

---

```

% Choose directory
rootdir = ['D0810B1F2_5/run1', 'D0810B1F2_5/run2', 'D0810B1F2_5/run3'];
%Load Data

for a=1:16:48
    DesiredEndEffectorVelocity = load(fullfile(rootdir(a:a
+15), "desiredEndEffectorVelocity.csv"));
    DesiredJointPosition = load(fullfile(rootdir(a:a
+15), "desiredJointPosition.csv"));
    JointPosition = load(fullfile(rootdir(a:a+15), "jointPosition.csv"));
    Time = load(fullfile(rootdir(a:a+15), "simulationTime.csv"));
    x = zeros(length(Time),3);
    xDot = zeros(length(Time)-1,3);

    for i =1:length(Time)
        [x(i,:), ~] = forwardKinematics(JointPosition(i,:));
    end
    for i =1:length(x)-1
        xDot(i,:) = (x(i+1,:) - x(i,:)) / (Time(i+1)-Time(i));
    end

    % Desired Path and Derivative of Path
    radius = 0.08;
    line_length = 0.1;
    alpha = (2*line_length)/(radius*pi) + 1;

    % Time vector
    sspace = linspace(0, alpha + 1, length(Time)); % Generate 100 points
    within the total time

    P_z = zeros(1,length(sspace));
    P_x = zeros(1,length(sspace));
    P_z_derivative = zeros(1,length(sspace));
    P_x_derivative = zeros(1,length(sspace));

    for i=1:length(sspace)
        s=sspace(i);
        if s <= 1
            % Calculate x and y coordinates of the first circle segment
            P_z(i) = -radius * cos(s*pi/2);
            P_x(i) = -radius * sin(s*pi/2);

            P_z_derivative(i) = radius * (pi/2) * sin(s * pi/2);
            P_x_derivative(i) = -radius * (pi/2) * cos(s * pi/2);

        elseif s <= alpha && s>1
            % Calculate the coordinates of the straight line segment
            P_z(i) = (s-1) * (radius * pi) / 2;
            P_x(i) = -radius * ones(1);

            P_z_derivative(1,i) = radius * pi / 2;
            P_x_derivative(1,i) = 0;

```

---

---

```

        else
            P_z(i) = radius * sin((s-alpha) * pi/2) + line_length;
            P_x(i) = -radius * cos((s-alpha) * pi/2);
            P_z_derivative(i) = radius * pi/2 * cos((s-alpha) * pi/2);
            P_x_derivative(i) = radius * pi/2 * sin((s-alpha) * pi/2);
        end

    end

end

% Plotting
figure(1)
hold on; grid on
plot(x(:,3),-x(:,1),'.')
if a == 33
    plot(P_z + 0.515509+radius,P_x + 0.0210774,'LineWidth',1.5)
end
xlabel('z direction (m)'); ylabel('x direction (m)')
title('Desired and Actual Position')
legend('Actual Path1', 'Actual Path2', 'Actual Path3','Desired Path' )
axis('equal');

figure(2)
hold on; grid on
plot(Time(10:end-1),xDot(10:end,1),'.')
if a == 33
    plot(Time,DesiredEndEffectorVelocity(:,1),'LineWidth',1.5)
end
xlabel('Time (s)'); ylabel('Velocity (m/s)')
title('Velocity in X direction')
legend('Actual Velocity1','Actual Velocity2','Actual Velocity3','Desired
Velocity')

figure(3)
hold on; grid on
plot(Time(10:end-1),xDot(10:end,3),'.')
if a==33
    plot(Time,DesiredEndEffectorVelocity(:,3),'LineWidth',1.5)
end
xlabel('Time (s)'); ylabel('Velocity (m/s)')
title('Velocity in Z direction')
legend('Actual Velocity1','Actual Velocity2','Actual Velocity3','Desired
velocity')

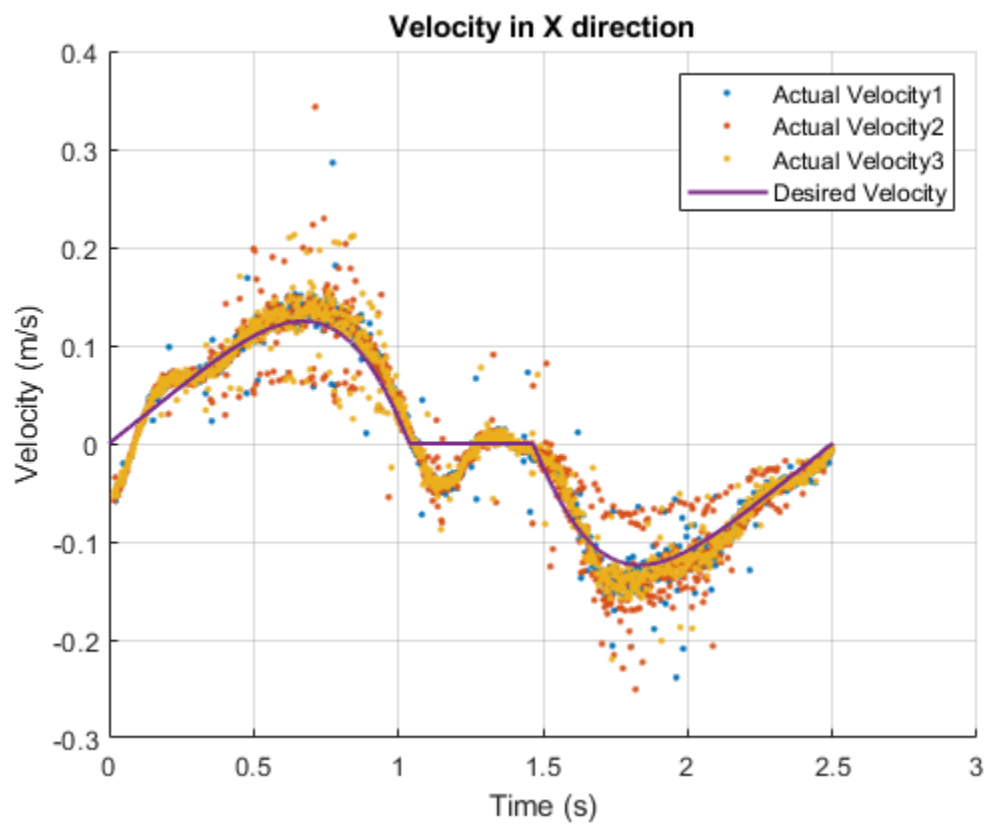
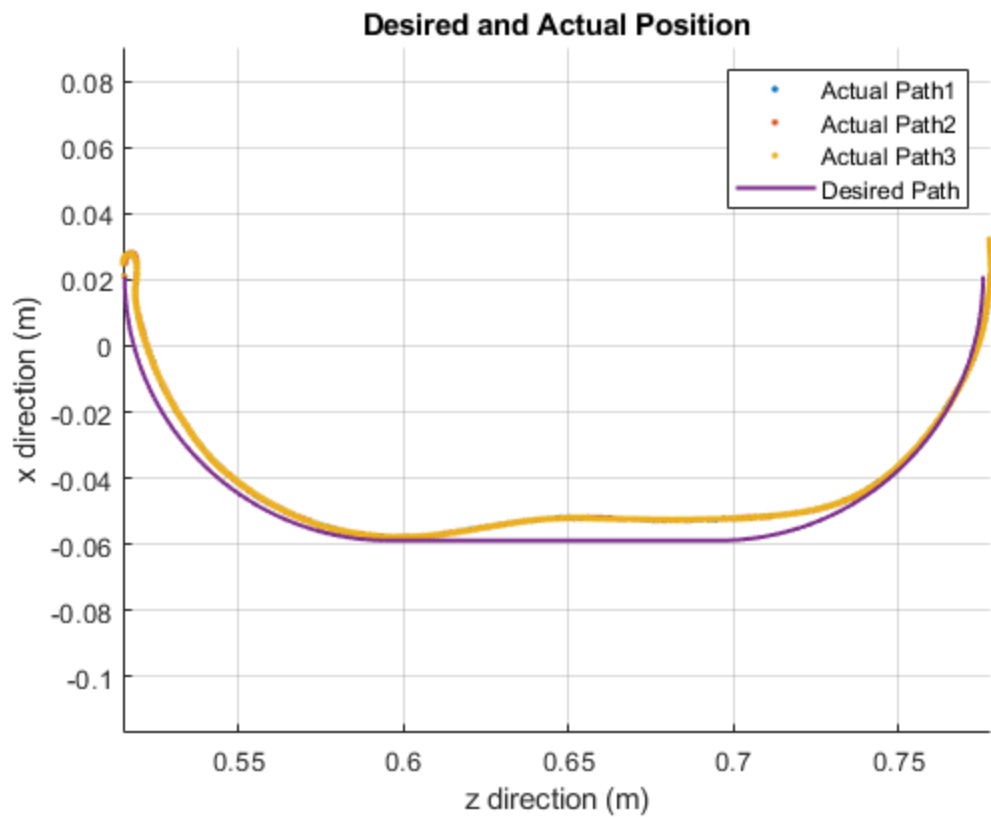
end

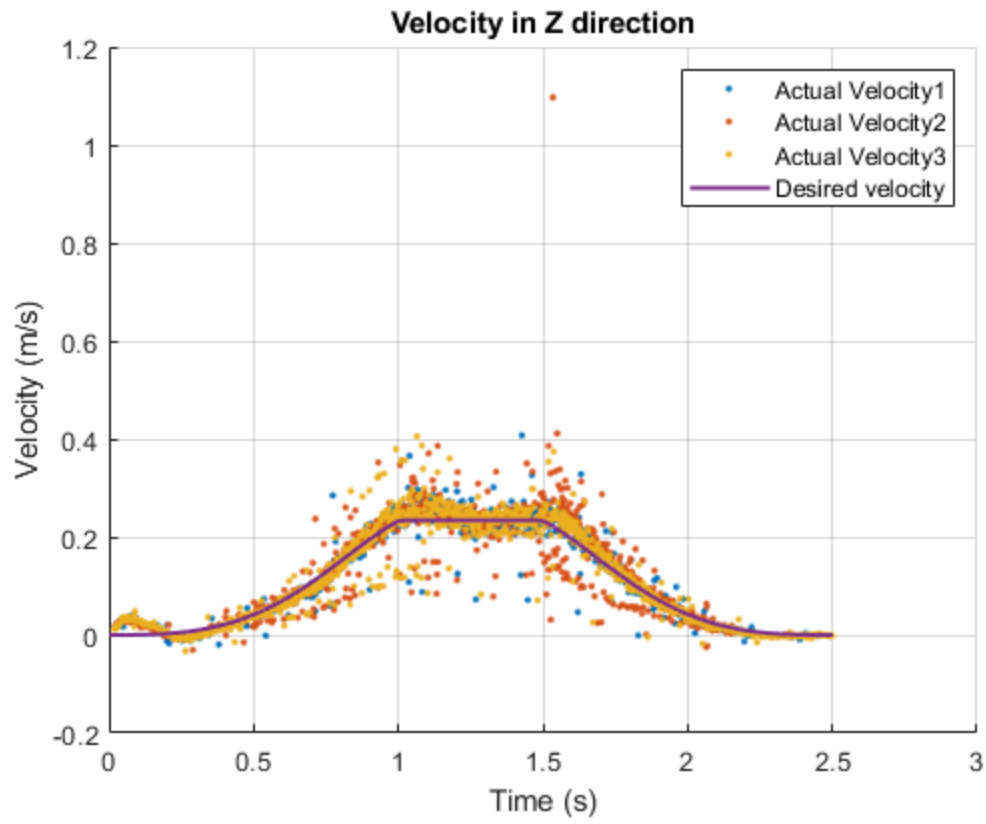
```

```

Warning: Ignoring extra legend entries.
Warning: Ignoring extra legend entries.
Warning: Ignoring extra legend entries.
Warning: Ignoring extra legend entries.
Warning: Ignoring extra legend entries.
Warning: Ignoring extra legend entries.

```





*Published with MATLAB® R2023a*