

Univerza v Ljubljani
Fakulteta za matematiko in fiziko

Gibanja togih teles

Matic Oskar Hajšen in Eva Zmazek

Ljubljana, 2019

Kazalo

1	Uvod	4
2	Teoretično ozadje	4
2.1	Homogene in kartezične koordinate	4
2.2	Zveza med koordinatami točk v fiksnem koordinatnem sistemu in ko- ordinatami točk v gibajočem se koordinatnem sistemu	4
2.3	Gibanje točk v času	5
2.4	Opis rotacij s kvaternioni	6
2.5	Bezierjeve krivulje	8
3	Implementacija	9
3.1	Kvaternioni	9
3.2	Bezierjeve krivulje	13
3.3	Kocka	17
3.4	Razvrsti	20

Listings

1	quat_vec	9
2	quatmultiply	9
3	conj_quat	10
4	quat_exp	10
5	quat_rot_mat	11
6	kot_v_kvrat	11
7	rot_vek_za_kot	12
8	bezier	13
9	decasteljau	13
10	sbezier	14
11	sdecasteljau	14
12	polepsaj_sbezier	15
13	translacija	15
14	izracunaj_vse	16
15	kocka	17
16	kocka_vek	17
17	risi_kocko	18
18	rotiraj_kocko	18
19	rotirana_kocka	18
20	plot_kontrolne_kocke	19
21	plot_tirnice	20
22	slerp	21
23	narisi_vse	22

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 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 :

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

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

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

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:

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

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 .

3×3 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 $\overline{\mathcal{A}} = (a_0, -\underline{a})$. S pomočjo konjugirane vrednosti nato definiramo tudi normo kvaterniona kot

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

Opomba 2.1 (Implementirane metode). V implementaciji vektorski del kvaterniona a pridobimo s funkcijo `quat_vec(a)`, skalararni del pa kar z ukazom `a(1)`. Kvaterniona a in b seštejemo z ukazom `a + b`, njun produkt pa kličemo s funkcijo `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 $\chi : \mathbb{H} \setminus \{0\} \rightarrow SO_3$ 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 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_kvaternion(φ, 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.

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

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

$$\begin{aligned}
m_{1,3} - m_{3,1} &= 4q_0q_2 \\
m_{2,1} + m_{1,2} &= 4q_1q_2 \\
m_{0,0} - m_{1,1} + m_{1,2} - m_{3,3} &= 4q_2^2 \\
m_{3,2} + m_{2,3} &= 4q_2q_3 \\
\\
m_{2,1} - m_{1,2} &= 4q_0q_3 \\
m_{1,3} + m_{3,1} &= 4q_1q_3 \\
m_{3,2} + m_{2,3} &= 4q_2q_3 \\
m_{0,0} - m_{1,1} - m_{1,2} + m_{3,3} &= 4q_0^2
\end{aligned}$$

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

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

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

3 Implementacija

3.1 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

```
1 function v = quat_vec(Q)
2
3 v = [Q(2) Q(3) Q(4)];
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

```
1 function c = quatmultiply(a,b)
2
3 c = zeros(1,4);
4 a_s = a(1);
5 b_s = b(1);
6 a_v = quat_vec(a);
7 b_v = quat_vec(b);
8 c(1) = a_s * b_s - a_v * b_v';
9 c_v = a_s * b_v + b_s * a_v + cross(a_v,b_v);
10 c(2) = c_v(1);
11 c(3) = c_v(2);
12 c(4) = c_v(3);
13 end
```

Konjugirana vrednost kvaterniona:

Input:

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

Output:

- kvaternion $Q = \bar{q}$

Listing 3: conj_quat

```
1 function Q = conj_quat(q)
2
3 Q = [q(1), -q(2:4)];
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

```
1 function e = quat_exp(q, t)
2
3 if t==1
4     e = conj_quat(q)/norm(q);
5 else
6     a = q(1);
7     v = quat_vec(q);
8     theta = acos(a/norm(q));
9     n = v/norm(v);
10    e = norm(q)^t*[cos(t*theta), n*sin(t*theta)];
11 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

```

1 function H = quat_rot_mat(Q)
2
3 H = zeros(3,3);
4 h = sum(Q.^2);
5
6 if h == 0
7     H = eye(3,3);
8 else
9     H(1,1) = Q(1)^2+Q(2)^2 - Q(3)^2 - Q(4)^2;
10    H(1,2) = 2*(Q(2)*Q(3) - Q(1)*Q(4));
11    H(1,3) = 2*(Q(2)*Q(4) + Q(1)*Q(3));
12
13    H(2,1) = 2*(Q(2)*Q(3) + Q(1)*Q(4));
14    H(2,2) = Q(1)^2 - Q(2)^2 + Q(3)^2 - Q(4)^2;
15    H(2,3) = 2*(Q(3)*Q(4) - Q(1)*Q(2));
16
17    H(3,1) = 2*(Q(2)*Q(4) - Q(1)*Q(3));
18    H(3,2) = 2*(Q(3)*Q(4) + Q(1)*Q(2));
19    H(3,3) = Q(1)^2 - Q(2)^2 - Q(3)^2 + Q(4)^2;
20
21    H = 1/h.*H;
22 end
23 end

```

Kvaternion glede na rotacijo za kot ϕ okrog osi e :

Input:

- kot ϕ
- vektor osi e

Output:

- kvaternion Q

Listing 6: kot_v_kvrat

```

1 function r = kot_v_kvrat(fi, e)
2
3 e = e/norm(e);
4 r = [cos(fi/2) sin(fi/2)*e(1) sin(fi/2)*e(2) sin(fi/2)*e(3)];
5 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

```
1 function v = rot_vek_za_kot(vec, kot_x,kot_y,kot_z)
2
3 q = angle2quat(kot_x, kot_y, kot_z);
4 v = quatmultiply(q, quatmultiply([0 vec], conj_quat(q)));
5 end
```

3.2 Bezierjeve krivulje

Bezier:

Input:

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

Output:

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

Listing 8: bezier

```
1 function b = bezier (B,t)
2
3 [n,d] = size(B);
4 k = length(t);
5 b = zeros(k,d);
6
7 for i=1:k
8     for j=1:d
9         D = decasteljau(B(:,j)',t(i));
10        b(i,j) = D(1,n);
11    end
12 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

```
1 function D = decasteljau (b,t)
2
3 n = length(b);
4 D = [b', NaN(n,n-1)];
5
6 for r=1:n
7     for i=0:n-r-1
8         D(i+1,r+1) = (1-t)*D(i+1,r) + t*D(i+2,r);
9     end
10 end
```

sBezier:

Input:

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

Output:

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

Listing 10: sbezier

```
1 function b = sbezier(Q,t)
2 k = length(t);
3
4 b = cell(k,1);
5
6 for K = 1:k
7     decast = sdecasteljau(Q,t(K));
8     b{K} = decast(1,end);
9 end
10 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

```
1 function D = sdecasteljau(Q,t)
2
3 [n,m] = size(Q);
4 n = n-1;
5 D = cell(n+1, n+1);
6 for i = 1:(n+1)
7     D{i,1} = Q(i,:);
8 end
9 for j=2:(n+1)
10     for i=1:(n+2-j)
11         D{i,j} = slerp(D{i,j-1},D{i+1,j-1},t);
12     end
13 end
14 end
```

Matrika iz celice:

Input:

- celica Q

Output:

- matrika Q

Listing 12: polepsaj_sbezier

```
1 function mat_Q = polepsaj_sbezier(Q)
2
3 n = length(Q);
4 mat_Q = zeros(n,4);
5 for i = 1:n
6     mat_Q(i,:) = Q{i}{1};
7 end
8 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

```
1 function c = translacija(w, Q, t)
2
3 n = length(t);
4 c = zeros(n,3);
5
6 for i = 1:n
7     nQt = norm(Q(i,:))^2;
8     c(i,:) = w(i,:)/nQt;
9 end
10 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

```
1 function [mat_Q,w,c] = izracunaj_vse(Q, B, t)
2
3 mat_Q = polepsaj_sbezier(sbezier(Q,t));
4 w = bezier(B,t);
5 c = translacija(w,mat_Q, t);
6 end
```


3.3 Kocka

Oglišča kocke z diagonalo d :

Input:

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

Output:

- koordinate oglišč kocke

Listing 15: kocka

```
1 function oglisca = kocka(T0, T1)
2
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 oglisca = [T0; T0+b; T0+a+b; T0+a; T0+c; T0+c+b; T0+a+b+c; T0+a+c];
7 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

```
1 function oglisca = kocka_vek(X, Y, Z, T0)
2
3 oglisca = [T0; T0+Y; T0+X+Y; T0+X; T0+Z; T0+Z+Y; T0+X+Y+Z; T0+X+Z];
4 end
```

Slika kocke:

Input:

- koordinate oglišč K
- barva *barva*

Output:

- slika kocke

Listing 17: risi_kocko

```

1 function risi_kocko(K, barva)
2
3 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];
4 patch('Vertices', K, 'Faces', ploskve, 'FaceColor', barva);
5 end

```

Rotacija kocke:

Input:

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

Output:

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

Listing 18: rotiraj_kocko

```

1 function K1 = rotiraj_kocko(K0, kot_x, kot_y, kot_z)
2
3 K1 = zeros(size(K0));
4 for i = 1:8
5     K1(i,:) = quat_vec(rot_vek_za_kot(K0(i,:), kot_x, kot_y, kot_z));
6 end
7 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 19: rotirana_kocka

```

1 function [K0, x0, y0, z0] = rotirana_kocka(x,y,z,Q)
2
3 H = quat_rot_mat(Q);
4 x0 = (H*x')';
5 y0 = (H*y')';
6 z0 = (H*z')';
7 K0 = kocka_vek(x0,y0,z0, [0 0 0]);
8
9 end

```

Gibanje kocke:

Input:

- vektorji x, y in z , i določajo začetno kocko.
- izhodišče kocke T_0
- matrika B kontrolnih kvaternionov
- translacijska funkcija c izhodišča gibajočega se koordinatnega sistema
- barve zac_barva , $vmes_barva$, kon_barva kontrolnih kvadrov
- argument $pavza$, ki pove, ali naj slika poteka v obliki gibanja

Output:

- slika gibanja kvadra

Listing 20: plot_kontrolne_kocke

```
1 function plot_kontrolne_kocke(x0,y0,z0,T0, B,c, zac_barva,
2   vmes_barva, kon_barva, pavza)
3 for i = 1:size(B,1)
4     if pavza
5         pause
6     end
7     switch i
8         case 1
9             barva = zac_barva;
10        case size(B,1)
11            barva = kon_barva;
12        otherwise
13            barva = vmes_barva;
14    end
15    Q = B(i,:);
16    H = quat_rot_mat(Q);
17    x = (H*x0')';
18    y = (H*y0')';
19    z = (H*z0')';
20    T = T0+c(i,:);
21    risi_kocko(kocka_vek(x, y, z, T),barva);
22 end
```

3.4 Razvrsti

:

Input:

-

Output:

-

Listing 21: `plot_tirnice`

```
1 function plot_tirnice(P, c)
2 %input:
3 % P          premaknjeni vektorji kvadra
4 % c          translacijska funkcija
5 %ouput:
6 % narise tirnice, po katerih se premikajo oglišca kvadra
7
8 n = size(c, 1);
9
10 %en vektor
11 V_x = zeros(n,3);
12 V_y = zeros(n,3);
13 V_z = zeros(n,3);
14 V_xy = zeros(n,3);
15 V_zy = zeros(n,3);
16 V_xz = zeros(n,3);
17 V = zeros(n,3);
18
19
20 for i = 1:n
21     Pi = P{i};
22     V_x(i,:) = Pi(1,:);
23     V_y(i,:) = Pi(2,:);
24     V_z(i,:) = Pi(3,:);
25 end
26
27 % dva vektorja
28 V_xy = V_x + V_y + c;
29 V_zy = V_z + V_y + c;
30 V_xz = V_x + V_z + c;
31
32 % vsi
33 V = V_x + V_y + V_z + c;
34
35 V_x = V_x + c;
36 V_y = V_y + c;
```

```

37 V_z = V_z + c;
38
39
40 plot3(V_x(:,1), V_x(:,2), V_x(:,3),...
41       V_y(:,1), V_y(:,2), V_y(:,3),...
42       V_z(:,1), V_z(:,2), V_z(:,3),...
43       V_xy(:,1), V_xy(:,2), V_xy(:,3),...
44       V_zy(:,1), V_zy(:,2), V_zy(:,3),...
45       V_xz(:,1), V_xz(:,2), V_xz(:,3),...
46       V(:,1), V(:,2), V(:,3),...
47       c(:,1), c(:,2), c(:,3),...
48       'LineWidth',1.5)
49
50
51 end

```

Listing 22: `slerp`

```

1 function Q = slerp(Q1, Q2, t)
2 %input:
3 % Q1, Q2          kvaterniona
4 % t               parameter v [0,1]
5 %output:
6 % Q               rezultat slerp-a med Q1 in Q2 na mestu t
7 %To je sfericna linearna interpolacija, ki naj bi zamenjala
8 %navadno v de Casteljauju.
9 % https://en.wikipedia.org/wiki/Slerp
10
11 if t == 0
12     Q = Q1;
13 else
14     if t == 1
15         Q = Q2;
16     else
17         % q = quatmultiply(Q1,Q2);
18         % if q(1) <= 0
19         %     display('dolga pot')
20         %     display(q)
21         %     [r1,r2,r3] = quat2angle(q)
22         %     %Q1 = -Q1; %tako naj bi izbrala krajso pot, se mi zdi
23         %     end
24         Q = quatmultiply(Q1, quat_exp(quatmultiply(quat_exp(Q1,-1),Q2
25                                     ),t));
26     end
27 end

```

:
Input:

•

Output:

•

Listing 23: `narisi_vse`

```
1 function narisi_vse(x0,y0,z0,T0, mat_Q,c, t, osi, prehod,tirnice,
   robovi)
2 %Tole bo narisalo vse
3 %input:
4 % x0,y0,z0          stranice kvadra
5 % T0                izhodišce kvadra
6 % mat_Q             matrika sfericnih rotacij
7 % c                 matrika normirane translacijske funkcije
8 % t                 parametri, pri katerih racunamo
9 % osi               dolzine osi
10 % prehod, tirnice, robovi 1 ali 0, kaj hocemo risati
11 n = length(t);
12
13 H = cell(n,1);
14 P = cell(n,1);
15 hold on
16 axis equal
17 axis([-osi osi -osi osi -osi osi])
18
19 % Rise prehajanje kvadrov
20
21 for i = 1:n
22     H{i} = quat_rot_mat(mat_Q(i,:));
23     x = (H{i}*x0')';
24     y = (H{i}*y0')';
25     z = (H{i}*z0')';
26     P{i} = [x;y;z];
27     if prehod
28         if (mod(i, 1) == 0 || i == 1)
29             risi_kocko(kocka_vek(x, y, z, c(i,:)), [(n-i)/n, 0, i/n])
30             ;
31         end
32     end
33 end
34 % Rise tirnice, po katerih se premikajo oglišca kvadra
35 if tirnice
```

```

36     plot_tirnice(P,c)
37     plot3(c(:,1), c(:,2), c(:,3),'LineWidth', 2, 'Color', 'k')
38 end
39 % Narise izracunano prvo in zadnje stanje
40 if robovi
41     H = quat_rot_mat(mat_Q(end,:));
42     x = (H*x0')';
43     y = (H*y0')';
44     z = (H*z0')';
45     T = T0+c(end,:);
46     risi_kocko(kocka_vek(x, y, z, T),'g');
47     H = quat_rot_mat(mat_Q(1,:));
48     x = (H*x0')';
49     y = (H*y0')';
50     z = (H*z0')';
51     T = T0+c(1,:);
52     risi_kocko(kocka_vek(x, y, z, T),'y');
53 end
54 end

```