

```
function [c, ceq] = mimp_opt(x)
    global nimp npar STM

    % Initial state
    X0 = [0 1 0 0 0 0]';
    alpha = x(npar);

    % Target final state
    thetaF = 2*pi;
    s = sin(thetaF + alpha);
    c0 = cos(thetaF + alpha);
    XF = [ s;
           2*c0;
           2*s;
           c0;
           -2*s;
           2*c0 ];

    % Initialize state
    Xm = X0;

    % Propagate through each impulse
    for k = 1:nimp
        idx = (k-1)*6 + (1:6);
        dv_plus = x(idx(1:3));
        dv_minus = x(idx(4:6));
        dv_net = dv_plus - dv_minus;
        Xp = Xm + [zeros(3,1); dv_net];
        Xm = STM * Xp;
    end

    % Final state after all impulses
    Xfinal = Xm;

    % No inequality constraints
    c = [];

    % Equality constraint: final state must match target
    ceq = Xfinal - XF;
end
```

```

close all; clear all; clc;
global nimp npar STM

nimp = 4; % Number of impulses
thetas = 2*pi/nimp;

% Continuous system
A = [0 0 0 1 0 0;
     0 0 0 0 1 0;
     0 0 0 0 0 1;
     3 0 0 0 2 0;
     0 0 0 -2 0 0;
     0 0 -1 0 0 0];
B = [zeros(3,3); eye(3)];

% STM for one step
STM = [4-3*cos(thetas) 0 0 sin(thetas) 2*(1-cos(thetas)) 0;
      -6*(thetas-sin(thetas)) 1 0 -2*(1-cos(thetas)) 4*sin(thetas)-3*thetas 0;
      0 0 cos(thetas) 0 0 sin(thetas);
      3*sin(thetas) 0 0 cos(thetas) 2*sin(thetas) 0;
      -6*(1-cos(thetas)) 0 0 -2*sin(thetas) -3+4*cos(thetas) 0;
      0 0 -sin(thetas) 0 0 cos(thetas)];

npar = nimp*6 + 1;
x = ones(npar,1);
xlow = zeros(npar,1);
xupp = inf(npar,1);
xupp(npar) = 2*pi;

options = optimset('Algorithm','active-set','TolFun',1e-10,'TolX',1e-10,'TolCon',1e-10);
[x,cost] = fmincon(@(x) sum(x(1:nimp*6)), x, [], [], [], [], xlow, xupp, 'mimp_opt', options);
alphaF = x(npar);

%%
theta_fine = pi/120;
theta_f = 2*pi;
n = theta_f/theta_fine;

STMF = [4-3*cos(theta_fine) 0 0 sin(theta_fine) 2*(1-cos(theta_fine)) 0;
      -6*(theta_fine-sin(theta_fine)) 1 0 -2*(1-cos(theta_fine)) 4*sin(theta_fine)-3*theta_fine 0;
      0 0 cos(theta_fine) 0 0 sin(theta_fine);
      3*sin(theta_fine) 0 0 cos(theta_fine) 2*sin(theta_fine) 0;
      -6*(1-cos(theta_fine)) 0 0 -2*sin(theta_fine) -3+4*cos(theta_fine) 0;
      0 0 -sin(theta_fine) 0 0 cos(theta_fine)];

x0 = [0 1 0 0 0 0]';
xm = x0;
X(:,1) = xm;
imp_angles = 0:thetas:(2*pi-thetas); % impulse angles for 4 burns
ii = 0;

for k = 1:n
    theta_now = (k-1)*theta_fine;
    if any(abs(theta_now - imp_angles) < 1e-3)
        ii = ii + 1;
        xp = xm + [zeros(3,1); (x((ii-1)*6+1:(ii-1)*6+3) - x((ii-1)*6+4:(ii-1)*6+6))];
    end
end

```

```

        DV(:,ii) = (x((ii-1)*6+1:(ii-1)*6+3) + x((ii-1)*6+4:(ii-1)*6+6));
    else
        xp = xm;
    end
    xm = STMF * xp;
    X(:,k+1) = xm;
end

```

```
%% Plot 2
```

```

figure;
plot(X(2,:), X(3,:), 'LineWidth', 1.5);
xlabel('y'); ylabel('z'); grid on;
title('Impulsive Trajectory (4-Impulse Case)');

```

```

figure;
stem(imp_angles, vecnorm(DV,1,1), 'filled');
set(gca, 'XTick', 0:pi/2:2*pi);
set(gca, 'XTickLabel', {'0','pi/2','pi','3pi/2','2pi'});
xlabel('\theta'); ylabel('||\Delta v||_1');
title('Impulse Magnitudes (4-Impulse Case)');
axis([0 2*pi 0 2.5]);

```

```
%% LQR
```

```

xref = @(th) [ ...
    sin(th + alphaF);
    2*cos(th + alphaF);
    2*sin(th + alphaF);
    cos(th + alphaF);
    -2*sin(th + alphaF);
    2*cos(th + alphaF)];

uref = @(th) [ -4*sin(th + alphaF) - 4*cos(th + alphaF); 0; 0];

```

```
% LQR on tracking error:
```

```

Qpos = diag([10 10 10]);
Qvel = 0.5*diag([1 1 1]);
Q = blkdiag(Qpos, Qvel);
R = 2*eye(3); % Optimize

```

```

[K,~,~] = lqr(A,B,Q,R);
Acl = (A - B*K);

```

```
% Integrate error dynamics
```

```

theta_span = [0 2*pi];
e0 = x0 - xref(0);
opts = odeset('RelTol',1e-9,'AbsTol',1e-12);

[Theta, E] = ode45(@(th,e) Acl*e, theta_span, e0, opts);

Xref = zeros(numel(Theta),6);
Uref = zeros(numel(Theta),3);
for k = 1:numel(Theta)
    Xref(k,:) = xref(Theta(k)).';
    Uref(k,:) = uref(Theta(k)).';
end
Xlqr = E + Xref;
Ulqr = Uref - (K*E).';

```

% Costs

```

dtheta = diff(Theta); dtheta = [dtheta; dtheta(end)];
xQx = sum((E*Q).*E,2);
uRu = sum((U_lqr*R).*U_lqr,2);
quad_cost = sum((xQx + uRu).*dtheta);
L1_thrust = sum(vecnorm(U_lqr,1,2).*dtheta);
fprintf('[Tracking LQR] Total Impulse = %.4f,    quadratic cost = %.4f\n', ...
        L1_thrust, quad_cost);

```

% LQR-only plot (y-z)

```

figure; plot(Xlqr(:,2), Xlqr(:,3), 'g', 'LineWidth', 1.8);
grid on; axis equal; xlim([-2 2.5]); ylim([-2 2]);
xlabel('y (LQR)'); ylabel('z (LQR)');
title('Continuous LQR Tracking Trajectory (y-z)');

```

% Impulsive vs LQR comparison

```

figure; hold on; grid on; axis equal;
plot(X(2,:), X(3,:), 'b', 'LineWidth', 1.8);
plot(Xlqr(:,2),Xlqr(:,3),'g', 'LineWidth', 1.8);
xlabel('y'); ylabel('z'); title('Comparison: Impulsive vs LQR (y-z)');
legend('Impulsive Control','LQR Tracking','Location','best');
xlim([-2 2.5]); ylim([-2 2]);

```

% LQR control magnitude

```

figure;
plot(Theta, vecnorm(U_lqr,2,2), 'LineWidth', 1.8);
hold on; grid on;

set(gca, 'XTick', 0:pi/2:2*pi);
set(gca, 'XTickLabel', {'0',' $\pi/2$ ',' $\pi$ ',' $3\pi/2$ ',' $2\pi$ '});

xlabel('\theta');
ylabel('||u(\theta)||2');
title('LQR Control Magnitude vs \theta');
axis([0 2*pi 0 6.5]);

```