Univerza v Ljubljani
Fakulteta za matematiko in fiziko

Gibanja togih teles

Matic Oskar Hajšen in Eva Zmazek

Ljubljana, 2019

Kazalo

Listings

1 Uvod

S seminarsko nalogo 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

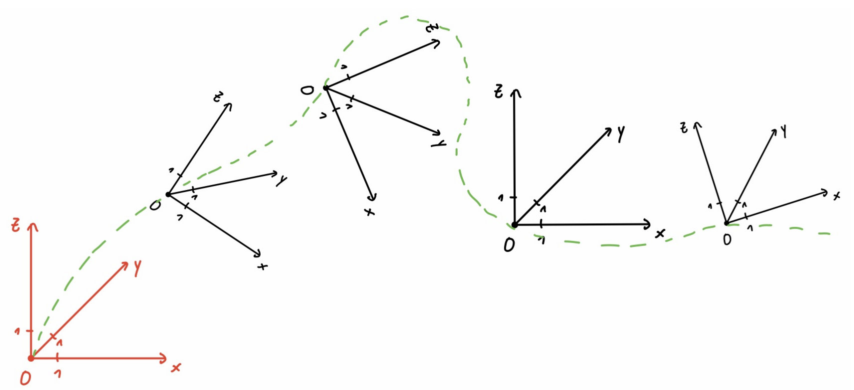
Naj bo p vektor v 3-dimenzionalnem prostoru s homogenimi koordinatami $p = (p_0, p_1, p_2, p_3)^T \in \mathbb{R}^4/\{(0,0,0,0)^T\}$. Če je prva komponenta p_0 neničelna, lahko za točko p definiramo prirejene kartezične koordinate $\underline{p} = (\underline{p_1}, \underline{p_2}, \underline{p_3})^T \in \mathbb{R}^3$, pri čemer velja $\underline{p_i} = \frac{p_i}{p_0}$ za $i = 1, 2, 3$. Na tak način vektorja p in λp opisujeta isto točko $\underline{p}$ za poljubno neničelno realno število λ. 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 $\underline{\hat{p}}$ pa glede na $\hat{E}^3$. Potrebujemo koordinatno transformacijo

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

$$\hat{p} \mapsto p.$$

Z uporabo homogenih koordinat, lahko transformacijo zapišemo s pomočjo matrike

$$M = \begin{bmatrix} m_{0,0} & m_{0,1} & m_{0,2} & m_{0,3} \\ m_{1,0} & m_{1,1} & m_{1,2} & m_{1,3} \\ m_{2,0} & m_{2,1} & m_{2,2} & m_{2,3} \\ m_{3,0} & m_{3,1} & m_{3,2} & m_{3,3} \end{bmatrix},$$

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

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

Vektor $\hat{p}$, ki ima neničelno prvo koordinato, lahko zapišemo kot $(1, b_M, c_M, d_M)^T$ (saj $\lambda\hat{p}$ in $\hat{p}$ predstavljata isti vektor v kartezičnih koordinatah). Vektor p je tako oblike

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

Ker je to vektor v homogenih koordinatah, lahko ponovno uporabimo lahtnost, da λp in p predstavljata isti vektor v kartezičnih koordinatah. Določimo, da ima vektor p prvo komponento enako $m_{0,0}$. Z matriko M to dosežemo tako, da $m_{0,1}$, $m_{0,2}$ in $m_{0,3}$ dodelimo vrednost 0.

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

Oglejmo si, kaj naredi matrika M z vektorjem $(1, b_M, c_M, d_M)^T$:

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

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

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

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

Vektor $(1,0,0,0)^T$, zapisan s homogenimi koordinatami, predstavlja koordinatno izhodišče. Vektorju $c = M(1,0,0,0)^T = (m_{0,0}, m_{1,0}, m_{2,3}, m_{3,0})^T$ 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. 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**. Ker stolpci matrike R predstavljajo transformacije koordinatnih osi $\vec{i}$, $\vec{j}$ in $\vec{k}$, za matriko R velja $RR^T = R^TR = I$ in $\det R = 1$. Hkrati velja, da je matrika R ortogonalna.Opazimo torej, da lahko transformacijo točke zapišemo kot vsoto transformacije koordinatnega izhodišča gibajočega se koordinatnega izhodišča ter rotacije okrog koordinatnega izhodišča koordinatnega sistema $\hat{E}^3$.

2.3 Gibanje točk v času

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

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

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

Krivulji $\underline{p}(t)$ pravimo **trajektorija** točke $\underline{\hat{p}}$. Če je $\underline{c}(t) = (0,0,0)$, potem trajektorija poljubne točke $\underline{\hat{p}}$ leži na sferi z radijem $||\underline{\hat{p}}||$ in središčem v koordinatnem izhodišču fiksnega 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 ortogonalne matrike R.

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: $\mathcal{A} = (a_0, \underline{a})$, kjer rečemo, da sta $a_0 \in \mathbb{R}$ skalarni del in $\underline{a} = (a_1, a_2, a_3)^T$ 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 kvaretniona $\mathcal{A} = (a_0, \underline{a})$ je definirana kot $\overline{\mathcal{A}} = (a_0, -\underline{a})$. S pomočjo konjugirane vrednosti nato definiramo tudi normo kvaterniona kot

$$||\mathcal{A}|| = \sqrt{\mathcal{A} \cdot \overline{\mathcal{A}}} = \sqrt{a_0^2 + a_1^2 + a_2^2 + a_3^2}.$$

Opomba 2.1 (Implementirane metode)**.** V implementaciji vektorski del kvaterniona a pridobimo s funkcijo $quat\_vec(a)$, sklararni del pa kar z ukazom $a(1)$. Kvaterniona a in b seštejemo z ukazom $a + b$, njun produkt pa kličemo s funkcjio $quatmultiply(a, b)$. Konjugirano vrednost kvaterniona a pridobimo s klicom funkcije $conj\_quat(a)$. Za kvaternion a normo izračunamo z ukazom $sum(a.^2)$.

Definicija 2.2. Preslikava $\chi : \mathbb{H}\backslash\{0\} \to SO_3$ oblike

$$Q \mapsto \frac{1}{q_0^2 + q_1^2 + q_2^2 + q_3^2} \begin{bmatrix} q_0^2 + q_1^2 - q_2^2 - q_3^2 & 2(q_1q_2 - q_0q_3) & 2(q_1q_3 + q_0q_2) \\ 2(q_1q_2 + q_0q_3) & q_0^2 - q_1^2 + q_2^2 - q_3^2 & 2(q_2q_3 - q_0q_1) \\ 2(q_1q_3 - q_0q_2) & 2(q_2q_3 + q_0q_2) & q_0^2 - q_1^2 - q_2^2 + q_3^2 \end{bmatrix}$$

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

se imenuje **kinematična preslikava**.

Matrika $\chi(Q)$ 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 interalu med 0 in 1. Vrednost q_0 in vektor $(q_1, q_2, q_3)^T$ lahko zato zapišemo v obliki:

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

in

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

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

Opomba 2.3 (Implementirane metode)**.** V implementaciji rotacijsko matriko, prirejeno kvaternionu a generiramo s funkcijo $quat\_rot\_mat(a)$. Če imamo podan kot ϕ in enotski vektor r, nam funkcija $kot\_v\_kvat(\phi, r)$ poda pripadajoč kvaternion.

Ker lahko vsako rotacijo zapišemo v tej obliki, lahko tako zapišemo tudi rotacijo iz poglavja 2.2. Če imamo podano preslikavo M, poiščimo, kako za to preslikavo definiramo kvaternion Q. Priemerjajmo matriki $\mathbb{R}$ in poglavja 2.2 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 $Q_0, Q_1, \ldots, Q_n$.

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

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 racionalno gibanje stopnje $2n$.

Gibanje koordinatnega izhodišča zapišemo v obliki

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

Opomba 2.4. Za računanje Bezierjeve krivulje rotacije uporabimo funkciji *sbezier* in *sdecasteljau*, za gibanje koordinatnega izhodišča pa *bezier* in *decasteljau*.

Bezierjeve krivulje za rotacije računamo na sferi, zato pride do manjših zapletov z že znanimi postopki. Pri De Casteljaujevem algoritmu se običajno naredi linearno interpolacijo $b = (1-t)b_0 + tb_1$ za parameter t med točkama b_0 in b_1, ker pa želimo gibanje po sferi in ne po premici med točkama, moramo uporabiti sferično linearno interpolacijo.

Definicija 2.5. Sferična linearna interpolacija, znana tudi kot **slerp**, je preslikava

$$\mathrm{slerp}(p_0, p_1, t) = \frac{\sin((1-t)\varphi)}{\sin\varphi} p_0 + \frac{\sin(t\varphi)}{\sin\varphi} p_1,$$

kjer je φ kot med p_0 in p_1, da velja $\cos\varphi = p_0 \cdot p_1$. Slerp torej vrne točko na krožnem loku med točkama p_0 in p_1, ki ustreza parameteru t.

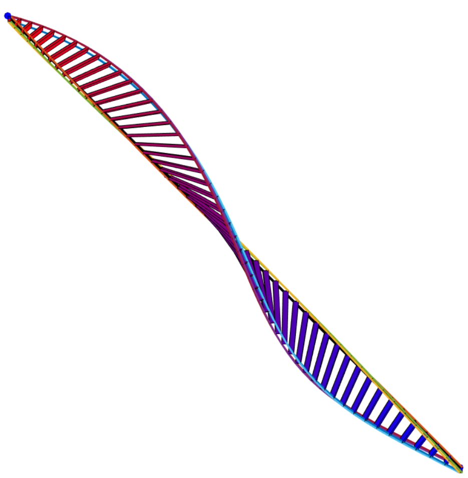
Sferično linearno interpolacijo lahko krajše zapišemo v kontekstu kvaternionov q_0 in q_1:

$$\begin{aligned}
\mathrm{slerp}(q_0, q_1, t) &= q_0(q_0^- 1 q_1)^t \\
&= q_1(q_1^- 1 q_0)^{1-t} \\
&= (q_0 q_1^- 1)^{1-t} q_1 \\
&= (q_1 q_0^- 1)^t q_0.
\end{aligned}$$

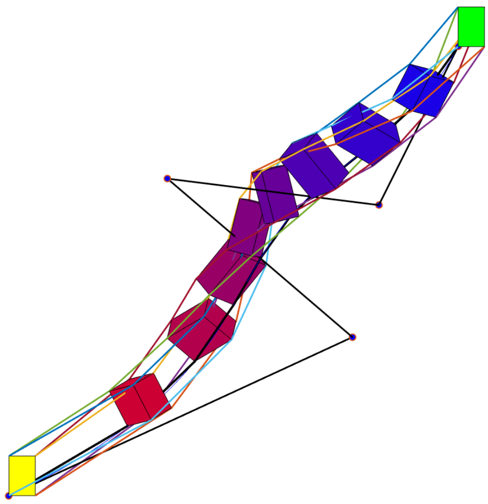
Pri računanju rotacije je bila uporabjena prva verzija formule.

3 Implementacija

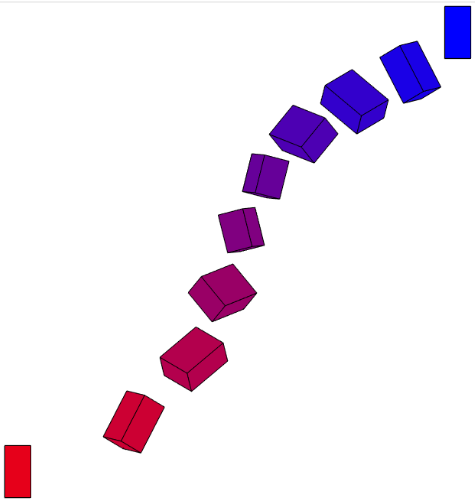
3.1 Primeri



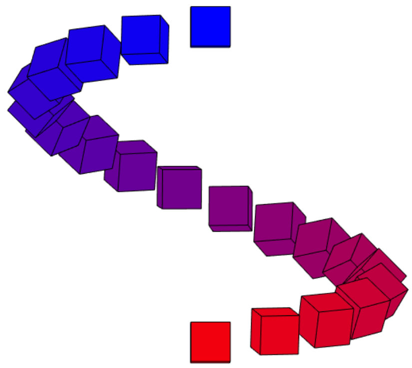
Gibanje palice (v obliki kvadra) s preprosto rotacijo



Gibanje kvadra po prostoru z izrisanimi tirnicami ogljišč



Gibanje kocke po spirali (prva perspektiva)



Gibanje kocke po spirali (druga perspektiva)

3.2 Kvaternioni

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

Vektorski del kvaterniona:

Input:

- kvaternion Q oblike $[q_0, q_1, q_2, q_3]$

Output:

- vektorski del v (q_1, q_2, q_3) kvaterniona Q

Listing 1: quat_vec

```
function v = quat_vec(Q)

v = [Q(2) Q(3) Q(4)];
end
```

Množenje kvaternionov:

Input:

- kvaternion a oblike $[a_1, a_2, a_3, a_4]$
- kvaternion b oblike $[b_1, b_2, b_3, b_4]$

Output:

- kvaternion $c = a \cdot b$

Listing 2: quatmultiply

```
function c = quatmultiply(a,b)

c = zeros(1,4);
a_s = a(1);
b_s = b(1);
a_v = quat_vec(a);
b_v = quat_vec(b);
c(1) = a_s * b_s - a_v * b_v';
c_v = a_s * b_v + b_s * a_v + cross(a_v,b_v);
c(2) = c_v(1);
c(3) = c_v(2);
c(4) = c_v(3);
end
```

Konjugirana vrednost kvaterniona:
Input:

- kvaternion q oblike $[q_1, q_2, q_3, q_4]$

Output:

- kvaternion $Q = \overline{q}$

Listing 3: conj_quat

```
function Q = conj_quat(q)

Q = [q(1), -q(2:4)];
end
```

Potenca kvaterniona q:
Input:

- kvaternion q oblike $[q_1, q_2, q_3, q_4]$
- eksponent t

Output:

- q^t

Listing 4: quat_exp

```
function e = quat_exp(q, t)

if t==-1
    e = conj_quat(q)/norm(q);
else
    a = q(1);
    v = quat_vec(q);
    theta = acos(a/norm(q));
    n = v/norm(v);
    e = norm(q)^t*[cos(t*theta), n*sin(t*theta)];
end
```

Rotacijska matrika:
Input:

- kvaternion q oblike $[q_1, q_2, q_3, q_4]$

Output:

- rotacijska matrika H za sfericno gibanje

Listing 5: quat_rot_mat

```
function H = quat_rot_mat(Q)

H = zeros(3,3);
h = sum(Q.^2);

if h == 0
    H = eye(3,3);
else
    H(1,1) = Q(1)^2+Q(2)^2 - Q(3)^2 - Q(4)^2;
    H(1,2) = 2*(Q(2)*Q(3) - Q(1)*Q(4));
    H(1,3) = 2*(Q(2)*Q(4) + Q(1)*Q(3));

    H(2,1) = 2*(Q(2)*Q(3) + Q(1)*Q(4));
    H(2,2) = Q(1)^2 - Q(2)^2 + Q(3)^2 - Q(4)^2;
    H(2,3) = 2*(Q(3)*Q(4) - Q(1)*Q(2));

    H(3,1) = 2*(Q(2)*Q(4) - Q(1)*Q(3));
    H(3,2) = 2*(Q(3)*Q(4) + Q(1)*Q(2));
    H(3,3) = Q(1)^2 - Q(2)^2 - Q(3)^2 + Q(4)^2;

    H = 1/h.*H;
end
end
```

Kvaternion glede na rotacijo za kot ϕ okrog osti e:

Input:

- kot fi
- vektor osi e

Output:

- kvaternion Q

Listing 6: kot_v_kvat

```
function r = kot_v_kvat(fi, e)

e = e/norm(e);
r = [cos(fi/2) sin(fi/2)*e(1) sin(fi/2)*e(2) sin(fi/2)*e(3)];
end
```

Rotacija vektorja okrog koordinatnih osi:

Input:

- vektor *vec*
- koti kot_x, kot_y, kot_z rotacije okrog x,y,z osi

Output:

- zarotiran vektor v

Listing 7: rot_vek_za_kot

```
function v = rot_vek_za_kot(vec, kot_x,kot_y,kot_z)

q = angle2quat(kot_x, kot_y, kot_z);
v = quatmultiply(q, quatmultiply([0 vec], conj_quat(q)));
end
```

3.3 Bezierjeve krivulje

Bezier:
Input:

- matrika B velikosti $(n+1)\times d$, ki predstavlja kontrolne točke Bezierjeve krivulje
- seznam parametrov t dolzine k, pri katerih računamo vrednost Bezierjeve krivulje

Output:

- matrika b, ki predstavlja točke na Bezierjevi krivulji pri parametrih iz t

Listing 8: bezier

```
function b = bezier (B,t)

[n,d] = size(B);
k = length(t);
b = zeros(k,d);

for i=1:k
    for j=1:d
        D = decasteljau(B(:,j)',t(i));
        b(i,j) = D(1,n);
    end
end
```

Decasteljau:
Input:

- seznam koordinat b kontrolnih točk Bezierjeve krivulje stopnje n
- parameter t, pri katerem računamo koordinato Bezierjeve krivulje

Output:

- Casteljaujeva shema D

Listing 9: decasteljau

```
function D = decasteljau (b,t)

n = length(b);
D = [b', NaN(n,n-1)];

for r=1:n
    for i=0:n-r-1
        D(i+1,r+1) = (1-t)*D(i+1,r) + t*D(i+2,r);
    end
end
```

sBezier:

Input:

- matrika Q velikosti $(n+1)\times d$, ki predstavlja kontrolne točke Bezierjeve krivulje
- seznam parametrov t dolzine k, pri katerih računamo vrednost Bezierjeve krivulje

Output:

- matrika b, ki predstavlja točke na Bezierjevi krivulji pri parametrih iz t

Listing 10: sbezier

```
function b = sbezier(Q,t)
k = length(t);

b = cell(k,1);

for K = 1:k
    decast = sdecasteljau(Q,t(K));
    b{K} = decast(1,end);
end
b;
```

sDecasteljau:

Input:

- seznam koordinat Q kontrolnih točk Bezierjeve krivulje stopnje n
- parameter t, pri katerem računamo koordinato Bezierjeve krivulje

Output:

- Casteljaujeva shema D

Listing 11: sdecasteljau

```
function D = sdecasteljau(Q,t)

[n,m] = size(Q);
n = n-1;
D = cell(n+1, n+1);
for i = 1:(n+1)
    D{i,1} = Q(i,:);
end
for j=2:(n+1)
    for i=1:(n+2-j)
        D{i,j} = slerp(D{i,j-1},D{i+1,j-1},t);
    end
end
end
```

Matrika iz celice:
Input:

- celica Q

Output:

- matrika Q

Listing 12: polepsaj_sbezier

```
function mat_Q = polepsaj_sbezier(Q)

n = length(Q);
mat_Q = zeros(n,4);
for i = 1:n
    mat_Q(i,:) = Q{i}{1};
end
end
```

Translacija izhodišča:
Input:

- translacijska funkcija izhodišča w, računana na t ($n \times 3$ matrika)
- sferična rotacijska Bezierjeva krivulja, računana na t ($n \times 4$ matrika)
- seznam parametrov t dolžine n, pri katerih računamo funkcijo.

Output:

- normirana translacija ($n \times 3$ matrika)

Listing 13: translacija

```
function c = translacija(w, Q, t)

n = length(t);
c = zeros(n,3);

for i = 1:n
    nQt = norm(Q(i,:))^2;
    c(i,:) = w(i,:)/nQt;
end
end
```

Celotno gibanje togega telesa:

Input:

- matrika kontrolnih kvaternionov Q za sferične rotacije
- matrika kontrolnih točk za Bezierjevo krivuljo gibanja izhodišča
- seznam parametrov t, pri katerih opazujemo gibanje

Output:

- matrika kvaternionov mat_Q, ki določajo sferične rotacije
- Bezierjeva krivulja w, ki določa gibanje koordinatnega izhodišča
- normirana translacijska funkcija c

Listing 14: izracunaj_vse

```
function [mat_Q,w,c] = izracunaj_vse(Q, B, t)

mat_Q = polepsaj_sbezier(sbezier(Q,t));
w = bezier(B,t);
c = translacija(w,mat_Q, t);
end
```

3.4 Kocka

Oglišča kocke z diagonalo d:
Input:

- krajišči diagonale $T0$ in $T1$ ($d = T0T1$)

Output:

- koorinate oglišč kocke

Listing 15: kocka

```
function oglisca = kocka(T0, T1)

a = [T1(1) - T0(1) 0 0];
b = [0 T1(2) - T0(2) 0];
c = [0 0 T1(3) - T0(3)];
oglisca = [T0; T0+b; T0+a+b; T0+a; T0+c; T0+c+b; T0+a+b+c; T0+a+c];
end
```

Kocka s stranicami in izhodiščem:
Input:

- vektorji stranic X,Y,Z
- izhodišče $T0$

Output:

- vsa oglišča kocke, ki se začne v T0

Listing 16: kocka_vek

```
function oglisca = kocka_vek(X, Y, Z, T0)

oglisca = [T0; T0+Y; T0+X+Y; T0+X; T0+Z; T0+Z+Y; T0+X+Y+Z; T0+X+Z];
end
```

Rotacija kocke:
Input:

- kocka $K0$
- koti kot_x, kot_y, kot_z

Output:

- $K1$ - zarotirana kocka $K0$ za podane kote

Listing 17: rotiraj_kocko

```
function K1 = rotiraj_kocko(K0, kot_x, kot_y, kot_z)

K1 = zeros(size(K0));
for i = 1:8
    K1(i,:) = quat_vec(rot_vek_za_kot(K0(i,:), kot_x, kot_y, kot_z));
end
end
```

Rotacija kocke:

Input:

- kvaternion Q
- vektorji stranic kocke x,y in z

Output:

- Oglišča zarotirane kocke z enim ogliščem v $[0, 0, 0]$

Listing 18: rotirana_kocka

```
function [K0, x0, y0, z0] = rotirana_kocka(x,y,z,Q)

H = quat_rot_mat(Q);
x0 = (H*x')';
y0 = (H*y')';
z0 = (H*z')';
K0 = kocka_vek(x0,y0,z0, [0 0 0]);

end
```