + c\sin\theta_c)$

$w_2 = -2h_2(g_2\cos\alpha_b + c\cos\theta_c)$

$C = b^2 - h_2^2 - (g_2\sin\alpha_b + c\sin\theta_c)^2 - (g_2\cos\alpha_b + c\cos\theta_c)^2$

3. $A_0B_0BAA_0$

$$g_1e^{i\alpha_a} + be^{i\theta_b} - h_1e^{i\theta_{h_1}} - ae^{i\theta_a}$$

Solving for θ_a:

$$\begin{Bmatrix} Re \\ Im \end{Bmatrix} = \begin{Bmatrix} h_1\cos\theta_{h_1} = (g_1\cos\alpha_a + b\cos\theta_b) - a\cos\theta_a \\ h_1\sin\theta_{h_1} = (g_1\sin\alpha_a + g_2\sin\theta_b) - a\sin\theta_a \end{Bmatrix}$$

$$h_1^2 = \left((g_1\cos\alpha_a + b\cos\theta_b) - a\cos\theta_a\right)^2 + \left((g_1\sin\alpha_a + g_2\sin\theta_b) - a\sin\theta_a\right)^2$$

$$\theta_a = 2\arctan\left(\frac{w_1 + \sqrt{w_1^2 + w_2^2 - C^2}}{w_2 + c}\right)$$

Where:

$w_1 = -2a(g_1\sin\alpha_a + b\sin\theta_b)$

$w_2 = -2a(g_1\cos\alpha_a + b\cos\theta_b)$

$C = h_1^2 - a - (g_1\sin\alpha_a + b\sin\theta_b)^2 - (g_1\cos\alpha_a + b\cos\theta_b)^2$

Solving for θ_{h_1}:

$$\begin{Bmatrix} Re \\ Im \end{Bmatrix} = \begin{Bmatrix} a\cos\theta_a = h_1\cos\theta_{h_1} - (g_1\cos\alpha_a + b\cos\theta_b) \\ a\sin\theta_a = h_1\sin\theta_{h_1} - (g_1\sin\alpha_a + g_2\sin\theta_b) \end{Bmatrix}$$

$$a^2 = \left(h_1\cos\theta_{h_1} - (g_1\cos\alpha_a + b\cos\theta_b)\right)^2 + \left(h_1\sin\theta_{h_1} - (g_1\sin\alpha_a + g_2\sin\theta_b)\right)^2$$

$$\theta_{h_1} = 2\arctan\left(\frac{w_1 - \sqrt{w_1^2 + w_2^2 - C^2}}{w_2 + c}\right)$$

Where:

$w_1 = -2h_1(g_1\sin\alpha_a + g_2\sin\theta_b)$

$w_2 = -2h_1(g_1\cos\alpha_a + b\cos\theta_b)$

$C = a^2 - h_1^2 - (g_1\sin\alpha_a + g_2\sin\theta_b)^2 - (g_1\cos\alpha_a + b\cos\theta_b)^2$

4. A_0APA_0

$$a\,e^{i\theta_a} - k_1 e^{i\theta_{k_1}} - x - i(mx + n) = 0$$

Solving for θ_{k_1}:

$$\begin{Bmatrix} Re \\ Im \end{Bmatrix} = \begin{Bmatrix} x = a\cos\theta_a - k_1\cos\theta_{k_1} \\ x = \left(a\sin\theta_a - k_1\sin\theta_{k_1} - n\right)/m \end{Bmatrix}$$

$$a\cos\theta_a - k_1\cos\theta_{k_1} = \frac{a\sin\theta_a - k_1\sin\theta_{k_1} - n}{m}$$

$$\theta_{k_1} = 2\arctan\left(\frac{w_1 + \sqrt{w_1^2 + w_2^2 - C^2}}{w_2 + c}\right)$$

Where:

$w_1 = -k_1$

$w_2 = mk_1$

$C = -a\sin\theta_a + ma\cos\theta_a + n$

Position of the input slider A_0AP

$$a\,e^{i\theta_a} - k_1 e^{i\theta_{k_1}} = P_x + iP_y$$

$$P_x = a\cos\theta_a - k_1\cos\theta_{k_1}$$

$$P_y = a\sin\theta_a - k_1\sin\theta_{k_1}$$

Absolute displacement of the input slider becomes

$$|P| = \left(a\cos\theta_a - k_1\cos\theta_{k_1}\right)^2 + \left(a\sin\theta_a - k_1\sin\theta_{k_1}\right)^2$$

$$|P| = a^2 + k_1^2 - 2ak_1\cos\left(\theta_a - \theta_{k_1}\right)$$

Parameter Optimization Code:

```
folder = '/Users/cemgungor/Documents/MATLAB/Solver/Project';
fullMatFileName = fullfile(folder,  's.mat');
if ~exist(fullMatFileName, 'file')
  message = sprintf('%s does not exist', fullMatFileName);
  uiwait(warndlg(message));
else
  s = load(fullMatFileName);
end

optimal = s.s.pos;  optimal(optimal==50)=[]; optimal(optimal==0)=[];
optimal = 5.2-(optimal/10);  optimal =
[optimal(1:length(optimal)/2),optimal(length(optimal)/2),optimal(length(optimal)/2+
1:end),optimal(end)];
optimal(167-20:167+20) = []; optimal =
[ones(1,20)*optimal(1),optimal,ones(1,20)*optimal(1)];
optimal = [optimal,optimal(1)];

step = 0.001;
LN = 0.0001;
tol = 1e-5;
record = [];

stdev_resultant = 5;
while stdev_resultant > 1 || isnan(stdev_resultant)

    stdev_init = 100;

    while stdev_init > 30
        starting_point = num2cell([40.5   10.3   50.4   12.3   -5.5   33.4   46
59   62.6    0.0928  -47.5907   -10   57   -0.9138   0]);
        [a,b,c,g1,g2,h1,h2,k1,k2,m,n,s_0,y,alpha_a,alpha_b] =
deal(starting_point{:});
        stdev_init =
stdev_val(a,b,c,g1,g2,h1,h2,k1,k2,m,n,s_0,y,alpha_a,alpha_b,optimal,s);
    end

    % 40.5   10.3   50.4   12.3   -5.5   33.4   46   59   62.6    0.0928  -47.5907
-10   57   -0.9138   0
    % [50*rand(1,9),0,-30*rand(1,1),0,30*rand(1,1),rad2deg(270),rad2deg(90)]

    err = 1; i = 0;

    while err > tol

    i = i+1;

        A1 =
[a(i),b(i),c(i),g1(i),g2(i),h1(i),h2(i),k1(i),k2(i),m(i),n(i),s_0(i),y(i),alpha_a(i
),alpha_b(i)];

        for index = 1:length(A1)

            A2 = zeros(1,length(A1)); A2(index) = step;

            A_u = A1 + A2; A_d = A1 - A2;
            A_up = num2cell(A_u); A_down = num2cell(A_d);
            [stdev_up,~] = stdev_val(A_up{:},optimal,s);
            [stdev_down,~] = stdev_val(A_down{:},optimal,s);

            gradient = (stdev_up-stdev_down)/step;
            G(index) = -LN*gradient;
            record = [record,stdev_up]; disp(stdev_up);
```

```
        end

        err = sum(abs(G))/length(G);
        N = A1 + G;
        N_cell = num2cell(N);


[a(i+1),b(i+1),c(i+1),g1(i+1),g2(i+1),h1(i+1),h2(i+1),k1(i+1),k2(i+1),m(i+1),n(i+1)
,s_0(i+1),y(i+1),alpha_a(i+1),alpha_b(i+1)] = deal(N_cell{:});

        N = [];

    end

    stoppoint =
[a(end),b(end),c(end),g1(end),g2(end),h1(end),h2(end),k1(end),k2(end),m(end),n(end)
,s_0(end),y(end),alpha_a(end),alpha_b(end)];
    stoppoint_cell = num2cell(stoppoint);

    [stdev_resultant,TR_resultant] = stdev_val(stoppoint_cell{:},optimal,s);
    clear a b c g1 g2 h1 h2 k1 k2 m n s_0 y alpha_a alpha_b

end

function [stdev,S] =
stdev_val(a,b,c,g1,g2,h1,h2,k1,k2,m,n,s_0,y,alpha_a,alpha_b,optimal,s)

    s = s.s; dds = s.acc; ds = s.vel; s = s_0 + s.pos; % Redefining the terms;

    theta_k2 = angle_solve2(-2*k2*(y - (g1 * sin(alpha_a) + g2 * sin(alpha_b))), -
2*k2*(s - (g1 * cos(alpha_a) + g2 * cos(alpha_b))),...
    (-k2^2 - (y - (g1 * sin(alpha_a) + g2 * sin(alpha_b))).^2 - (s - (g1 *
cos(alpha_a) + g2 * cos(alpha_b))).^2 + c^2),s);

    theta_c =  angle_solve(-2*c*(y - (g1 * sin(alpha_a) + g2 * sin(alpha_b))), -
2*c*(s - (g1 * cos(alpha_a) + g2 * cos(alpha_b))),...
        (-c^2 - (y - (g1 * sin(alpha_a) + g2 * sin(alpha_b))).^2 - (s - (g1 *
cos(alpha_a) + g2 * cos(alpha_b))).^2 + k2^2),s);

    theta_h2  = angle_solve2(-2*h2*(g2 * sin(alpha_b) + c * sin(theta_c)),-2*h2*(g2
*...
        cos(alpha_b) + c * cos(theta_c)),(-h2^2 - (g2 * sin(alpha_b) +...
        c * sin(theta_c)).^2 - (g2 * cos(alpha_b) + c * cos(theta_c)).^2 + b^2),s);

    theta_b = angle_solve(-2*b*(g2 * sin(alpha_b) + c * sin(theta_c)),-2*b*(g2 *...
        cos(alpha_b) + c * cos(theta_c)),(-b^2 - (g2 * sin(alpha_b) +...
        c * sin(theta_c)).^2 - (g2 * cos(alpha_b) + c * cos(theta_c)).^2 +
h2^2),s);

    theta_h1 = angle_solve2(-2*h1*(g1 * sin(alpha_a) + b * sin(theta_b)),-2*h1*(g1
*...
        cos(alpha_a) + b * cos(theta_b)),(-h1^2 - (g1 * sin(alpha_a) +...
        b * sin(theta_b)).^2 - (g1 * cos(alpha_a) + b * cos(theta_b)).^2 + a^2),s);

    theta_a  = angle_solve(-2*a*(g1 * sin(alpha_a) + b * sin(theta_b)),-2*a*(g1
*...
        cos(alpha_a) + b * cos(theta_b)),(-a^2 - (g1 * sin(alpha_a) +...
        b * sin(theta_b)).^2 - (g1 * cos(alpha_a) + b * cos(theta_b)).^2 +
h1^2),s);
    theta_k1 = angle_solve(-k1,m*k1,-a*sin(theta_a)+m*a*cos(theta_a)+n,s);


    P = [(a*cos(theta_a) - k1*cos(theta_k1));(a*sin(theta_a) - k1*sin(theta_k1))];
    Pd = P - P(:,1); Pd = sqrt(Pd(1,:).^2 + Pd(2,:).^2);
```

```
    if max(theta_a) == 0 || max(theta_b) == 0

        stdev = 30;
        S = zeros(1,length(s));

    else
        dp = Pd - [Pd(end),Pd(1:end-1)];
        ds = s - [s(end),s(1:end-1)];
        TR = abs(ds./dp);
        TR(isnan(TR)) = []; optimal = optimal(1:length(TR));

        diff = abs(TR-optimal);
        stdev = std(diff)+mean(diff);
        S = TR;
    end
end

function theta = angle_solve(w1,w2,c,s)
    r = w1.^2+w2.^2-c.^2;

    if min(r) > 0
        theta      = 2 * (atan2(w1 - sqrt(r), w2 + c)+pi);

    else
        theta      = zeros(1,length(s)); disp("the sys has no valid solution!!");

    end
end

function theta = angle_solve2(w1,w2,c,s)
    r = w1.^2+w2.^2-c.^2;

    if min(r) > 0
        theta      = 2 * (atan2(w1 + sqrt(r), w2 + c)+pi);

    else
        theta      = zeros(1,length(s)); disp("the sys has no valid solution!!");

    end
end
```

Animation code + generation of relevant graphs:

```
% Establishing kinematic parameters:
starting_point = num2cell([43 14 52 21 -6 22 49 20 50 -0.3 -44.4 -22 43 deg2rad(-
70) deg2rad(5)]);
[a,b,c,g1,g2,h1,h2,k1,k2,m,n,s_0,y,alpha_a,alpha_b] = deal(starting_point{:});
clear starting_point;

% Loading the embossing slider position data:
folder = '/Users/cemgungor/Documents/MATLAB/Solver/Project';
fullMatFileName = fullfile(folder,  's.mat');
if ~exist(fullMatFileName, 'file')
  message = sprintf('%s does not exist', fullMatFileName);
  uiwait(warndlg(message));
else
  s = load(fullMatFileName);
end

folder = '/Users/cemgungor/Documents/MATLAB/Solver/Project';
fullMatFileName = fullfile(folder,  'adams.mat');
if ~exist(fullMatFileName, 'file')
  message = sprintf('%s does not exist', fullMatFileName);
  uiwait(warndlg(message));
else
  adams = load(fullMatFileName);
end
adams = table2array(adams.Adamsdatas2); adams = adams(1:1000);
s = s.s; dds = s.acc; ds = s.vel; s = s_0 + s.pos; % Redefining the terms;

theta_k2 = angle_solve2(-2*k2*(y - (g1 * sin(alpha_a) + g2 * sin(alpha_b))), -
2*k2*(s - (g1 * cos(alpha_a) + g2 * cos(alpha_b))),...
    (-k2^2 - (y - (g1 * sin(alpha_a) + g2 * sin(alpha_b))).^2 - (s - (g1 *
cos(alpha_a) + g2 * cos(alpha_b))).^2 + c^2),s);

theta_c =  angle_solve(-2*c*(y - (g1 * sin(alpha_a) + g2 * sin(alpha_b))), -2*c*(s
- (g1 * cos(alpha_a) + g2 * cos(alpha_b))),...
    (-c^2 - (y - (g1 * sin(alpha_a) + g2 * sin(alpha_b))).^2 - (s - (g1 *
cos(alpha_a) + g2 * cos(alpha_b))).^2 + k2^2),s);

theta_h2  = angle_solve2(-2*h2*(g2 * sin(alpha_b) + c * sin(theta_c)),-2*h2*(g2
*...
    cos(alpha_b) + c * cos(theta_c)),(-h2^2 - (g2 * sin(alpha_b) +...
    c * sin(theta_c)).^2 - (g2 * cos(alpha_b) + c * cos(theta_c)).^2 + b^2),s);

theta_b = angle_solve(-2*b*(g2 * sin(alpha_b) + c * sin(theta_c)),-2*b*(g2 *...
    cos(alpha_b) + c * cos(theta_c)),(-b^2 - (g2 * sin(alpha_b) +...
    c * sin(theta_c)).^2 - (g2 * cos(alpha_b) + c * cos(theta_c)).^2 + h2^2),s);

theta_h1 = angle_solve2(-2*h1*(g1 * sin(alpha_a) + b * sin(theta_b)),-2*h1*(g1 *...
    cos(alpha_a) + b * cos(theta_b)),(-h1^2 - (g1 * sin(alpha_a) +...
    b * sin(theta_b)).^2 - (g1 * cos(alpha_a) + b * cos(theta_b)).^2 + a^2),s);

theta_a  = angle_solve(-2*a*(g1 * sin(alpha_a) + b * sin(theta_b)),-2*a*(g1 *...
    cos(alpha_a) + b * cos(theta_b)),(-a^2 - (g1 * sin(alpha_a) +...
    b * sin(theta_b)).^2 - (g1 * cos(alpha_a) + b * cos(theta_b)).^2 + h1^2),s);
theta_k1 = angle_solve(-k1,m*k1,-a*sin(theta_a)+m*a*cos(theta_a)+n,s);

P = [(a*cos(theta_a) - k1*cos(theta_k1));(a*sin(theta_a) - k1*sin(theta_k1))];
Pd = sqrt(a^2+k1^2-2*a*k1*cos(theta_a-theta_k1)); Pd = abs(Pd-Pd(end));

alpha = atan2(m,1)+2*pi; V = Pd(200)-Pd;
```

```
gamma_init = deg2rad(-43); Rb = 30; Rf = 5; % cam

Sc = sqrt((V(1) - V).^2 + Rb^2 + 2*Rb*(V(1) - V)*cos(gamma_init));

% Draw the Roller at point A
theta = linspace(0, 2*pi, 100);



%%
clc; close all

A = a*exp(1i*theta_a);
B = A + h1*exp(1i*theta_h1);
C = B + h2*exp(1i*theta_h2);
B_0 = g1*exp(1i*alpha_a);
C_0 = B_0 + g2*exp(1i*alpha_b);
S = C + k2*exp(1i*theta_k2);
P_p = A - k1*exp(1i*theta_k1);

%cam propertier


cam_center = [ P(1,1) + Rb * cos(gamma_init) , P(2,1) + Rb * sin(gamma_init)];

%Dont change, needed to rotate
z2p = linspace(0,2*pi,1000);


distance = zeros(1,length(A));
angle = flip(linspace(0, 2*pi, length(A)))+pi;


for j = 1:length(P_p)

    %calculate distances for each angle
    distance(j) = sqrt( (real(P_p(j)) - cam_center(1) ).^2 + ( imag(P_p(j)) -
cam_center(2) ).^2 );

    %compensate for angle change due to swing
    angle(j) = angle(j) - atan2( (imag(P_p(j)) - cam_center(2)) , abs(real(P_p(j))
- cam_center(1)));


end


%figure
iter = 0;

while iter < 1

    for i = 1:length(theta_k1)

        if mod(i,10) == 0

            %current linkage directions
            A_i = [real(A(i)) imag(A(i))];
            B_i = [real(B(i)) imag(B(i))];
            C_i = [real(C(i)) imag(C(i))];
            B0_i = [real(B_0) imag(B_0)];
            C0_i = [real(C_0) imag(C_0)];
            S_i = [real(S(i)) imag(S(i))];
            P_pi = [real(P_p(i)) imag(P_p(i))];

            [x_prf, y_prf] = pol2cart(angle+z2p(i), distance);
```

```
            [x_prf, y_prf] = camprofile(x_prf,y_prf,Rf);


            % Radius of the circle
            x_circle = P_pi(1) + Rf * cos(theta);
            y_circle = P_pi(2) + Rf * sin(theta);
            plot(x_circle, y_circle,'k','LineWidth', 2); hold on; % Circle at end
of input link
            plot([0, A_i(1)], [0, A_i(2)],'k','LineWidth', 2);

            %cam plot
            plot(x_prf + cam_center(1),y_prf + cam_center(2),'k','LineWidth', 2)

            plot([A_i(1) B_i(1)],[A_i(2) B_i(2)],'k','LineWidth', 2)
            plot([B_i(1) C_i(1)],[B_i(2) C_i(2)],'k','LineWidth', 2)
            plot([C_i(1) S_i(1)],[C_i(2) S_i(2)],'k','LineWidth', 2)
            plot([0 B0_i(1)],[0 B0_i(2)],'k--','LineWidth', 2)
            plot([B0_i(1) C0_i(1)],[B0_i(2) C0_i(2)],'k--','LineWidth', 2)
            plot([C0_i(1) C_i(1)],[C0_i(2) C_i(2)],'k','LineWidth', 2)
            plot([B0_i(1) B_i(1)],[B0_i(2) B_i(2)],'k','LineWidth', 2)
            plot([A_i(1) P_pi(1)],[A_i(2) P_pi(2)],'k','LineWidth', 2)
            plot([-1000,1000],[-1000*m+n, 1000*m+n],'k--','LineWidth', 1)

            %change window limits
            xlim([-80 150])
            ylim([-170 60])
            %plot([cam_center(1), cam_center(1)+100], [cam_center(2),
cam_center(2)])
            axis equal;
            hold off;

            pause(0.00001)

        end
    end
    iter = iter+1;
end

PA = pressure_angle(x_prf,y_prf,m);
rho = radiusofcurv(x_prf,y_prf);

Pc = contact_pressure(170,PA,2e+11,0.02,0.005,rho);


close all;

plot(0:(360/1000):360-(360/1000),Pc,'k','LineWidth', 1); hold on; xlim([0 360])
plot(0:(360/1000):360-(360/1000),350*ones(1,1000),'r--','LineWidth', 2);

xlabel('Master angle [deg]');
ylabel('Contact pressure [MPa]');

figure();
plot(0:(360/1000):360-(360/1000),PA);

disp(max(Pc));



%% Functions
function theta = angle_solve(w1,w2,c,s)
    r = w1.^2+w2.^2-c.^2;

    if min(r) > 0
        theta      = 2 * (atan2(w1 - sqrt(r), w2 + c)+pi);
```

```
    else
        theta      = zeros(1,length(s)); disp("the sys has no valid solution!!");

    end
end

function theta = angle_solve2(w1,w2,c,s)
    r = w1.^2+w2.^2-c.^2;

    if min(r) > 0
        theta      = 2 * (atan2(w1 + sqrt(r), w2 + c)+pi);

    else
        theta      = zeros(1,length(s)); disp("the sys has no valid solution!!");

    end
end

function [x_cam,y_cam] = camprofile(xprofile,yprofile,Rr)

    dx = gradient(xprofile);
    dy = gradient(yprofile);

    magnitude = sqrt(dx.^2 + dy.^2);

    tangent_vectors = [dx ./ magnitude; dy ./ magnitude];

    normal_vectors = [-tangent_vectors(2, :); tangent_vectors(1, :)];

    x_cam = xprofile - Rr * normal_vectors(1, :);
    y_cam = yprofile - Rr * normal_vectors(2, :);

end

function [P] = pressure_angle(x, y, m)
    x = flip(x);
    y = flip(y);

    pressure_angles = zeros(size(x));

    dx = gradient(x);
    dy = gradient(y);

    magnitude = sqrt(dx.^2 + dy.^2);

    tangent_vectors = [dx ./ magnitude; dy ./ magnitude];
    normal_vectors = [-tangent_vectors(2, :); tangent_vectors(1, :)];

    follower_direction = [cos(atan(m)), sin(atan(m))];

    for i = 1:size(pressure_angles, 2)

        theta = 2 * pi * (i - 1) / 1000;
        normal = normal_vectors(:, i);

        % Rotation matrix for clockwise rotation
        R = [cos(theta), sin(theta); -sin(theta), cos(theta)];
        rotated_normal = R * normal_vectors(:, i);

        follower_dir = follower_direction;

        % Compute the dot product
        dot_product = dot(rotated_normal, follower_dir);
        cross_product = cross([rotated_normal;0],[follower_dir,0]);

        % Compute the pressure angle (in radians)
```

```
        pressure_angle_rad = atan2(sqrt(sum(cross_product.^2)),dot_product);

        % Convert to degrees
        pressure_angle_deg = rad2deg(pressure_angle_rad);

        % Store the pressure angle
        pressure_angles(i) = pressure_angle_deg;
    end
    P = flip(pressure_angles);
end

function [R] = radiusofcurv(x,y)

    dx = gradient(x);
    dy = gradient(y);
    xp = x;
    yp = y;
    d1x = gradient(xp);
    d1y = gradient(yp);
    d2x = gradient(d1x);
    d2y = gradient(d1y);


    curvature_radius_vector = zeros(1,1000);
    for i=1:1000
        idx = i;
        curvature_i = abs(dx(idx) * d2y(idx) - dy(idx) * d2x(idx)) / ((dx(idx)^2 +
dy(idx)^2)^(3/2));
        radius_of_curvature_i = 1 / curvature_i;
        curvature_radius_vector(i) = radius_of_curvature_i;
    end
    R = curvature_radius_vector;
end

function [Pc] = contact_pressure(F,P,E,b,Rr,rho)

    S = F./cos(deg2rad(P));
    Pc = sqrt(0.175*((S.*E)./(b*Rr)).*(1+(Rr./rho))) .* 1e-6; % [MPa]

end
```