2 Trajectory Optimization

In this problem you will use trajectory optimization to perform motion planning in 2-DoF environments. As before, the mobile robot's position is $\theta = [x, y]^T$.

2.1 (15 points)

Implement the trajectory optimization algorithm. Your code should be able to work with an arbitrary number of waypoints and circular obstacles. Set the initial trajectory ξ^0 as:

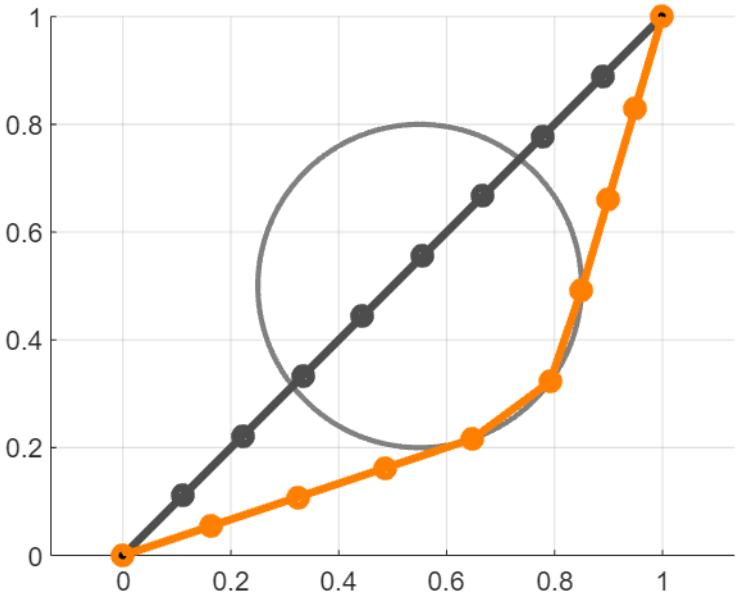
```
xi_0 = [linspace(theta_start(1), theta_goal(1), k);
        linspace(theta_start(2), theta_goal(2), k)];
```

```
clear
close all

% Start and Goal orientations
theta_start = [0;0];
theta_goal = [1;1];

% Initial trajectory variables
n = 2; % No. of joints / 2-D trajectory
k = 10; % No. of waypoints

% Obstacle Parameters - add obstacles as needed
% Each row: [center_x, center_y, radius]
obstacles = [
    0.55, 0.5, 0.3;  % First obstacle
    % Add more obstacles here, e.g., [x, y, r]
];

xi_0 = [linspace(theta_start(1), theta_goal(1), k);
        linspace(theta_start(2), theta_goal(2), k)];

xi_0_vec = reshape(xi_0, [], 1);

% Equality constraints for start and goal positions
A = [eye(n), zeros(n,n*(k-1)) ;...
     zeros(n,n*(k-1)), eye(n) ];
B = [theta_start;theta_goal];

% Nonlinear optimization
options = optimoptions('fmincon','Display','iter',...
    'Algorithm','sqp','MaxFunctionEvaluations',1e4);
xi_star_vec = fmincon(@(xi) cost(xi, obstacles), xi_0_vec, ...
    [], [], A, B, [], [], @(xi) nonlcon(xi, obstacles), options);
```

```
 Iter  Func-count           Fval   Feasibility   Step Length       Norm of   First-order
                                                                      step    optimality
    0          21    2.406330e+03     2.442e-01     1.000e+00     0.000e+00     9.306e+04
    1          65    1.363684e+03     2.844e-03     2.737e-04     2.567e+01     1.033e+02
    2          88    1.336296e+03     0.000e+00     4.900e-01     3.312e+01     4.955e+01
    3         110    9.947275e+02     0.000e+00     7.000e-01     2.670e+01     3.330e+01
    4         132    9.093469e+02     0.000e+00     7.000e-01     2.865e+01     2.909e+01
    5         154    7.748821e+02     0.000e+00     7.000e-01     2.120e+01     4.364e+01
    6         176    5.391986e+02     0.000e+00     7.000e-01     1.920e+01     3.179e+01
    7         198    3.956808e+02     0.000e+00     7.000e-01     1.285e+01     2.076e+01
    8         220    3.798429e+02     0.000e+00     7.000e-01     9.370e+00     1.782e+01
    9         242    3.623666e+02     0.000e+00     7.000e-01     7.150e+00     2.639e+01
   10         264    3.412506e+02     0.000e+00     7.000e-01     7.378e+00     3.076e+01
   11         285    3.304941e+02     0.000e+00     1.000e+00     5.843e+00     2.229e+01
   12         308    3.226864e+02     0.000e+00     4.900e-01     2.806e+00     2.048e+01
   13         330    3.175471e+02     0.000e+00     7.000e-01     2.991e+00     2.423e+01
   14         353    3.159786e+02     0.000e+00     4.900e-01     1.581e+00     2.583e+01
   15         374    3.159722e+02     0.000e+00     1.000e+00     1.960e+00     2.443e+01
   16         396    3.149716e+02     0.000e+00     7.000e-01     1.337e+00     2.346e+01
   17         417    3.141597e+02     0.000e+00     1.000e+00     1.608e+00     2.410e+01
   18         438    3.135818e+02     0.000e+00     1.000e+00     1.145e+00     2.481e+01
   19         459    3.129914e+02     0.000e+00     1.000e+00     8.481e-01     2.418e+01
   20         480    3.126799e+02     0.000e+00     1.000e+00     3.258e-01     2.413e+01
   21         501    3.120199e+02     0.000e+00     1.000e+00     5.074e-01     2.420e+01
   22         522    3.100772e+02     0.000e+00     1.000e+00     1.502e+00     2.412e+01
   23         543    3.065493e+02     0.000e+00     1.000e+00     2.335e+00     2.396e+01
   24         564    2.990809e+02     0.000e+00     1.000e+00     3.968e+00     2.342e+01
   25         585    2.838973e+02     0.000e+00     1.000e+00     5.688e+00     2.213e+01
   26         606    2.538013e+02     0.000e+00     1.000e+00     7.847e+00     1.958e+01
   27         627    1.982382e+02     0.000e+00     1.000e+00     1.033e+01     1.472e+01
   28         648    1.150687e+02     0.000e+00     1.000e+00     1.392e+01     7.462e+00
   29         669    3.811974e+01     0.000e+00     1.000e+00     1.616e+01     7.956e+00
 Iter  Func-count           Fval   Feasibility   Step Length       Norm of   First-order
                                                                      step    optimality
   30         690    5.580687e+00     0.000e+00     1.000e+00     1.185e+01     4.798e+00
   31         711    6.208616e-01     0.000e+00     1.000e+00     3.783e+00     1.222e+00
   32         732    3.138023e-01     0.000e+00     1.000e+00     6.718e-01     3.266e-01
   33         753    2.879849e-01     0.000e+00     1.000e+00     2.517e-01     1.548e-01
   34         774    2.784429e-01     0.000e+00     1.000e+00     1.505e-01     5.104e-02
   35         795    2.750657e-01     0.000e+00     1.000e+00     6.938e-02     1.499e-02
   36         816    2.742908e-01     0.000e+00     1.000e+00     1.253e-02     3.333e-03
   37         837    2.742592e-01     0.000e+00     1.000e+00     9.383e-04     3.245e-04
   38         858    2.742592e-01     0.000e+00     1.000e+00     2.124e-04     4.465e-05
   39         879    2.742592e-01     0.000e+00     1.000e+00     2.595e-05     1.755e-05
   40         903    2.742592e-01     0.000e+00     3.430e-01     9.282e-06     1.054e-05
   41         924    2.742592e-01     0.000e+00     1.000e+00     9.153e-06     2.044e-06
   42         945    2.742592e-01     0.000e+00     1.000e+00     5.645e-11     3.949e-07

Local minimum found that satisfies the constraints.

Optimization completed because the objective function is non-decreasing in
feasible directions, to within the value of the optimality tolerance,
and constraints are satisfied to within the value of the constraint tolerance.

<stopping criteria details>
```

```
xi_star = reshape(xi_star_vec, 2, []);

% Plotting
figure
grid on
hold on
```

```
axis([0, 1, 0, 1])
axis equal
% Plot Obstacles
for i = 1:size(obstacles, 1)
    viscircles(obstacles(i, 1:2), obstacles(i, 3), 'Color', [0.5, 0.5, 0.5]);
end
plot(theta_start(1), theta_start(2), 'ko', 'MarkerFaceColor', 'k')
plot(theta_goal(1), theta_goal(2), 'ko', 'MarkerFaceColor', 'k')
% Plot Result
plot(xi_0(1,:), xi_0(2,:), 'o-', 'Color', [0.3, 0.3, ...
 0.3], 'LineWidth', 3);
plot(xi_star(1,:), xi_star(2,:), 'o-', 'Color', [1, 0.5, 0], 'LineWidth', 3);
```




```
% Cost function to minimize
```

```
function C = cost(xi, obstacles)
    gamma = 20;
    xi = reshape(xi, 2, []);
    C = 0;

    for idx = 2:length(xi)
        Urep = 0;
        for obs_idx = 1:size(obstacles, 1)
            r = obstacles(obs_idx, 3);
            center = obstacles(obs_idx, 1:2)';

            if norm(center - xi(:, idx)) <= r
                Urep = Urep + 0.5 * gamma * ((1 / norm(center - xi(:, idx))) - (1 /
r))^2;
            end
```

```
        end

        C = C + norm(xi(:, idx) - xi(:, idx - 1))^2 + Urep;
    end
end

% Nonlinear constraints (optional)
function [c, ceq] = nonlcon(xi, obstacles)
    xi = reshape(xi, 2, []);
    c = [];
    for idx = 1:length(xi)
        for obs_idx = 1:size(obstacles, 1)
            r = obstacles(obs_idx, 3);
            center = obstacles(obs_idx, 1:2)';
            dist = norm(xi(:, idx) - center);

            % Add a constraint for each obstacle
            c = [c; r - dist];
        end
    end
    ceq = [];
end
```