PPKE ITK

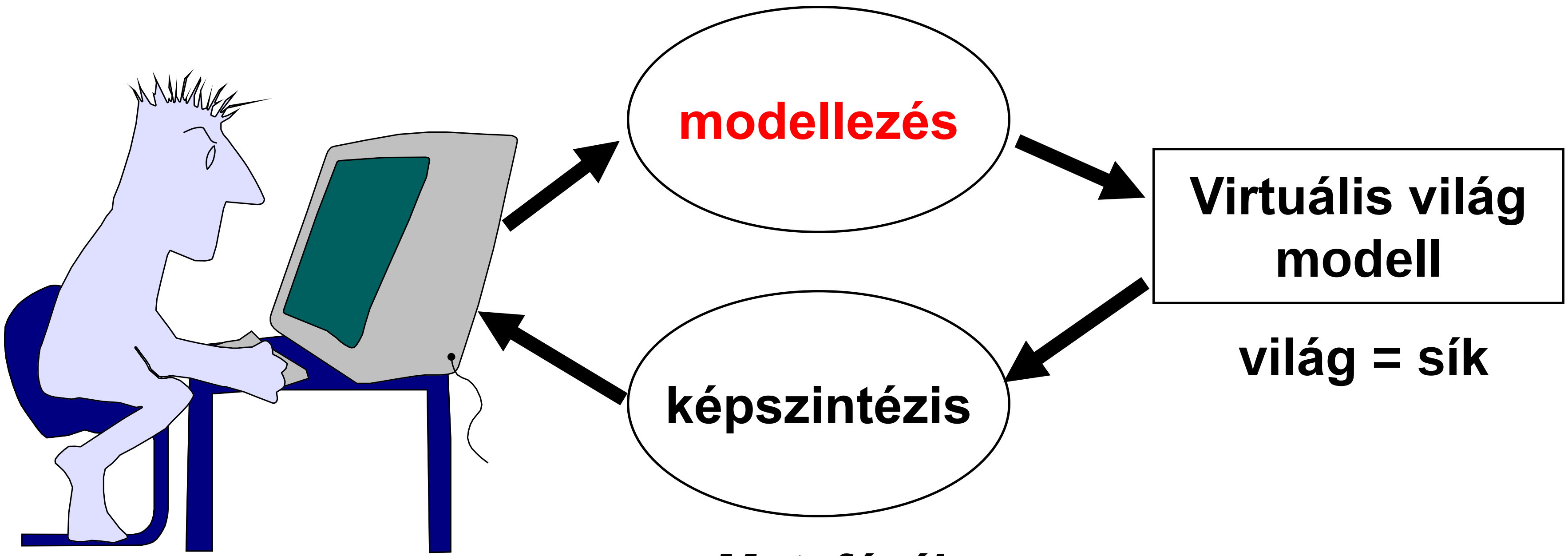
A számítógépes grafika alapjai

Animációk/1

Előadó: Benedek Csaba

Tananyag : Szirmay-Kalos László, Benedek Csaba

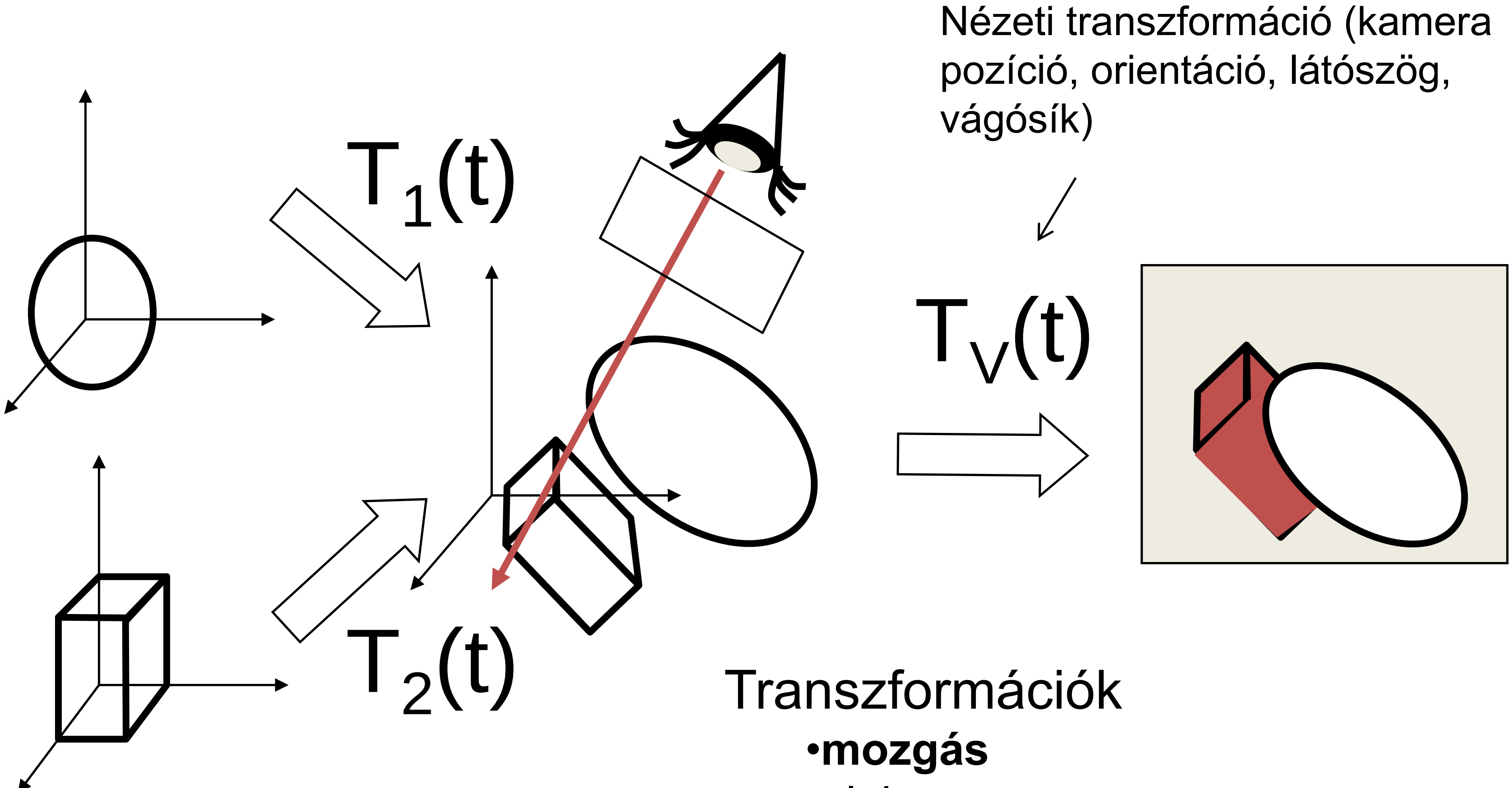
ANIMÁCIÓ



Metafórák:

- 2D rajzolás

Animáció = időfüggés



Transzformációk

- **mozgás**
- alak
- szín
- megjelenítési attribútumok, stb
- **kameranézet**

Valós idejű animáció

Legalább 15 ciklus másodpercenként

Óra inicializálás (t_{start})
do
    t = Óra leolvasás
    Nézeti transzformáció: $T_V = T_V(t)$
    for each object o:
        modellezési transzf $T_{M,\,o} = T_{M,\,o}(t)$
    endfor
    Képszintézis
while ($t < t_{end}$)

Folyamatás mozgatás - OpenGL

- Követelmények:
  - animáció folyamatosan fusson (azaz a felhasználónak ne kelljen minden lépésnél „léptetni”)
  - a rendszer reagáljon a felhasználói beavatkozásra (pl leállítás, lövés…)
    - nem jó egyszerű végtelen ciklus-hurok!
- Megoldás:
  - olyan ciklus, amely felváltva hajt végre egy-egy lépést a rendszer eseménykezelő hurkából és a program szimulációs hurkából

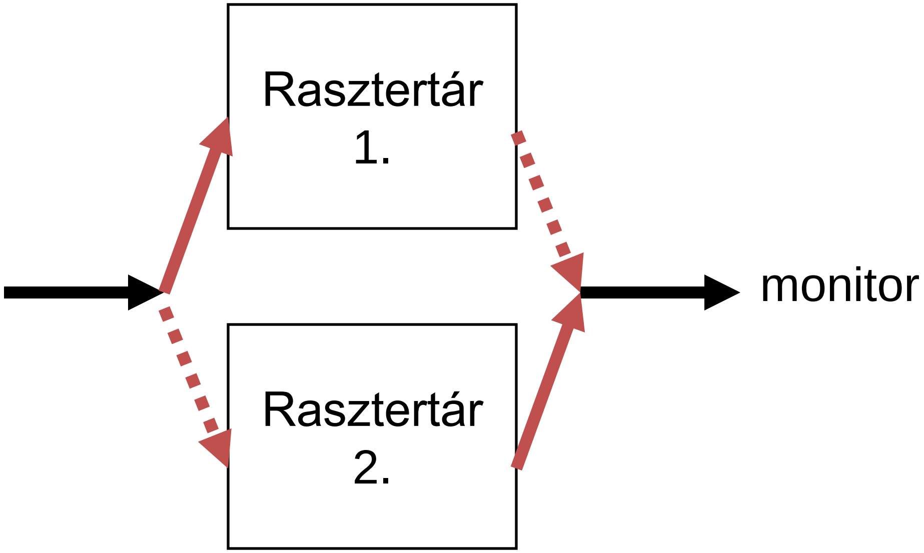
GLUT

- Üresjárati eseménykezelő függvény:
  - **glutIdleFunc**(myIdleFunc)
- Idő lekérdezés:
  - **glutGet**(GLUT_ELAPSED_TIME)
- Minta program:

```
long oldTime;
void IdleFunc(void) {
  long newTime=glutGet(GLUT_ELAPSED_TIME);
  myStepFunction(newTime-oldTime);
  oldTime=newTime;
  }
```

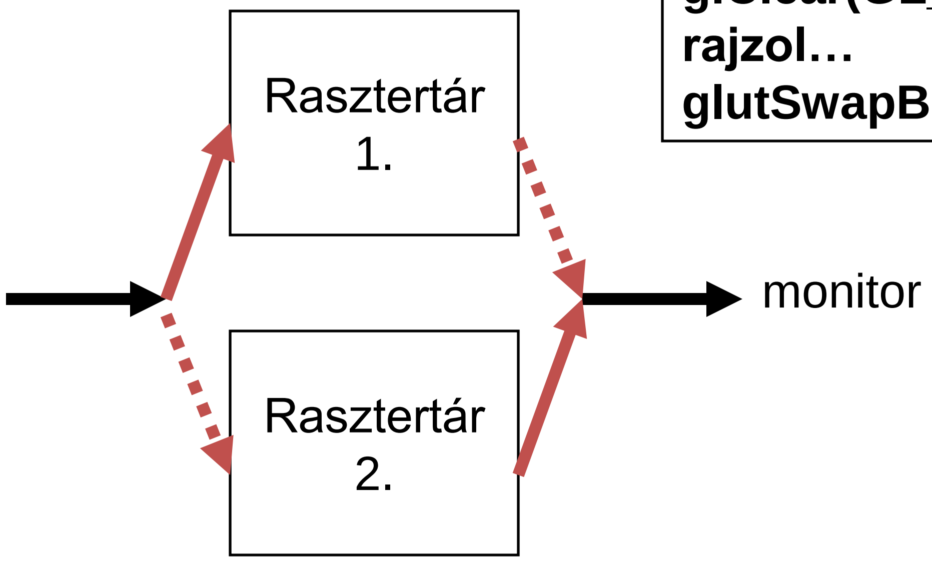
Képszintézis és megjelenítés

- Animációs hurok: képek előállítása és megjelenítése ciklikusan ismételve
  - Inkrementális képszintézis eljárások villogáshoz vezethetnek (képet fokozatosan építjük fel)
  - Megoldás: két külön rasztertár
    - Egyikben készül a kép, míg a másikat jelenítjük meg
    - Új képkocka megjelenítése: a két rasztertár gyors kicserélése



Dupla buffer animációhoz (GLUT)

Inicializálás:
glutInitDisplayMode(GLUT_RGBA | GLUT_DOUBLE);

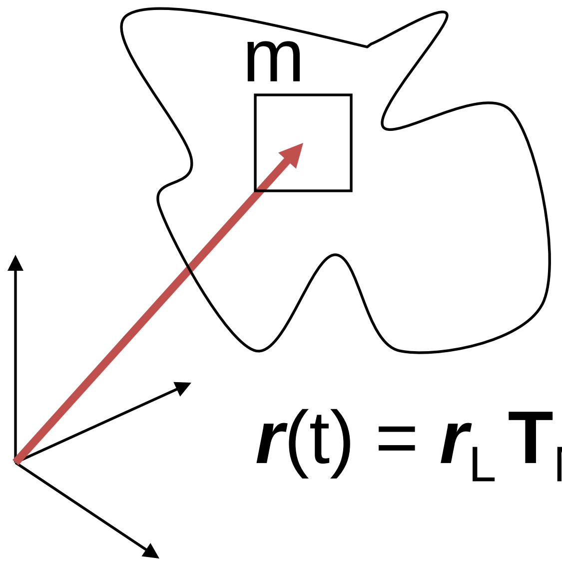


glClear(GL_COLOR_BUFFER_BIT);
rajzol…
glutSwapBuffers( );

Valószerű mozgás

- Fizikai törvények:
  - Newton törvény
  - ütközés detektálás és válasz: impulzus megmaradás
- Fiziológiai törvények
  - csontváz nem szakad szét
  - meghatározott szabadságfokú ízületek
  - bőr rugalmasan követi a csontokat
- Energiafelhasználás minimuma

Newton törvény



$$\boldsymbol{r}(t) = \boldsymbol{r}_L\,\mathbf{T}_M(t)$$

$$\mathbf{F}/m = \frac{d^2}{dt^2}\,\boldsymbol{r}$$

$$= \boldsymbol{r}_L \frac{d^2}{dt^2}\,\mathbf{T}_M(t)$$

Az erő rugalmas mechanizmuson keresztül hat, azaz folytonosan változik

$$\boxed{\mathbf{T}_M(t)\ C^2\ \text{folytonos}}$$

$T_M(t)$: Mozgástervezés

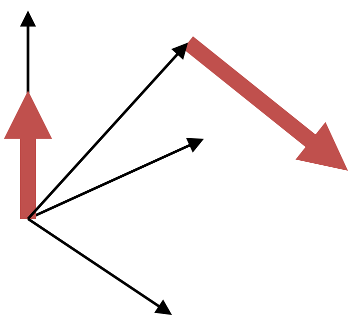
- Követelmény: ált. C^2 , néha (C^1,C^0) folytonosság
- Mozgás = a transzformációs elemek időbeli változtatása
- Tetszőleges pozíció+orientáció megadható az alábbi mátrixszal:

$$\mathbf{T}_M(t) = \begin{bmatrix} a11 & a12 & a13 & 0 \\ a21 & a22 & a23 & 0 \\ a31 & a32 & a33 & 0 \\ px & py & pz & 1 \end{bmatrix}$$

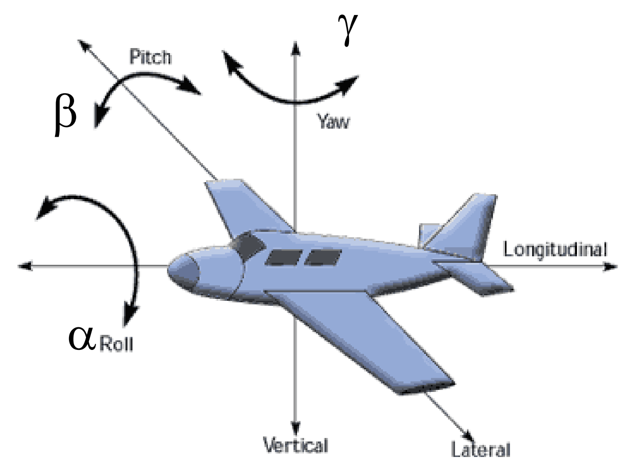
  - De az orientáció (A mátrix) szabadsági foka csak 3!
    - szabályos orientáció: sorvektorok egymásra merőleges egységvektorok

$T_M(t)$: Mozgástervezés

- **$T_M(t)$** mátrixelemek nem függetlenek!
  - Tervezés független paraméterek terében



pozíció: px, py, pz
orientáció: α, β, γ



$$\mathbf{p}(t)=[px, py, pz, \alpha, \beta, \gamma](t)$$

$$\mathbf{T}_M(t)=$$

$$\begin{bmatrix} \cos\alpha & \sin\alpha & & \\ -\sin\alpha & \cos\alpha & & \\ & & 1 & \\ & & & 1 \end{bmatrix} \begin{bmatrix} \cos\beta & & -\sin\beta & \\ & 1 & & \\ \sin\beta & & \cos\beta & \\ & & & 1 \end{bmatrix} \begin{bmatrix} 1 & & & \\ & \cos\gamma & \sin\gamma & \\ & -\sin\gamma & \cos\gamma & \\ & & & 1 \end{bmatrix} \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ px, & py, & pz, & 1 \end{bmatrix}$$

Orientáció tervezés – gondok…

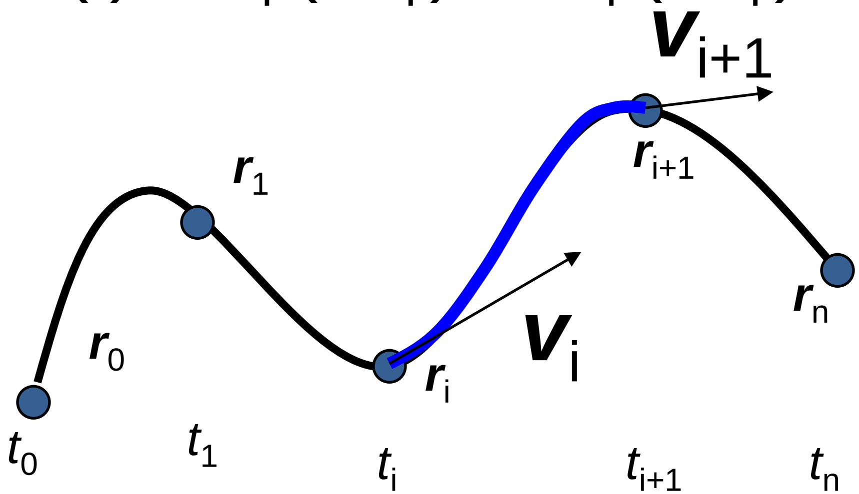
- Változó orientáció-animálás a α, β, γ csavaró-billentő-forduló szögek független interpolációjával
  - PRO: minden pillanatban érvényes orientációt kapunk
  - KONTRA: képzeletbeli tengelyek körül forgatunk, ezért a mozgás nem lesz valósszerű ☹
    - a paraméterek egyenletes változtatása egyenetlen mozgást eredményez, a képzeletbeli tengelyek láthatóvá válnak
    - megoldás: interpoláció kvaterniókkal
      - (érdeklődőknek részletek: Szirmay-Kalos László et. al. „Háromdimenziós grafika, animáció és játékfejlesztés” 312 oldal – lásd könyvtárban)

Mozgástervezés a paramétertérben

- **p**(t) elemei ált. C^2 , néha (C^1,C^0) folytonosak
- **p**(t) elemeinek a definíciója:
  - görbével direkt módon (**spline**)
  - képlettel: **script animation**
    - pl: origóból (v_x , v_y) kezdősebességgel kilőtt lövedék mozgása
      $x(t)=v_x t$, $y(t) = v_y t - g t^2/2$
  - kulcsokból interpolációval: **keyframe animation**
  - görbével indirekt módon: **path animation**
  - mechanikai modellből az erők alapján: **physical anim.**
  - mérésekből: **motion capture animation**

Interpoláció: 3-d rendű spline

$$\boldsymbol{r}(t) = \boldsymbol{a}_i (t - t_i)^3 + \boldsymbol{b}_i (t - t_i)^2 + \boldsymbol{c}_i (t - t_i)^1 + \boldsymbol{d}_i \qquad \text{ha } t_i \le t < t_{i+1}$$



$$\boldsymbol{r}(t_i) = \boldsymbol{r}_{i,} \qquad \boldsymbol{r}(t_{i+1}) = \boldsymbol{r}_{i+1}$$

$$\boldsymbol{r}'(t_i) = \boldsymbol{v}_i \qquad \boldsymbol{r}'(t_{i+1}) = \boldsymbol{v}_{i+1}$$

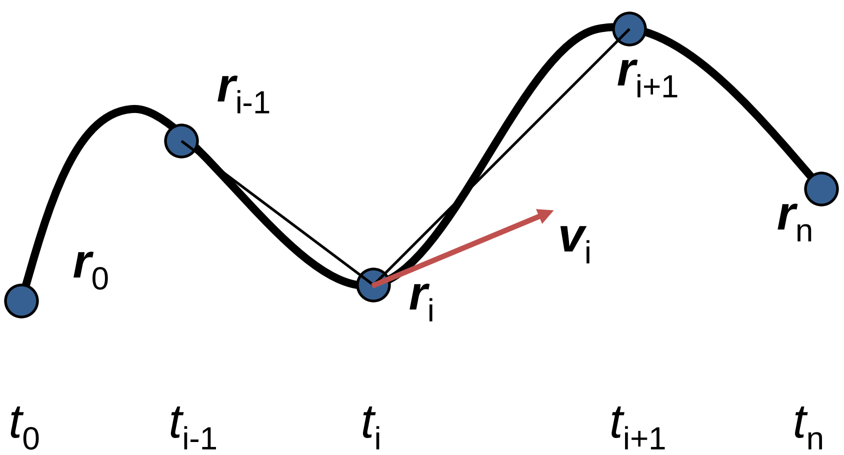
- C^2 folytonosság követelményéből: spline
- <u>Ismeretlen $\boldsymbol{v}_i$ -k meghatározása:</u>
  - $\boldsymbol{r}_i''(t_{i+1}) = \boldsymbol{r}_{i+1}''(t_{i+1})$ + sebesség a kezdő és végpontban
    - bonyolult lineáris egyenletrendszer megoldását igényli ☹
- Tervezési paraméterek alapján: Kohanek-Bartels, <u>Catmull-Rom</u>
  - Feladjuk a C^2 folytonosság követelményét a görbeszegmensek kapcsolódási pontjaiban
  - Legalább szép sima legyen a pálya…

Catmull-Rom „spline”

$$\boldsymbol{r}(t) = \boldsymbol{a}_i (t - t_i)^3 + \boldsymbol{b}_i (t - t_i)^2 + \boldsymbol{c}_i (t - t_i)^1 + \boldsymbol{d}_i \qquad \text{ha } t_i \le t < t_{i+1}$$

Sebességek előírása:

$$\boldsymbol{r}'(t_i) = \boldsymbol{v}_i \qquad \boldsymbol{r}'(t_{i+1}) = \boldsymbol{v}_{i+1}$$



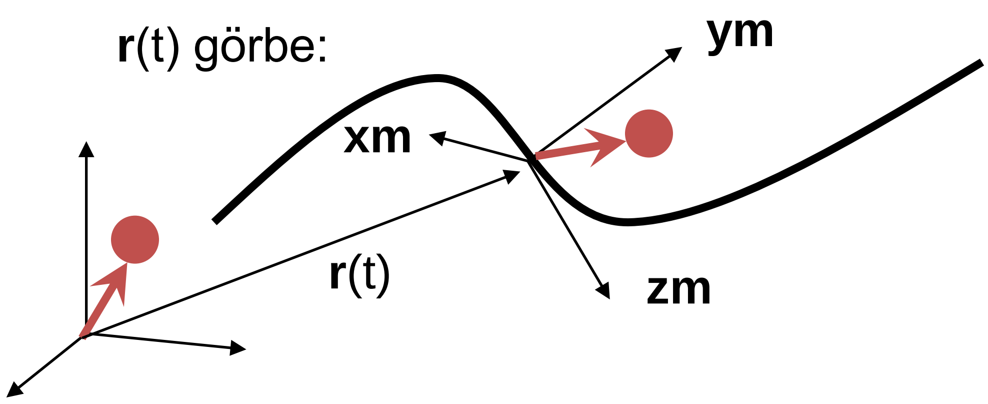
$$\boldsymbol{a}_i = \frac{\boldsymbol{v}_{i+1} + \boldsymbol{v}_i}{(t_{i+1} - t_i)^2} - \frac{2(\boldsymbol{r}_{i+1} - \boldsymbol{r}_i)}{(t_{i+1} - t_i)^3}$$

$$\boldsymbol{b}_i = \frac{3(\boldsymbol{r}_{i+1} - \boldsymbol{r}_i)}{(t_{i+1} - t_i)^2} - \frac{\boldsymbol{v}_{i+1} + 2\boldsymbol{v}_i}{(t_{i+1} - t_i)}$$

$$\boldsymbol{c}_i = \boldsymbol{v}_i \qquad \boldsymbol{d}_i = \boldsymbol{r}_i$$

$$\boldsymbol{v}_i = \frac{1}{2}\left(\frac{\boldsymbol{r}_{i+1} - \boldsymbol{r}_i}{t_{i+1} - t_i} + \frac{\boldsymbol{r}_i - \boldsymbol{r}_{i-1}}{t_i - t_{i-1}}\right)$$

Pálya animáció: Transzformáció



$$T_M = \begin{pmatrix} \mathbf{xm}^0(t) & 0 \\ \mathbf{ym}^0(t) & 0 \\ \mathbf{zm}^0(t) & 0 \\ \mathbf{r}(t) & 1 \end{pmatrix}$$

<u>Explicit up vektor</u>

zm = **r**’(t)
xm = **zm** ⊗ up
ym = **zm** ⊗ **xm**

<u>Frenet keretek:</u>

zm = **r**’(t)
xm = **zm** ⊗ **r**’’(t) =**r**’(t) ⊗ **r**’’(t)
ym = **zm** ⊗ **x**

A függőleges,
amerre az erő hat