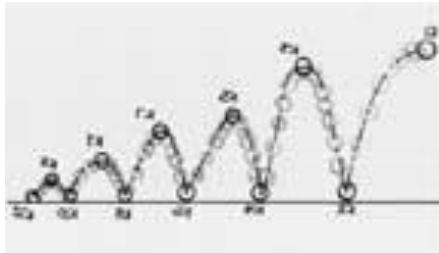


1. Definiranje objekata, putanja i pokreta

- Objekte putanje i pokrete potrebno je prenijeti iz stvarnog svijeta u računalo
 - različite putanje objekata, orijentacije, izobličavanje, promjene boje
- Način ostvarivanja objekata za animaciju
 - definiranje objekata i putanja
 - uzorkovanje objekata i putanja
 - proceduralno modeliranje objekata



1.1 Alati za definiranje animacije

- alati za izgradnju objekata, scene, animacije i pripadni zapisi datoteka
 - različiti alati za različite namjene
 - definiranje objekata, scene, ([AutoCAD](#), [Microstation](#), [Catia](#)) - strojarstvo
 - definiranje pokreta (3DSMax, SoftImage, Maya) - RG
 - izrada prikaza (MentalRay, [RendermanPixar](#))
 - specijalni vizualni učinci - dodaci alatima, skriptni jezici,
 - alati za izradu igara, [Unity](#) <http://threejsexporter.nickjanssen.com/>, Unreal Engine [UE4](#)
- primjeri alata:
 - [Maya](#) '98 – Alias (/Wavefront) '95 SGI (objekti, scena, pokreti ...)
 - [3DS MAX](#)
 - [LightWave](#), [Modo](#)
 - [Cinema 4D](#), [Houdini](#),
 - [SoftImage](#) (objekti, scena, pokreti, Behavior, Batch)
 - [MentalRay](#) ('88 samostalan paket za izradu prikaza Nvidia), [Renderman](#)-Pixar,
 - (free, open source -), [Blender](#) (modeliranje), [PowRay](#) (izrada prikaza)
 - [Godot](#) (game engine)
 - Web alati za modeliranje [Clara.io](#), API za web aplikacije [Three.js](#)
 - <https://cecropia.github.io/thehallaframe/> CTRL+ALT+I (AFrame)

- objekti u animaciji - primjeri

- modeliranje objekata

- dijelovi objekata

- ishodište (engl. origin point) lokalni sustav
 - pivot točka ishodište za transformacije
 - orijentacija (početni položaj)
 - svojstva (npr. teksture)
 - ograničenja u pokretljivosti

- modeliranje povezanosti dijelova objekta

- stvaranje kostura - skeleta (engl. rigging)
 - povezivanje kože s kosturom (engl. skinning)

- https://threejs.org/examples/#webgl_animation_skinning_morph

- modeliranje pokreta

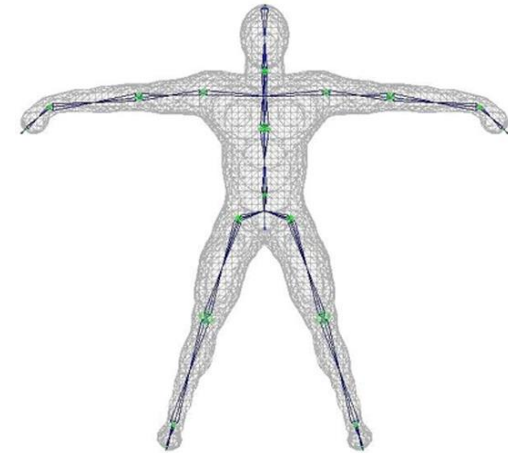
- tehnike niže razine – npr:

- animator treba odrediti kako će se objekt gibati u vremenu i prostoru (putanju), pokrete pomicanjem kostiju kostura
 - definiranje ključnih okvira (Blender Timeline, Dope Sheet, Graph Editor)

- http://threejs.org/examples/webgl_animation_skinning_blending.html

- http://alteredqualia.com/three/examples/webgl_animation_skinning_tf2.html

- tehnike više razine https://threejs.org/examples/webgl_animation_skinning_ik.html



- tehnike niže razine
 - animator ima ideju koju želi ostvariti
 - zadaje ključne položaje i dovoljni informacije uz te položaje
 - interpolacija (in-betweening)
- tehnike više razine
 - pravila i ograničenja koja utječu na ponašanje objekata, te postavlja početne uvjete
 - kretanje objekata je upravljano algoritmom ili modelom
 - proceduralno generiranje objekata, scene, kretanja (Čvorovi geometrije)
<https://demo.marpi.pl/spiders/>
 - fizikalno temeljeni pokreti, sudari (proceduralna animacija)
http://graphics.stanford.edu/~fedkiw/animations/Burning_Paper.avi
 - <http://webgl-fire.appspot.com/html/fire.html> Chrome
- podjela na tehnike više i niže razine više je pedagoška
 - u praksi je to kombinacija obje tehnike

1.2 Zadavanje i praćenje putanje

- vrijednosti parametara u ključnim okvirima (položajima) koje je potrebno interpolirati)
 - trajektorija objekta ili kamere
 - kutovi između spojnih dijelova robota
 - žarišna daljina kamere, boja izvora
- odabir odgovarajuće krivulje
 - modeliranje krivulje
 - aproksimacijska krivulja
 - Bezier, B-spline
 - uzorkovanje putanje
 - interpolacijska krivulja
 - Carmull-Rom splajn, Hermitova krivulja
- odabir kontinuiteta krivulje – prva, druga derivacija
- poželjan lokalni nadzor

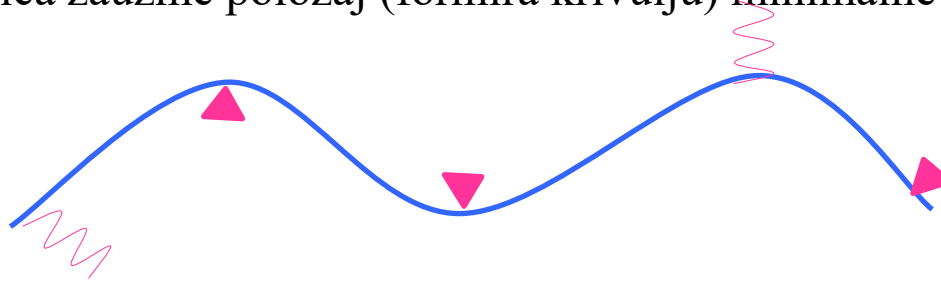
1.2.1 Zadavanje putanje

Bezier-ova krivulja nije dovoljno dobra → SEGMENTIRANJE KRIVULJE

<https://stemkoski.github.io/Three.js/ParticleSystem-PathMovement.html>

<https://math.hws.edu/graphicsbook/demos/c2/cubic-bezier.html>

- povezivanje segmenata uz očuvanje kontinuiteta na spojevima segmenata C^0 , C^1 , C^2 ... <http://www.cs.vu.nl/~eliens/papers/media/@vr-vrml-math-FlyDemo.wrl>
-
- korištenje crtaće letvice <http://www.cs.vu.nl/~eliens/media/@vr-vrml-math-FlyInt.wrl>
 - letvica se učvrsti i optereti utezima tako da postigne željeni oblik
 - crtač zatim iscrta krivulju
 - letvica zauzme položaj (formira krivulju) minimalne potencijalne energije



letvica se savija po zakonu progiba opterećene grede:

$$\frac{d^2 y}{dx^2} = \frac{M}{EI} \Rightarrow d^2 y = \frac{M}{EI} dx^2 \quad \text{rješenje će dati kubnu funkciju}$$

M – moment koji djeluje na gredu

E – Yungov modul elastičnosti, I – moment tromosti

B - KRIVULJA

B - elastična krivulja ima svojstvo elastične letvice (bazni “spline”)

- kontinuitet je postignut dijeljenjem kontrolnih točaka između više segmenata

- prirodni splajn se može prikazati kao težinska suma baznih funkcija

– APROKSIMACIJSKA B-KRIVULJA

- k stupanj krivulje (broj kontrolnih točaka ne utječe na stupanj)
- r_i kontrolne točke - ukupno ih ima $n+1$
- $N_{i,k}$ bazne (težinske) funkcije - polinomi stupnja k
- u_i vrijednosti uzlova (engl. knot values)
- $U_{KNOT} = \{u_i\}$ vektor uzlova

$$\vec{p}(u) = \sum_{i=0}^n \vec{r}_i N_{i,k}(u)$$

Određivanje baznih funkcija

$$N_{i,0}(u) = \begin{cases} 1 & \text{za } u_i \leq u < u_{i+1} \\ 0 & \text{inače} \end{cases}$$

$$N_{i,k}(u) = \frac{u - u_i}{u_{i+k} - u_i} N_{i,k-1}(u) + \frac{u_{i+k+1} - u}{u_{i+k+1} - u_{i+1}} N_{i+1,k-1}(u)$$

- Ako je nazivnik jednak nuli vrijednost razlomka je nula.
- Uzlovi mogu biti višestruki.
- $u_{i+1} - u_i = \text{konst.}$ Krivulja se naziva UNIFORMNA krivulja.

Inače krivulja je NEUNIFORMNA.

Vektor uzlova

- veza između broja točaka stupnja krivulje i vektora uzlova

$$U_{KNOT} = \left[\underbrace{0 \ 0 \ .. \ 0}_{k+1} \ \underbrace{1 \ 2 \ 3 \ .. \ (m-2k-1)}_{m-2k-1} \ \underbrace{(m-2k) \ .. \ (m-2k)}_{k+1} \right]$$

- $n + 1$ - broj kontrolnih točaka
- k - stupanj krivulje
- $m + 1$ - broj vrijednosti u vektoru uzlova

$$m = n + k + 1$$

- broj segmenata krivulje

$$\text{broj segmenata krivulje} = n - k + 1 = m - 2k$$

- specijalan slučaj $m - 2k - 1 = 0 \rightarrow$ KRIVULJA BEZIERA
(preko Bernsteinovih polinoma)

* PRIMJER

Neperiodička kvadratna B-krivulja određene je sa šest kontrolnih točaka.

Odrediti segmente krivulje.

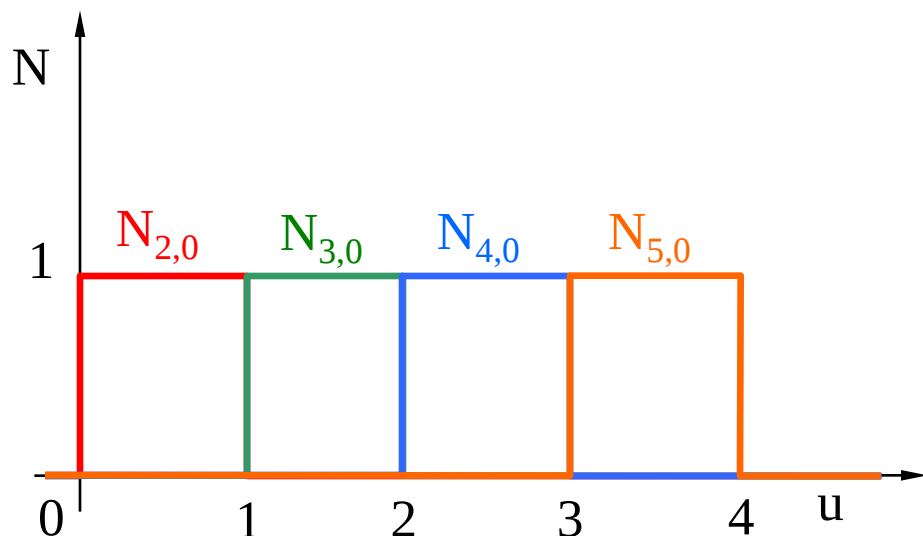
- $k = 2$
- $n + 1 = 6$
- $m = n + k + 1 = 8$

$$U_{KNOT} = \begin{bmatrix} u_0 & u_1 & u_2 & u_3 & u_4 & u_5 & u_6 & u_7 & u_8 \\ \underbrace{0 \quad 0 \quad 0}_{k+1} & \underbrace{1 \quad 2 \quad 3}_{m-2k-1} & \underbrace{4 \quad 4 \quad 4}_{k+1} \end{bmatrix}$$

$$\vec{p}(u) = \sum_{i=0}^5 \vec{r}_i N_{i,0}(u)$$

1. Korak:

$$\begin{array}{ll} N_{0,0} = 1|_{[0]} & N_{1,0} = 1|_{[0]} \\ N_{2,0} = 1|_{[0,1]} & N_{3,0} = 1|_{[1,2]} \\ N_{4,0} = 1|_{[2,3]} & N_{5,0} = 1|_{[3,4]} \\ N_{6,0} = 1|_{[4]} & N_{7,0} = 1|_{[4]} \end{array}$$



2. Korak:

$$U_{KNOT} = \begin{bmatrix} u_0 & u_1 & u_2 & u_3 & u_4 & u_5 & u_6 & u_7 & u_8 \\ \underbrace{0 \quad 0 \quad 0}_{k+1} & \underbrace{1 \quad 2 \quad 3}_{m-2k-1} & \underbrace{4 \quad 4 \quad 4}_{k+1} \end{bmatrix}$$

$$N_{i,1}(u) = \frac{u - u_i}{u_{i+1} - u_i} N_{i,0}(u) + \frac{u_{i+2} - u}{u_{i+2} - u_{i+1}} N_{i+1,0}(u)$$

$$N_{0,1} = \frac{u - u_0}{u_1 - u_0} N_{0,0} + \frac{u_2 - u}{u_2 - u_1} N_{1,0} = 0 \Big|_{[0]}$$

$$N_{1,1} = \frac{u - u_1}{u_2 - u_1} N_{1,0} + \frac{u_3 - u}{u_3 - u_2} N_{2,0} = (1 - u) \Big|_{[0,1]}$$

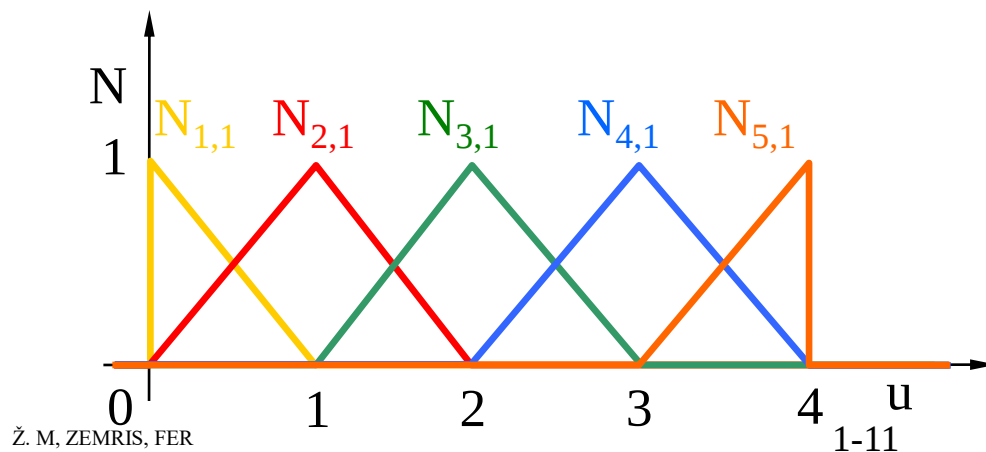
$$N_{2,1} = \frac{u - u_2}{u_3 - u_2} N_{2,0} + \frac{u_4 - u}{u_4 - u_3} N_{3,0} = u \Big|_{[0,1]} + (2 - u) \Big|_{[1,2]}$$

$$N_{3,1} = (u - 1) \Big|_{[1,2]} + (3 - u) \Big|_{[2,3]}$$

$$N_{4,1} = (u - 2) \Big|_{[2,3]} + (4 - u) \Big|_{[3,4]}$$

$$N_{5,1} = (u - 3) \Big|_{[3,4]} + 0 \Big|_{[4]}$$

$$N_{6,1} = 0 \Big|_{[4]}$$



3. Korak:

$$N_{i,2}(u) = \frac{u - u_i}{u_{i+2} - u_i} N_{i,1}(u) + \frac{u_{i+3} - u}{u_{i+3} - u_{i+1}} N_{i+1,1}(u)$$

$$U_{KNOT} = \begin{bmatrix} u_0 & u_1 & u_2 & u_3 & u_4 & u_5 & u_6 & u_7 & u_8 \\ \underbrace{0 \quad 0 \quad 0}_{k+1} & \underbrace{1 \quad 2 \quad 3}_{m-2k-1} & \underbrace{4 \quad 4 \quad 4}_{k+1} \end{bmatrix}$$

$$N_{0,2} = \frac{u - u_0}{u_2 - u_0} N_{0,1} + \frac{u_3 - u}{u_3 - u_1} N_{1,1} = (1 - u)^2 \Big|_{[0,1]}$$

$$N_{1,2} = \frac{u - u_1}{u_3 - u_1} N_{1,1} + \frac{u_4 - u}{u_4 - u_2} N_{2,1} = u(1 - u) \Big|_{[0,1]} + \frac{(2 - u)u}{2} \Big|_{[0,1]} + \frac{(2 - u)^2}{2} \Big|_{[1,2]}$$

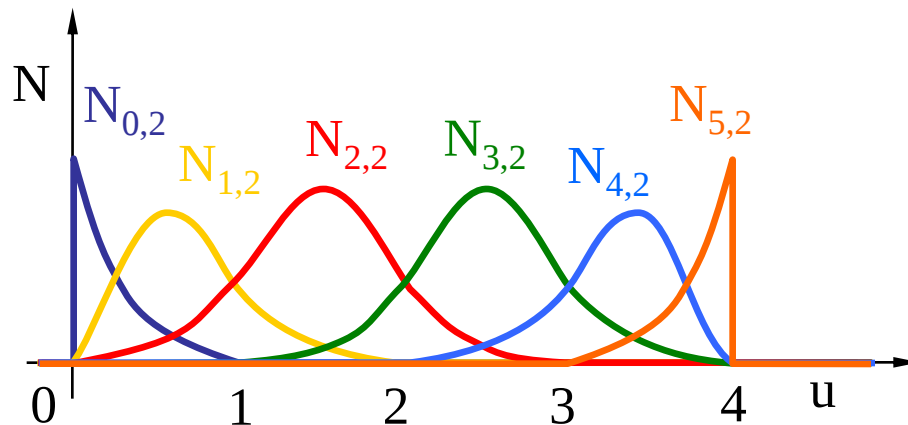
$$= \frac{u}{2} (4 - 3u) \Big|_{[0,1]} + \frac{(2 - u)^2}{2} \Big|_{[1,2]}$$

$$N_{2,2} = \frac{u^2}{2} \Big|_{[0,1]} + \frac{-2u^2 + 6u - 3}{2} \Big|_{[1,2]} + \frac{(3 - u)^2}{2} \Big|_{[2,3]}$$

$$N_{3,2} = \frac{(u-1)^2}{2} \Big|_{[1,2]} + \frac{-2u^2+10u+11}{2} \Big|_{[2,3]} + \frac{(4-u)^2}{2} \Big|_{[3,4]}$$

$$N_{4,2} = \frac{(u-2)^2}{2} \Big|_{[2,3]} + \left(-\frac{3}{2}u^2 + 10u - 16 \right) \Big|_{[3,4]}$$

$$N_{5,2} = (u-3)^2 \Big|_{[3,4]}$$



Određivanje segmenata krivulje:

$$\vec{p}(u) = \sum_{i=0}^5 \vec{r}_i N_{i,2}(u)$$

$N_{i,2}(u)$	[0, 1]	[1, 2]	[2, 3]	[3, 4]
$\vec{r}_0$	$u^2 - 2u + 1$			
$\vec{r}_1$	$-\frac{3}{2}u^2 + 2u$	$\frac{1}{2}u^2 - 2u + 2$		
$\vec{r}_2$	$\frac{u^2}{2}$	$-u^2 + 3u - \frac{3}{2}$	$\frac{1}{2}u^2 - 3u + \frac{9}{2}$	
$\vec{r}_3$		$\frac{1}{2}u^2 - u + \frac{1}{2}$	$-u^2 + 5u + \frac{11}{2}$	$\frac{1}{2}u^2 - 4u + 8$
$\vec{r}_4$			$\frac{1}{2}u^2 - 2u + 2$	$-\frac{3}{2}u^2 + 10u - 16$
$\vec{r}_5$				$u^2 - 6u + 9$

Parametar u mijenja se od 0-1, 1-2, 2-3, 3-4

Segment 1: $u \in [0, 1]$

$$\vec{p}_1(u) = \begin{bmatrix} u^2 & u & 1 \end{bmatrix} \begin{bmatrix} 1 & -\frac{3}{2} & \frac{1}{2} \\ -2 & 2 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \vec{r}_0 \\ \vec{r}_1 \\ \vec{r}_2 \end{bmatrix}$$

Segment 2: $u \in [1, 2]$

Reparametrizacija $u=t+1, t \in [0, 1]$:

$$\vec{p}_2(u) = \begin{bmatrix} u^2 & u & 1 \end{bmatrix} \begin{bmatrix} \frac{1}{2} & -1 & \frac{1}{2} \\ -2 & 3 & -1 \\ 2 & -\frac{3}{2} & \frac{1}{2} \end{bmatrix} \begin{bmatrix} \vec{r}_1 \\ \vec{r}_2 \\ \vec{r}_3 \end{bmatrix}$$

$$\vec{p}_2(t) = \begin{bmatrix} t^2 & t & 1 \end{bmatrix} \begin{bmatrix} \frac{1}{2} & -1 & \frac{1}{2} \\ -1 & 1 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 \end{bmatrix} \begin{bmatrix} \vec{r}_1 \\ \vec{r}_2 \\ \vec{r}_3 \end{bmatrix}$$

Segment 3: $u \in [2, 3]$

Reparametrizacija $u=t+2, t \in [0, 1]$:

$$\vec{p}_3(u) = \begin{bmatrix} u^2 & u & 1 \end{bmatrix} \begin{bmatrix} \frac{1}{2} & -1 & \frac{1}{2} \\ -3 & 5 & -2 \\ \frac{9}{2} & \frac{4}{2} & 2 \end{bmatrix} \begin{bmatrix} \vec{r}_2 \\ \vec{r}_3 \\ \vec{r}_4 \end{bmatrix} \quad \vec{p}_3(t) = \begin{bmatrix} t^2 & t & 1 \end{bmatrix} \begin{bmatrix} \frac{1}{2} & -1 & \frac{1}{2} \\ -1 & 1 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 \end{bmatrix} \begin{bmatrix} \vec{r}_2 \\ \vec{r}_3 \\ \vec{r}_4 \end{bmatrix}$$

Segment 4: $u \in [3, 4]$

Reparametrizacija $u=t+3, t \in [0, 1]$:

$$\vec{p}_4(u) = \begin{bmatrix} u^2 & u & 1 \end{bmatrix} \begin{bmatrix} \frac{1}{2} & -\frac{3}{2} & 1 \\ -4 & 10 & -6 \\ 8 & -16 & 9 \end{bmatrix} \begin{bmatrix} \vec{r}_3 \\ \vec{r}_4 \\ \vec{r}_5 \end{bmatrix} \quad \vec{p}_4(t) = \begin{bmatrix} t^2 & t & 1 \end{bmatrix} \begin{bmatrix} \frac{1}{2} & -\frac{3}{2} & 1 \\ -1 & 1 & 0 \\ \frac{1}{2} & \frac{1}{2} & 0 \end{bmatrix} \begin{bmatrix} \vec{r}_3 \\ \vec{r}_4 \\ \vec{r}_5 \end{bmatrix}$$

$n-3k+3=2$ periodička segmenta segmenti 2 i 3 su isti!

<http://www.martin-kraus.org/LiveGraphics3D/examples/parametrized/cagd/chap8fig1.html>

Periodički segment

- Unutar neperiodičke B-krivulje stupnja k postoje periodički segmenti:

$$n \geq 3k - 2 \quad \text{ili} \quad m \geq 4k - 1$$

- kada je zadan veliki broj točaka središnji dio krivulje jednostavnije se može računati uporabom izraza za periodički segment

Npr: periodički segment kubne B-krivulje ($k=3$):

$$\vec{p}_i(t) = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \frac{1}{6} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 3 & -6 & 3 & 0 \\ -3 & 0 & 3 & 0 \\ 1 & 4 & 1 & 0 \end{bmatrix} \begin{bmatrix} \vec{r}_{i-1} \\ \vec{r}_i \\ \vec{r}_{i+1} \\ \vec{r}_{i+2} \end{bmatrix}$$

- <http://www.martin-kraus.org/LiveGraphics3D/examples/parametrized/cagd/chap8fig2.html> <http://intanto.net/geox.html>
- broj periodičkih segmenata

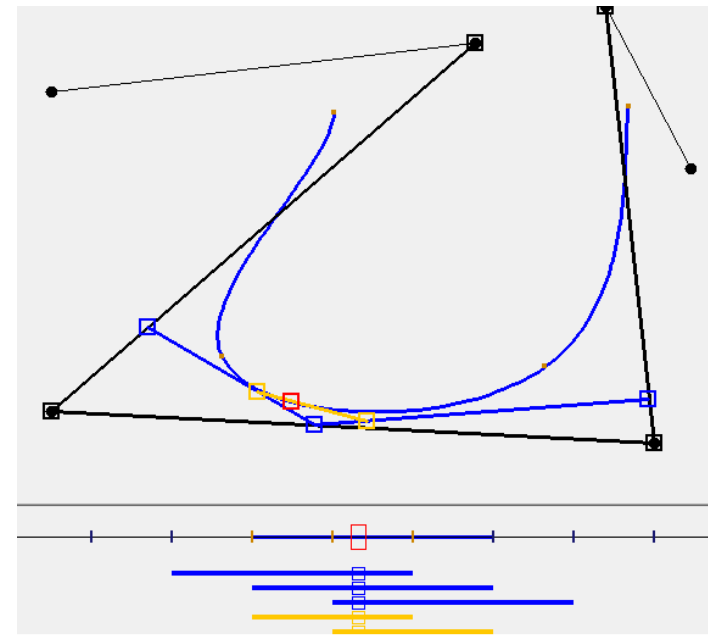
$$n - 3k + 3 \quad \text{ili} \quad m - 4k + 2$$

Geometrijska svojstva

- Krivulja leži unutar **konveksne ljuske** kontrolnih točaka.
To je posljedica baricentrične kombinacije težinskih funkcija.

$$\sum_{i=0}^n N_{i,k}(u) = 1$$

- **i - ti segment** je u konveksnoj ljusci pripadnih kontrolnih točaka
- $k = 1$ **linearna** interpolacija - krivulja je jednaka **karakterističnom poligonu**
- **lokalni** nadzor - pomak jedne točke utječe najviše na $k+1$ segment
- u uzlu krivulja ima **neprekinutost** C^{k-q} , q višestrukost uzla
- *** http://www.cs.utah.edu/~dav/curve_ed/
- <https://www.ibiblio.org/e-notes/Splines/nurbs.html>
- Konstrukcija krivulje – De Boor schema
- <https://www.food4rhino.com/en/app/de-boors-algorithm>

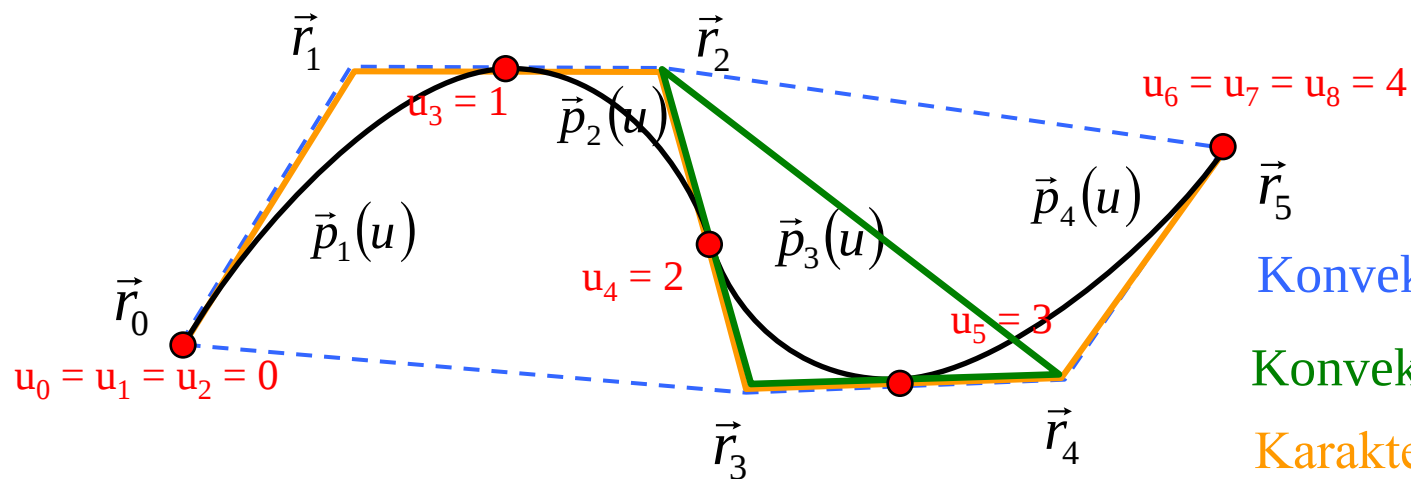
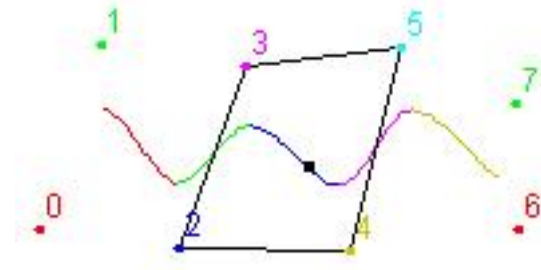


- Krivulja prolazi kroz početnu i završnu točku

$$\vec{p}_1(0) = \vec{r}_0, \quad \vec{p}_{kraj}(1) = \vec{r}_n$$

- Derivacije u početnoj i krajnjoj točki

$$\vec{p}'_1(0) = \frac{k(\vec{r}_1 - \vec{r}_0)}{u_{k+1}}, \quad \vec{p}'_{kraj}(1) = \frac{k(\vec{r}_n - \vec{r}_{n-1})}{1 - u_{m-k-1}}$$



Konveksna ljuska

Konveksna ljuska p_3

Karakteristični poligon

Oblikovanje krivulje

- Višestruke kontrolne točke
- Višestruke vrijednosti uzlova

⇒ smanjenje kontinuiteta - krivulja se približava kontrolnim točkama

- Fantomske točke - dodaju se kolinearno s derivacijama u krajnjim točkama tako da krivulja prolazi tim točkama $(\vec{r}_{-1} \text{ sa } \vec{r}_0 \vec{r}_1)$
- Poznate derivacije *** <https://www.ibiblio.org/e-notes/Splines/none.html>
- <https://www.ibiblio.org/e-notes/Splines/basis.html>

Višestrukost	Kontrolne točke	Uzla
1	$C^2 \ G^2$	$C^2 \ G^2$
2	$C^2 \ G^1$	$C^1 \ G^1$
3	$C^2 \ G^0$	$C^0 \ G^0$
4	$C^2 \ G^0$	diskontinuitet - odvojene točke

INTERPOLACIJSKI OBLIK B-KRIVULJA

- k stupanj krivulje
- p_i točke kroz koje želimo da krivulja prolazi - ukupno ih ima $n + 1$
- $\Rightarrow$ potrebno je odrediti točke kontrolnog poligona r_j tako da krivulja prolazi točkama p_i . Kada odredimo točke r_j načinimo aproksimacijsku krivulju određenu točkama r_j

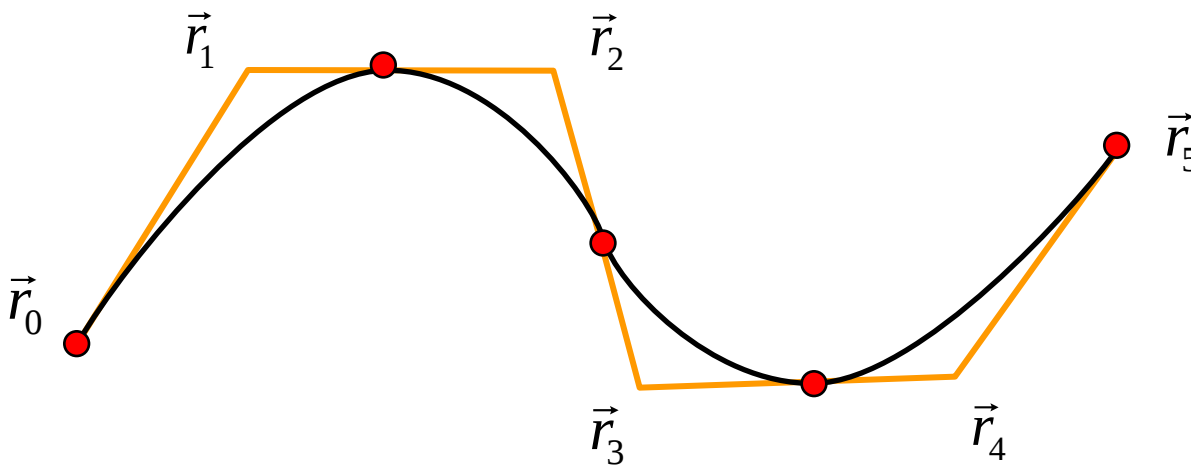
	r_j	Broj uvjeta:
Zatvorene periodičke krivulje	$j = 0 \dots n - 1$	n
Otvorene neperiodičke krivulje	$j = 0 \dots n + k - 1$	$n + k$

Kod zatvorenih periodičkih krivulja krajnje točke se preklape s početnima. Npr. 2 točke preklopimo kod kvadratne, 3 kod kubne krivulje. Kod otvorenih periodičkih krivulja poznato je $n + 1$ interpolacijskih točaka i treba još $k - 1$ dodatnih uvjeta.

* PRIMJER

Poznato je pet točaka. Odrediti kvadratnu interpolacijsku otvorenu neperiodičku B-krivulju.

- $k = 2$
- poznato je 5 točaka kroz koje krivulja treba prolaziti
- treba nam još $k-1=1$ dodatni uvjet
 - neka je dodatni uvjet derivacija u početnoj točki
- kontrolni poligon će imati 6 točaka $\vec{r}_j \quad j = 0..5$



Segment 1: $u \in [0, 1]$

$$\vec{p}_1(0) = \begin{bmatrix} u^2 & u & 1 \end{bmatrix} \begin{bmatrix} 1 & -\frac{3}{2} & \frac{1}{2} \\ -2 & 2 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \vec{r}_0 \\ \vec{r}_1 \\ \vec{r}_2 \end{bmatrix} \bigg|_0 = \vec{r}_0$$

$$\vec{p}'_1(0) = \begin{bmatrix} 2u & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & -\frac{3}{2} & \frac{1}{2} \\ -2 & 2 & 0 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \vec{r}_0 \\ \vec{r}_2 \\ \vec{r}_3 \end{bmatrix} \bigg|_0 = -2\vec{r}_0 + 2\vec{r}_1$$

Segment 2: $u = 1$ ili u reparametriziranom obliku $t = 0$:

$$\vec{p}_2(0) = \frac{\vec{r}_1}{2} + \frac{\vec{r}_2}{2}$$

Segment 3:

$$\vec{p}_3(0) = \frac{\vec{r}_2}{2} + \frac{\vec{r}_3}{2}$$

Segment 4:

$$\vec{p}_4(0) = \frac{\vec{r}_3}{2} + \frac{\vec{r}_4}{2}$$

$$\vec{p}_4(1) = \vec{r}_5$$

Svih 6 uvjeta zapisano matrično:

$$\begin{bmatrix} \vec{p}_1(0) \\ \vec{p}_1'(0) \\ \vec{p}_2(0) \\ \vec{p}_3(0) \\ \vec{p}_4(0) \\ \vec{p}_4(1) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ -2 & 2 & 0 & 0 & 0 & 0 \\ 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 & 0 \\ 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{2} & \frac{1}{2} & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \vec{r}_0 \\ \vec{r}_1 \\ \vec{r}_2 \\ \vec{r}_3 \\ \vec{r}_4 \\ \vec{r}_5 \end{bmatrix}$$

<https://www.ibiblio.org/e-notes/Splines/b-int.html>

U dobivenom sustavu potrebno je invertirati matricu (ili riješiti LU dekompozicijom, Gaussovom eliminacijom) i odrediti točke r_i kontrolnog poligona.

Točke r_i određuju kontrolni poligon tako da aproksimacijska krivulja prolazi zadanim p_i točkama.

Daljnje proširenje **B**-krivulja → NURBS <http://nurbscalculator.in/>

- **NeUniformne** <https://www.ibiblio.org/e-notes/Splines/none.html>
- **Racionalne** <https://www.ibiblio.org/e-notes/Splines/nurbs.html>

B – površine

<https://www.ibiblio.org/e-notes/Splines/bezier3d.html>

<https://www.ibiblio.org/e-notes/Splines/animation.html>

Zašto B-spline

- krivulje, površine
- interpolacija
- kontrola oblika
- obrada površina CAD



- <https://www.ibiblio.org/e-notes/webgl/deflate/zmaj.html>
- <https://sketchfab.com/models/72b5d3713bda4dc08f6b6f20fb2e6b7c> (ModelInspector/Wireframe)

SPLAJN CATMULL-ROMA

Kada želimo da krivulja glatko interpolira niz 3D točaka koje interaktivno zadajemo, možemo koristiti ovu interpolaciju.

- interpolira niz točaka P_1 do P_{m-1} na osnovi sekvence P_0 do P_m .

Vektor tangente u točki P_i paralelan je s dužinom $P_{i-1} P_{i+1}$.

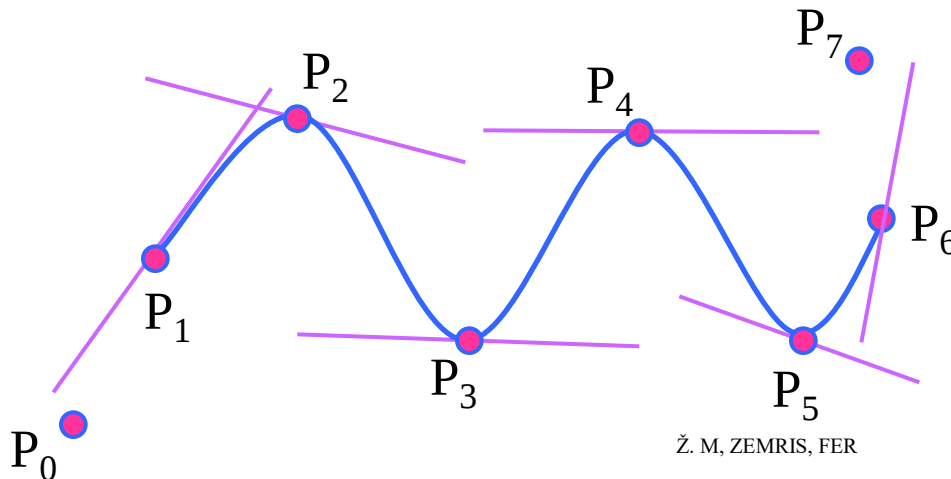
Nema svojstvo konveksne ljuske.

$$\vec{p}_i(t) = \begin{bmatrix} t^3 & t^2 & t & 1 \end{bmatrix} \frac{1}{2} \begin{bmatrix} -1 & 3 & -3 & 1 \\ 2 & -5 & 4 & -1 \\ -1 & 0 & 1 & 0 \\ 0 & 2 & 0 & 0 \end{bmatrix} \begin{bmatrix} \vec{r}_{i-3} \\ \vec{r}_{i-2} \\ \vec{r}_{i-1} \\ \vec{r}_i \end{bmatrix}$$

http://www.hiwis.com/demo/three/examples/webgl_geometry_spline_editor.html

http://threejs.org/examples/webgl_geometry_spline_editor

<https://curvyeditor.com/demo> https://threejs.org/examples/webgl_modifier_curve.html



1.2.2 Kontrola vremena duž putanje

- kontrola gibanja duž krivulje
 - oblik krivulje
 - definiranje brzine kretanja duž krivulje
- zadavanje oblika krivulje
 - kontrolne točke – kada uvrstimo u kubnu krivulju Beziera (ili B-splajn) dobit ćemo

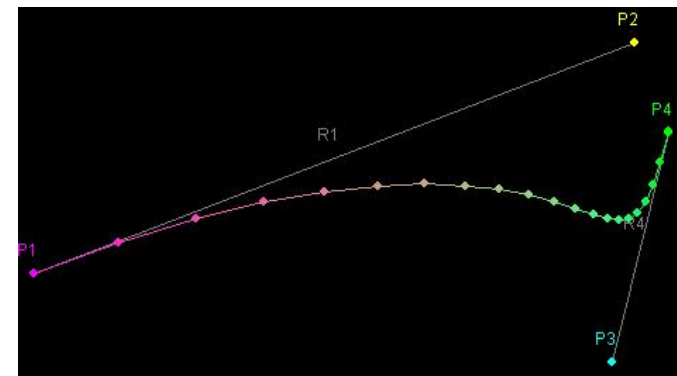
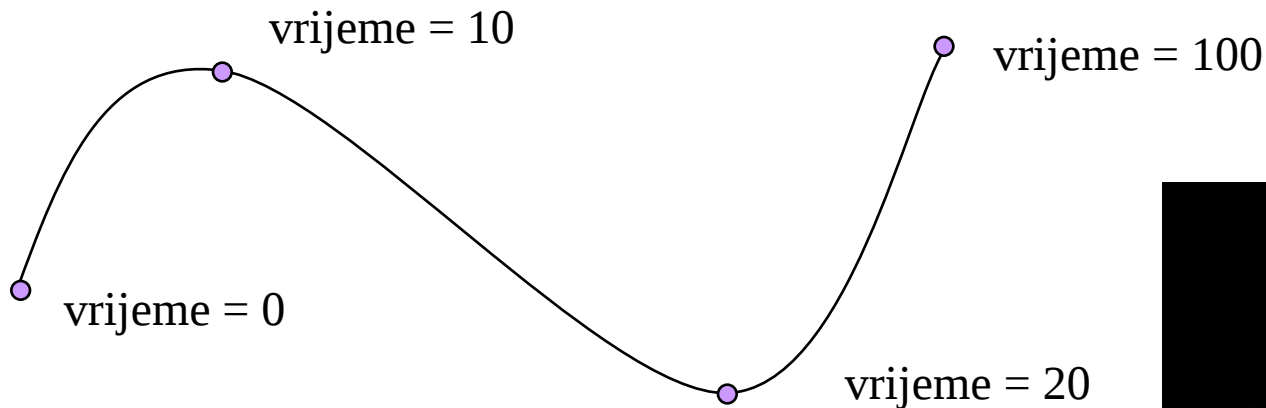
$$\vec{p}(u) = \sum_{i=0}^n \vec{r}_i N_{i,k}(u)$$
$$\vec{p}(u) = \begin{bmatrix} p_x(u) \\ p_y(u) \\ p_z(u) \end{bmatrix} = \begin{bmatrix} a_x u^3 + b_x u^2 + c_x u + d_x \\ a_y u^3 + b_y u^2 + c_y u + d_y \\ a_z u^3 + b_z u^2 + c_z u + d_z \end{bmatrix}$$

- kontrola brzine kretanja objekta duž krivulje
 - ako mijenjamo vrijednosti uzlova u utjecat ćemo na oblik krivulje
 - veza između parametra u i pripadne udaljenosti duž krivulje (arc length)
 - prvi korak – pomaci po krivulji za jednolike korake
 - reparametrizacija duljinom luka (parameterization by arc length)
 - kontrola puta (duljine), brzine, akceleracije

Parametrizacija duljinom luka (arc length)

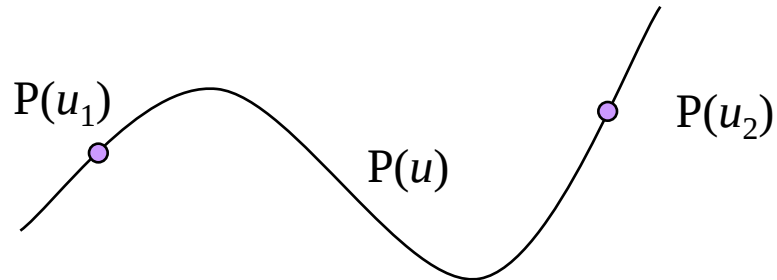
- nezavisna kontrola oblika krivulje i brzine kretanja objekta duž krivulje
- izravno korištenje parametra u kako bi dobili promjene u vremenu često daje neželjene brzine kretanja
- brzine kretanja duž pojedinih segmenata su različite

<https://webglfundamentals.org/webgl/lessons/resources/bezier-curve-diagram.html?maxDepth=0&showCurve=true&showTolerance=true> / -duljine segmenata za uniformnu promjenu t



Određivanje duljine luka (engl. arc length)

- analitički
- procjena



- analitički:

$$s = \int_{u_1}^{u_2} |dP(u)/du| du$$

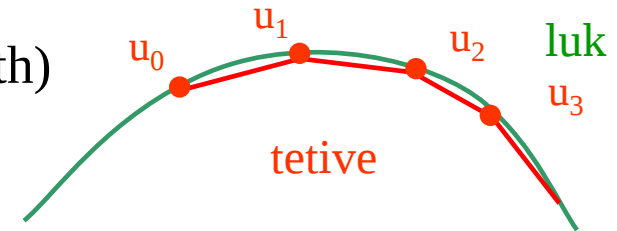
$$|dP(u)/du| = \sqrt{(dp_x(u)/du)^2 + (dp_y(u)/du)^2 + (dp_z(u)/du)^2}$$

$$(dp_x(u)/du)^2 = (3a_x u^2 + 2b_x u + c_x)^2 = Au^4 + Bu^3 + Cu^2 + Du + E$$

- kada odredimo promjenu parametra tako da se pomičemo po krivulji za jednolike vremenske razmake lako možemo kontrolirati brzinu i akceleraciju

- procjena duljine luka
 - zbrajanje duljina tetiva (engl. chord length)

- https://mathinsight.org/applet/helix_arc_length
- ([arclength.mov](#))
- <http://verbnurbs.com/examples/curveDivide.html>



- izrada tablice duljine krivulje
 - $G(u)$ kumulativna vrijednost duljina tetiva
 - NPR početna tablica (jednolika promjena parametra):

Parametar u	$G(u)$	- linearno (kubno) možemo interpolirati
0.00	0.000	za zadani u naći pripadni $G(u)$
0.05	0.080	ili obrnuto za $G(u)$ pripadni u
0.10	0.150	
0.15	0.230	$G(u) = G(u_i) + \frac{u - u_i}{u_{i+1} - u_i} (G(u_{i+1}) - G(u_i))$
0.20	0.320	
...	...	$u = 0.18,$
0.95	0.998	$G(0.18) = 0.23 + \frac{0.18 - 0.15}{0.20 - 0.15} (0.32 - 0.23)$
1.00	1.000	

- kontrola brzine

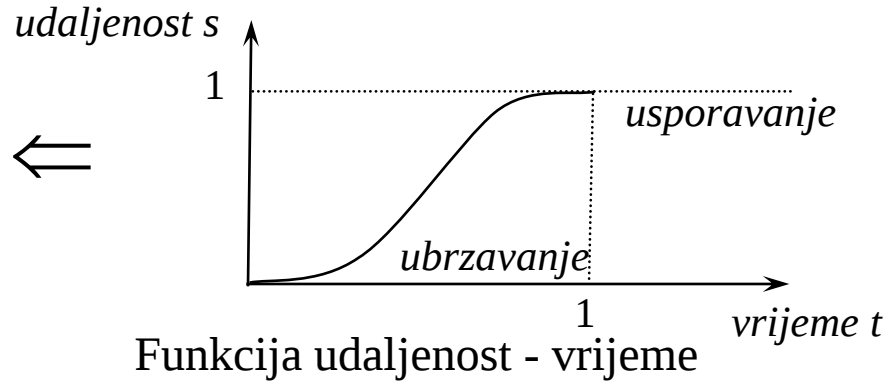
- praćenje putanje konstantnom brzinom, za jednolike razmake po krivulji $s = G(u_{i+1}) - G(u_i)$, naći pripadne vrijednosti u_i
- $\Rightarrow$ kontrolirano kretanje krivuljom
- često je potrebno postupno ubrzavanje/usporavanje (engl. ease-in/ease-out)

tablica duljine krivulje

u	$G(u)$	.
...		



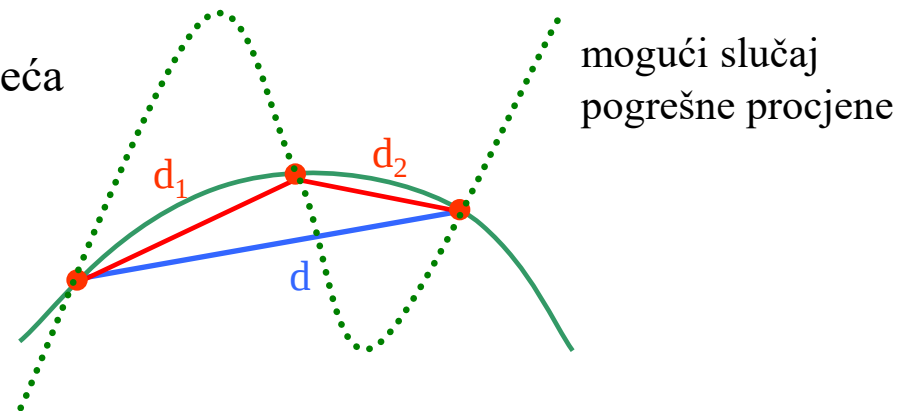
u



točka na krivulji u prostoru za vrijeme t
 Prostorna krivulja

- duljina dijela krivulje $s = G(u_2) - G(u_1)$
- prednosti tablične procjene duljine luka
 - intuitivno,
 - jednostavno za implementaciju,
 - brzo za izvođenje
- obično se normira tako da je parametar $0 \leq u \leq 1$ i $0 \leq G(u) \leq 1$
- poboljšanja
 - više uzoraka interpolacija bolja od linearne (ne osigurava manju pogrešku)
 - adaptivan pristup gustoći uzoraka

NPR: više uzoraka gdje je pogreška veća
 ako je $(d_1 + d_2) - d > \varepsilon$
 dalje se dijeli



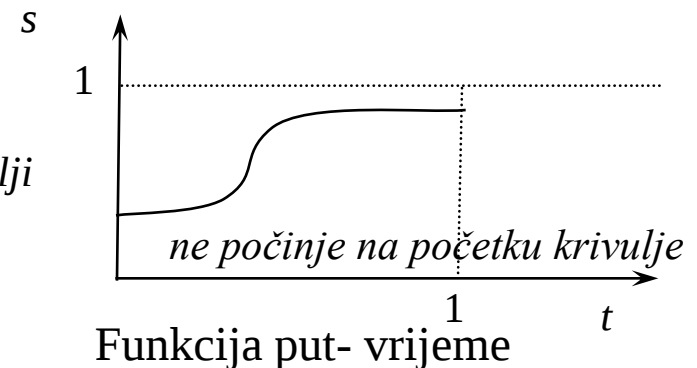
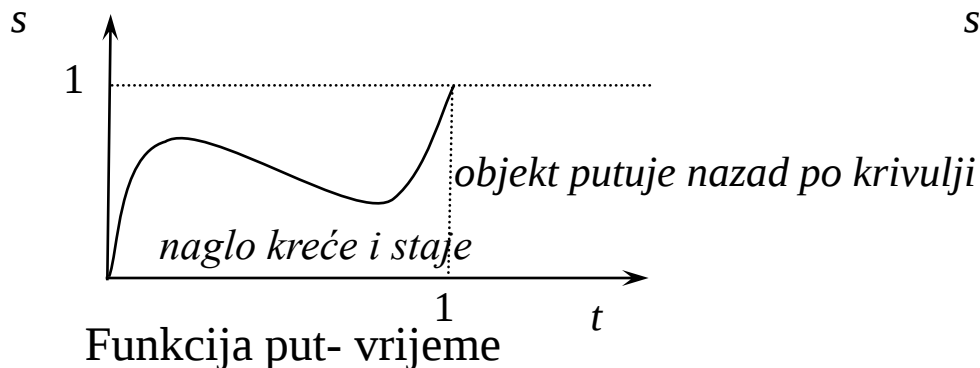
početno se načini grublja linearna podjela,
 a dalje se binarno dijeli, ako postoje infleksije moguće su pogrešne procjene

Na ovaj način postigli smo razdvojenu kontrolu pozicije na krivulji i vremena

- KADA - funkcija udaljenost – vrijeme $\rightarrow$ za neki trenutak t odredimo s
- GDJE - prostorna krivulja $\rightarrow$ iz tablice pomoću s (tj. $G(u)$) odredimo u i točku krivulje <https://nccastaff.bournemouth.ac.uk/jmacey/WebGL/Interpolations/>

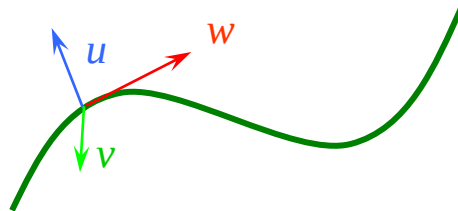
- funkcija udaljenost - vrijeme

- može se zadati eksplicitno (krivuljom)
- analitički – preko krivulje brzine i/ili akceleracije (uz pripadne međuzavisnosti)
- ova funkcija je obično
 - monotonno rastuća
 - kontinuirana
 - $s(0.0) = 0.0$, $s(1.0) = 1.0$,



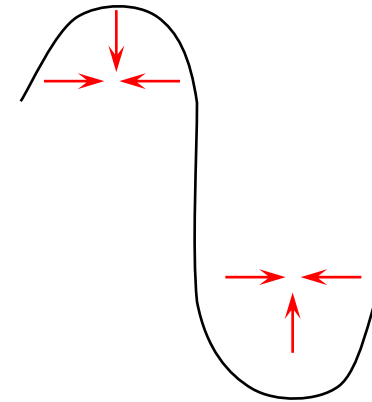
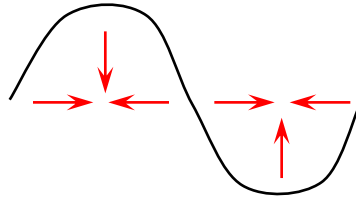
1.2.3 Praćenje putanje (engl. path following)

- objekt (kamera) prati putanju
- ukoliko su zadani položaji dobiveni uzorkovanjem potrebno je načiniti filtriranje – izgladiti putanju
- putanja po površini terena – složeniji problem
 - orijentacija duž putanje
 - lokalni koordinatni sustav (u , v , w) Frenetov okvir (engl. Frenet frame)
 - w smjer tangente $w = P'(s)$
 - u smjer normale $u = P'(s) \times P''(s)$
 - v smjer binormale (up-vector) $v = w \times u$



- <https://avikdas.com/2020/09/08/rendering-curves-in-3d.html>
- http://pages.cs.wisc.edu/~lizzy/mrdoob-three.js-ef5f05d/examples/webgl_geometry_extrude_splines.html
- <https://mattbradley.github.io/dash/>

- problemi pri korištenju Frenet-ovog okvira
 - moguć diskontinuirani skok (druge derivacije)
 - <https://avikdas.com/2020/09/08/rendering-curves-in-3d.html> <https://pomax.github.io/bezierinfo/#pointvectors> (15)



- ravni segment nema zakrivljenost $P''(s) = 0$
- rezultirajuća gibanja često su ekstremna, izgledaju neprirodno (normala naglo mijenja smjer) iako je matematički egzaktno > tada se radi RM frame (engl. Rotation Minimizing Frame)
- tangenta nije dobar izbor za usmjeravanje pogleda
- u – može su iskoristiti u simulaciji djelovanja sile u zavoju
- <https://zz85.github.io/threejs-path-flow/flow.html>
- <https://stemkoski.github.io/MathBox/graph3d-curve.html> radius

- putanja kamere
 - potrebno je postaviti gledište COI (engl. center of interest)
 - postavljanje jedne ili više točaka
 - COI određuje smjer pogleda – vektor w
 - $u = w \times y_{os}$
 - $v = w \times u$
 - postavljanje pomične točke COI na putanji ispred kamere
 - $P(s)$ – putanja, $P(s + \delta s)$ – gledište
 - potrebna je parametrizacija duljinom luka inače će se mijenjati udaljenost do gledišta https://threejs.org/examples/webxr_vr_rollercoaster
 - definiranje posebne putanje za COI
 - veća kontrola
 - automatsko upravljanje kamerom
 - pogled “preko ramena” <https://stemkoski.github.io/Three.js/Viewports-Dual.html>
 - važno je uhvatiti središte zbivanja
 - kamera:
 - https://threejs.org/examples/webgl_geometry_extrude_splines (Look Ahead, Camera Helper)
 - <https://www.benno-lueders.de/ThreeJSRollercoaster/index.html> s

1.3 Animacija 6DOF (pozicija i orijentacija)

- reprezentacija proizvoljne pozicije i orijentacije
 - 6DOF 6 stupnjeva slobode (engl. Six Degrees of Freedom)
 - tri koordinate u prostoru - pozicija x, y, z
 - rotacijski položaj objekta - orijentacija α, β, γ

Pozicija

- zadavanje i praćenje putanje

Bezier-ova krivulja opisuje segment – povezivanje segmenata $\rightarrow$ B - krivulja

Orijentacija

- pogreške pri rotaciji
- orijentacija objekta (interpolacija između različitih orijentacija)

(vrlo često – orijentacija objekta/ruke, kamere, normale, 3D koordinatni sustav)

A) Eulerovi kutovi (sustav objekta)

B) rotacijska matrica

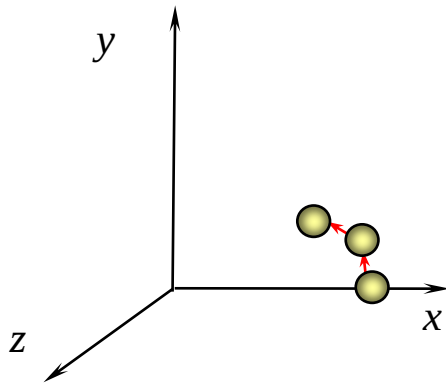
C) os rotacije i kut rotacije

D) kvaternioni (engl. quaternions)

1.3.1 Pogreške pri rotaciji

- tri načina rotacije oko osi y u koracima za po 5° ,
I – jedinična matrica R_{dy} - matrica rotacije oko y za 5°

1. pogreška zaokruživanja akumulira se tijekom rotacije u koordinatama
gubi se koplanarnost - sjenčanje, deformacije

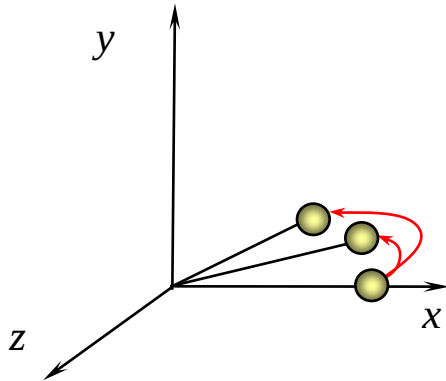


ponavljaj {
 za svaku točku {
 $P = P * R_{dy};$
 }
 crtaj (P);
}

oko y osi za kut β suprotno smjeru kazaljke na satu

$$R_y = \begin{bmatrix} \cos \beta & 0 & -\sin \beta & 0 \\ 0 & 1 & 0 & 0 \\ \sin \beta & 0 & \cos \beta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

2. modificiramo transformacijsku matricu koja transformira objekt iz prostora objekta u prostor scene



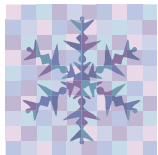
```
R = I
ponavljaj {
    za svaku točku {
         $P' = P * R$  ;
    }
    crtaj (P');
     $R = R_{dy} R$ 
}
```

<http://bigwww.epfl.ch/demo/jshiftlinear/start.php>

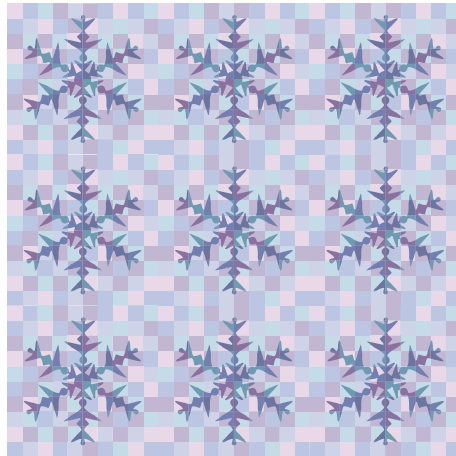
3. u 2. postupku greška se akumulira u transformacijskoj matrici

- pojava smika
- popravljjanje ortonormiranosti matrice – greška se distribuira
 - prvi red se normira
 - vektorski produkt 1. i 2. retka se normira i smjesti u 3. redak
 - vektorski produkt 3. i 1. retka se normira i smjesti u 2. redak
- postupak ortonormiranja Gram–Schmidt

motif()



D



0

L

```
glPushMatrix();  
glLoadIdentity();  
  
for (int row = 0; row < 3; row++)  
{  
    glPushMatrix();  
  
    for (int col = 0; col < 3; col++)  
    {  
        motif();  
        translate2D (L, 0);  
    }  
  
    glPopMatrix();  
    translate2D (0, D);  
}  
  
glPopMatrix();
```

Primjer očuvanja koordinata primjenom stoga kako bi se izbjeglo nakupljanje pogreške.

Na ovaj način koordinate objekta se čuvaju u lokalnom koordinatnom sustavu, a u sustavu scene se ostvaruje prikaz transformiranjem tih koordinata.

1.3.2 Rotacija

- devet članova koji određuju rotaciju (9DOF)
 - možemo li odrediti iz matrice o kojoj rotaciji (transformaciji) se radi?
 - zadavanje rotacije nije jednoznačno
- ortonormiranost (rigid body)
 - po stupcima (redcima) duljina je **1** (ako skaliramo onda je $s_{x,y,z}$) jer je $\sin^2\alpha + \cos^2\alpha = 1$

$$\sqrt{r_{00}^2 + r_{01}^2 + r_{02}^2} = 1$$

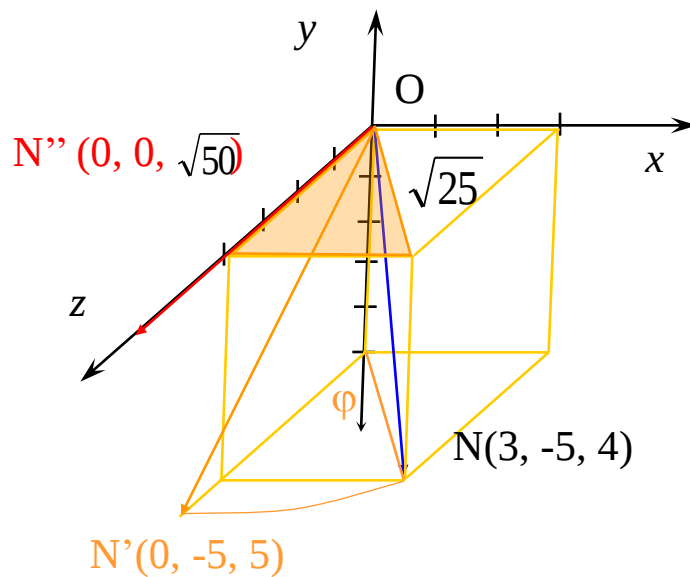
- međusobno okomiti
(9DOF - 3DOF - 3DOF = 3DOF)
(ako nema smika i skaliranja)

$$R = \begin{bmatrix} r_{00} & r_{01} & r_{02} & 0 \\ r_{10} & r_{11} & r_{12} & 0 \\ r_{20} & r_{21} & r_{22} & 0 \\ t_x & t_y & t_z & 1 \end{bmatrix}$$

$$\begin{bmatrix} r_{00} & r_{01} & r_{02} \end{bmatrix} \times \begin{bmatrix} r_{10} & r_{11} & r_{12} \end{bmatrix} = \begin{bmatrix} r_{20} & r_{21} & r_{22} \end{bmatrix}$$

Orijentacija objekta u sustavu scene

Npr. A:



Rotacija $ON \rightarrow ON''$ (up vektor u smjeru y)

$$O = (0 \ 0 \ 0 \ 1) \quad N = (3 \ -5 \ 4 \ 1)$$

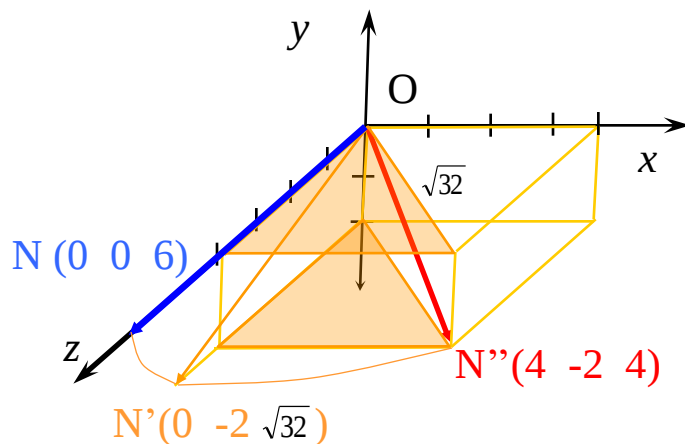
1. Rotacija oko y osi $ON \rightarrow ON'$

$$R_y = \begin{bmatrix} \frac{4}{\sqrt{25}} & 0 & \frac{3}{\sqrt{25}} & 0 \\ 0 & 1 & 0 & 0 \\ -\frac{3}{\sqrt{25}} & 0 & \frac{4}{\sqrt{25}} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad N' = \begin{pmatrix} 0 & -5 & \frac{25}{\sqrt{25}} & 1 \end{pmatrix}$$

2. Rotacija oko x osi $ON' \rightarrow ON''$

$$R_x = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{\sqrt{25}}{\sqrt{50}} & -\frac{5}{\sqrt{50}} & 0 \\ 0 & \frac{5}{\sqrt{50}} & \frac{\sqrt{25}}{\sqrt{50}} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad N'' = \begin{pmatrix} 0 & 0 & \sqrt{50} & 1 \end{pmatrix}$$

Npr. B:



Rotacija **ON** → **ON''** (up vektor u smjeru y)

$$O = (0 \ 0 \ 0 \ 1) \quad N = (0 \ 0 \ 6 \ 1)$$

1. Rotacija oko x osi **ON** → **ON'**

$$R_x = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \frac{\sqrt{32}}{6} & \frac{2}{6} & 0 \\ 0 & -\frac{2}{6} & \frac{\sqrt{32}}{6} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad N' = (0 \ -2 \ \sqrt{32} \ 1)$$

2. Rotacija oko y osi **ON'** → **ON''**

$$R_y = \begin{bmatrix} \frac{4}{\sqrt{32}} & 0 & -\frac{4}{\sqrt{32}} & 0 \\ 0 & 1 & 0 & 0 \\ \frac{4}{\sqrt{32}} & 0 & \frac{4}{\sqrt{32}} & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad N'' = (4 \ -2 \ 4 \ 1)$$

Ovo je inverzan postupak

A) Eulerovi kutovi

- svaka rotacija može biti ostvarena kao kompozicija tri rotacije oko tri primarne osi R_x , R_y , R_z (lokalnog ili globalnog sustava)
- Eulerov teorem: proizvoljna orijentacija može biti postignuta kao tri uzastopne rotacije oko primarnih osi, ako proizvoljno možemo odabrati redoslijed rotacija
- problemi i nedostaci takvog načina izvođenja rotacije
 - interpolacija između dva položaja u postupaka animacije
 - važan je redoslijed rotacija
 - zastoj osi - Gimbal-lock (kada se dvije od tri osi poklope)

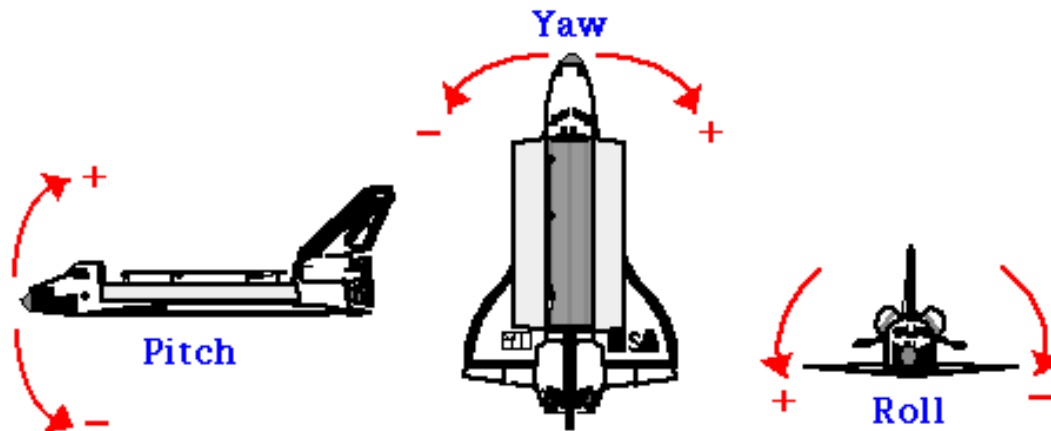
Terminologija iz aeronautike

– rotacije u koordinatnom sustavu objekta

DUBINA

SMJER

NAGIB



Zastoj žiroskopskih osi (engl. Gimbal lock)

- engl. gimbal - prsten u žiroskopu
- poravnavanje osi uzrokuje gubitak stupnja slobode (DOF) – rotacija oko prve i treće os su iste

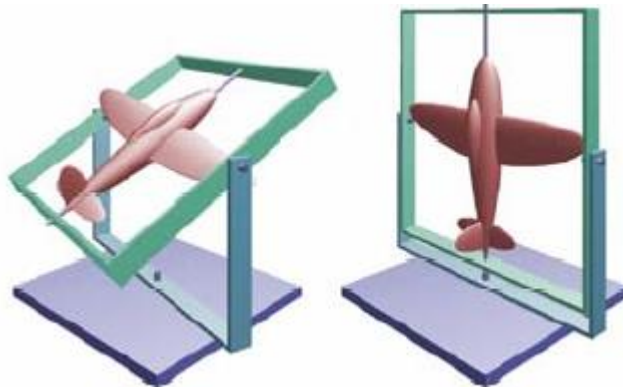
Moguće rotacije

1. rotacija oko osi fiksnog koordinatnog sustava,
2. rotacija oko rotiranog sustava
zadan redoslijed rotacija npr: x, y, z

- Gimbal lock ($y - 90$): http://www.realtimerendering.com/udacity/?load=demo/unit4-euler_angles.js

(A Fourth Gimbal for Christmas)

- <https://history.nasa.gov/alsj/gimbals.html>



Ž. M. ZEMRIS, FER

[NPR](#) – demo Euler



kada se zeleni i ljubičasti podudare
ljubičasta i plava rotacija su iste
1-47

Zastoj žiroskopskih osi (engl. gimbal lock)

- ako je redoslijed rotacija zadan, za određene položaje može doći do problema
- problem interpolacije zadanih orijentacijskih položaja ne može se načiniti kvalitetno (za opći slučaj) korištenjem uzastopnih rotacija oko tri primarne osi

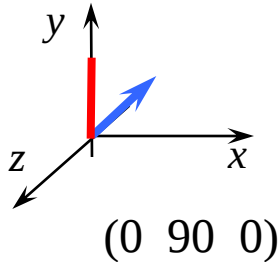
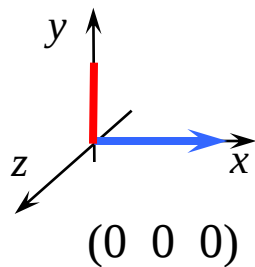
<http://www.cs.vu.nl/~eliens/papers/media/@vr-vrml-math-FlyDemo.wrl> [Demo1](#)

- dodavanje još jednog stupnja slobode DOF, uz aktivno vođenje vanjskog prstena tako da bude okomit na unutarnji (inače bi se opet mogli podudariti)
- rotacija zadana preko osi i kuta (4DOF)
- rotacija zadana kvaternionima

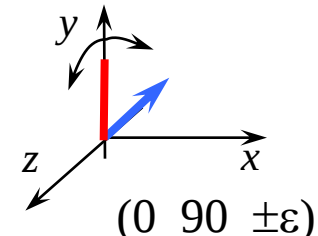
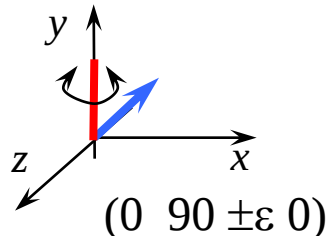
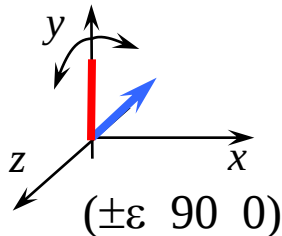


Rotacija oko osi fiksnog koordinatnog sustava scene

- fiksni redoslijed tri rotacije koje se primjenjuju Npr. $x y z$
- problem zastoja osi (engl. gimbal lock) - dvije od tri osi se poklope

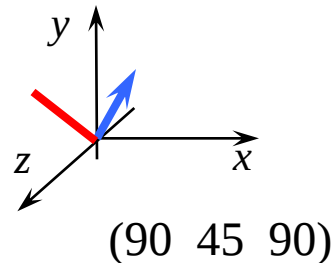
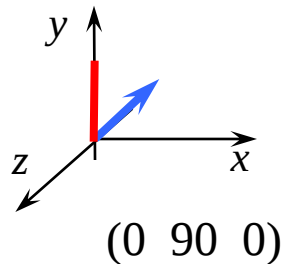


rotacija oko x i z daju isto,
ako je oko y osi 90



Ista rotacija može se zapisati na više različitih načina npr. $0 \ 90 \ 10$ je isto što i $350 \ 90 \ 0$

- Npr. rotacija između dva položaja
(0 90 0) i (90 45 90)
 - položaj (90 45 90) neintuitivan je u ostvarivanju
 - interpolacija između zadanih položaja – između zadanih kutova ne će dati dobar rezultat (0 → 90 90 → 45 0 → 90) za pola je (45 67.5 45)



- postavi yaw ← → na 90, roll Insert/PageUp i pitch ↑↓ = daju isto
- <http://www.gamedev.net/reference/programming/features/qpowers/CameraEuler.zip> EULER
- <http://www.gamedev.net/reference/programming/features/qpowers/CameraQuat.zip> QUAT
- http://www.iwant2study.org/lookangejss/math/ejss_model_glimbal8wee02/glimbal8wee02_Simulation.xhtml P90

- na primjer rotacije oko $x(\varphi)$, $y(\theta)$, $z(\psi)$ daje ukupnu matricu rotacije:

$$R = \begin{bmatrix} \cos(\theta) \cos(\psi) & -\cos(\varphi) \sin(\psi) + \sin(\varphi) \sin(\theta) \cos(\psi) & \sin(\varphi) \sin(\psi) + \cos(\varphi) \sin(\theta) \cos(\psi) & 0 \\ \cos(\theta) \sin(\psi) & \cos(\varphi) \cos(\psi) + \sin(\varphi) \sin(\theta) \sin(\psi) & -\sin(\varphi) \cos(\psi) + \cos(\varphi) \sin(\theta) \sin(\psi) & 0 \\ -\sin(\theta) & \sin(\varphi) \cos(\theta) & \cos(\varphi) \cos(\theta) & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- za rotaciju φ , $\theta=\pi/2$, ψ matrica je (gimbal lock):

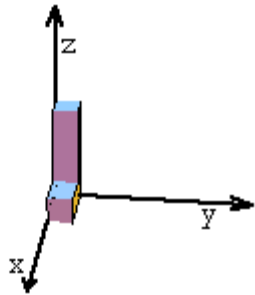
$$R = \begin{bmatrix} 0 & \sin(\varphi - \psi) & \cos(\varphi - \psi) & 0 \\ 0 & \cos(\varphi - \psi) & -\sin(\varphi - \psi) & 0 \\ -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

- postoji ovisnost samo o kutu $(\varphi - \psi)$, odnosno o jednom a ne dva parametra

Dvije konvencije za Eulerove kutove obzirom na redoslijed rotacija:

- Rotacija oko osi
 - koordinatnog sustava
 - Eulerovi kutovi (α , β , γ)
-
- oko z - osi za $(-30)^\circ$, α
 - oko y - osi za $(-60)^\circ$, β
 - oko z - osi za $(-45)^\circ$, γ

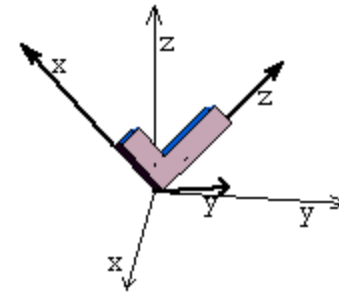
Starting position



Rotacija oko osi
koordinatnog sustava tijela
Eulerovi kutovi (α , β , γ)

1. oko z_L - osi za 30° , α
2. oko y_L - osi za 60° , β
3. oko z_L - osi za 45° , γ

Starting position



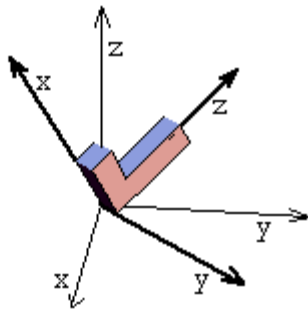
rotacija se može izvoditi oko

- fiksnih osi $R_x(\alpha)$
- rotiranih osi $R'_x(\alpha)$ (npr. rotiramo oko x osi, tako dobijemo rotiranu y os oko koje zatim rotiramo ...)
- to su ekvivalentni sustavi uz obrnuti redoslijed rotacija

$$R'_y(\beta) R'_x(\alpha) = (R_x(\alpha) R_y(\beta) R_x(-\alpha)) R_x(\alpha) = R_x(\alpha) R_y(\beta)$$

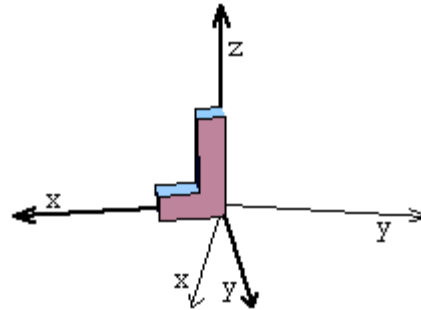
α se mijenja (ROLL)
NAGIB

1. $0^\circ < \alpha \leq 360^\circ$,
2. $\beta = -60^\circ$,
3. $\gamma = -45^\circ$.



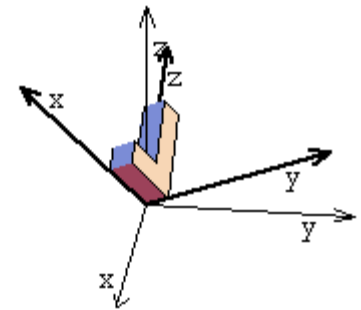
β se mijenja (PITCH)
DUBINA

1. $\alpha = -30^\circ$,
2. $0^\circ < \beta \leq 360^\circ$,
3. $\gamma = -45^\circ$.



γ se mijenja (YAW)
SMJER

1. $\alpha = -30^\circ$,
2. $\beta = -60^\circ$,
3. $0^\circ < \gamma \leq 360^\circ$.



B) Rotacijska matrica

- želimo na jedinstven način zapisati (opisati) orijentaciju objekta – čvrstog tijela
 - Točku $V_i(x,y)$ rotiramo oko ishodišta za kut φ matricom $V'_i = V_i R(\varphi)$.
- Točka čini objekt koji se nalazi u lokalnom koordinatnom sustavu objekta.

- Promatramo rotaciju lokalnog koordinatnog sustava a time i objekta (osi koordinatnog sustava $\vec{x}$, $\vec{y}$)

$$\vec{x} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \vec{y} = \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

- ako rotiramo jedinične vektore koji određuju osi koord. sustava tada te vektore množimo inverznom matricom:

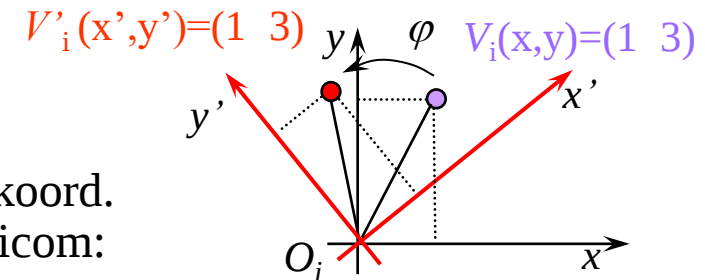
$$\vec{x}' = R^{-1}(\varphi) \vec{x}, \quad \vec{y}' = R^{-1}(\varphi) \vec{y}, \quad (\text{za 2D vrijedi } R^{-1}(\varphi) = R(-\varphi))$$

$$[\vec{x}' \quad \vec{y}'] = R^{-1}(\varphi) [\vec{x} \quad \vec{y}] = R^{-1}(\varphi) \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = R^{-1}(\varphi)$$

- osi rotiranog koordinatnog sustava izravno određuju stupce inverzne rotacijske matrice

(DCM – direction cosine matrix) zove se kosinusna jer kosinus kuta između rotiranih jediničnih vektora i pripadnih izvornih određuje elemente DCM matrice

- komponente rotiranih vektora projekcije su na pojedine osi



Primjer:

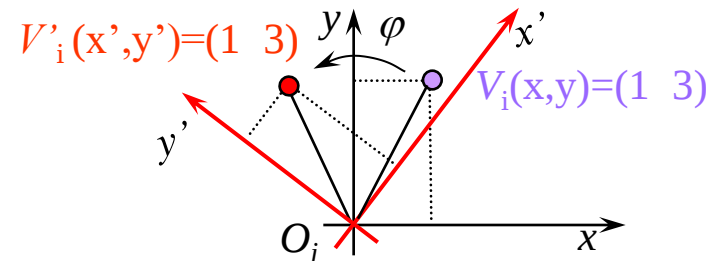
Znamo osi zarotiranog koordinatnog sustava $\bar{x}' \ \bar{y}'$ (lokalni koordinatni sustav objekta).

Želimo odrediti matricu rotacije kojom trebamo pomnožiti točke objekta tako da objekt bude rotiran.

(promatrano iz koordinatnog sustava objekta, koordinate će ostati iste) Neka je:

$$\bar{x}' = \begin{bmatrix} 1 \\ 2 \\ 1 \\ 2 \end{bmatrix}, \quad \bar{y}' = \begin{bmatrix} -1 \\ 2 \\ 1 \\ 2 \end{bmatrix} \quad \text{Prvo vektore normiramo} \quad \bar{x}' = \begin{bmatrix} \frac{\sqrt{2}}{2} \\ \frac{2}{\sqrt{2}} \\ \frac{2}{2} \end{bmatrix}, \quad \bar{y}' = \begin{bmatrix} -\frac{\sqrt{2}}{2} \\ \frac{2}{\sqrt{2}} \\ \frac{2}{2} \end{bmatrix}$$

$$R^{-1}(\varphi) = [\bar{x}' \ \bar{y}'] = \begin{bmatrix} \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} \\ \frac{2}{\sqrt{2}} & \frac{2}{\sqrt{2}} \\ \frac{2}{2} & \frac{2}{2} \end{bmatrix} \quad R(\varphi) = \begin{bmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \\ \frac{2}{\sqrt{2}} & \frac{2}{\sqrt{2}} \\ -\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} \end{bmatrix}$$

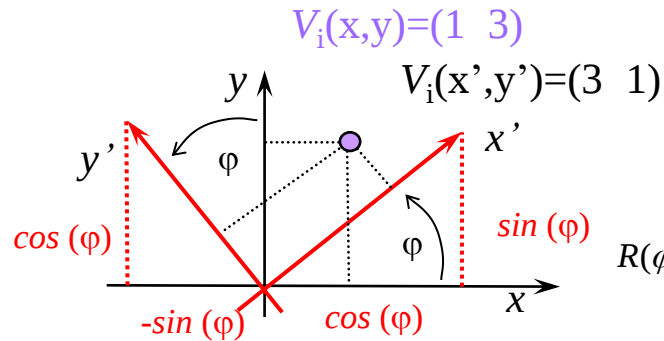


Npr. $V=(1 \ 3)$ interpretiramo kao točku ili kao vektor iz ishodišta $\overrightarrow{O\mathbf{V}}$

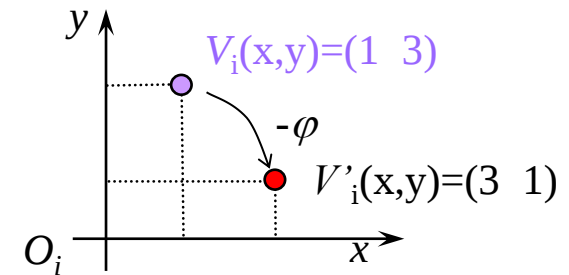
$$\mathbf{V} \cdot R(\varphi) = \begin{bmatrix} 1 & 3 & 1 \end{bmatrix} \cdot \begin{bmatrix} \frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & 0 \\ -\frac{\sqrt{2}}{2} & \frac{\sqrt{2}}{2} & 0 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} -\sqrt{2} & 2\sqrt{2} & 1 \end{bmatrix} \quad R^{-1}(\varphi) \cdot \overrightarrow{O\mathbf{V}} = \begin{bmatrix} \frac{\sqrt{2}}{2} & -\frac{\sqrt{2}}{2} & 0 \\ \frac{2}{\sqrt{2}} & \frac{2}{\sqrt{2}} & 0 \\ 0 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 \\ 3 \\ 0 \end{bmatrix} = \begin{bmatrix} -\sqrt{2} \\ 2\sqrt{2} \\ 0 \end{bmatrix}$$

Rotacijska matrica

- ako je $R(-\varphi)$ matrica rotacije kojom rotiramo vektore, za rotaciju točaka je $V' = VR(\varphi)$
 - promatramo rotaciju koordinatnog sustava za $R(-\varphi)$, a točka V je fiksna
 - rotiramo točku $R(-\varphi)$ a koordinatni sustav je fiksna



$$R(\varphi) = \begin{bmatrix} \cos(\varphi) & \sin(\varphi) & 0 \\ -\sin(\varphi) & \cos(\varphi) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$



Znači ako rotiramo koordinatni sustav za $R(-\varphi)$, koordinate točke $T(x,y)$ koju nismo rotirali ali ju sada promatramo u rotiranom sustavu bit će iste kao da smo rotirali točku za $R(-\varphi)$ i promatramo njene koordinate u izvornom sustavu

Radi se o ortonormiranim matricama pa je $R^{-1} = R^T$

- u 3D prostoru vrijedi isto
$$[\vec{x}' \mid \vec{y}' \mid \vec{z}'] = R^{-1}(\varphi) = \begin{bmatrix} \begin{bmatrix} x_x \\ x_y \\ x_z \end{bmatrix} & \begin{bmatrix} y_x \\ y_y \\ y_z \end{bmatrix} & \begin{bmatrix} z_x \\ z_y \\ z_z \end{bmatrix} \end{bmatrix}$$
- TBN (tangenta, binormala, normala) matrica

Rotacijska matrica

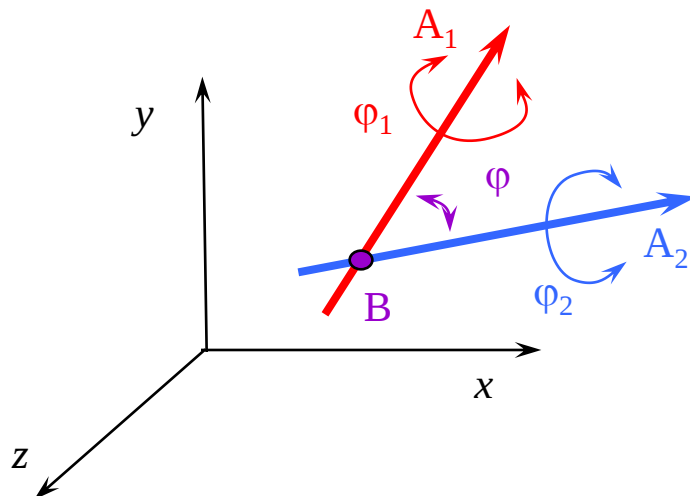
- za DCM (9 elemenata) vrijedi da ima 3 stupnja slobode (CTM - Current Transformation Matrix)
 - to je matrica s realnim koeficijentima, ortogonalna je $A^{-1}=A^T$, $AA^T=I$ (čuva duljinu i kutove, transformacija čvrstog tijela)
 - determinanta matrice je jednaka 1, ako ne postoji skaliranje (determinanta odgovara površini / volumenu) ako je determinanta negativna došlo je do promjene desnog $\leftrightarrow$ lijevog koordinatnog sustava
 - svojstvene vrijednosti matrice su $\{1, \cos(\varphi) \pm i \sin(\varphi)\}$ tj. $\{1, e^{\pm i\varphi}\}$,
 - svojstveni vektor koji odgovara svojstvenoj vrijednosti 1 je os rotacije, (množenje s 1 pri transformaciji će čuvati vrijednosti, npr rotacija oko y osi ne mijenja y vrijed.)
 - stupci inverzne rotacijske matrice su ortonormirani vektori koji razapinju rotirani prostor

<http://www.falstad.com/matrix/>

- problem je ako imamo zadane dvije orijentacije i između njih želimo kontinuirano interpolirati orijentaciju <https://thenumb.at/Exponential-Rotations/>

C) Os rotacije i kut rotacije

- rotacija zadana pomoću osi oko koje se rotira i kuta za koji se rotira



$$B = A_1 \times A_2$$

$$\cos(\varphi) = \frac{A_1 \cdot A_2}{|A_1||A_2|}$$

interpolacija, t parametar:

$$\varphi_t = (1-t)\varphi_1 + t\varphi_2$$

$$A_t = R_B(t\varphi) \cdot A_1$$

- TRANSFORMATION

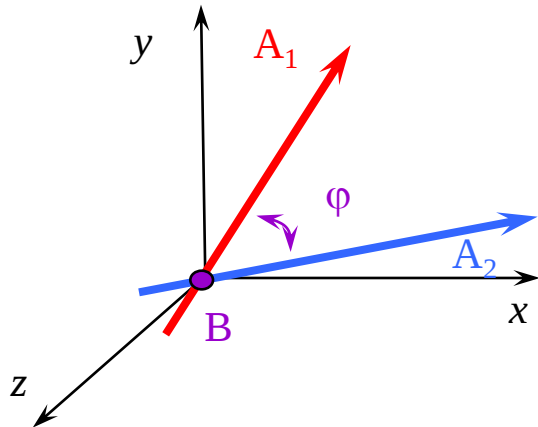
φ_1 rotacija oko osi A_1 , φ_2 rotacija oko osi A_2

$R_B(\varphi)$ rotacija oko osi B za kut φ , B je os okomita na A_1 i A_2

4 DOF, problemi se javljaju u sastavljanju niza rotacija – nije praktično

Os rotacije i kut rotacije

- matrica rotacije za zadani kut φ oko osi $\mathbf{v}$, (ako nam nije dostupan OpenGL)



$$R(\mathbf{v}, \varphi) = \begin{pmatrix} v_x^2 u + c & v_x v_y u - v_z s & v_z v_x u + v_y s \\ v_x v_y u + v_z s & v_y^2 u + c & v_z v_y u - v_x s \\ v_x v_z u - v_y s & v_y v_z u + v_x s & v_z^2 u + c \end{pmatrix}$$

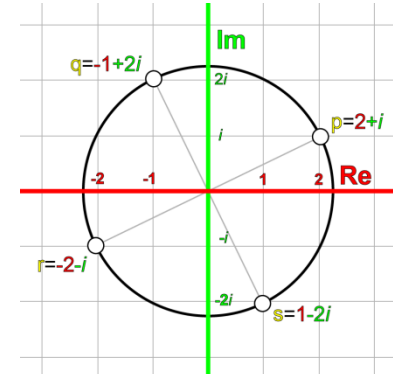
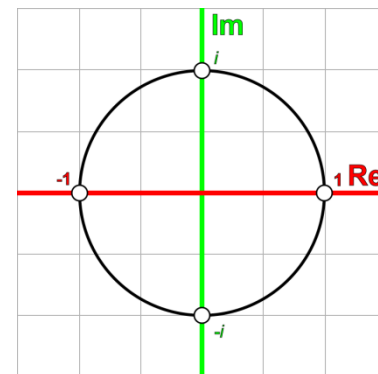
- gdje je $s = \sin(\varphi)$, $c = \cos(\varphi)$, $u = 1 - \cos(\varphi)$
- ili drugi zapis rotacija točke $\mathbf{v}$ za kut φ oko osi $\boldsymbol{\omega}$
- $\mathbf{v}_{rot} = \mathbf{v} \cos(\varphi) + \boldsymbol{\omega} \times \mathbf{v} \sin(\varphi) + \boldsymbol{\omega} (\boldsymbol{\omega} \cdot \mathbf{v}) (1 - \cos(\varphi))$

D) Kvaternioni

- proširenje kompleksnih brojeva (4DOF) $i^2 = -1$, $z = a + bi$, $a, b \in \mathbb{R}$

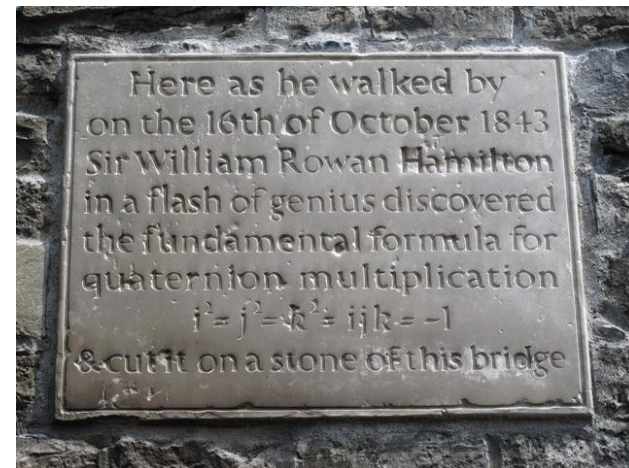
Kompleksni brojevi:

- zbrajanje / oduzimanje $(a_1 + b_1 i) + (a_2 + b_2 i) = (a_1 + a_2) + (b_1 + b_2) i$
 $(a_1 + b_1 i) - (a_2 + b_2 i) = (a_1 - a_2) + (b_1 - b_2) i$
- množenje skalarom $\lambda(a_1 + b_1 i) = \lambda a_1 + \lambda b_1 i$
- konjugirano kompleksni broj $z = a_1 + b_1 i$ $z' = a_1 - b_1 i$
- množenje $(a_1 + b_1 i) \cdot (a_2 + b_2 i) = a_1 a_2 - b_1 b_2 + (a_1 b_2 + b_1 a_2) i$
- uzastopno množenje 1 s i daje niz 1 i -1 $-i$ 1 i -1 $-i$... koji odgovara rotaciji
- uzastopno množenje $2+i$ s i daje niz koji odgovara rotaciji
- polarni zapis kompleksnih brojeva $|z|e^{i\varphi}$



Kvaternioni (engl. quaternions)

- W.R. Hamilton uklesao je osnovnu formulu u kamen na mostu u Irskoj



$$i^2 = j^2 = k^2 = i j k = -1$$

- dio općenitije klase hiperkompleksnih brojeva (tvore grupu)

$$H = \begin{bmatrix} z & w \\ -\bar{w} & \bar{z} \end{bmatrix} = \begin{bmatrix} a+bi & c+di \\ -c+di & a-bi \end{bmatrix}, \quad a, b, c, d \in \mathbb{R}, \quad z, w \in \mathbb{C}$$

- kvaternion se može interpretirati kao skalar plus vektor

$$q = a_1 + a_2 i + a_3 j + a_4 k = (a_1 \mathbf{a})$$

- zadane rotacije izrazimo u obliku kvaterniona, koordinate tijela množimo kvaternionima.

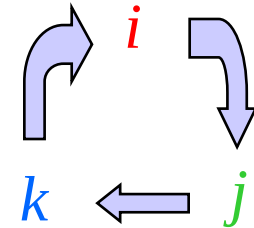
- osnovne operacije s kvaternionima

- množenje osnovnih vektora (slično vektorskom produktu u 3D)

$$i \cdot i = -1 \quad (i = \sqrt{-1}) \quad i \cdot j = -j \cdot i = k$$

$$j \cdot j = -1 \quad j \cdot k = -k \cdot j = i$$

$$k \cdot k = -1 \quad k \cdot i = -i \cdot k = j$$



- konjugirani broj

$$q = a_1 + a_2 i + a_3 j + a_4 k \quad q' = a_1 - a_2 i - a_3 j - a_4 k$$

- modul (apsolutna vrijednost), jedinični kvaternion, inverzni element

$$\|q\| = \sqrt{a_1^2 + a_2^2 + a_3^2 + a_4^2}, \quad q^{-1} = q' / (q \cdot q') = q' / \|q\|^2, \quad \text{za } \|q\| = 1 \Rightarrow q' = q^{-1},$$

- jesu asocijativni, **nisu** komutativni

$$q_1(q_2 q_3) = (q_1 q_2) q_3 \quad q_1 q_2 \neq q_2 q_1$$

- zbroj

$$q_1 + q_2 = (s_1 \quad \mathbf{v}_1) + (s_2 \quad \mathbf{v}_2) = ((s_1 + s_2) \quad [(v_{1x} + v_{2x}) \quad (v_{1y} + v_{2y}) \quad (v_{1z} + v_{2z})])$$

- produkt kvaterniona (dobijemo umnoškom pojedinih komponenti)

$$q_1 q_2 = (s_1 \quad \mathbf{v}_1)(s_2 \quad \mathbf{v}_2) = (s_1 s_2 - \mathbf{v}_1 \cdot \mathbf{v}_2 \quad s_1 \mathbf{v}_2 + s_2 \mathbf{v}_1 + \mathbf{v}_1 \times \mathbf{v}_2)$$

- reprezentacija rotacije oko vektora $\mathbf{u}$ za kut θ kvaternionom:

$$q = (s \quad \mathbf{v}) \quad s = \cos(\theta/2), \mathbf{v} = \hat{\mathbf{u}} \cdot \sin(\theta/2), \|\hat{\mathbf{u}}\| = 1$$

- rotacija točke $\mathbf{P}$, točku predstavimo kvaternionom $p = (0, \mathbf{P})$ (*Sandwiching*)

$$p_{rotirana} = q p q^{-1}, \text{ ako je } q \text{ normiran} \quad q' = q^{-1}$$

- dvije uzastopne rotacije q_1, q_2 :

$$p_{rotirana} = q_2 (q_1 p q_1^{-1}) q_2^{-1} = (q_2 q_1) p (q_1^{-1} q_2^{-1}) = (q_2 q_1) p (q_2 q_1)^{-1}$$

- matrični oblik rotacije $P = [x_p \ y_p \ z_p]$, za kvaternion $q = [w \ (x \ y \ z)]$:

$$P_{rotirana} = P Q_{rot} = \begin{bmatrix} x_p & y_p & z_p & 1 \end{bmatrix} Q_{rot} =$$

$$= P \begin{bmatrix} w^2 + x^2 - y^2 - z^2 & 2xy + 2wz & 2xz - 2wy & 0 \\ 2xy - 2wz & w^2 - x^2 + y^2 - z^2 & 2yz + 2wx & 0 \\ 2xz + 2wy & 2yz - 2wx & w^2 - x^2 - y^2 + z^2 & 0 \\ 0 & 0 & 0 & w^2 + x^2 + y^2 + z^2 \end{bmatrix}$$

- čisti kvaternion: $q = (0 \quad \mathbf{v})$ skalarni dio je 0 $q_1 q_1 = (0 \quad \mathbf{v}_1 \times \mathbf{v}_2)$

- Npr. točku $P = (0 \ 1 \ 1)$ treba rotirati za 90° oko y osi

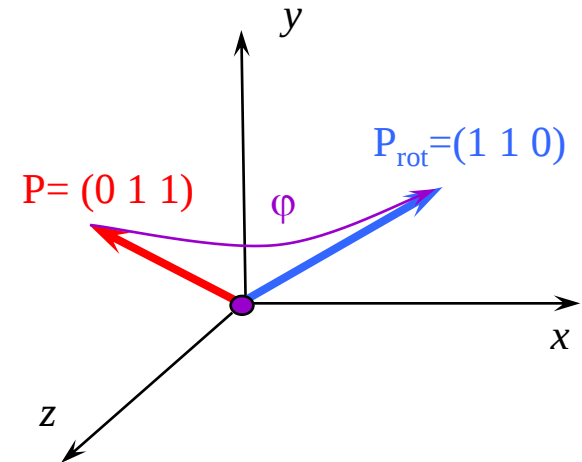
Kvaternion za točku P je $\mathbf{P} = (0 \ 0 \ 1 \ 1)$.

Rotacija za 90° oko y osi predstavljena kvatrnionom je

$$q = \left(\cos(90^\circ / 2) \quad \hat{\mathbf{u}} \sin(90^\circ / 2) \right), \quad \hat{\mathbf{u}} = (0 \ 1 \ 0)$$

$$q = \begin{pmatrix} \frac{\sqrt{2}}{2} & 0 & \frac{\sqrt{2}}{2} & 0 \end{pmatrix}$$

$$q^{-1} = \begin{pmatrix} \frac{\sqrt{2}}{2} & 0 & -\frac{\sqrt{2}}{2} & 0 \end{pmatrix}$$

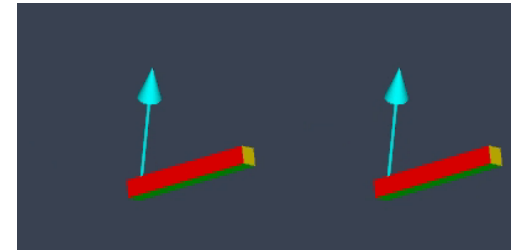


$$p_{rot} = q p q^{-1} = \left(\left(\frac{\sqrt{2}}{2} + 0\mathbf{i} + \frac{\sqrt{2}}{2}\mathbf{j} + 0\mathbf{k} \right) (0 + 0\mathbf{i} + \mathbf{j} + \mathbf{k}) \right) \left(\frac{\sqrt{2}}{2} - 0\mathbf{i} - \frac{\sqrt{2}}{2}\mathbf{j} - 0\mathbf{k} \right)$$

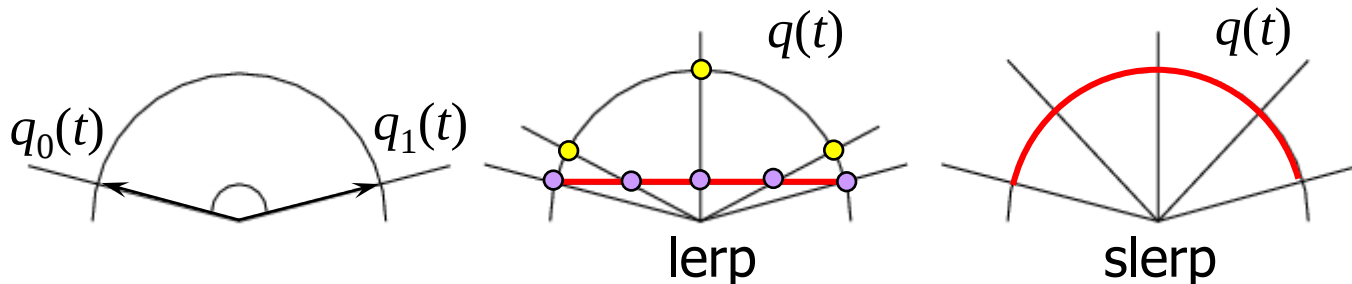
$$p_{rot} = q p q^{-1} = \left(-\frac{\sqrt{2}}{2} + \frac{\sqrt{2}}{2}\mathbf{i} + \frac{\sqrt{2}}{2}\mathbf{j} + \frac{\sqrt{2}}{2}\mathbf{k} \right) \left(\frac{\sqrt{2}}{2} - 0\mathbf{i} - \frac{\sqrt{2}}{2}\mathbf{j} - 0\mathbf{k} \right)$$

$$= (0 + \mathbf{i} + \mathbf{j} + 0\mathbf{k})$$

Interpolacija kvaterniona



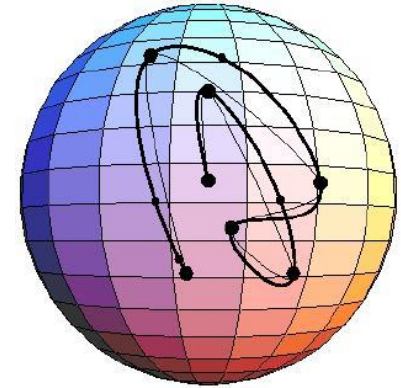
- potrebno je očuvati jediničnu duljinu
- ne interpolira se linearno, linearnom interpolacijom nećemo ostvariti konstantnu brzinu <http://allenchou.net/2018/05/game-math-swing-twist-interpolation-slerp/>
- LERP – linearna interpolacija (engl. **linear interpolation**)
- SLERP – linearna interpolacija po površini sfere (engl. **Spherical linear interpolation**) <https://markaren.github.io/WebGL-demos/test.html>
- dualni kvaternioni – čuvaju poziciju (translaciju) i orijentaciju
- <https://nccastaff.bournemouth.ac.uk/jmacey/WebGL/QuatSlerp/>
- vs. http://www.iwant2study.org/lookangejss/math/ejss_model_glimbal8wee02/glimbal8wee02_Simulation.xhtml
-



- SLERP – linearna interpolacija po površini sfere (engl. **S**pherical **l**inear **i**nter**p**olation)
- potrebno je interpolirati između zadanih položaja na sferi

$$\text{slerp}(q_0, q_1, t) = q(t) = \frac{q_0 \sin((1-t)\theta) + q_1 \sin(t\theta)}{\sin(\theta)}, \quad 0 \leq t \leq 1$$

$$\theta = \cos^{-1}(q_0 \cdot q_1)$$



- pojedine vrijednosti $q(t)$ se moraju normirati
- negativan kvaternion $(-q_1) = (-s -\mathbf{v})$ predstavlja istu orijentaciju
- interpolaciju ćemo obavljati između q_0 i q_1 ili q_0 i $(-q_1)$ ovisno o kutu između tih kvaterniona (ako je $\text{kut}(q_0, q_1)$ pozitivan, taj put je kraći i on se uzima u obzir, ako ne uzima se negativna vrijednost q_1)
- https://threejs.org/examples/webgl_math_orientation_transform
- za ostvarivanje krivulje po sferi, potrebna je interpolacija (SQUAD)

- svojstvo kvaterniona interesantno u primjeni je:

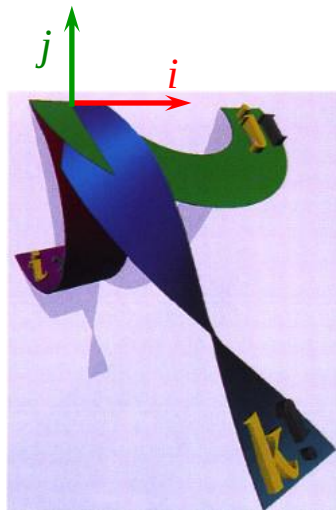
$$t = 0.5, \quad \text{slerp}(q_0, q_1, t = 0.5) = q_0 + q_1$$

- fraktalni skupovi (quaternioni)

- [6_obj1.mpg](#)

- <https://www.shadertoy.com/view/MsfGRr>

Vizualizacija rotacije kvaternionima pomoću pojasa učvršćenog u ishodištu:



- rotacija odgovara množenju s kvaternionom
(npr. rotacija oko osi x (i) za 180° odgovara množenju s $q=(0 \ 1 \ 0 \ 0)$, $q^{-1}=(0 \ -1 \ 0 \ 0)$).
- rotacija za 360° daje početno stanje ali uvrnuto (-1)
- „Dirac's belt trick” – invarijantnost na rotaciju 720° ali nije invarijantno na 360°

[7_belt-trick-win32\application-belt-trick-win32\BeltTrick.win32.exe](#) (srednji miš)

<https://eater.net/quaternions>

- <https://nccastaff.bournemouth.ac.uk/jmacey/WebGL/Bottle/>

REZIME:

A) Eulerovi kutovi 3DOF

- rotacija u sustavu scene $9\text{DOF} - 3\text{DOF} - 3\text{DOF} = 3\text{DOF}$
- zastoj osi

B) Rotacijska matrica

- znamo početni i krajnji položaj – problem kao interpolirati između

C) os i kut 4DOF

- nepraktični za interpolaciju niza zadanih rotacija

D) Kvaternioni 4DOF