```
clear;
clc;

t_0 = 0;
num_steps = 64;

disp('Initial Optimization');
out = hw8_1_RUN();
u_0 = out.CONTROLS; % initial controls
clc;

BEGIN_ACADO;                                % Always start with "BEGIN_ACADO".

    acadoSet('problemname', 'hw8_2');

    % (a) Define states and controls
    Parameter T;
    DifferentialState px py theta delta v;
    Control u_a u_delta;


    % (a) Differential Equation
    f = acado.DifferentialEquation(0, T);
    f.add( dot(px) == v * cos(theta) );
    f.add( dot(py) == v * sin(theta) );
    f.add( dot(theta) == v * tan(delta) );
    f.add( dot(v) == u_a );
    f.add( dot(delta) == u_delta );

    % (b) Optimal Control )
    ocp = acado.OCP(0.0, T, num_steps);

    % (b) Minimize control effort
    ocp.minimizeLSQ( u_delta, 0 )
    ocp.minimizeMayerTerm(T);

    % (c) Path constraints
    ocp.subjectTo( f ); % dynamics
    ocp.subjectTo( -5 <= v <= 5 );
    ocp.subjectTo( -5 <= u_a <= 5 );
    ocp.subjectTo( -pi/4 <= delta <= pi/4 );
    ocp.subjectTo( -pi/6 <= u_delta <= pi/6 );

    % obstacle
    ocp.subjectTo( (px+7)^2 + py^2 >= 1);



    % (d) Initial Conditions
    ocp.subjectTo( 'AT_START', px == -10.0 );
    ocp.subjectTo( 'AT_START', py == 1.0 );
    ocp.subjectTo( 'AT_START', v == 0.0 );
    ocp.subjectTo( 'AT_START', theta == 0.0 );
    ocp.subjectTo( 'AT_START', delta == 0.0 );

    % (d) Final boundary conditions
    ocp.subjectTo( 'AT_END', px == 0.0 );
    ocp.subjectTo( 'AT_END', py == 0.0 );
    ocp.subjectTo( 'AT_END', v == 0.0 );
    ocp.subjectTo( 'AT_END', theta == 0.0 );
    ocp.subjectTo( 3 <= T <= 10 );


    % (e) Optimization Algorithm
    algo = acado.OptimizationAlgorithm( ocp );

    % algorithm parameters
    algo.set( 'KKT_TOLERANCE', 1e-8 );
    algo.set( 'DISCRETIZATION_TYPE', 'MULTIPLE_SHOOTING' );
    algo.set( 'MAX_NUM_ITERATIONS', 500 );

    % initializations
    algo.initializeControls(u_0);


END_ACADO;           % Always end with "END_ACADO".
                     % This will generate a file problemname_ACADO.m.
                     % Run this file to get your results. You can
                     % run the file problemname_ACADO.m as many
                     % times as you want without having to compile again.

% Run the test
out = hw8_2_RUN();

% Plotting
x = out.STATES;
u = out.CONTROLS;
fprintf('optimal final time: %f', x(end,1));

figure;
title('Trajectory');
% plot obstacle
i = 0: 0.1 : 2*pi;
plot(-7 + cos(i),  0 + sin(i), '-k','LineWidth',2);
hold on;
plot(x(:,2), x(:,3),'.r', 'DisplayName', 'trajectory');
hold on;
plot(x(1,2), x(1,3), 'b*', 'DisplayName', 'start');
hold on;
plot(x(end,2), x(end,3), 'g*', 'DisplayName', 'end');
hold off;
grid on;
legend();
xlabel('x');
ylabel('y');
axis equal;
```

```
figure;
title('States');
plot(x(:,1), x(:,4),'-r', 'DisplayName', 'theta');
hold on;
plot(x(:,1), x(:,5), '-g', 'DisplayName', 'v');
hold on;
plot(x(:,1), x(:,6), '-b', 'DisplayName', 'delta');
hold on;
plot(x(:,1), x(:,2),'-k', 'DisplayName', 'x');
hold on;
plot(x(:,1), x(:,3), '-c', 'DisplayName', 'y');
hold on;
grid on;
legend();
xlabel('t');
ylabel('value');

figure;
title('Controls');
plot(u(:,1), u(:,2), '-r','DisplayName', 'u_a');
hold on;
plot(u(:,1), u(:,3), '-b','DisplayName', 'u_\delta');
hold off;
grid on;
legend();
xlabel('t');
ylabel('value');
```

Published with MATLAB® R2021b