Lab 2: Optimization

Objectives

In this lab, students will solve two optimization problems using MATLAB. The first problem is an unconstrained optimization problem where three different algorithms are used to minimize a function. Students will also learn the importance of the selection of the starting points. The second problem is in relation to the synthesis or design of mechanisms for motion, function and path generation. Students will solve these engineering design problems as optimization problems.

Part I. Comparison of Optimization Methods

A common function used in optimization to evaluate the characteristics of an algorithm is Himmelblau's function (see below). Write a Matlab function called Lab_2_Fun.m that includes Himmelblau's function with a vector input **x** = [x_0, y_0] (in the function `x = x(1)` and `y=x(2)`) and an output `z = f(x,y)`.

$$f(x,y) = (x^2 + y - 11)^2 + (x + y^2 - 7)^2$$

Himmelblau's function has four local minima. To find them all, you must use different starting points. Three unconstrained optimization numerical methods will be employed to minimize this function.

Main Code

Create a main code, Lab_2_main.m in which you can call each method and plot the function. Use the percentage character `%` before each line to comment the methods that you are not testing.

```
%Define call function numerically
[x1,y1] = meshgrid(-5:0.1:5,-5:0.1:5);
f = fx(x1,y1);

% Plot surface
surface = figure; figure(surface);
surf(x1,y1,f); shading interp;

x0=[0; 0] %Initial Guess
%Call Methods (Use % before each line to comment the methods you are not testing)
[X,traj,Z,k,Err] = %Call Method

% Add trajectory for finding minima (3D plot)
plot3(traj(:,1), traj(:,2), Z(:,1) ,'-k+')

%Plot contour (2D plot)
contour_graph = figure; figure(contour_graph);
contour(x1,y1,f,100); hold on;
plot(X(1,:), X(2,:),'-k+')
```

I.1 Steepest Descent Method.

- Write a code for the steepest ascent method (Lab_2_sdm.m).

$$\mathbf{x}_{i+1} = \mathbf{x}_i - \nabla(\mathbf{x}_i)h$$

where $\mathbf{x} = [x, y]^T$, and an approximation of the step size h is given by

$$h = \left|\frac{\nabla(\mathbf{x})^T\nabla(\mathbf{x})}{\nabla(\mathbf{x})^T\mathbf{H}(\mathbf{x})\nabla(\mathbf{x})}\right|; \quad \text{with } \nabla(\mathbf{x}) = \begin{bmatrix} \frac{\partial f}{\partial x} \\ \frac{\partial f}{\partial y} \end{bmatrix} \quad \text{and } \mathbf{H}(\mathbf{x}) = \begin{bmatrix} \frac{\partial^2 f}{\partial x^2} & \frac{\partial^2 f}{\partial x \partial y} \\ \frac{\partial^2 f}{\partial x \partial y} & \frac{\partial^2 f}{\partial y^2} \end{bmatrix}$$

You can find the gradient and Hessian by hand or you can use Matlab. Type `syms x y` in the command window and the function `f=(x^2 + y - 11)^2 + (x + y^2 - 7)^2`. Type `dx=diff(f,x)` yielding the first component of the gradient. Repeat to find the other component of the gradient and the elements of the Hessian, e.g., `ddx=diff(dx,x`.

Create two functions Lab_2_Grad.m and Lab_2_Hess.m in which the input is the x vector (include both x and y) and outputs are the gradient and the Hessian matrix, respectively.

While iterating use a tolerance error of $\varepsilon_s = 10^{-4}$. The approximation error is defined as the norm between the new and old values of vector x, i.e., $\varepsilon_a = \|\mathbf{x}_{new} - \mathbf{x}_{old}\|$. A pseudocode of the steepest descent is shown below, where x0 is the initial guess and tol is the tolerance.

```
function [X,traj,f,k,Err] = Lab2_sdm(x0,tol)
k = 0; ea = 1;
X = x0; traj=[];
f(1) = Evaluate function;
Err=NaN;
while ea > tol,
   Evaluate Gradient
   Evaluate Hessian;
   Evaluate Step Size;
   xnew=Evaluate Steepest Descent Formula
   traj = [traj xnew]; %Store solution in a vector
   k = k+1;
   ea = norm(xnew-x); %Evaluate error
   Err = [Err ea];
   x = xnew;
   f(k+1) = Evaluate Function
end
```

The output gives `X`, the final solution, `traj` the vector with all the estimated solutions, `f` the vector of the function evaluated at each iteration, (`k`) number of iterations and (`Err`) the error evaluated at each iteration.

- Use as four starting points, [0, 0], [-1, 0], and the other two of your choice in such a way that the other two local minima are found.
- Report for every starting point, a table that includes number of iterations (about 20 iterations spread around the search), estimated points (x,y), function evaluation (`f`) and approximate error after each iteration (`Err`). Add a picture of the path followed and its contour.
- Discuss the convergence of this method

I.2 Newton's Method.

- Write a code for Newton's method (Lab_2_Newton.m). Simply modify the equation for `xnew` from the steepest descent method and erase the step size line.
- Use the same four starting points.
- Report for every starting point, a table that includes number of iterations (about 20 iterations spread around the search), estimated points (x,y), function evaluation (f) and approximate error after each iteration (Err). Add a picture of the path followed and its contour.
- Discuss the convergence of this method.

I.3 MATLAB's Function

- Use MATLAB function `fminunc` as follows

```
options  =  optimoptions('fminunc','GradObj','on','Algorithm','trust-region');
[X,FVAL,EXITFLAG,OUTPUT] = fminunc(@Lab2_Fun,x0)
```

- Report only the number of iterations, final estimate point (x,y) and function evaluation (f) for each starting point.
- Discuss the convergence of this method.

Part II. Synthesis of Mechanisms

In this part, mechanism will be designed to accomplish a particular task for motion generation, function generation and path generation. All three tasks will require the mechanism to pass through four precision points. Thus the goal is to find the length of all the links, as well as the location of the ground pivots.

Under the laboratory folder on Canvas, download the following Matlab files:

MSE_211_Lab_2_motion_generation.m
MSE_211_Lab_2_function_generation.m
MSE_211_Lab_2_path_generation.m

These files are partially written, this is you cannot execute them as they are, you would have to complete the files where indicated in the script and execute them

II. 1 Motion Generation

It is desired to transport a box through four precision points (position and orientation). Shown below is an illustration of the task that is required:

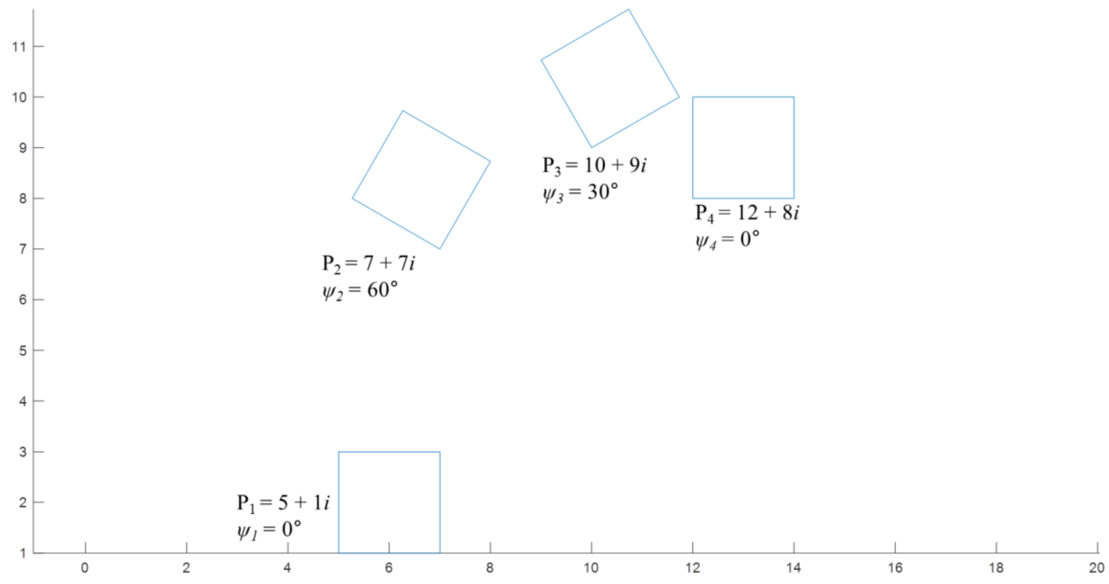


Figure 1. Definition of Precision Points for Motion Generation

Define Precision Points

Define the precision points (use the same point on the box) and then determine the 2nd, 3rd, and 4th precision points relative to the 1st precision point. The relative position is $\boldsymbol{\delta_j}$ and the relative rotation α_j, for $j=1,2,3$. For example,

$$\boldsymbol{\delta_1 = P_2 - P_1} \qquad\qquad \alpha_1 = \psi_2 - \psi_1$$

Thus, $\boldsymbol{\delta_1}$ represents the position vector from $\boldsymbol{P_1}$ to $\boldsymbol{P_2}$ and α_1 between these vectors. Similarly, determine $\boldsymbol{\delta_2}$, α_2, $\boldsymbol{\delta_3}$ and α_3.

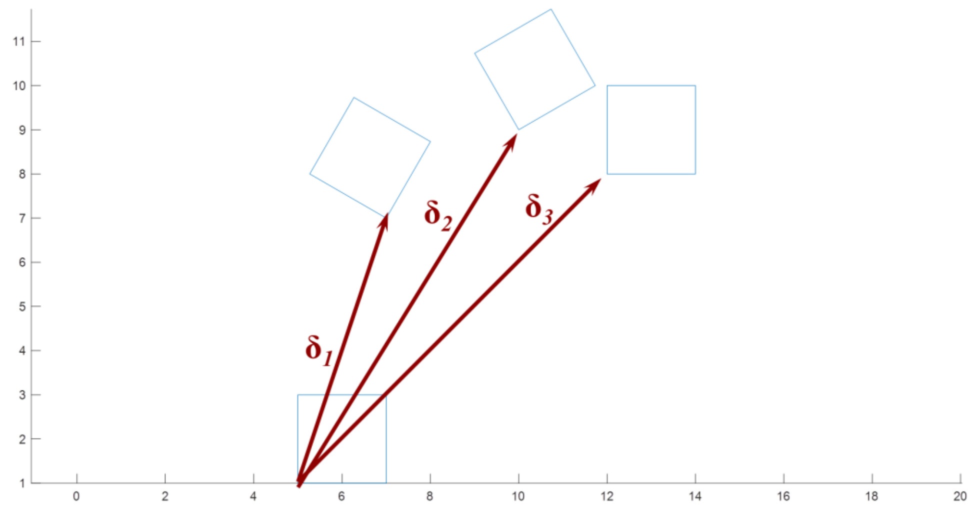
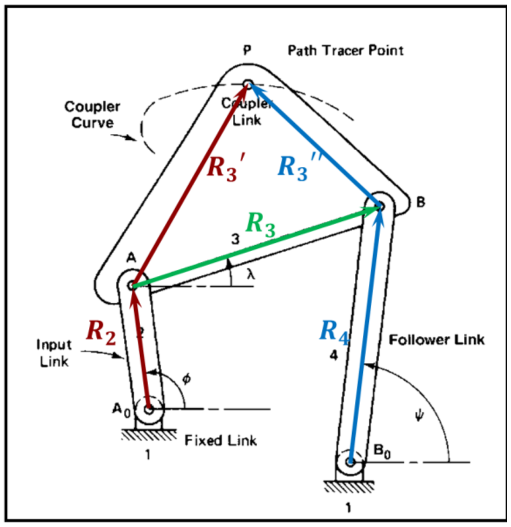


Figure 2. Relative Displacement between Precision Points

Define Optimization Problem

The vector notation that will be used is shown below. The left side (red) and the right side (blue) are solved independently. Each set of vector $\mathbf{R}_i$ involves two components $\mathbf{R}_i = [\, r_{ix}, r_{iy} \,]$.



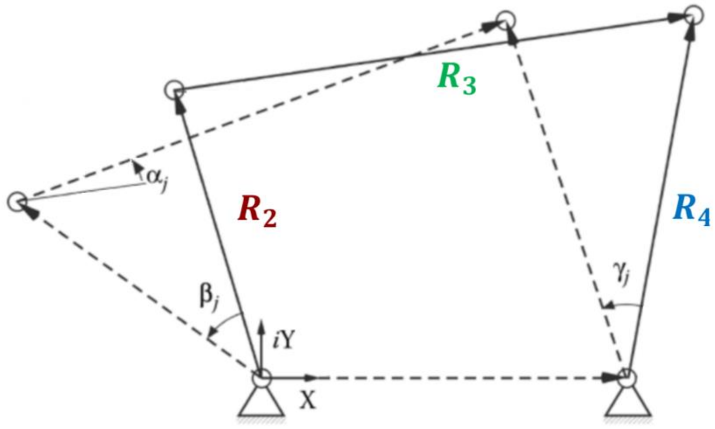


Figure 4. Relative Rotation between Precision Points

Figure 3. Vector Representation

In Fig. 3, $\mathbf{R_2}$ is the vector that represent the input link in the first precision point. $\mathbf{R_3}$**'** is the vector that connects the mobile pin at the end of $\mathbf{R_2}$ to the first precision point P_1. Similarly, $\mathbf{R_4}$ is the vector that represents the output link in the first precision point. $\mathbf{R_3}$**"** is the vector that connect the pin at the end of $\mathbf{R_4}$ to the first precision point P_1.

In Fig. 4, the angle α_j is the relative displacement of $\mathbf{R_3}$ between the configuration in the first precision point (solid line) and the j^{th} precision point (dashed line). Similarly, β_j is the relative displacement of $\mathbf{R_2}$ between the first and the j^{th} precision point configurations and γ_j is the relative displacement of $\mathbf{R_4}$ between the first and the j^{th} precision point configurations.

Designing a four-bar mechanism that has to pass through a number of precision points requires solving a set of equations. For motion generation, each side of the mechanism is solved independently.

Left side analysis:

Shown below is the relative rotation of the left side of the mechanism and its vector representation

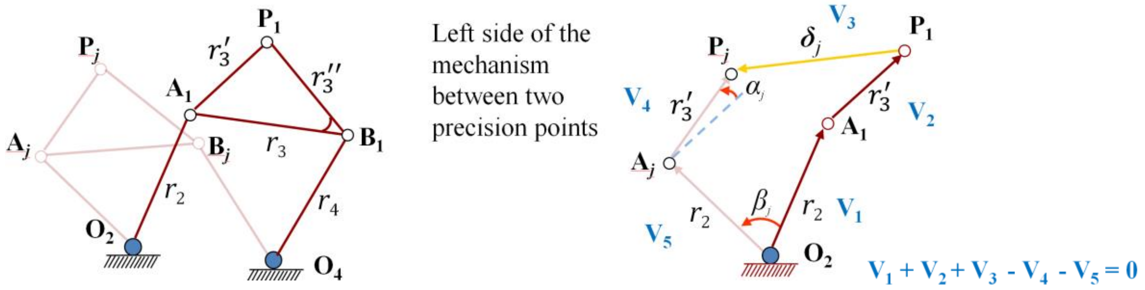


Figure 5. Relative Rotation from First Precision Point to j^{th} Precision Point

The vector representation can be expanded as:

$$f_{jx} = r_{2x} + r'_{3x} + \delta_{jx} - r'_{3x}\cos(\alpha_j) + r'_{3y}\sin(\alpha_j) - r_{2x}\cos(\beta_j) + r_{2y}\sin(\beta_j) = 0$$

$$f_{jy} = r_{2y} + r'_{3y} + \delta_{jy} - r'_{3y}\cos(\alpha_j) - r'_{3x}\sin(\alpha_j) - r_{2y}\cos(\beta_j) - r_{2x}\sin(\beta_j) = 0$$

where *j=1,2,3,* i.e., a set of three equations.

From this set of equations, only the vectors $\boldsymbol{\delta}_j$ and the angles α_j, for *j=1,2,3*, are known. There are seven unknown variables: **R**$_2$, **R**$_3'$, β_1, β_2, β_3, with each of the vectors **R** being defined by two components.

Since there are three sets of equations (*j = 1, 2, 3*) and all these equations must be equal to zero, an objective function of the following form can be established,

$$F = \sqrt{(f_{1x})^2 + (f_{1y})^2 + (f_{2x})^2 + (f_{2y})^2 + (f_{3x})^2 + (f_{3y})^2}$$

Estimate starting points for the seven unknowns, i.e.,

$r_{2x} = a$ $r_{2y} = b$ $r'_{3x} = c$ $r'_{3y} = d$ $\beta_1 = e$ $\beta_2 = f$ $\beta_3 = g$

where the *a, b, c, d , e, f,* and *g*, are the estimated numerical values.

Use `fminunc` to minimize the objective function *F*. Note that if *F = 0*, then the resulting length of the links will reach the desired precision points.

Right side analysis:

The right side of the mechanism is solved similarly. The rotation of the follower link between the first and *j*th precision point is γ_j.

$$g_{jx} = r_{4x} + r''_{3x} + \delta_{jx} - r''_{3x}\cos(\alpha_j) + r''_{3y}\sin(\alpha_j) - r_{4x}\cos(\gamma_j) + r_{4y}\sin(\gamma_j) = 0$$

$$g_{jy} = r_{4y} + r''_{3y} + \delta_{jy} - r''_{3y}\cos(\alpha_j) - r''_{3x}\sin(\alpha_j) - r_{4y}\cos(\gamma_j) - r_{4x}\sin(\gamma_j) = 0$$

Note that vectors δ_j and angles α_j are known, and the other variables r_{4x} , r_{4y}, r''_{3x}, r''_{3y} and γ_j, for *j = 1, 2, 3*, are unknown.

Estimate starting points for the seven unknowns, and use `fminunc` to minimize the objective function *G*.

$$G = \sqrt{(g_{1x})^2 + (g_{1y})^2 + (g_{2x})^2 + (g_{2y})^2 + (g_{3x})^2 + (g_{3y})^2}$$

Verify that the links pass through the precision points with the same assembly mode, as the **solution may not be feasible**, i.e., the mechanism must be reassembled to reach a particular precision point.

Laboratory Requirements

- Design a mechanism that will move the box as shown in Fig. 1. Use the partially written Matlab file MSE_211_Lab_2_motion_generation.m
- Write two sets of optimization problems (left and right side of the mechanism). This requires the estimation of the initial values of the seven variables ($\mathbf{R_2}$, $\mathbf{R_3}'$, β_1, β_2, β_3 for the left side) and ($\mathbf{R_4}$, $\mathbf{R_3}''$, γ_1, γ_2, γ_3 for the right side). Also, write the objective functions to be minimized.
- Include in your report the set of initial values that were assumed, a print out of your objective functions, and a screenshot of the designed mechanism

Note: Make sure your objective functions are equal to zero. Also, this method does not guarantee that the design mechanism will work. The assembly modes might change among precision points (unfeasible design).

II. 2 Path Generation

Assume the case that you want to build a walking spider that climbs a staircase. The tip of the leg must pass through four precision points as a function of the input angle

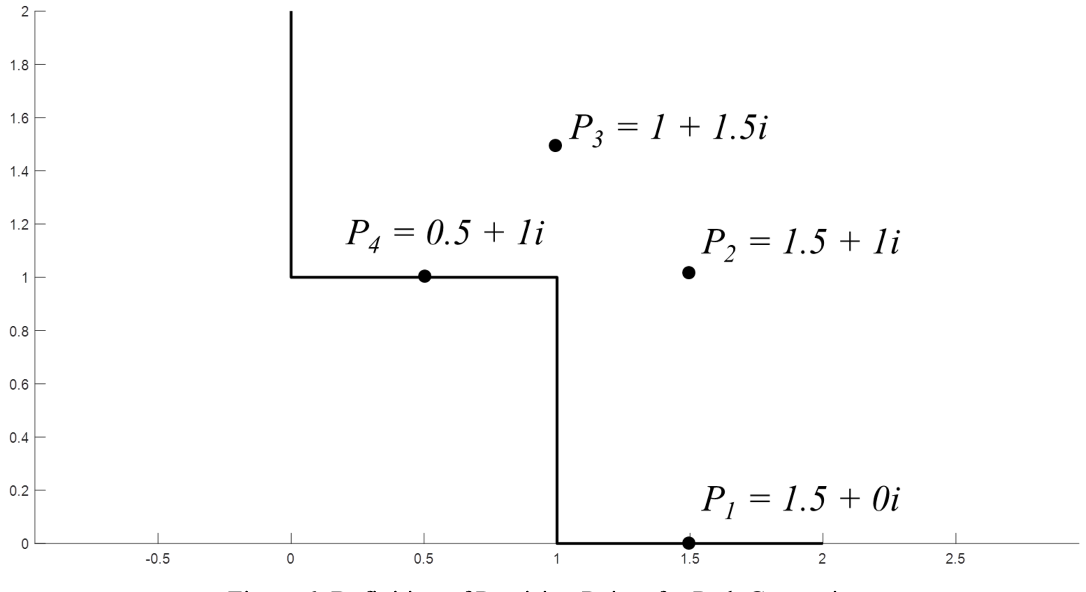


Figure 6. Definition of Precision Points for Path Generation

In addition, the relative rotation of the input link must move 60º between each precision point, i.e., the relative angles with respect to the first precision point are $\beta_1 = 60^o$, $\beta_2 = 120^o$, and $\beta_3 = 180^o$. In doing so, a second leg can be designed to move the spider through the other 180º.

The problem is similar to Motion Generation, however here the left side of the mechanism has to be solved first.

Left side analysis:

The vector equations are expanded as:

$$f_{jx} = r_{2x} + r'_{3x} + \delta_{jx} - r'_{3x}\cos(\alpha_j) + r'_{3y}\sin(\alpha_j) - r_{2x}\cos(\beta_j) + r_{2y}\sin(\beta_j) = 0$$

$$f_{jy} = r_{2y} + r'_{3y} + \delta_{jy} - r'_{3y}\cos(\alpha_j) - r'_{3x}\sin(\alpha_j) - r_{2y}\cos(\beta_j) - r_{2x}\sin(\beta_j) = 0$$

where *j=1,2,3,* i.e., there is a set of three equations. Note that vectors δ_j and angles β_j are known, for *j = 1, 2, 3*, and the other variables r_{2x} , r_{2y}, r'_{3x}, r'_{3y} and α_j, for *j = 1, 2, 3*, are unknown

Since there are three sets of equations (6 in total) and all these equations must be equal to zero, an objective function of the following form can be established.

$$F = \sqrt{(f_{1x})^2 + (f_{1y})^2 + (f_{2x})^2 + (f_{2y})^2 + (f_{3x})^2 + (f_{3y})^2}$$

Estimate starting points for the seven unknowns, i.e.,

$r_{2x} = a$ $r_{2y} = b$ $r'_{3x} = c$ $r'_{3y} = d$ $\alpha_1 = e$ $\alpha_2 = f$ $\alpha_3 = g$

where the *a, b, c, d , e, f,* and *g*, are the estimated numerical values. Use `fminunc` to minimize the objective function *F*. Note that if *F = 0*, then the resulting length of the links will reach the desired precision points.

Right side analysis:

For the right side the equations are:

$$g_{jx} = r_{4x} + r''_{3x} + \delta_{jx} - r''_{3x}\cos(\alpha_j) + r''_{3y}\sin(\alpha_j) - r_{4x}\cos(\gamma_j) + r_{4y}\sin(\gamma_j) = 0$$

$$g_{jy} = r_{4y} + r''_{3y} + \delta_{jy} - r''_{3y}\cos(\alpha_j) - r''_{3x}\sin(\alpha_j) - r_{4y}\cos(\gamma_j) - r_{4x}\sin(\gamma_j) = 0$$

where *j=1,2,3.* For this side, the α values that were determined in the left side analysis become known parameters. Thus, vectors δ_j and angles α_j are known, and the other variables r_{4x} , r_{4y}, r''_{3x}, r''_{3y} and γ_j, for *j = 1, 2, 3*, are unknown.

Estimate starting points for the seven unknowns, and use `fminunc` to minimize the objective function *G*.

$$G = \sqrt{(g_{1x})^2 + (g_{1y})^2 + (g_{2x})^2 + (g_{2y})^2 + (g_{3x})^2 + (g_{3y})^2}$$

Verify that the links pass through the precision points with the same assembly mode, as the **solution may not be feasible**.

Laboratory Requirements

- Design a mechanism that will move the leg of the spider as shown in Fig. 6 while the prescribed rotation of the input link is $\beta_1 = 60^o$, $\beta_2 = 120^o$, and $\beta_3 = 180^o$. Use the partially written Matlab file MSE_211_Lab_2_path_generation.m
- Write two sets of optimization problems (left and right side of the mechanism). This requires the estimation of the initial values of the seven variables ($\mathbf{R_2}$, $\mathbf{R_3}'$, α_1, α_2, α_3 for the left side) and ($\mathbf{R_4}$, $\mathbf{R_3}''$, γ_1, γ_2, γ_3 for the right side). Also, write the objective functions to be minimized.
- Include in your report the set of initial values that were assumed, a print out of your objective functions, and a screenshot of the designed mechanism

Note: Make sure your objective functions are equal to zero. Also, this method does not guarantee that the design mechanism will work. The assembly modes might change among precision points (unfeasible design).

II. 2 Function Generation

Assume you want to control the motion of the output link (Link 4) as a function of the input link (Link 2). Let $\beta_1 = 60^o$, $\beta_2 = 180^o$, and $\beta_3 = 270^o$ and $\gamma_1 = 90^o$, $\gamma_2 = 120^o$, and $\gamma_3 = 210^o$. Thus, while the input link travels 60^o from configuration 1 to configuration 2, the output link travels 90^o.

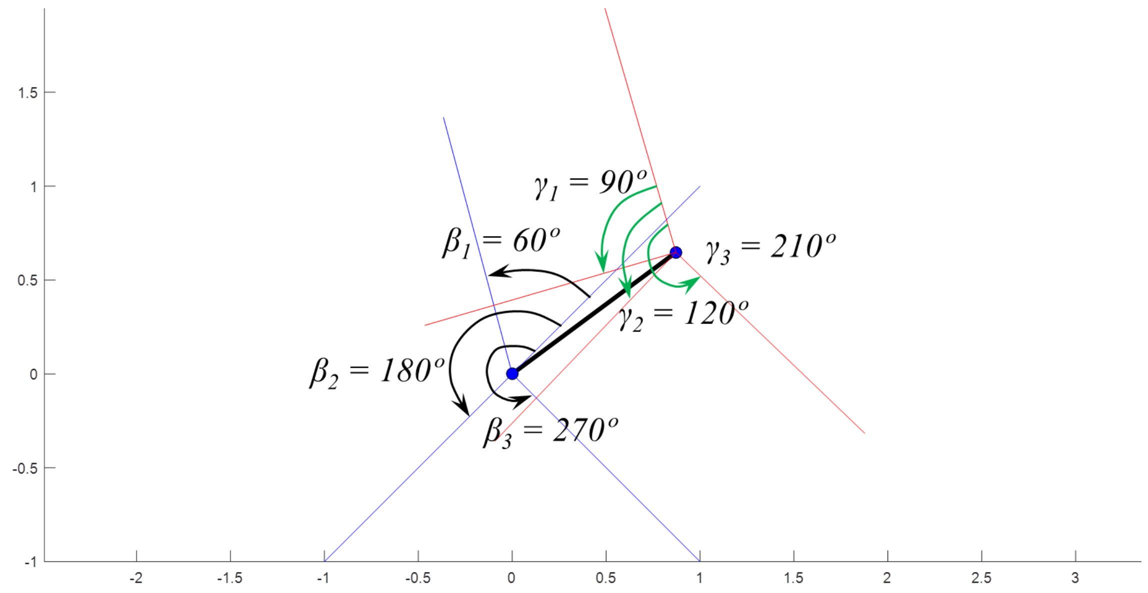


Figure 7. Definition of Precision Points for Function Generation

In Function Generation, there is a single set of equations (left and right sides are combined). Usually, vector $\mathbf{R_2}$ is also prescribed so the mechanism is scaled. Let $\mathbf{R_2} = [1, 1]$

Combined analysis:

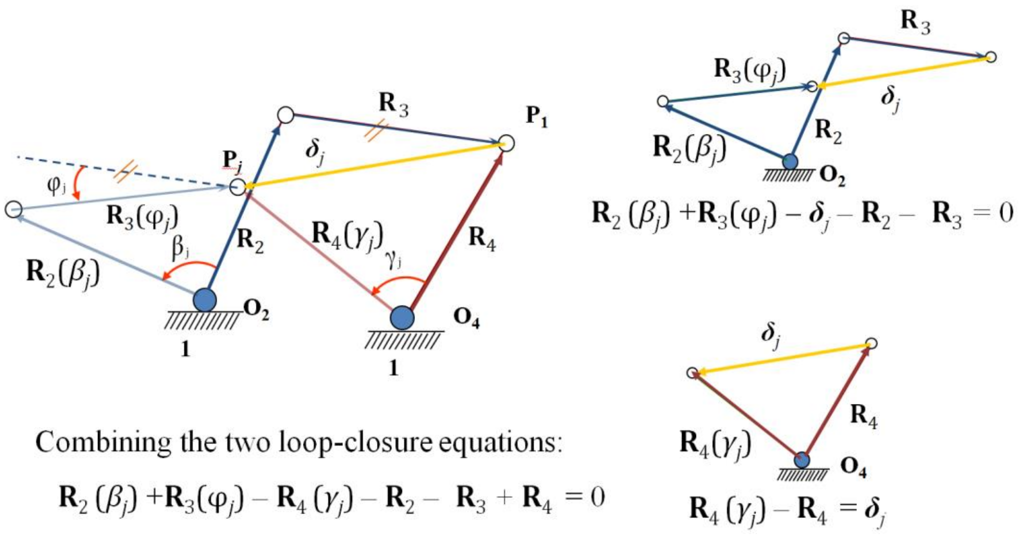

R3
P1
Pj
δj
φj
R3(φj)
R2
R4(γj)
R4
βj
γj
R2(βj)
O2
O4
1
1
R3
R3(φj)
δj
R2
R2(βj)
O2
R2 (βj) +R3(φj) – δj – R2 – R3 = 0
Combining the two loop-closure equations:
R2 (βj) +R3(φj) – R4 (γj) – R2 – R3 + R4 = 0
δj
R4
R4(γj)
O4
R4 (γj) – R4 = δj

Figure 8. Vector Representation for Function Generation

The vector representation is expanded as:

$$f_{jx} = -r_{2x}\cos(\beta_j) + r_{2y}\sin(\beta_j) - r_{3x}\cos(\varphi_j) + r_{3y}\sin(\varphi_j) + r_{4x}\cos(\gamma_j) - r_{4y}\sin(\gamma_j) + r_{2x} + r_{3x} - r_{4x} = 0$$

$$f_{jy} = -r_{2x}\sin(\beta_j) - r_{2y}\cos(\beta_j) - r_{3x}\sin(\varphi_j) - r_{3y}\cos(\varphi_j) + r_{4x}\sin(\gamma_j) + r_{4y}\cos(\gamma_j) + r_{2y} + r_{3y} - r_{4y} = 0$$

where *j=1,2,3,* i.e., a set of three equations. Note that vectors r_{2x} and r_{2y} are defined in order to scale the mechanism, and the angles β_j and γ_j are established with the precision points. All the other variables, r_{4x} , r_{4y}, r_{3x} , r_{3y} and φ_j , for *j = 1, 2, 3*, are unknown.

Since there are three sets of equations (6 in total) and all these equations must be equal to zero, an objective function of the following form can be established.

$$F = \sqrt{(f_{1x})^2 + (f_{1y})^2 + (f_{2x})^2 + (f_{2y})^2 + (f_{3x})^2 + (f_{3y})^2}$$

Estimate starting points for the seven unknowns, i.e.,

$r_{3x} = a$ $r_{3y} = b$ $r_{4x} = c$ $r_{4y} = d$ $\varphi_1 = e$ $\varphi_2 = f$ $\varphi_3 = g$

where the *a, b, c, d , e, f*, and *g*, are the estimated numerical values. Use `fminunc` to minimize the objective function *F*. Note that if *F = 0*, then the resulting length of the links will reach the desired precision points.

Verify that the links pass through the precision points with the same assembly mode, as the **solution may not be feasible**.

Laboratory Requirements

- Design a mechanism that will control the output link as a function of the input link as shown in Fig. 7. Use the partially written Matlab file MSE_211_Lab_2_function_generation.m
- Write the optimization problem of the mechanism. This requires the estimation of the initial values of the seven variables ($\mathbf{R}_3$, $\mathbf{R}_4$, φ_1, φ_2, φ_3 for the combined analysis) Also, write the objective function to be minimized.
- Include in your report the set of initial values that were assumed, a print out of your objective function, and a screenshot of the designed mechanism

Note: Make sure your objective functions results in $G = 0$. Also, this method does not guarantee that the design mechanism will work. The assembly modes might change among precision points (unfeasible design).