
```
% Clear the workspace
clc
clear all
close all
addpath("Api")

% Initialize Communication with CoppeliaSim
[ret_status, sim, clientID] = initializeComm();

% Make sure that initialization is succesful
if (ret_status == 0)
    % Collect all the variables

    [returnCode, Quad] = getObjectReference(sim, clientID, 'Quadricopter');
    % Reference the 'Quadcopter_target' object in CoppeliaSim as 'target' in
    MATLAB
    [returnCode, target] = getObjectReference(sim,
clientID, 'Quadricopter_target');
    % Return Quad's position as quad_pos
    [returnCode, quad_pos] = getObjectPosition(sim, clientID, Quad, 1);
    % Get the position of the green sphere (target) from the coppeliaSim
    [returnCode, target_position] = getObjectPosition(sim, clientID, target,
1);

    % sphere position
    p_x_star = target_position(1);
    p_y_star = target_position(2);
    p_z_star = target_position(3);

    % Current Positions
    p_x = [];
    p_y = [];
    p_z = [];

    % Iterations
    i = 0;
    j = 0;
    k = 1;

    % seemed to need this, so they exist here first
    targX =0;
    targXinv =0;
    targZ =0;
    targZinv =0;

    % Set a tolerance bc we can't get to targets without an overshoot
    err = 0.25;

    % Add some time
    % variables for time/ when to stop sim
```

```

T = 35; % How long you want to collect the data for

t=clock;
startTime=t(6);
currentTime=t(6);

% While sim is on
while(currentTime-startTime < T)
    % if the sim is running do this
    if(sim.simxGetConnectionId(clientID) ~= -1)

        % Insert code here:
        % Collect info on position
        [returnCode, quad_pos] = getObjectPosition(sim, clientID, Quad, 0);
        p_x = [p_x; quad_pos(1)];
        p_y = [p_y; quad_pos(2)];
        p_z = [p_z; quad_pos(3)];

        % Tell Target to move, if statements in xz
        % x moves: 0 -> 1
        %   0<x<1      z = 1      y = 0
        % z moves: 1 -> 2
        %   x = 1      1<z<2      y = 0
        % x moves: 1 -> 0
        %   1>x>0      z = 2      y = 0
        % z moves: 2 -> 1
        %   x = 0      2<z<1      y = 0

        % Set constant
        p_y_star = 0;

        if(quad_pos(1) <= 1 && quad_pos(3) < 2-err)
            % px < 1.05 && pz < 2.05 at (1,0,1)
            targX      = 1+err;
            targXinv = 0;
            p_z_star = 1;
            p_x_star = i/(i+1)*targX + 1/(i+1)*targXinv;
            i = i + 0.0001;
        elseif(quad_pos(1) > 1 && quad_pos(3) < 2)
            targZ      = 2+err;
            targZinv = 1;
            p_x_star = 1;
            p_z_star = k/(i+1)*targZ + 1/(k+1)*targZinv;
            k = k + 0.0001;
        elseif(quad_pos(1) > 0 && quad_pos(3) >= 2)
            targX      = 0-err+0.01;
            targXinv = 1;
            p_z_star = 2;
            p_x_star = j/(j+1)*targX + 1/(j+1)*targXinv;
            j = j + 0.0001;
            % (0-0.25, 0, 2)
        elseif( quad_pos(1) < 0 && quad_pos(3) > 1 )
            p_x_star = 0;

```

```

        p_z_star = 1-err;

        % Read current time
        t = clock;
        currentTime = t(6);
        % final send for 1 square
        position = [p_x_star,p_y_star,p_z_star];
        [returnCode] = setObjectPosition(sim, clientID, target,
position);

        pause(1)
        % reset to remove accidental KICK to the system
        i=0;
        j=0;
        k=0;

end

% Read current time
t = clock;
currentTime = t(6);

% send to target
position = [p_x_star,p_y_star,p_z_star];
[returnCode] = setObjectPosition(sim, clientID, target, position);

%-----

% position = [0,0,2]

% % Send to target
% [returnCode] = setObjectPosition(sim, clientID, target, position);

% position = [1,0,2]

% % Send to target
% [returnCode] = setObjectPosition(sim, clientID, target, position);

% pause(1)
% position = [1,0,1]

% % Send to target
% [returnCode] = setObjectPosition(sim, clientID, target, position);

% pause(1)
% position = [0,0,1]

% % Send to target
% [returnCode] = setObjectPosition(sim, clientID, target, position);

% [returnCode, quad_pos] = getObjectPosition(sim, clientID, Quad, 0);
% p_x = [p_x; quad_pos(1)];
% p_y = [p_y; quad_pos(2)];

```

```

    % p_z = [p_z; quad_pos(3)];

    % pause(10)

    end
end
% plot
positions = [p_x, p_y, p_z];
figure(1)
plot3(positions(:,1),positions(:,2),positions(:,3),'linewidth',3);
title('3D Quadrotor path from CoppeliaSim')
xlabel('x [m]')
ylabel('y [m]')
zlabel('z [m]')
saveas(gcf,'Part2Box_3d.png');
figure(2)
plot(positions(:,1),positions(:,3))
title('2D Quadrotor path from CoppeliaSim')
xlabel('x [m]')
ylabel('z [m]')
saveas(gcf,'Part2Box_2d.png');

% Kill the connection to CoppeliaSim
uninitializeComm(sim, clientID)

else
    disp('Unable to connect to CoppeliaSim')
end

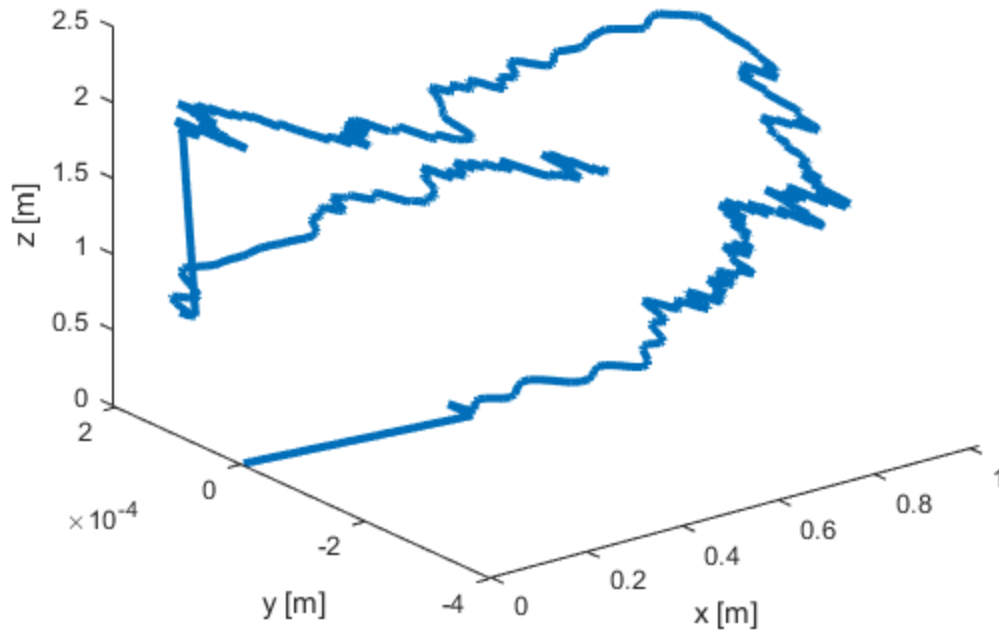
Note: always make sure you use the corresponding remoteApi library
(i.e. 32bit Matlab will not work with 64bit remoteApi, and vice-versa)
Connected to CoppeliaSim

ans =

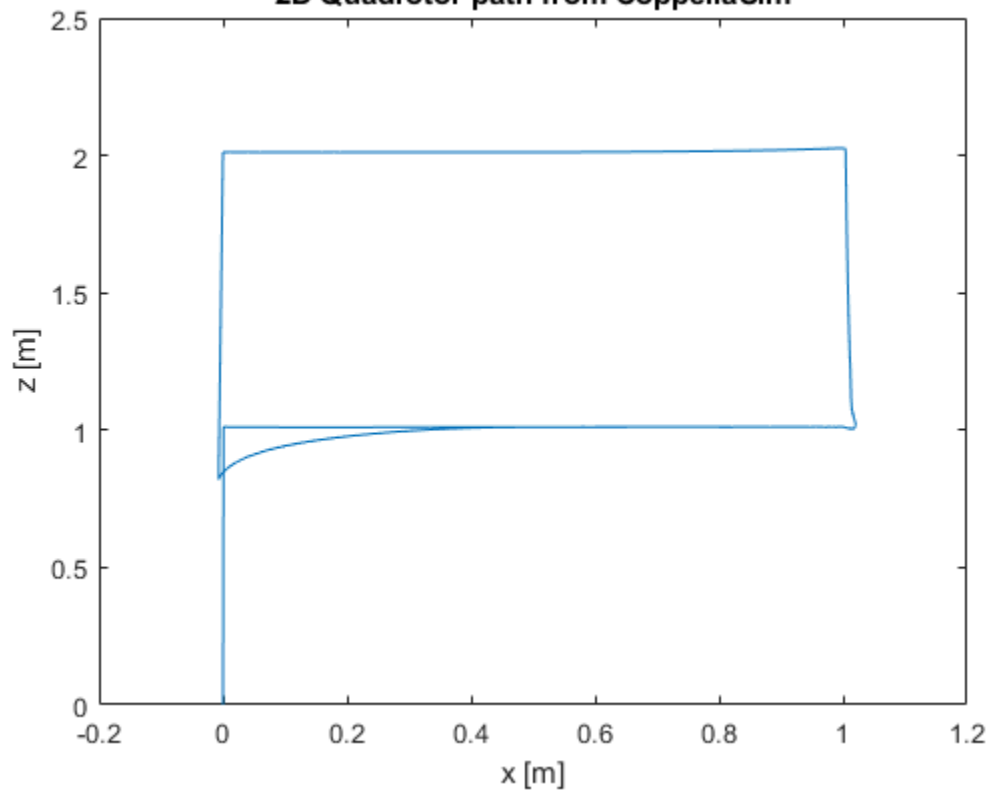
    0

```

3D Quadrotor path from CoppeliaSim



2D Quadrotor path from CoppeliaSim



Published with MATLAB® R2022b