Ρομποτική Ι: Ανάλυση – Έλεγχος – Εργαστήριο

7ο Εξάμηνο 2022 – 2023

Εξαμηνιαία Εργασία

Ζαρίφης Στέλιος – el20435

Email: el20435@mail.ntua.gr

Contents

B. Κινηματική Προσομοίωση 1
Ερώτημα 6 1
Kinematic Simulation 3

B. Κινηματική Προσομοίωση

Ερώτημα 6

Γνωρίζουμε τις οριακές συνθήκες της κίνησης (αρχική και τελική θέση) και χρησιμοποιούμε πολυωνυμικές συναρτήσεις παρεμβολής για το σχεδιασμό της τροχιάς.

Επειδή γνωρίζουμε αρχική και τελική θέση (έστω $P_A = [x_A, y_A, z_A], P_B = [x_B, y_B, z_B]$) και επίσης θεωρούμε αρχική και τελική ταχύτητες μηδενικές αφού είναι τα ακραία σημεία της τροχιάς (τότε αλλάζει η φορά κίνησης άρα μηδενίζεται στιγμιαία η ταχύτητα), θεωρούμε πολυωνυμική συνάρτηση 3^{ου} βαθμού (4 οριακές συνθήκες – 4 άγνωστοι):

$$S(t) = a_3 t^3 + a_2 t^2 + a_1 t + a_0$$

Οριακές συνθήκες:

$$S(0) = P_A \Rightarrow a_0 = [x_A, y_A, z_A],$$
$$S(t_f) = P_B \Rightarrow a_3 t_f^3 + a_2 t_f^2 + a_1 t_f + [x_A, y_A, z_A] = [x_B, y_B, z_B]$$
$$\dot{S}(0) = [0,0,0] \Rightarrow a_1 = [0,0,0],$$
$$\dot{S}(t_f) = [0,0,0] \Rightarrow 3a_3 t_f^2 + 2a_2 t_f + [0,0,0] = [0,0,0] \Rightarrow 3a_3 t_f^2 + 2a_2 t_f = [0,0,0]$$

Καταλήγουμε, λοιπόν:

$$\begin{cases} a_0 = [x_A, y_A, z_A] \\ a_3 t_f^3 + a_2 t_f^2 + [x_A, y_A, z_A] = [x_B, y_B, z_B] \\ a_1 = [0,0,0] \\ 3a_3 t_f^2 + 2a_2 t_f = [0,0,0] \end{cases} \Rightarrow \begin{cases} a_0 = [x_A, y_A, z_A] \\ a_3 t_f^3 + a_2 t_f^2 + [x_A, y_A, z_A] = [x_B, y_B, z_B] \\ a_1 = [0,0,0] \\ \alpha_2 = -\frac{3}{2}\alpha_3 t_f \end{cases} \Rightarrow$$

$$\Rightarrow \begin{cases} a_0 = [x_A, y_A, z_A] \\ a_3 t_f^3 - \frac{3}{2}\alpha_3 t_f^3 + [x_A, y_A, z_A] = [x_B, y_B, z_B] \\ a_1 = [0,0,0] \\ \alpha_2 = -\frac{3}{2}\alpha_3 t_f \end{cases} \Rightarrow \boxed{\begin{cases} a_0 = [x_A, y_A, z_A] \\ a_1 = [0,0,0] \\ \alpha_2 = \frac{3}{t_f^2}[x_B - x_A, y_B - y_A, z_B - z_A] \\ a_3 = -\frac{2}{t_f^3}[x_B - x_A, y_B - y_A, z_B - z_A] \end{cases}}$$

Η τροχιά στο χώρο που ικανοποιεί τις προδιαγραφές είναι η εξής:

$$S(t) = P_A + \frac{3}{t_f^2}(P_B - P_A)t^2 - \frac{2}{t_f^3}(P_B - P_A)t^3$$

Kinematic Simulation

Για την προσομοίωση στο MATLAB εργαζόμαστε ως εξής:

Ορίζουμε τα μήκη των συνδέσμων (αυθαίρετα), την περίοδο δειγματοληψίας του Ρομπότ και την ημιπερίοδο της κίνησης (χρόνος μετατόπισης τελικού σημείου δράσης από το σημείο A στο σημείο B) και το διάνυσμα του χρόνου.

```
l0 = 5.0;  %% in cm
l1 = 10.0;
l2 = 8.0;
l3 = 12.0;

%% Sampling period
dt = 0.001;

Tf=10.0;     % 10sec semi period
t=0:dt:Tf;
```

Επιπλέον, δηλώνουμε και τα σημεία A, B στο χώρο.

```
% Initial/final position of task - space trajectory
xd0 = 0.0;
xd1 = 10.0;
yd0 = -12.00;
yd1 = 12.00;
zd0 = 20.00;
zd1 = 20.00;
Pa = [xd0, yd0, zd0];
Pb = [xd1, yd1, zd1];
```

Ακόμα, φτιάχνουμε τα διανύσματα που αναπαριστούν την επιθυμητή τριδιάστατη τροχιά σε σχέση με το χρόνο, όπως υπολογίσαμε στην αρχή του Β' Μέρους.

```
% Compute kmax based on Tf and dt
kmax = floor(Tf/dt) + 1;

% Initialize xd, yd, and zd vectors
xd = zeros(kmax, 1);
yd = zeros(kmax, 1);
zd = zeros(kmax, 1);

% Fill in the first two elements of xd and yd vectors
xd = [xd0; xd0 + 3/(Tf^2) * (xd1 - xd0) * dt^2 - 2/(Tf^3) * (xd1 - xd0) * dt^3];
yd = [yd0; yd0 + 3/(Tf^2) * (yd1 - yd0) * dt^2 - 2/(Tf^3) * (yd1 - yd0) * dt^3];
zd = [zd0; zd0];

% Fill in the rest of xd, yd, and zd vectors using the specified trajectories
for k = 3:kmax
    tNew = (k-2) * dt;
    xd(k) = xd0 + 3/(Tf^2) * (xd1 - xd0) * tNew^2 - 2/(Tf^3) * (xd1 - xd0) * tNew^3;
    yd(k) = yd0 + 3/(Tf^2) * (yd1 - yd0) * tNew^2 - 2/(Tf^3) * (yd1 - yd0) * tNew^3;
    zd(k) = zd0;
end
```

Από την αντίστροφη κινηματική ανάλυση υπολογίζουμε τις περιστροφές των αρθρώσεων σε σχέση με την επιθυμητή θέση του τελικού σημείου δράσης

```
qd(:, 3) = acos((xd.^2 + yd.^2 + (zd - l0).^2 - l1.^2 - l2.^2 - l3.^2)./(2*l2.*l3));
qd(:, 2) = asin(yd./sqrt((l2 + l3.*cos(qd(:, 3))).^2 + (l3.*sin(qd(:, 3))).^2)) -
atan2(l3.*sin(qd(:, 3)), (l2 + l3.*cos(qd(:, 3))));
qd(:, 1) = -asin(xd./sqrt((l3.*cos(qd(:, 2)+qd(:, 3)) + l2.*cos(qd(:, 2))).^2 +
l1.^2)) + atan2((l3.*cos(qd(:, 2)+qd(:, 3)) + l2.*cos(qd(:, 2))), l1);
```

Για ευκολία συμβολίζουμε:

```
c1 = cos(qd(:, 1));
s1 = sin(qd(:, 1));
c2 = cos(qd(:, 2));
s2 = sin(qd(:, 2));
c3 = cos(qd(:, 3));
s3 = sin(qd(:, 3));
```

Υπολογίζουμε τις θέσεις των αρθρώσεων στο χώρο σχετικά με το χρόνο, κοιτώντας τους μετασχηματισμούς από το Α’ Μέρος και τη ρομποτική διάταξη.

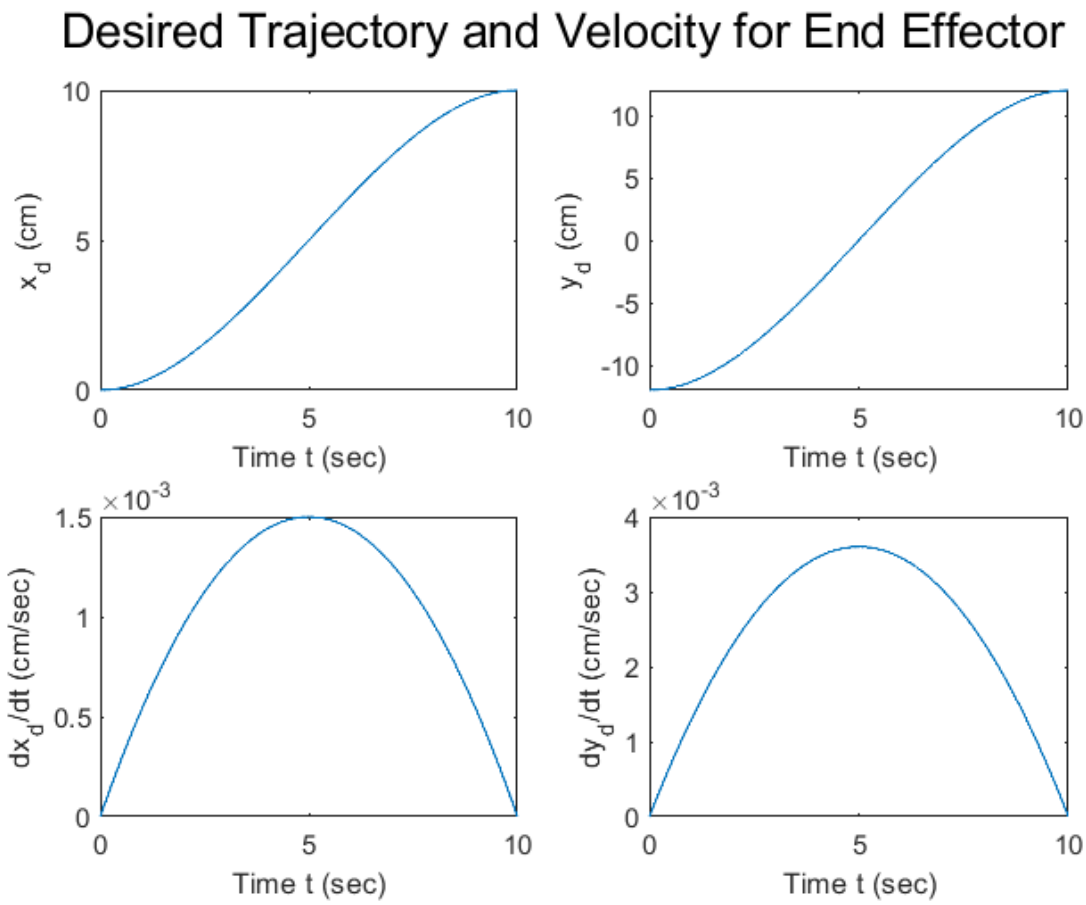
```
xd1 = zeros(length(qd(:, 1)), 1);
yd1 = zeros(length(qd(:, 1)), 1);
zd1 = l0*ones(length(xd1));

xd2 = xd1 - l1*s1;
yd2 = yd1;
zd2 = zd1 + l1*c1;

xd3 = l2*c1.*c2 - l1*s1;
yd3 = l2*s2;
zd3 = l2*s1.*c2 + l0;

xd4 = l3*c1.*c2.*c3 - l3*c1.*s2.*s3 + l2*c1.*c2 - l1*s1;
yd4 = l3*s2.*c3 + l3*c2.*s3 + l2*s2;
zd4 = l3*s1.*c2.*c3 - l3*s1.*s2.*s3 + l2*s1.*c2 + l1*c1 + l0;
```

Σχεδιάζουμε την επιθυμητή τροχιά και ταχύτητα του End – Effector

```
fig1 = figure;
subplot(2,2,1);
plot(t,xd);
ylabel('x_d (cm)');
xlabel('Time t (sec)');

subplot(2,2,2);
plot(t,yd);
ylabel('y_d (cm)');
xlabel('Time t (sec)');

subplot(2,2,3);
plot(t,dxdds);
ylabel('dx_d/dt (cm/sec)');
xlabel('Time t (sec)');

subplot(2,2,4);
plot(t,dydds);
ylabel('dy_d/dt (cm/sec)');
xlabel('Time t (sec)');

% Add a title to the entire figure
sgtitle('Desired Trajectory and Velocity for End Effector')

% Adjust the font size of the title
sgtitle('Desired Trajectory and Velocity for End Effector','FontSize',16)
```

Τα αποτελέσματα φαίνονται στο επόμενο διάγραμμα. Βλέπουμε πως οι συνθήκες της τροχιάς πληρούνται:

1. Αρχική θέση $x_0 = 0 \wedge y_0 = -10$
2. Τελική θέση $x_f = 10 \wedge y_f = 10$
3. Αρχική ταχύτητα $v_{x0} = 0 \wedge v_{y0} = 0$
4. Τελική ταχύτητα $v_{xf} = 0 \wedge v_{yf} = 0$
5. Ημιπερίοδος $T = 10\ sec$

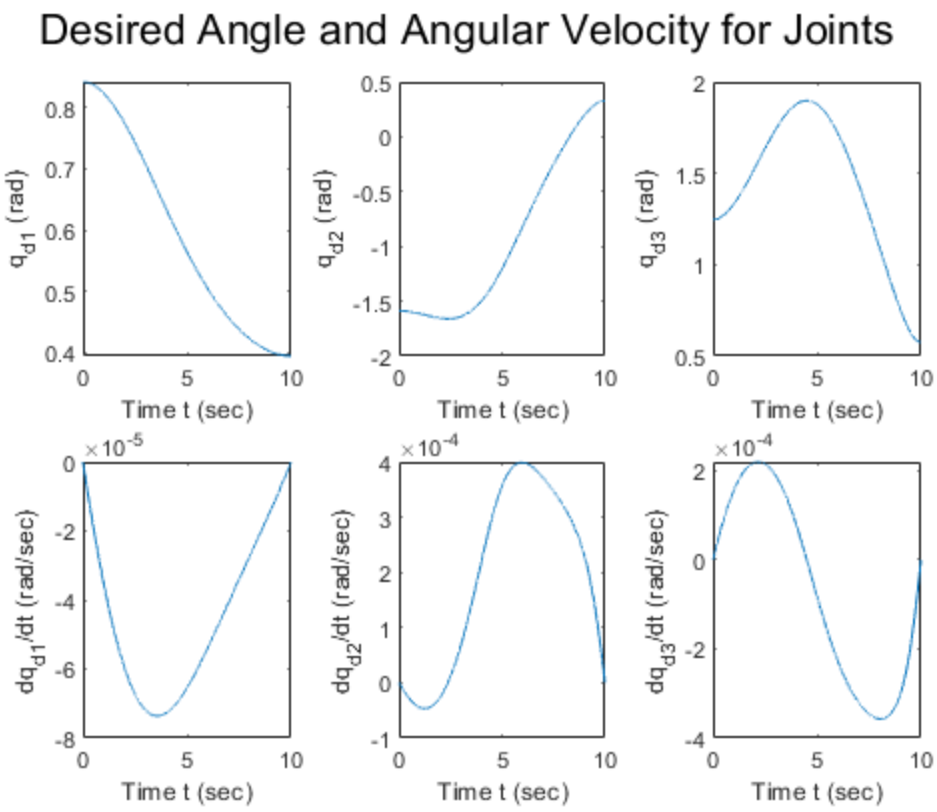

Desired Trajectory and Velocity for End Effector
x_d (cm)
y_d (cm)
dx_d/dt (cm/sec)
dy_d/dt (cm/sec)
×10^-3
Time t (sec)

Σχεδιάζουμε τις επιθυμητές γωνίες στροφής και γωνιακές ταχύτητες των αρθρώσεων.

```
fig2 = figure;
subplot(2,3,1);
plot(t,qd(:,1));
ylabel('q_{d1} (rad)');
xlabel('Time t (sec)');

subplot(2,3,2);
plot(t,qd(:,2));
ylabel('q_{d2} (rad)');
xlabel('Time t (sec)');

subplot(2,3,3);
plot(t,qd(:,3));
ylabel('q_{d3} (rad)');
xlabel('Time t (sec)');

subplot(2,3,4);
plot(t,dqds(:,1));
ylabel('dq_{d1}/dt (rad/sec)');
xlabel('Time t (sec)');

subplot(2,3,5);
plot(t,dqds(:,2));
ylabel('dq_{d2}/dt (rad/sec)');
xlabel('Time t (sec)');

subplot(2,3,6);
plot(t,dqds(:,3));
ylabel('dq_{d3}/dt (rad/sec)');
xlabel('Time t (sec)');

% Add a title to the entire figure
sgtitle('Desired Angle and Angular Velocity for Joints')

% Adjust the font size of the title
sgtitle('Desired Angle and Angular Velocity for Joints','FontSize',16)
```

Τα αποτελέσματα φαίνονται στο επόμενο διάγραμμα. Βλέπουμε πως οι συνθήκες της τροχιάς πληρούνται:

1. Αρχική γωνιακή ταχύτητα $\omega_{d1}(t=0)=0 \wedge \omega_{d2}(t=0)=0 \wedge \omega_{d3}(t=0)=0$
2. Τελική γωνιακή ταχύτητα $\omega_{d1}(t=t_f)=0 \wedge \omega_{d2}(t=t_f)=0 \wedge \omega_{d3}(t=t_f)=0$
3. Ημιπερίοδος $T=10\ sec$



Τα γραφήματα θέσης – ταχύτητας και γωνίας – γωνιακής ταχύτητας είναι αποθηκευμένα ως `desiredTrajectoryTaskSpace.png` και `desiredTrajectoryJointSpace.png`.

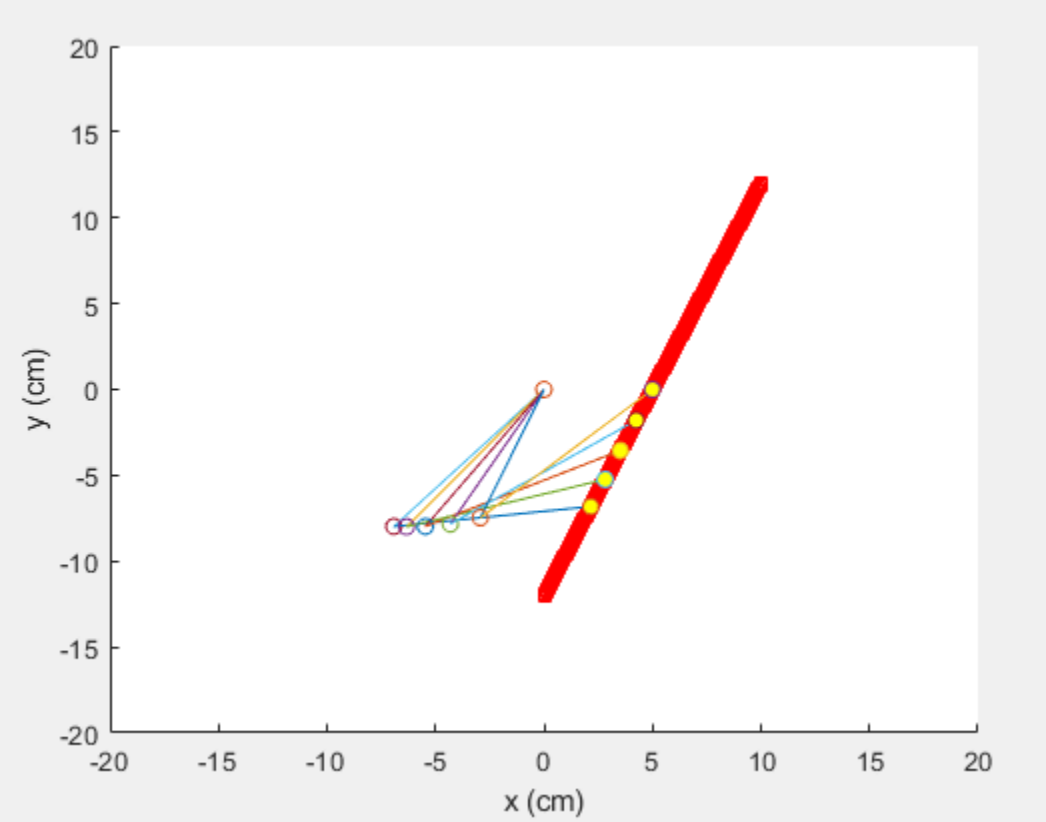
Ακόμα, σχεδιάζουμε 1 περίοδο της προβολής της κίνησης του ρομπότ στο xy επίπεδο και την αποθηκεύουμε στο αρχείο 2DSimulation.gif.

```
fig3 = figure;
axis([-20 20 -20 20]) %%set xy plot axes (caution: square axes, i.e. dx=dy)
axis on
hold on
xlabel('x (cm)');
ylabel('y (cm)');
plot(xd,yd,'rs');
dtk=500; %% plot robot position every dtk samples, to animate its motion
plot([0],[0],'o');

n_plots = 0; % counter for number of plots
window_size = 10; % number of plots to display at a time
handles = zeros(window_size, 1); % circular buffer to store plot handles
idx = 1; % index to the oldest plot handle

plot_handles = {};
for tk=1:dtk:kmax    %%%
    pause(0.1);     %% pause motion to view successive robot configurations

    h1 = plot([0,xd3(tk)],[0,yd3(tk)]);
    h2 = plot([xd3(tk)],[yd3(tk)],'o');
    h3 = plot([xd3(tk),xd4(tk)],[yd3(tk),yd4(tk)]);
    h4 = plot([xd4(tk)],[yd4(tk)],'o','MarkerFaceColor','yellow');

    % add current plot handles to the end of the queue
    plot_handles{end+1} = [h1, h2, h3, h4]; %, h5];

    % check if queue has reached its maximum size
    if length(plot_handles) > 5
        % get handle of oldest plot
        oldest_handle = plot_handles{1};
        % delete oldest plot
        delete(oldest_handle);
        % remove oldest plot handle from queue
        plot_handles(1) = [];
    end

    % Capture the current frame
    frame = getframe(fig3);
    im = frame2im(frame);
    [imind, cm] = rgb2ind(im, 256);

    % Write the frame to the GIF file
    if tk == 1
        imwrite(imind, cm, '2DSimulation.gif', 'gif', 'Loopcount', inf, 'DelayTime',
0.1);
    else
        imwrite(imind, cm, '2DSimulation.gif', 'gif', 'WriteMode', 'append',
'DelayTime', 0.1);
    end
end
```

```
for tk=kmax:-dtk:1    %%%
    pause(0.1);     %% pause motion to view successive robot configurations

    h1 = plot([0,xd3(tk)],[0,yd3(tk)]);
    h2 = plot([xd3(tk)],[yd3(tk)],'o');
    h3 = plot([xd3(tk),xd4(tk)],[yd3(tk),yd4(tk)]);
    h4 = plot([xd4(tk)],[yd4(tk)],'o','MarkerFaceColor','yellow');

    % add current plot handles to the end of the queue
    plot_handles{end+1} = [h1, h2, h3, h4]; %, h5];

    % check if queue has reached its maximum size
    if length(plot_handles) > 5
        % get handle of oldest plot
        oldest_handle = plot_handles{1};
        % delete oldest plot
        delete(oldest_handle);
        % remove oldest plot handle from queue
        plot_handles(1) = [];
    end

    % Capture the current frame
    frame = getframe(fig3);
    im = frame2im(frame);
    [imind, cm] = rgb2ind(im, 256);

    imwrite(imind, cm, '2DSimulation.gif', 'gif', 'WriteMode', 'append', 'DelayTime',
0.1);

end

% get handle of oldest plot
oldest_handle = plot_handles{1};
% delete oldest plot
delete(oldest_handle);
% remove oldest plot handle from queue
plot_handles(1) = [];

% get handle of oldest plot
oldest_handle = plot_handles{1};
% delete oldest plot
delete(oldest_handle);
% remove oldest plot handle from queue
plot_handles(1) = [];

% get handle of oldest plot
oldest_handle = plot_handles{1};
% delete oldest plot
delete(oldest_handle);
% remove oldest plot handle from queue
plot_handles(1) = [];

% get handle of oldest plot
oldest_handle = plot_handles{1};
% delete oldest plot
```

```
delete(oldest_handle);
% remove oldest plot handle from queue
plot_handles(1) = [];
```

Εδώ βλέπουμε ένα στιγμιότυπο της κίνησης.



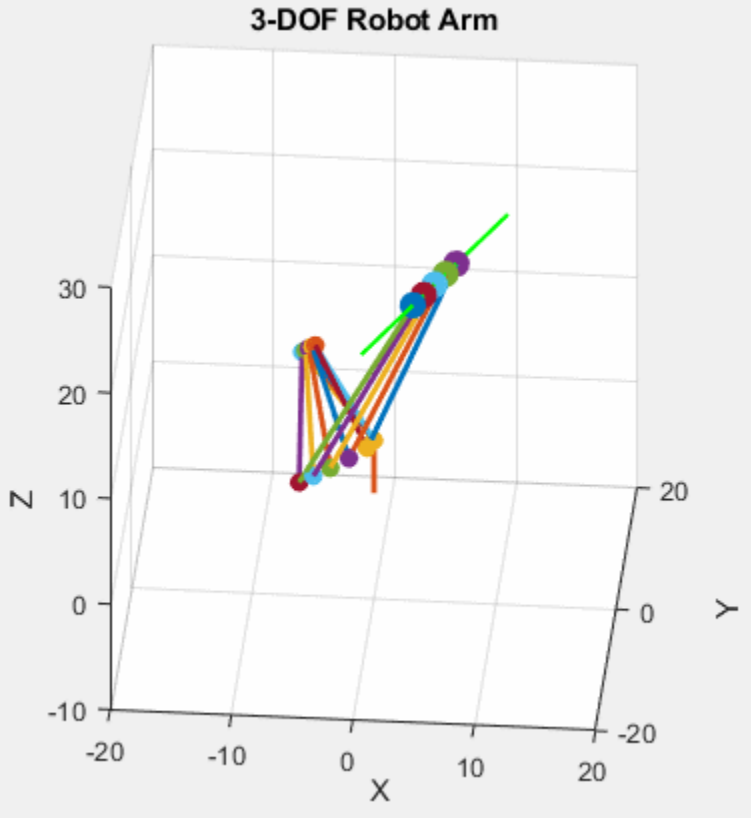
Τέλος, σχεδιάζουμε 2 περιόδους κίνησης του ρομπότ στο χώρο και αποθηκεύουμε την κίνηση στο αρχείο 3DSimulation.gif.

```
fig4 = figure;
hold on;
axis equal;
grid on;

% Set axis limits
xlim([-20, 20])
ylim([-20, 20])
zlim([-10, 30])
plot_handles = {};
plot3([Pa(1), Pb(1)], [Pa(2), Pb(2)], [Pa(3), Pb(3)], 'linewidth', 1.5, 'color',
'g');
plot3([0, xd1(tk)], [0, yd1(tk)], [0, zd1(tk)], 'linewidth', 2);
scatter3(xd1(tk), yd1(tk), zd1(tk), 50, 'filled');

for iter = 1:2
    for tk=1:dtk:kmax    %%%
        pause(0.1); %% pause motion to view successive robot configurations

        % Plotting the robot arm links
        h1 = plot3([xd1(tk), xd2(tk)], [yd1(tk), yd2(tk)], [zd1(tk), zd2(tk)],
'linewidth', 2);
        h2 = plot3([xd2(tk), xd3(tk)], [yd2(tk), yd3(tk)], [zd2(tk), zd3(tk)],
'linewidth', 2);
        h3 = plot3([xd3(tk), xd4(tk)], [yd3(tk), yd4(tk)], [zd3(tk), zd4(tk)],
'linewidth', 2);

        % Plotting the joint locations
        h4 = scatter3(xd2(tk), yd2(tk), zd2(tk), 50, 'filled');
        h5 = scatter3(xd3(tk), yd3(tk), zd3(tk), 50, 'filled');
        h6 = scatter3(xd4(tk), yd4(tk), zd4(tk), 100, 'filled');

        % Adding labels and title
        xlabel('X');
        ylabel('Y');
        zlabel('Z');
        title('3-DOF Robot Arm');

        % Setting view angle
        view(5, 30);
        % add current plot handles to the end of the queue
        plot_handles{end+1} = [h1, h2, h3, h4, h5, h6];

        % check if queue has reached its maximum size
        if length(plot_handles) > 5
            % get handle of oldest plot
            oldest_handle = plot_handles{1};
            % delete oldest plot
            delete(oldest_handle);
            % remove oldest plot handle from queue
            plot_handles(1) = [];
        end
```

```
        % Capture the current frame
        frame = getframe(fig4);
        im = frame2im(frame);
        [imind, cm] = rgb2ind(im, 256);

        % Write the frame to the GIF file
        if iter == 1
            if tk == 1
                imwrite(imind, cm, '3DSimulation.gif', 'gif', 'Loopcount', inf,
'DelayTime', 0.1);
            else
                imwrite(imind, cm, '3DSimulation.gif', 'gif', 'WriteMode', 'append',
'DelayTime', 0.1);
            end
        end
    end

    for tk=kmax:-dtk:1    %%%
        pause(0.1); %% pause motion to view successive robot configurations

        % Plotting the robot arm links
        h1 = plot3([xd1(tk), xd2(tk)], [yd1(tk), yd2(tk)], [zd1(tk), zd2(tk)],
'linewidth', 2);
        h2 = plot3([xd2(tk), xd3(tk)], [yd2(tk), yd3(tk)], [zd2(tk), zd3(tk)],
'linewidth', 2);
        h3 = plot3([xd3(tk), xd4(tk)], [yd3(tk), yd4(tk)], [zd3(tk), zd4(tk)],
'linewidth', 2);

        % Plotting the joint locations
        h4 = scatter3(xd2(tk), yd2(tk), zd2(tk), 50, 'filled');
        h5 = scatter3(xd3(tk), yd3(tk), zd3(tk), 50, 'filled');
        h6 = scatter3(xd4(tk), yd4(tk), zd4(tk), 100, 'filled');

        % Adding labels and title
        xlabel('X');
        ylabel('Y');
        zlabel('Z');
        title('3-DOF Robot Arm');

        % Setting view angle
        view(5, 30);
        % add current plot handles to the end of the queue
        plot_handles{end+1} = [h1, h2, h3, h4, h5, h6];

        % check if queue has reached its maximum size
        if length(plot_handles) > 5
            % get handle of oldest plot
            oldest_handle = plot_handles{1};
            % delete oldest plot
            delete(oldest_handle);
            % remove oldest plot handle from queue
            plot_handles(1) = [];
        end
```

```
        % Capture the current frame
        frame = getframe(fig4);
        im = frame2im(frame);
        [imind, cm] = rgb2ind(im, 256);

        if iter == 1
            imwrite(imind, cm, '3DSimulation.gif', 'gif', 'WriteMode', 'append',
'DelayTime', 0.1);
        end
    end
end

% get handle of oldest plot
oldest_handle = plot_handles{1};
% delete oldest plot
delete(oldest_handle);
% remove oldest plot handle from queue
plot_handles(1) = [];

% get handle of oldest plot
oldest_handle = plot_handles{1};
% delete oldest plot
delete(oldest_handle);
% remove oldest plot handle from queue
plot_handles(1) = [];

% get handle of oldest plot
oldest_handle = plot_handles{1};
% delete oldest plot
delete(oldest_handle);
% remove oldest plot handle from queue
plot_handles(1) = [];

% get handle of oldest plot
oldest_handle = plot_handles{1};
% delete oldest plot
delete(oldest_handle);
% remove oldest plot handle from queue
plot_handles(1) = [];

disp('...DONE');
```

Και ένα στιγμιότυπο της κίνησης.



Το αρχείο `el20435_partB.m` παράγει τα γραφήματα και την προσομοίωση. Προσοχή, οι προσομοιώσεις αποθηκεύονται στον τρέχοντα φάκελο του MATLAB!