

Univerza v Ljubljani  
Fakulteta za matematiko in fiziko

# Geometrijsko zvezna gibanja togih teles

Matic Oskar Hajšen in Eva Zmazek

Ljubljana, 2019



Kazalo

Listings

# 1 Uvod

Z najino seminarsko bova prikazala, kako se da znanje, pridobljeno pri tem predmetu, uporabiti pri upodobitvi gibanja togih teles, ki se uporabljajo pri računalniških animacijah in v robotiki. Za opis teh gibanj bomo uporabljali kvaternione in bezierjeve krivulje na kvaternionih.

## 2 Teoretično ozadje

### 2.1 Homogene in kartezične koordinate

Imejmo vektor  $p$  v 3-dimenzionalnem prostoru s homogenimi koordinatami  $p = (p_0, p_1, p_2, p_3)^T \in \mathbb{R}^4 / \{(0, 0, 0, 0)^T\}$ . Če je prva komponenta  $p_0$  neničelna, lahko za točko  $p$  definiramo prirejene kartezične koordinate  $\underline{p} = (\underline{p}_1, \underline{p}_2, \underline{p}_3)^T \in \mathbb{R}^3$ , pri čemer velja  $\underline{p}_i = \frac{p_i}{p_0}$  za  $i = 1, 2, 3$ . Na tak način vektorja  $p$  in  $\lambda p$  opisujeta isto točko  $\underline{p}$  za poljubno neničelno realno število  $\lambda$ . Vektorjem z ničelno prvo komponento priredimo točke v neskončnosti.

### 2.2 Zveza med koordinatami točk v fiksnem koordinatnem sistemu in koordinatami točk v gibajočem se koordinatnem sistemu

Definirajmo dva koordinatna sistema v  $\mathbb{R}^3$ :

- fiksen koordinatni sistem  $E^3$  (običajen koordinatni sistem)
- gibajoč se koordinatni sistem  $\hat{E}^3$

Točke lahko predstavimo v enem ali drugem.

Označimo s  $\underline{p}$  točko glede na fiksen koordinatni sistem  $E^3$ , s  $\hat{\underline{p}}$  pa glede na  $\hat{E}^3$ . Potrebujemo koordinatno transformacijo

$$\hat{E}^3 \rightarrow E^3$$

$$\hat{\underline{p}} \mapsto \underline{p}$$

Z uporabo homogenih koordinat, lahko transformacijo zapišemo s pomočjo matrike

$$M = \left[ \begin{array}{c|ccc} m_{0,0} & 0 & 0 & 0 \\ \hline m_{1,0} & m_{1,1} & m_{1,2} & m_{1,3} \\ m_{2,0} & m_{2,1} & m_{2,2} & m_{2,3} \\ m_{3,0} & m_{3,1} & m_{3,2} & m_{3,3} \end{array} \right],$$

kjer velja  $m_{0,0} \neq 0$ . Preslikavo v homogenih koordinatah lahko torej zapišemo kot:

$$\hat{\underline{p}} \mapsto \underline{p} = M\hat{\underline{p}}$$

Vektorju  $c = M(1, 0, 0, 0)^T = (m_{0,0}, m_{1,0}, m_{2,0}, m_{3,0})^T$  zapisanemu v homogenih koordinatah pripada vektor  $\underline{c} = (\frac{m_{1,0}}{m_{0,0}}, \frac{m_{2,0}}{m_{0,0}}, \frac{m_{3,0}}{m_{0,0}})^T$ , zapisan v kartezičnih koordinatah.

Ta vektor opisuje položaj koordinatnega izhodišča gibajočega se koordinatnega sistema  $\hat{E}^3$  glede na koordinatni sistem  $E^3$ .

$3 \times 3$  matrika

$$\underline{R} = \frac{1}{m_{0,0}} \begin{bmatrix} m_{1,1} & m_{1,2} & m_{1,3} \\ m_{2,1} & m_{2,2} & m_{2,3} \\ m_{3,1} & m_{3,2} & m_{3,3} \end{bmatrix}$$

opisuje orientacijo gibajočega se koordinatnega sistema  $\hat{E}^3$ . Pravimo ji **rotacijska matrika**.

Oglejmo si, kaj naredi matrika  $M$  z vektorjem  $[1, b_M, c_M, d_M]$ :

$$M \cdot \begin{bmatrix} 1 \\ b_M \\ c_M \\ d_M \end{bmatrix} = \begin{bmatrix} m_{0,0} \\ m_{1,0} \\ m_{2,0} \\ m_{3,0} \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ m_{1,1}b_M & m_{1,2}c_M & m_{1,3}d_M \\ m_{2,1}b_M & m_{2,2}c_M & m_{2,3}d_M \\ m_{3,1}b_M & m_{3,2}c_M & m_{3,3}d_M \end{bmatrix}$$

Dobimo vektor v homogeni obliki, ki ima na prvi komponenti vrednost  $m_{0,0}$ , preostale tri komponente pa predstavlja vektor

$$\begin{bmatrix} m_{1,0} \\ m_{2,0} \\ m_{3,0} \end{bmatrix} + \begin{bmatrix} m_{1,1} & m_{1,2} & m_{1,3} \\ m_{2,1} & m_{2,2} & m_{2,3} \\ m_{3,1} & m_{3,2} & m_{3,3} \end{bmatrix} \cdot \begin{bmatrix} b_M \\ c_M \\ d_M \end{bmatrix}$$

Ker je to vektor v homogeni obliki in ker je prva komponenta neničelna ( $m_{0,0} \neq 0$ ), je njemu prirejen vektor v kartezični obliki enak

$$\frac{1}{m_{0,0}} \begin{bmatrix} m_{1,0} \\ m_{2,0} \\ m_{3,0} \end{bmatrix} + \frac{1}{m_{0,0}} \begin{bmatrix} m_{1,1} & m_{1,2} & m_{1,3} \\ m_{2,1} & m_{2,2} & m_{2,3} \\ m_{3,1} & m_{3,2} & m_{3,3} \end{bmatrix} \cdot \begin{bmatrix} b_M \\ c_M \\ d_M \end{bmatrix},$$

ki pa je enak vsoti  $\underline{c} + R \cdot \hat{\underline{p}}$

Transformacijo  $\hat{\underline{p}} \mapsto \underline{p}$  v kartezičnih koordinatah zapišemo kot:

$$\underline{p} = \underline{c} + R\hat{\underline{p}}$$

## 2.3 Gibanje točk v času

Kadar je  $\underline{c} = \underline{c}(t)$  in  $R = R(t)$ , govorimo o gibanju togega telesa:

$$\hat{E}^3 \times I \rightarrow E^3$$

$$(\hat{\underline{p}}, t) \mapsto \underline{c}(t) + R(t)\hat{\underline{p}} =: \underline{p}(t)$$

Krivulji  $\underline{p}(t)$  pravimo **trajektorija** točke  $\hat{\underline{p}}$

Če je  $\underline{c}(t) = (0, 0, 0)$ , potem trajektorija poljubne točke  $\hat{\underline{p}}$  leži na sferi z radijem  $||\hat{\underline{p}}||$  in središčem v koordinatnem izhodišču fiksne koordinatnega sistema  $E^3$ . Rotacijski del gibanja  $R(t)$  opisuje gibanje po enotski sferi, zato se imenuje tudi **sferični del gibanja togega telesa**. Problem je konstrukcija matrike  $R$ , ki mora biti ortogonalna. ( $RR^T = R^T R = I$ ,  $\det R = 1$ ).

## 2.4 Opis rotacij s kvaternioni

Pri opisovanju rotacij si lahko pomagamo s **kvaternioni**. Prostor kvaternionov  $\mathbb{H}$  je 4-dimenzionalni vektorski prostor s standardno bazo

$$\underline{1} = (1, (0, 0, 0)^T)$$

$$\underline{i} = (0, (1, 0, 0)^T)$$

$$\underline{j} = (0, (0, 1, 0)^T)$$

$$\underline{k} = (0, (0, 0, 1)^T)$$

Vsak kvaternion  $\mathcal{A}$  lahko zapišemo kot:

$$\mathcal{A} = (a_0, \underline{a}), \quad a_0 \in \mathbb{R} \text{ skalarni del, } \underline{a} = (a_1, a_2, a_3)^T \text{ vektorski del}$$

Na kvaternionih sta definirana seštevanje in množenje kot:

$$\mathcal{A} + \mathcal{B} = (a_0, \underline{a}) + (b_0, \underline{b}) = (a_0 + b_0, \underline{a} + \underline{b})$$

$$\mathcal{A} \cdot \mathcal{B} = (a_0 \cdot b_0 - \underline{a} \cdot \underline{b}, a_0 \underline{b} + b_0 \underline{a} + \underline{a} \times \underline{b})$$

Konjugirana vrednost kvaterniona  $\mathcal{A} = (a_0, \underline{a})$  je definirana kot  $\overline{\mathcal{A}} = (a_0, -\underline{a})$ .

**Definicija 2.1.** Preslikava  $\chi : \mathbb{H} \setminus \{0\} \rightarrow SO_3$  oblike

$$Q \mapsto \frac{1}{q_0^2 + q_1^2 + q_2^2 + q_3^2} \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1 q_2 - q_0 q_3) & 2(q_1 q_3 + q_0 q_2) \\ 2(q_1 q_2 + q_0 q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2 q_3 - q_0 q_1) \\ 2(q_1 q_3 - q_0 q_2) & 2(q_2 q_3 + q_0 q_2) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix}$$

$$Q = (q_0, (q_1, q_2, q_3)^T)$$

se imenuje **kinematična preslikava**.

Matika  $\chi(Q)$  je rotacijska matrika. Velja pa tudi obratno. Vsako rotacijsko matriko  $R$  lahko zapišemo v zgornji obliki, to je, lahko jo preslikamo v dva **antipodna kvaterniona** oblike

$$\pm Q = \pm(q_0, (q_1, q_2, q_3)^T),$$

$$q_0^2 + q_1^2 + q_2^2 + q_3^2 = 1$$

**Kinematična preslikava** poda korespondenco med 3D rotacijami in parom antipodnih točk na 4D enotski sferi  $S^3 \subseteq R^4$ .

Ker velja  $q_0^2 + q_1^2 + q_2^2 + q_3^2 = 1$ , so vrednosti  $|q_i|$ ;  $i = 0, 1, 2, 3$  na zaprtem intervalu med 0 in 1. Vrednost  $q_0$  in vektor  $(q_1, q_2, q_3)^T$  lahko zato zapišemo v obliki:

$$q_0 = \cos\left(\frac{\phi}{2}\right)$$

in

$$\begin{bmatrix} q_1 \\ q_2 \\ q_3 \end{bmatrix} = \sin\left(\frac{\phi}{2}\right) \cdot \vec{r}; \quad \vec{r} \text{ enotski vektor}$$

Če kvaternion  $Q$  zapišemo v tej obliki, ima rotacija, prirejena temu kvaternionu lepo geometrijsko interpretacijo. Predstavlja namreč rotacijo za kot  $\phi$  okrog osi  $\vec{r}$ .

Ker lahko vsako rotacijo zapišemo v tej obliki, lahko tako zapišemo tudi rotacijo iz poglavja ???. Če imamo podano preslikavo  $M$ , poiščimo, kako za to preslikavo definiramo kvaternion  $Q$ . Priemerjajmo matriki  $\mathbb{R}$  in poglavja ??? in  $\mathbb{R}$ , zapisanega s kvaternioni.

$$\begin{aligned} m_{0,0} + m_{1,1} + m_{1,2} + m_{3,3} &= 4q_0^2 \\ m_{3,2} - m_{2,3} &= 2 \cdot (q_2q_3 - q_0q_1) - 2 \cdot (q_2q_3 + q_0q_1) = 4q_0q_1 \\ m_{1,3} - m_{3,1} &= 2 \cdot (q_1q_3 + q_0q_2) - 2 \cdot (q_1q_3 + q_0q_2) = 4q_0q_2 \\ m_{2,1} - m_{1,2} &= 2 \cdot (q_1q_2 + q_0q_4) - 2 \cdot (q_1q_2 + q_0q_4) = 4q_0q_3 \end{aligned}$$

$$\begin{aligned} m_{3,2} - m_{2,3} &= 4q_0q_1 \\ m_{0,0} + m_{1,1} - m_{1,2} - m_{3,3} &= 4q_1^2 \\ m_{2,1} + m_{1,2} &= 4q_1q_2 \\ m_{1,3} + m_{3,1} &= 4q_1q_3 \end{aligned}$$

$$\begin{aligned} m_{1,3} - m_{3,1} &= 4q_0q_2 \\ m_{2,1} + m_{1,2} &= 4q_1q_2 \\ m_{0,0} - m_{1,1} + m_{1,2} - m_{3,3} &= 4q_2^2 \\ m_{3,2} + m_{2,3} &= 4q_2q_3 \end{aligned}$$

$$\begin{aligned} m_{2,1} - m_{1,2} &= 4q_0q_3 \\ m_{1,3} + m_{3,1} &= 4q_1q_3 \\ m_{3,2} + m_{2,3} &= 4q_2q_3 \\ m_{0,0} - m_{1,1} - m_{1,2} + m_{3,3} &= 4q_0^2 \end{aligned}$$

Opazimo, da v vsakem sklopu razmerja med vrednostmi enaka

$$q_0 : q_1 : q_2 : q_3$$

Ker  $q_0, q_1, q_2, q_3$  niso hkrati enaki 0, bo vsaj eno izmed zgornjih razmerij različno od  $0 : 0 : 0 : 0$ . Tisto razmerje nato uporabimo kot razmerje  $q_0 : q_1 : q_2 : q_3$ . Skupaj z enakostjo  $q_0^2 + q_1^2 + q_2^2 + q_3^2 = 1$  nato izračunamo kvaternion  $Q = (q_0, q_1, q_2, q_3)^T$  (bolj natančno sta v množici rešitev dva antipodna kvaterniona).

## 2.5 Bezierjeve krivulje

Z uporabo kinematične preslikave lahko za konstrukcijo sferičnih gibanj uporabimo Bezierjeve krivulje. Izberemo kontrolne kvaternione  $Q_0, Q_1, \dots, Q_n$ .

$$Q(t) = \sum_{i=0}^n Q_i B_i^n(t)$$

Bezierjeva krivulja  $Q(t)$  v času  $t$  opiše kvaternion, ki mu priredimo rotacijo  $R(t)$ :

$$\chi(Q(t)) = R(t)$$

Rotacija, ki je določena z Bezierjevo krivuljo  $Q(t) = \sum_{i=0}^n Q_i B_i^n(t)$  stopnje  $n$ , je sferično razionalno gibanje stopnje  $2n$ .

Gibanje koordinatnega izhodišča zapišemo v obliki

$$\underline{c}(t) = \frac{w(t)}{\|Q(t)\|^2}; \quad w(t) := (w_1(t), w_2(t), w_3(t)).$$



## 3 Implementacija

### 3.1 Kvaternioni

Ker si pri opisovanju rotacij pomagamo s kvaternioni, sva na kvaternionih definirala naslednje funkcije:

Listing 1: array2quat

```
1 function Q = array2quat(a, b, c, d)
2 %ARRAY2QUAT prejme 4 parametre, ki jih pretvori v kvaternion (
   vektorsko obliko)
3 %input:
4 %a, b, c, d      komponente
5 %output:
6 %Q               kvaternion
7
8 Q = [a, b, c, d];
9 end
```

Listing 2: quat\_vec

```
1 function v = quat_vec(Q)
2 %QUAT_VEC poda vektorski del kvaterniona Q
3 %input:
4 %Q               kvaternion (q0, q1, q2, q3)
5 %output:
6 %v               vektorski komponenta Q-ja, (q1, q2, q3)
7
8 v = [Q(2) Q(3) Q(4)];
9 end
```

Listing 3: quatmultiply

```
1 function c = quatmultiply(a,b)
2 %QUATMULTIPLY izracuna produkt dveh kvaternionov, kot je opisano v
   clanku
3 %
4 % Input:
5 % a ... prvi kvaternion
6 % b ... drugi kvaternion
7 %
8 % Output:
9 % c ... produkt (nov kvaternion)
10
11 c = zeros(1,4);
12 a_s = a(1);
13 b_s = b(1);
14
```

```

15 a_v = quat_vec(a);
16 b_v = quat_vec(b);
17
18 c(1) = a_s * b_s - a_v * b_v';
19 c_v = a_s * b_v + b_s * a_v + cross(a_v,b_v);
20
21 c(2) = c_v(1);
22 c(3) = c_v(2);
23 c(4) = c_v(3);
24 end

```

Listing 4: conj\_quat

```

1 function Q = conj_quat(q)
2 %CONJ_QUAT vrne konjugirano vrednost podanega kvaterniona q
3 %input:
4 %q          kvaternion
5 %output:
6 %Q          konjugiran qvaternion
7
8 Q = [q(1), -q(2:4)];
9 end

```

Listing 5: quat\_exp

```

1 function e = quat_exp(q, t)
2 %QUAT_EXP vrne potenco z osnovo q (kvaternion) in eksponentom t
3 %input:
4 % q          kvaternion [a, b, c, d]
5 % t          eksponent
6 %output:
7 % e          rezultat q^t
8
9 if t==1
10     e = conj_quat(q)/norm(q);
11 else
12     a = q(1);
13     v = quat_vec(q);
14     theta = acos(a/norm(q));
15     n = v/norm(v);
16
17     e = norm(q)^t*[cos(t*theta), n*sin(t*theta)];
18 end

```

Listing 6: quat\_rot\_mat

```

1 function H = quat_rot_mat(Q)
2 %QUAT_ROT_MAT oblikuje rotacijsko matriko, prirejeno kvaternionu Q

```

```

3 %input:
4 %Q          kvaternion
5 %
6 %output:
7 %H          rotacijska matrika za sfericno gibanje
8
9 H = zeros(3,3);
10 h = sum(Q.^2);
11
12 if h == 0
13     H = eye(3,3);
14 else
15     H(1,1) = Q(1)^2+Q(2)^2 - Q(3)^2 - Q(4)^2;
16     H(1,2) = 2*(Q(2)*Q(3) - Q(1)*Q(4));
17     H(1,3) = 2*(Q(2)*Q(4) + Q(1)*Q(3));
18
19     H(2,1) = 2*(Q(2)*Q(3) + Q(1)*Q(4));
20     H(2,2) = Q(1)^2 - Q(2)^2 + Q(3)^2 - Q(4)^2;
21     H(2,3) = 2*(Q(3)*Q(4) - Q(1)*Q(2));
22
23     H(3,1) = 2*(Q(2)*Q(4) - Q(1)*Q(3));
24     H(3,2) = 2*(Q(3)*Q(4) + Q(1)*Q(2));
25     H(3,3) = Q(1)^2 - Q(2)^2 - Q(3)^2 + Q(4)^2;
26
27     H = 1/h.*H;
28 end
29
30 end

```

## 3.2 Kocka

Listing 7: kocka

```

1 function oglisca = kocka(T0, T1)
2 %Vrne kocko, definirano z diagonalo (T0,T1)
3 a = [T1(1) - T0(1) 0 0];
4 b = [0 T1(2) - T0(2) 0];
5 c = [0 0 T1(3) - T0(3)];
6
7 oglisca = [T0; T0+b; T0+a+b; T0+a; T0+c; T0+c+b; T0+a+b+c; T0+a+c];
8 %ploskve = [1 2 3 4; 2 6 7 3; 4 3 7 8; 1 5 8 4; 1 2 6 5; 5 6 7 8];
9 end

```

Listing 8: kocka\_vek

```

1 function oglisca = kocka_vek(X, Y, Z, T0)
2 %input:

```

```

3 % x,y,z      vektorji stranic x,y,z stranic
4 % T0        izhodišce
5 %output:
6 % oglišca    vsa oglišca kocke, ki se začne v T0
7
8 oglišca = [T0; T0+Y; T0+X+Y; T0+X; T0+Z; T0+Z+Y; T0+X+Y+Z; T0+X+Z];
9 %ploskve = [1 2 3 4; 2 6 7 3; 4 3 7 8; 1 5 8 4; 1 2 6 5; 5 6 7 8];
10 end

```

Listing 9: kot\_v\_kvrat

```

1 function r = kot_v_kvrat(fi, e)
2 %input:
3 % fi        kot
4 % e         vektor osi
5 %output:
6 % Q         kvaternion
7 %
8 % sprejme kot fi in os e ter vrne kvaternion, ki predstavlja rotacijo
9 % za kot fi okoli osi e
10
11 e = e/norm(e);
12 r = [cos(fi/2) sin(fi/2)*e(1) sin(fi/2)*e(2) sin(fi/2)*e(3)];
13 end

```

### 3.3 Razvrsti

```

1 function b = bezier (B,t)
2 % Opis :
3 % bezier vrne točke na Bezierjevi krivulji pri danih
4 % parametrih
5 %
6 % Definicija :
7 % b = bezier (B,t)
8 %
9 % Vhodna podatka :
10 % B      matrika velikosti n+1 x d, ki predstavlja kontrolne
11 %        točke Bezierjeve krivulje stopnje n v
12 %        d– dimenzionalnem prostoru ,
13 % t      seznam parametrov dolžine k, pri katerih računamo
14 %        vrednost Bezierjeve krivulje
15 %
16 % Izhodni podatek :
17 % b      matrika velikosti k x d, kjer i–ta vrstica
18 %        predstavlja točko na Bezierjevi krivulji pri

```

```

19 %      parametru iz t na i- tem mestu
20
21 [n,d] = size(B);
22 k = length(t);
23 b = zeros(k,d);
24
25 for i=1:k
26     for j=1:d
27         D = decasteljau(B(:,j)',t(i));
28         b(i,j) = D(1,n);
29     end
30 end

```

```

1 function D = decasteljau (b,t)
2 % Opis :
3 % decasteljau vrne shemo de Casteljaujevega postopka za dan
4 % seznam koordinat b pri danem parametru t
5 %
6 % Definicija :
7 % D = decasteljau (b,t)
8 %
9 % Vhodna podatka :
10 % b      seznam koordinat kontrolnih tock Bezierjeve krivulje
11 %        stopnje n,
12 % t      parameter , pri katerem racunamo koordinato
13 %        Bezierjeve krivulje
14 %
15 % Izhodni podatek :
16 % D      tabela velikosti n+1 x n+1, ki predstavlja de
17 %        Casteljaujevo shemo za koordinate b pri parametru t
18 %        ( element na mestu (1,n +1) je koordinata Bezierjeve
19 %        krivulje pri parametru t, elementi na mestih (i,j)
20 %        za i > n-j+2 so NaN )
21
22 n = length(b);
23 D = [b', NaN(n,n-1)];
24
25 for r=1:n
26     for i=0:n-r-1
27         D(i+1,r+1) = (1-t)*D(i+1,r) + t*D(i+2,r);
28     end
29 end

```

```

1  % definicija kocke
2
3  % STRANICE KOCKE
4  x0 = [1 0 0];
5  y0 = [0 1.5 0];
6  z0 = [0 0 2];
7  T0 = [0 0 0];
8  K = kocka_vek(x0,y0,z0, T0);
9
10
11 % KVATERNIONI ZA OBRACANJE KOCKE
12
13 % Q0 = angle2quat(0, 0, 0);
14 % Q1 = angle2quat(pi/2, 0, 0);
15 % Q2 = angle2quat(pi/2, pi/4, 0);
16 % Q3 = angle2quat(pi/2, pi/4, pi/3);
17
18 % KVATERNIONI ZA OBRACANJE KOCKE
19
20 Q0 = kot_v_kvrat(0, [1,0,0]);
21 Q1 = kot_v_kvrat(-pi/2, [1,0,0]);
22 Q2 = kot_v_kvrat(-pi/2, [0,0,1]);
23 Q3 = kot_v_kvrat(3*pi/4, [1,0,-1]);
24
25 Q = [Q0; Q1; Q2; Q3];
26
27
28 % PRIMER: pri temle naj bi sel po dolgi poti, vendar se mi zdi
29 % da je brez popravka boljse (glej funkcijo slerp)
30 % Q0 = kot_v_kvrat(0, [1,0,0]);
31 % Q1 = kot_v_kvrat(pi/4, [1,0,0]);
32 % Q2 = kot_v_kvrat(-pi/2, [0,0,1]);
33 % Q3 = kot_v_kvrat(3*pi/4, [1,0,-1]);
34
35
36
37 % BEZIERJEVA KRIVULJA ZA TRANSLACIJO
38 b0 = [-9 -9 -9];
39 b1 = [6 2 5];
40 b2 = [-8 6 9];
41 b3 = [5 5 -5];
42 B = [b0; b1; b2; b3];
43

```

```

44 n = 15;
45 t = linspace(0,1,n);
46 os = 10;
47
48 [mat_Q, w, c] = izracunaj_vse(Q,B,t);
49 narisi_vse(x0,y0,z0,T0, mat_Q,c,t, os, 1, 1, 1)
50 %plot3(B(:,1),B(:,2),B(:,3))
51 %scatter3(B(:,1),B(:,2),B(:,3))
52
53 % ZLEPKI?
54 % to sta ze dva premika, ki se nadaljujeta
55 % samo C^0 zveznost
56
57 QQ = [Q3; kot_v_kvrat(pi/4, [-1,0,1]); kot_v_kvrat(pi/2, [1,0,0]);
        kot_v_kvrat(pi/3, [1,0,1])];
58 BB = [b3; 2 3 4; 5 6 7; -5 -5 5];
59 [mat_QQ, ww, cc] = izracunaj_vse(QQ,BB,t);
60 narisi_vse(x0,y0,z0,T0, mat_QQ,cc,t, os, 1, 1, 1)
61 %plot3(BB(:,1),BB(:,2),BB(:,3))
62 %scatter3(BB(:,1),BB(:,2),BB(:,3))
63
64 QQQ = [QQ(end,:); kot_v_kvrat(-pi/3, [1,0,1]); kot_v_kvrat(pi/2,
        [0,0,1]); kot_v_kvrat(-pi/2, [1,0,0])];
65 BBB = [BB(end,:); 0 9 0; 0 0 -9; 0 0 0];
66 [mat_QQQ, www, ccc] = izracunaj_vse(QQQ,BBB,t);
67 narisi_vse(x0,y0,z0,T0, mat_QQQ,ccc,t, os, 1, 1, 1)
68 %plot3(BBB(:,1),BBB(:,2),BBB(:,3))
69 %scatter3(BBB(:,1),BBB(:,2),BBB(:,3))
70
71 % ce zdruzimo kontrolne poligone, dobimo precej drugacne premik
72 % N = 5*n;
73 % skupaj_Q = [Q; QQ(2:end-1,:); QQQ];
74 % skupaj_B = [B; BB(2:end-1,:); BBB];
75 % [mat_skuQ, skuw, skuc] = izracunaj_vse(skupaj_Q,skupaj_B,linspace
        (0,1,N));
76 % narisi_vse(x0,y0,z0,T0, mat_skuQ,skuc,linspace(0,1,N), os, 1, 1, 1)

```

Ker sva rotacijo želela izraziti s kotom  $\phi$  in osjo  $\vec{r}$ , sva definirala funkcijo, ki kot  $\phi$  in enotski vektor  $e$ , ki leži na osi  $r$ , pretvori v ustrezen kvaternion  $Q$ .

Listing 10: For educational purposes

```

1 function r = kot_v_kvrat(fi, e)
2 %input:
3 % fi          kot
4 % e           vektor osi
5 %output:
6 % Q           kvaternion

```

```

7 %
8 % sprejme kot fi in os e ter vrne kvaternion, ki predstavlja rotacijo
9 % za kot fi okoli osi e
10
11 e = e/norm(e);
12 r = [cos(fi/2) sin(fi/2)*e(1) sin(fi/2)*e(2) sin(fi/2)*e(3)];
13 end

```

Listing 11: For educational purposes

```

1 % example of while loop using placeholders
2 while <condition>
3     if <something-bad-happens>
4         break
5     else
6         % do something useful
7     end
8     % do more things
9 end

```