UNIVERZITET U TUZLI
FAKULTET ELEKTROTEHNIKE
Akademska 2023/2024.

Zadaca 1

Robotika i Masinska vizija

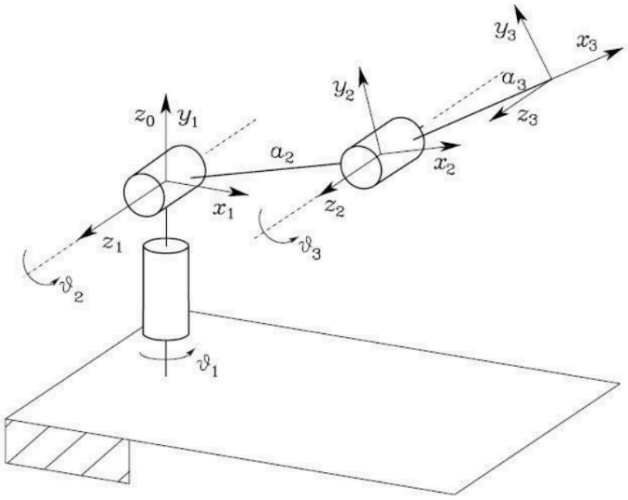
Antonija Mrkonjic

Sadržaj

GitHub repository: https://github.com/antonijaMr/RMV

Zadatak 1

Za antropomorfnu ruku kreirati kinematski model korištenjem Robotics toolboxa. Isplanirati zajednički trajektoriju zglobova u tranjanju od tri sekunde da manipulator iz konfiguracije $q_o = [0\ 0\ 0]^T$ pređe u konfiguraciju $q_o = [\pi/2\ \ -\ \pi/3\ \ \pi/2]^T$. Skicirati isplanirane trajektorije i animirati kretanje manipulatora.



Rješenje:
Kako bi kreirali model antropomorfne ruke potrebno je da:
Definisemo kinematski model koristeci Denavit-Hartenbergovu reprezentaciju za definiranje zglobova i segmenata ruke. Zatim cemo isplanirati trajektoriju koristecu funkcije za interpolaciju izmedju pocetne i krajnje kofiguracije koje su zadate.

```
L1 = Link('d', 0, 'a', 1, 'alpha', pi/2);  % Prvi zglob
L2 = Link('d', 0, 'a', 1, 'alpha', 0);    % Drugi zglob
L3 = Link('d', 0, 'a', 1, 'alpha', 0);    % Treći zglob
```

Prvo smo definisali tri linka koji cine antropomorfnu ruku. Ruka koju modelujemo ima tri stepena slobode. Linkove smo napravili pozivajuci Link funkciju cije argumnti su: d - duljina pomaka na z osi, a - duljina pomaka na x osi, i alpha - stepen rotacije oko x ose

```
robot = SerialLink([L1 L2 L3], 'name', 'Antropomorfna Ruka');
```

Pomocu funkcije SerialLink cemo napraviti SerialLink objekat koji ima metode i atribute pomocu kojih mozemo da analiziramo ponasanje robota.

```
% Korak 2: Planiranje trajektorije
% Početna i krajnja konfiguracija
q0 = [0 0 0];           % Početna konfiguracija
qf = [pi/2 -pi/3 pi/2]; % Krajnja konfiguracija

T = 3; % trajanje u sekundama
t = linspace(0, T, 100); % Vremenski vektor
```

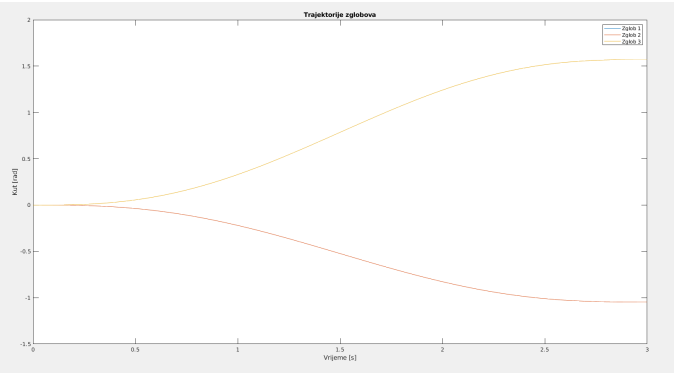
Definisemo pocenu i krajnju konfiguraciju robota (q0 i qf). Konfiguracije su zadate kao vektori zglobova.

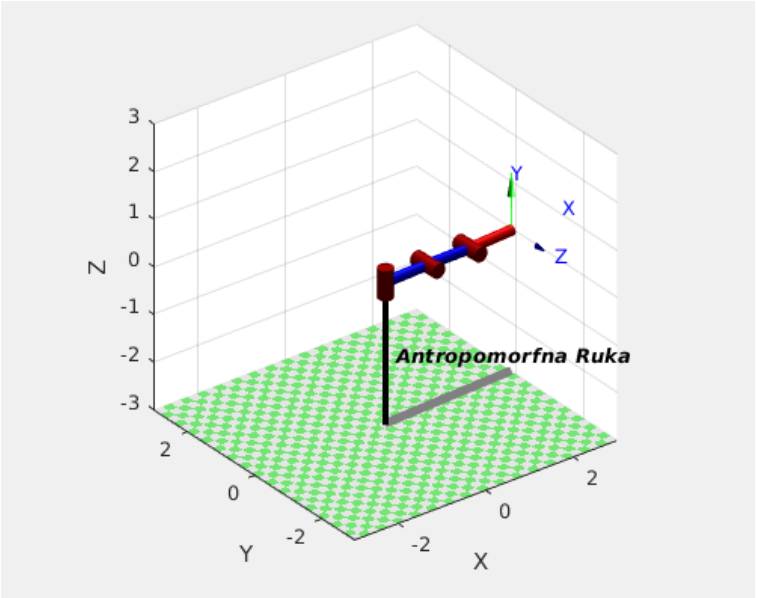
```
[q, qd, qdd] = jtraj(q0, qf, t);
```

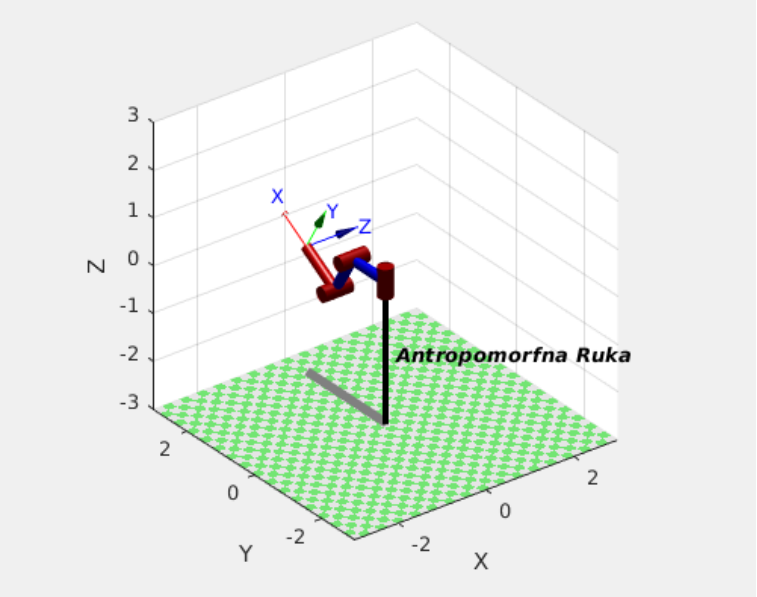
Generisemo trajektoriju za svaki zglob robota koristeci funkciju ‘jtraj’ koja koristi interpolaciju kako bi generisala glatku putanju izmedju pocetne i krajnje tacke.

```
figure;
plot(t, q);
title('Trajektorije zglobova');
xlabel('Vrijeme [s]');
ylabel('Kut [rad]');
legend('Zglob 1', 'Zglob 2', 'Zglob 3');

figure;
robot.plot(q);
```

Sada mozemo iscrtati trajektorije pojednih zglobova i vidjeti animaciju antropomorfne ruke.







Zadatak 2

Neka je vrh Puma560 manipulatora opisuje Kartezijansku trajektoriju i neka su početna i kranja tačka trajektorije vrha šake date sljedećim homogenim matricama transformcije:

$$T_i = \begin{bmatrix} 1 & 0 & 0 & 0.6 \\ 0 & 1 & 0 & -0.5 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$T_f = \begin{bmatrix} 1 & 0 & 0 & 0.4 \\ 0 & 1 & 0 & 0.5 \\ 0 & 0 & 1 & 0.2 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Rješavanjem problema inverzne kinematike, skicirati i animirati trajektorije svih zglobova manipulatora u toku putanje.

Rješenje:

Kako bi riješili ovaj problem koristeći matlab robotics toolbox prvo cemo definirati početne i krajnje homogene matrice transformacije.

```
Ti = [1 0 0 0.6;
      0 1 0 -0.5;
      0 0 1 0;
      0 0 0 1];

Tf = [1 0 0 0.4;
      0 1 0 0.5;
      0 0 1 0.2;
      0 0 0 1];
```

Ti je pocenta homogena matrica transformacije. Definise pocetnu poziciju i orijentaciju krajnje tacke manipulatora u prostoru.
Prvih 3x3 elementata (jedinice) predstavljaju rotacionu matricu (orijentaciju). Posljednja kolona (0.6, -0.5, 0) predstavljaju pozicije krajnje tacke.

Tf je krajnja homogena matrica transformacije. Definise krajnju poziciju i orijentaciju krajnje tacke manipulatora u prostoru.
Prvih 3x3 elementa (jednice) predstavljaju rotacionu matricu (orijentaciju). Posljednja kolona (0.4, 0.5, 0.2) predstavlja poziciju krajnje tacke.

Nakon ovog koraka cemo ucitati Puma560 model iz Robotics Toolbox-a pomocu sljedece komande:

```
mdl_puma560;
```

Ovim izrazom pozivamo skriptu mdl-puma560 koja definise kinematicke i dinamicke parametre naseg manipulatora.

Sada mozemo korisiti inverznu kinematiku za pocetnu tacku. Na osnovu krajnje tacke koju zelimo da zauzme krajnja tacka robota pomocu inverzne kinematike odredjujemo potrebne uglove zglobova.

Pomocu funkcije ‘ikine’ cemo izracunati zglobne uglove za pocetnu homogenu matricu transformacije Ti.

```
q_initial = p560.ikine(Ti);
```

Nakon ovog cemo uraditi isto i za krajnju tacku.

```
q_final = p560.ikine(Tf);
```

Sada je potrebno definisati vrijeme trajanja i broj tacaka na putanji, to cemo uraditi pomocu funkcije linspace.

```
t = linspace(0, 2, 100);
```

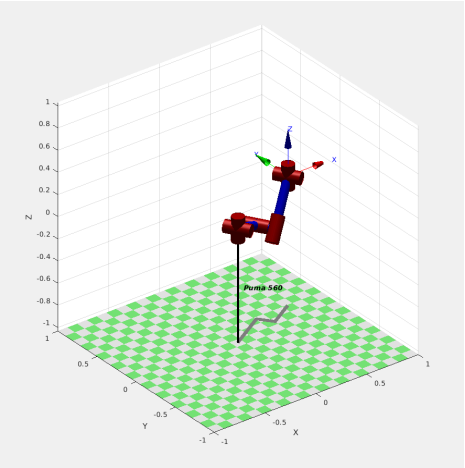
Ovime generisemo niz od 100 jednako rasporedjenih tacaka izmedju 0 i 2 sekunde. Mozemo ih definisati u skladu sa potrebama sistema koji kontrolisemo.

```
q_traj = jtraj(q_initial, q_final, t);
```

‘jtraj’ funkcija generise zglobne putanje sa zadatim pocetnim i krajnjim uglovima, koje smo definisali ranije, tokom definisanog vremena t.
Ovim smo dobili q_traj matricu gdje svaki red predstavlja uglove zglobova manipulatora u razlicitim vremenskim trenutcima. Ovu matricu cemo koristit za animaciju manipulatora.

```
figure;
p560.plot(q_traj);
```

Nad modelom p560 pozivamo funkciju plot koja animira manipulator Puma560 koji prati putanju q_traj.



Zadatak 3

Neka je početna konfiguracija Puma560 manipulatora data sa:
$q_z = [0\ 0\ 0\ 0\ 0\ 0]^T$ i neka na zglobove ne djeluje niti jedna druga sila osim gravitacije.
Skicirati trajektorije zglobova i animirati kretanje robota.

Rješenje:

U ovom primjeru cemo posmatrati kako moment gravitacione sile djeluju na kretanje robota. Neki roboti koriste utege ili opruge kako bi lakse savladavali djelovanje ove sile te time postigli da su motori manji i jeftiniji.
Kako bi vidjeli opterecenje sile korisiti cemo metodu gravload.

```
mdl_puma560;
qz=[0 0 0 0 0 0];
tau = p560.gravload(qz);

tau =

         0   37.4837    0.2489         0         0         0
```

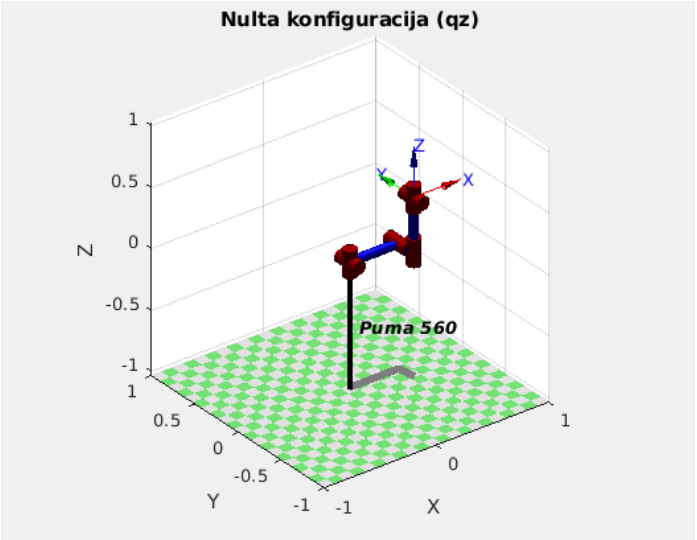
Okretni moment koji djeluje na zglob zbog gravitacije ovisi od poze robota. Ocito je tda je momenat koji djeluje na rameni zglob veci kada je ruka vodoravno ispruzena nego kada pokazuje prema gore.

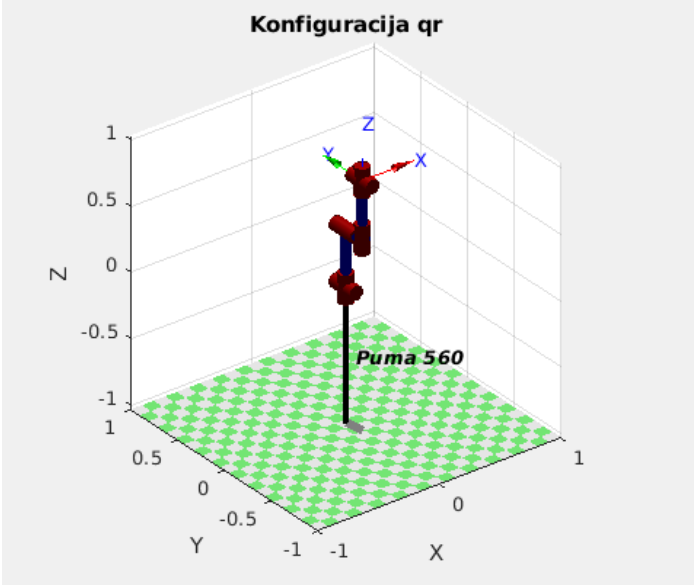
```
qs = [0, 0, -pi/2, 0, 0, 0];
qr = [0, pi/2, -pi/2, 0, 0, 0];
```

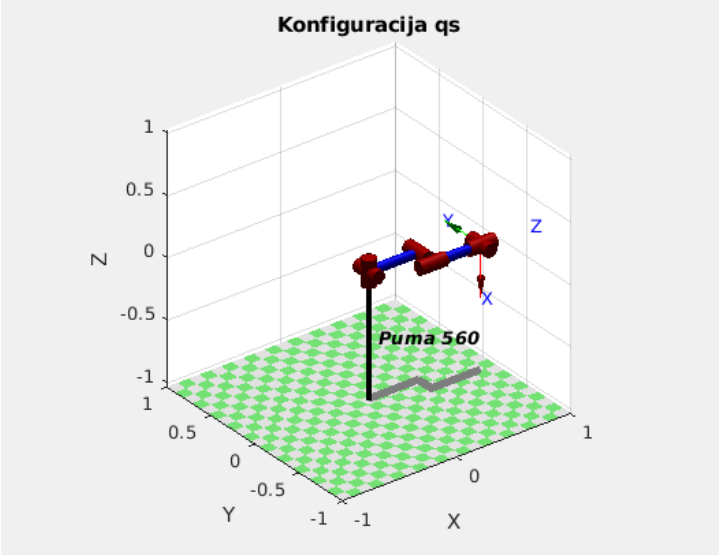
```
qz_tau = p560.gravload(qz);
qs_tau = p560.gravload(qs);
qr_tau = p560.gravload(qr);

disp('Gravitacijski moment za konfiguraciju qz:');
disp(qz_tau);

disp('Gravitacijski moment za konfiguraciju qs:');
disp(qs_tau);

disp('Gravitacijski moment za konfiguraciju qr:');
disp(qr_tau);

Gravitacijski moment za konfiguraciju qz:
         0   37.4837    0.2489         0         0         0

Gravitacijski moment za konfiguraciju qs:
   -0.0000   46.0069    8.7722    0.0000    0.0283         0

Gravitacijski moment za konfiguraciju qr:
         0   -0.7752    0.2489         0         0         0
```

Simuliranje polozaja robota za konfiguraciju qz, qs i qr

```
% Simuliranje kretanja robota za konfiguraciju qz
p560.plot(qz);
title('Nulta konfiguracija (qz)');
pause(2);

% Simuliranje kretanja robota za konfiguraciju qs
p560.plot(qs);
title('Konfiguracija qs');
pause(2);

% Simuliranje kretanja robota za konfiguraciju qr
p560.plot(qr);
title('Konfiguracija qr');
pause(2);
```

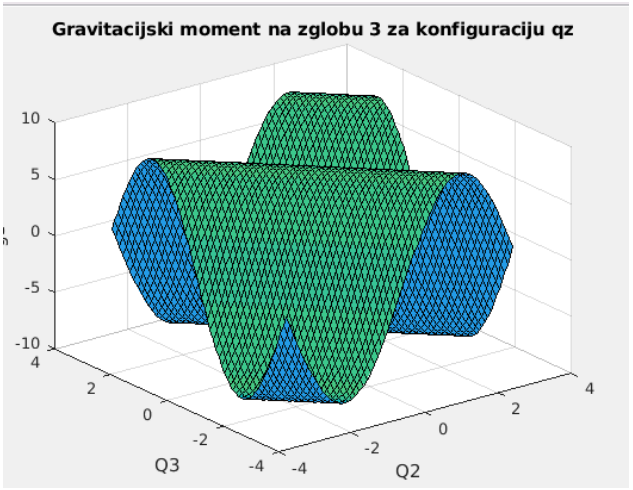




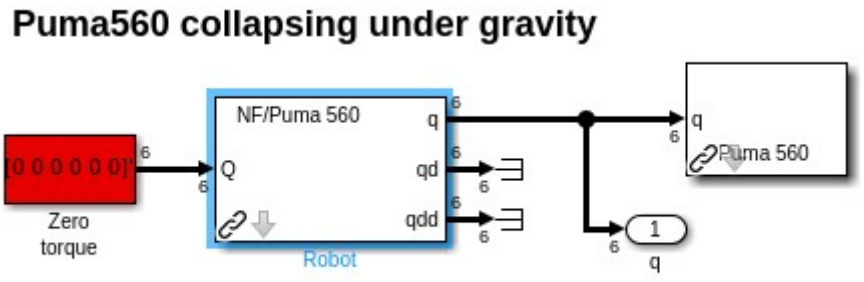


Sada cemo izracunati gravitacijiski momenta na dva zgloba za neutralnu konfiguraciju..

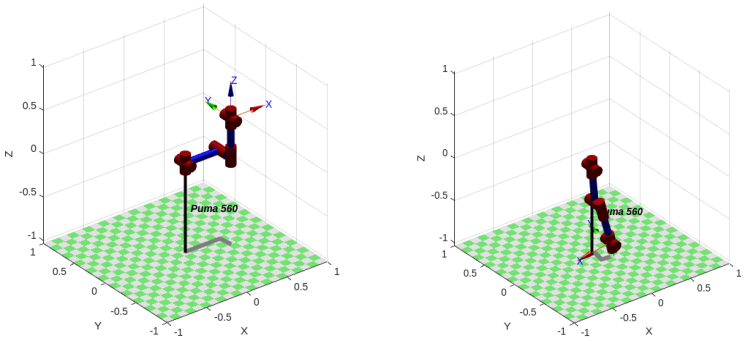
```
% Generisanje mreže vrijednosti za Q2 i Q3
[Q2, Q3] = meshgrid(-pi:0.1:pi, -pi:0.1:pi);

% Inicijalizacija matrica za čuvanje gravitacijskih momenata
g2_qz = zeros(size(Q2));
g3_qz = zeros(size(Q2));

% Petlja za izračunavanje gravitacijskih momenata za qz
konfiguraciju
for i = 1:numel(Q2)
    g_qz = p560.gravload([qz(1), Q2(i), Q3(i), qz(4), qz(5),
qz(6)]);
    g2_qz(i) = g_qz(2);
    g3_qz(i) = g_qz(3);
end

% Prikazivanje rezultata za qz
figure;
surfl(Q2, Q3, g2_qz);
title('Gravitacijski moment na zglobu 2 za konfiguraciju qz');
xlabel('Q2');
ylabel('Q3');
zlabel('g2');

figure;
surfl(Q2, Q3, g3_qz);
title('Gravitacijski moment na zglobu 3 za konfiguraciju qz');
xlabel('Q2');
ylabel('Q3');
zlabel('g3');
```



Pomocu funkcije sl_ztorque mozemo dobiti simulink model robota na koji djeluju samo gravitacione sile na njega.



Te cemo dobiti robota koji ce iz polozaja na prvoj slici zavrsiti u polozaju na drugoj slici



Zadatak 4

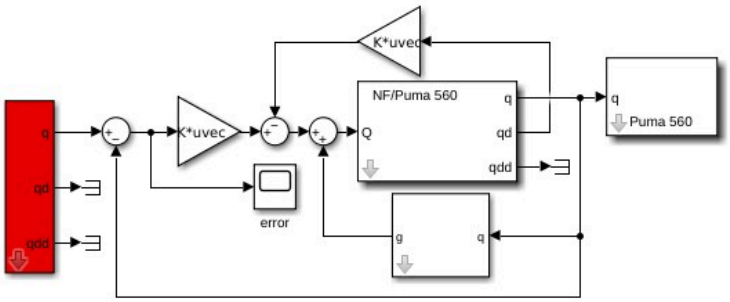
Implementirati PD regulaciju sa kompenzacijom gravitacije za Puma 560 manipulator u Simulinku.
Zglobovi treba da prate referentnu trajektoriju koja je interpolacija sljedećih tačaka:

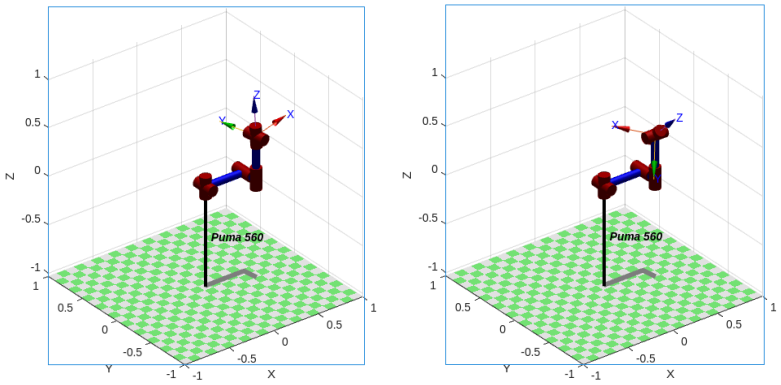
$$q(t = 0) = [0\ 0\ 0\ 0\ 0\ 0]^T$$
$$q(t = 3) = [0\ 0\ 0\ \pi/2\ \pi/4\ \pi/3]^T$$

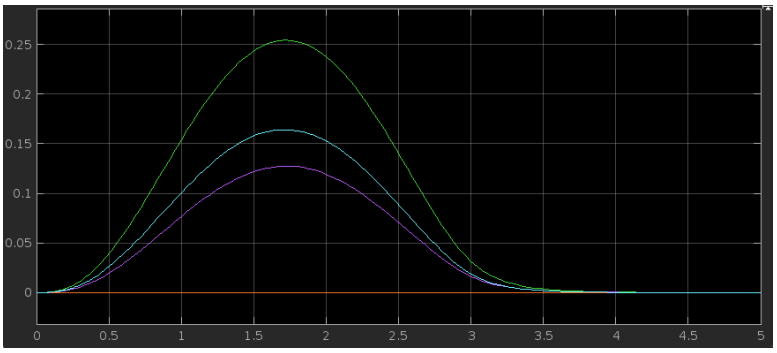
Potrebno je kreirati i animaciju kretanja manipulatora. Dati analizu regulatora te ispitati utjecaj tereta na odziv manipulatora i na poremećaj tipa početnih uslova.

Rješenje:

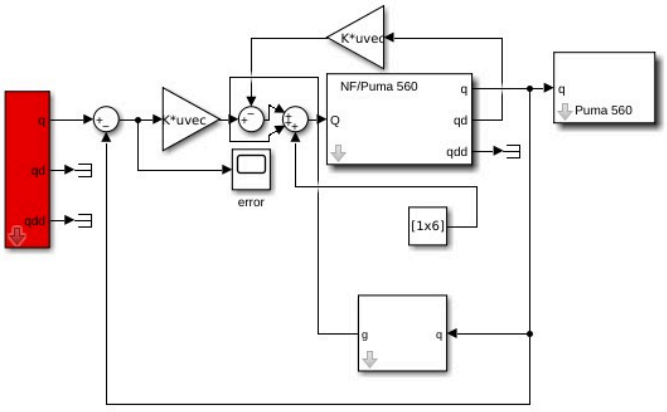
Pomoću modela u simulinku možemo vidjeti ponašanje robota

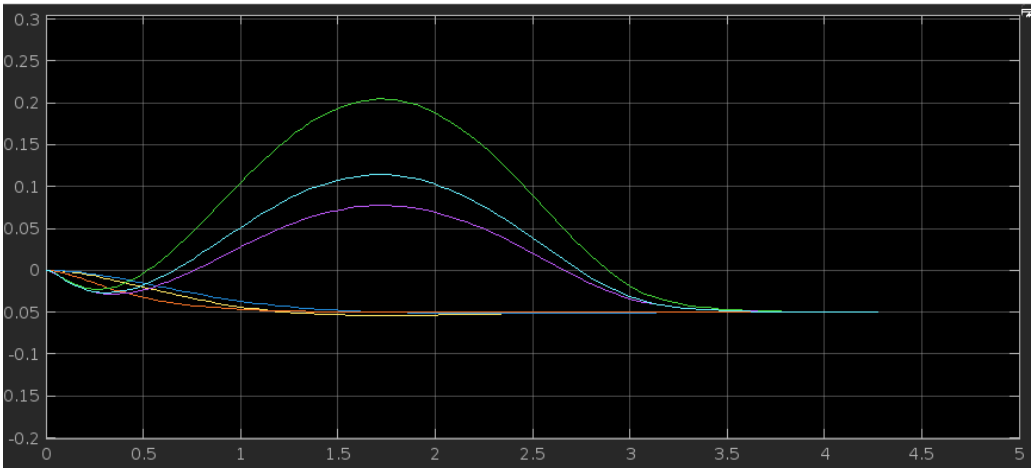






Ukoliko dodamo teret kao na slici, dobiti cemo sljedeci odziv.





Zadatak 5

Implementirati regulaciju inverzijom dinamike za Puma 560 manipulator u Simulinku. Zglobovi treba da prate referentnu trajektoriju koja je interpolacija sljedećih tačaka:

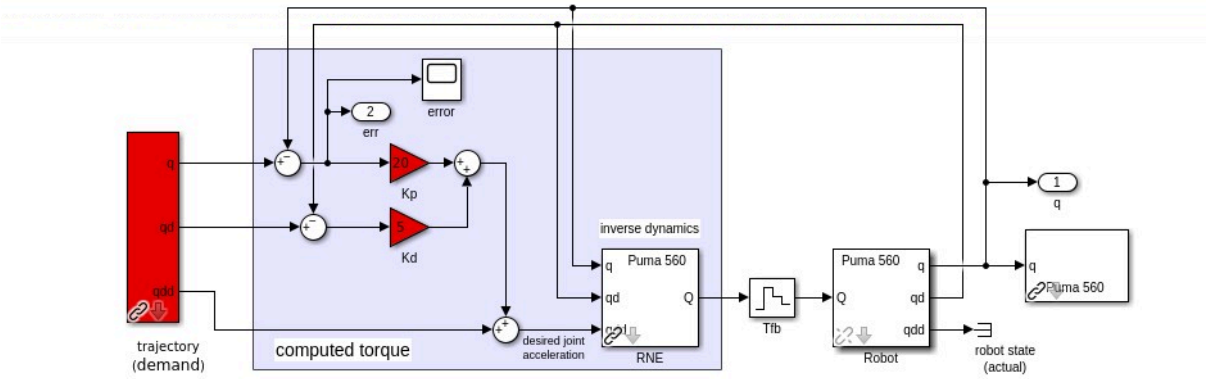
$q(t = 0) = [0\ 0\ 0\ 0\ 0\ 0]^T$

$q(t = 3) = [0\ 0\ 0\ \pi/2\ \pi/4\ \pi/3]^T$

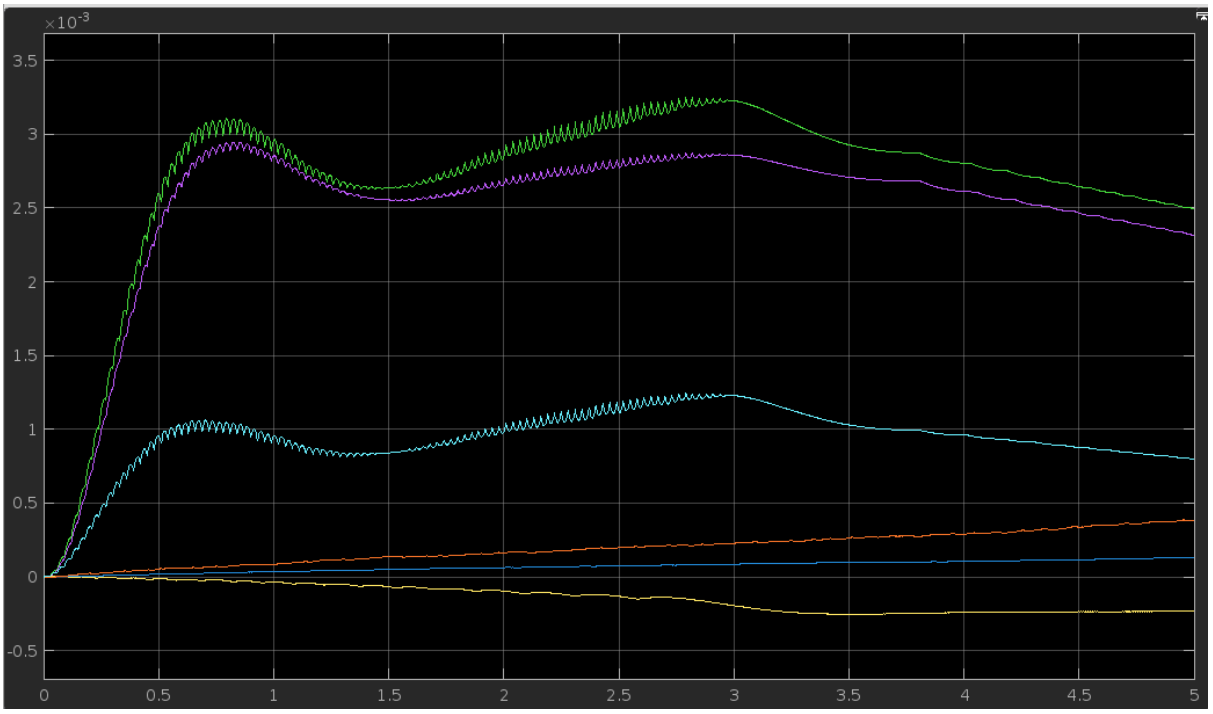
Potrebno je kreirati i animaciju kretanja manipulatora. Dati analizu regulatora te ispitati utjecaj tereta na odziv manipulatora i na poremećaj tipa početnih uslova.

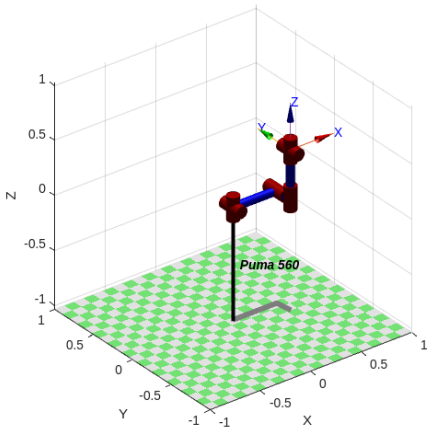
Rješenje:

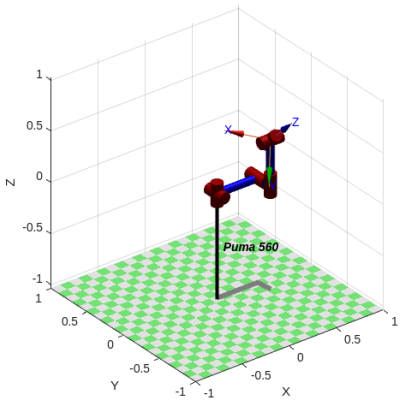
Pomocu funkcije sl_ctorque smo dobili ovu shemu. Namjestili smo pocetne uslove i dobili odziv na slici i animaciju kretanja robota.



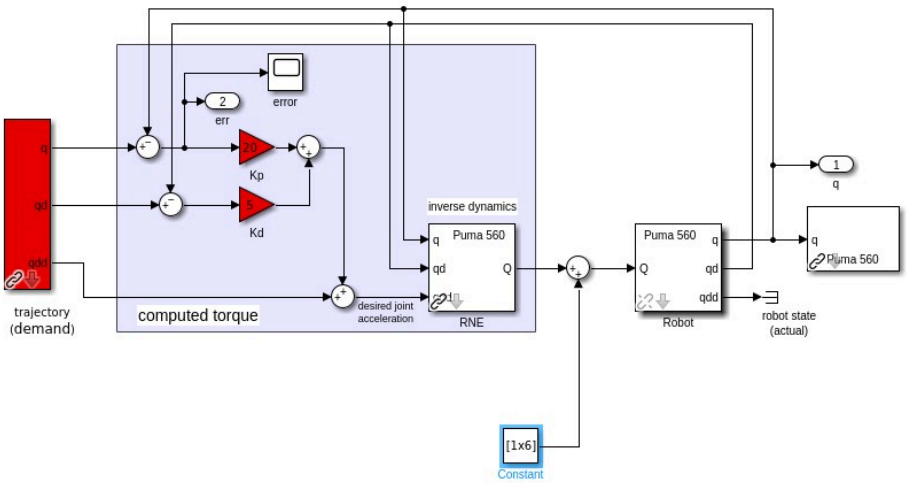
This is different compared to RVC1 (Fig 9.20) and RVC2 (Fig 9.21).
The q and qd inputs to the inverse dynamics block come directlu from the outputs of the robot model.

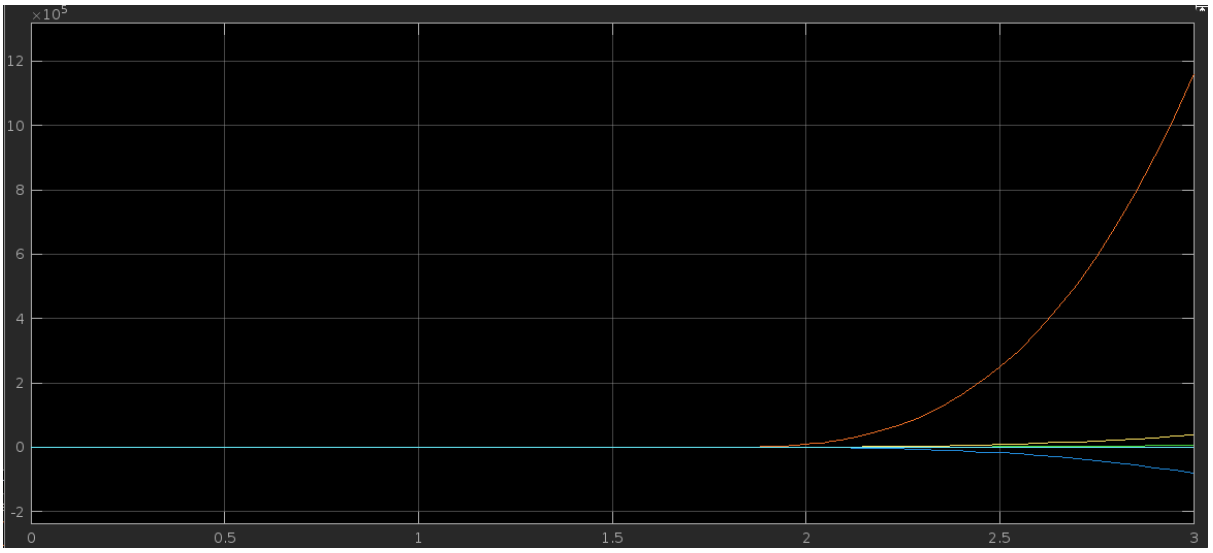


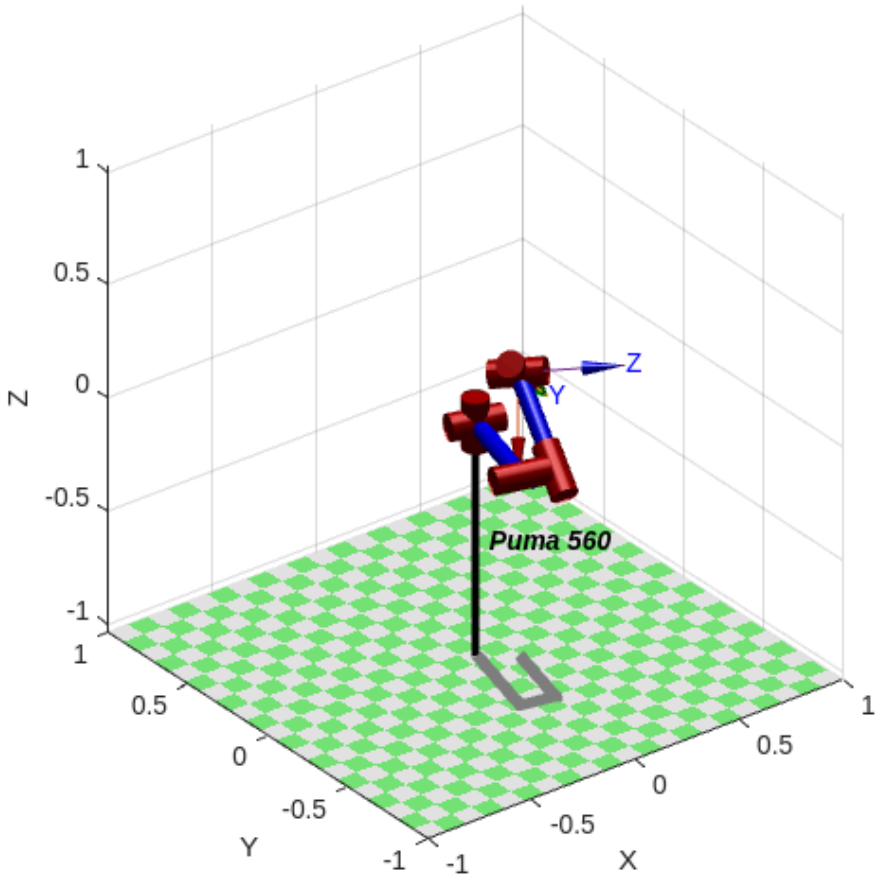




Ukoliko dodamo teret kao na slici dobiti cemo sljedeci odziv:




×10^5
12
10
8
6
4
2
0
-2
0
0.5
1
1.5
2
2.5
3


Z
Y
Puma 560
1
0.5
0
-0.5
-1
Z
Y
X