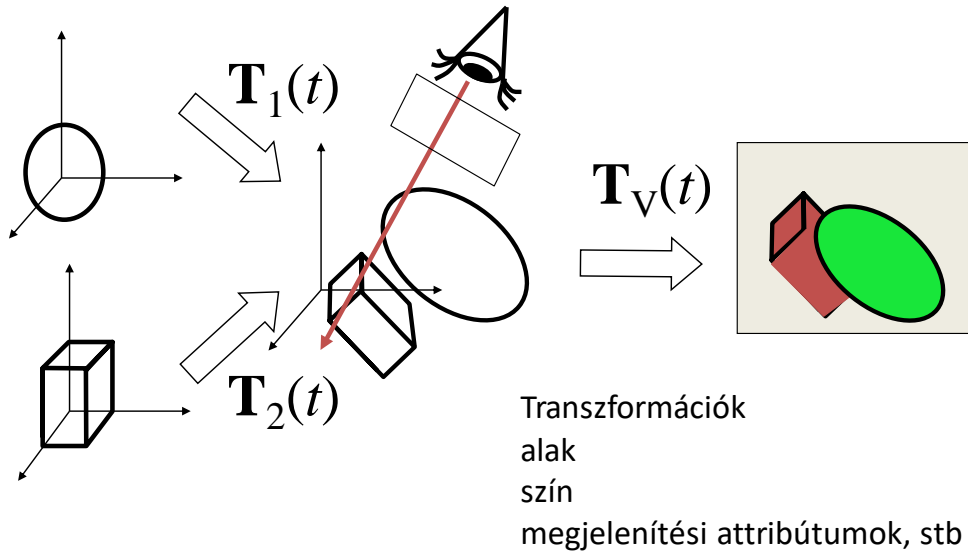


Animáció

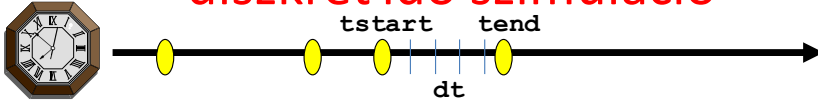
Szirmay-Kalos László

Animáció = időfüggés



Animation means that the properties of objects, light sources, or the camera change in time. Any property may change, but the most important case is when transformations are functions of time. If modeling transformation depends on time, we animate objects. If camera transformations are time dependent, we animate the camera.

Valós idejű animáció és diszkrét idő szimuláció



```
void IdleFunc( ) { // idle call back
    static float tend = 0;
    const float dt = 0.01; // dt is "infinitesimal"
    float tstart = tend;
    tend = glutGet(GLUT_ELAPSED_TIME)/1000.0f;

    for(float t = tstart; t < tend; t += dt) {
        float Dt = min(dt, tend - t);
        for (Object * obj : objects) obj->Control(Dt);
        for (Object * obj : objects) obj->Animate(Dt);
    }
    glutPostRedisplay();
}
```

Animation also means that when some time elapses, the state of the virtual world catches up with the elapsed time. This should be happening even if the user does not even touch the computer. The event handling system provides the idle callback for this purpose. So in an idle call back, the elapsed times is computed, and the time interval [tstart,tend] elapsed since the last idle callback is simulated. As there is no upper limit for the length of the simulated interval, it is decomposed to dt steps that are sufficiently small. Here small means that time differentials can be well approximated differences and in dt we can suppose that the velocity and acceleration are constants.

The simulation of an object is subdivided to control and animate. Control means the preparation for the state change, and animate means the execution of the state change.

If we merged the two operations together, then the simulation would depend on the processing order of objects.

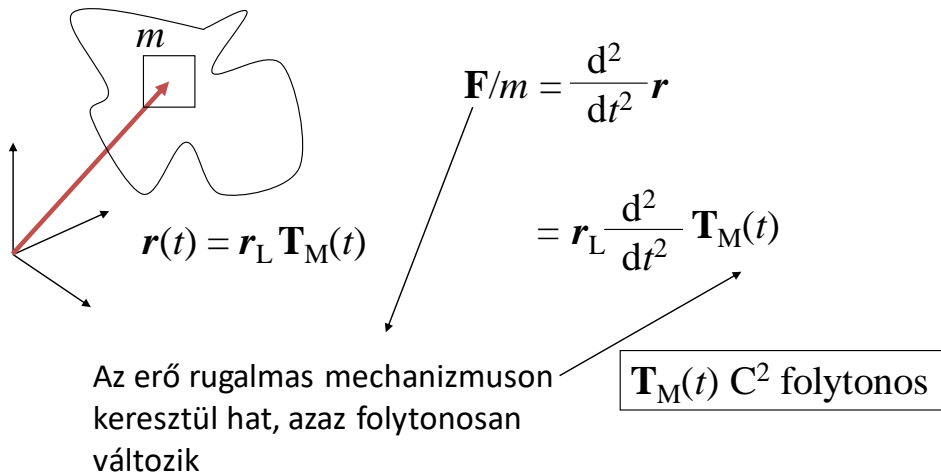
Valószerű mozgás

- Fizikai törvények:
 - Newton törvény
 - Impulzus deriváltja az erő; Perdület deriváltja a forgatónyomaték
 - ütközés detektálás és válasz:
 - impulzus és perdület megmaradás
 - energia részleges megmaradás
- Fiziológiai törvények
 - csontváz nem szakad szét
 - adott szabadságfokú ízületek
 - bőr rugalmasan követi a csontokat
- Energiafelhasználás minimuma

VideoMach unregistered

The task of animation is the definition of the time dependence of transformations. We hope for realistic animations that do not contradict to the physical laws (acceleration is proportional to the force, linear momentum and angular momentum are conserved) or to the physiological laws (bones are connected by joints that do not allow bones to separate).

Newton törvény



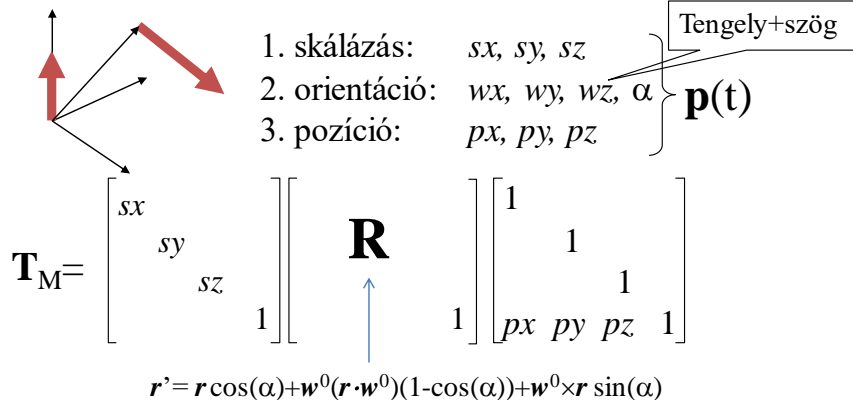
According to the Newton's law, acceleration, i.e. the second derivative of the motion is proportional to the force. As forces act on some elastic mechanism, they cannot change abruptly. So generally, the path should have a continuous second derivative, i.e. of C^2 type curve.

However, if the body is really rigid (and not elastic), then force can change abruptly. In case of collisions very large, i.e. infinite forces may occur. So in special situations, the path can be of C^1 or C^0 type.

T_M(t): Mozgástervezés

- Követelmény: ált. C², néha (C¹, C⁰) folytonosság
- Mátrixelemek nem függetlenek

– Tervezés független paraméterek terében



So we need motion curves that are generally of C² type, occasionally of C¹ and C⁰. Motion is the time dependence of 4x4 homogeneous linear transformation matrices, having 16 elements. However, typical motions like translation, translation+rotation etc. Have less degrees of freedom, translation has 3, rotation has also 3, thus the 16 matrix elements are not independent. Should we interpolate them independently, the transformed object would be distorted. Therefore, we never specify time dependency directly for the transformation matrix elements. Instead, the space of independent motion parameters is found, the independent motion parameters are given time functions and are calculated first, then the matrix elements are obtained from the independent motion parameters. For rigid body motion, the independent motion parameters include the Cartesian coordinates of the translation, the direction vector of the rotation axis, and the angle of rotation. If the size can also change in time, three additional scaling parameters are added. From these, the modeling transformation matrix can be obtained.

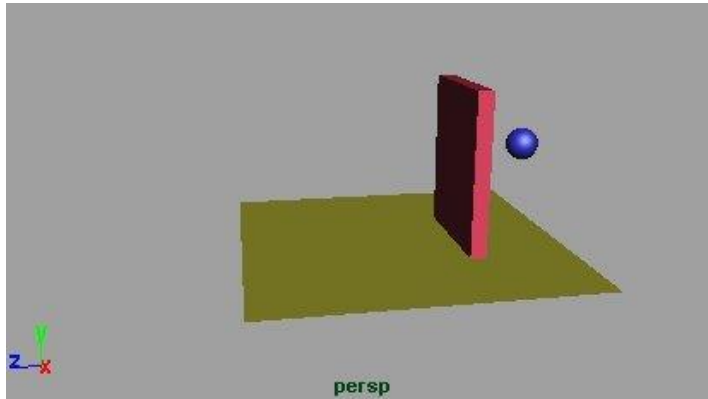
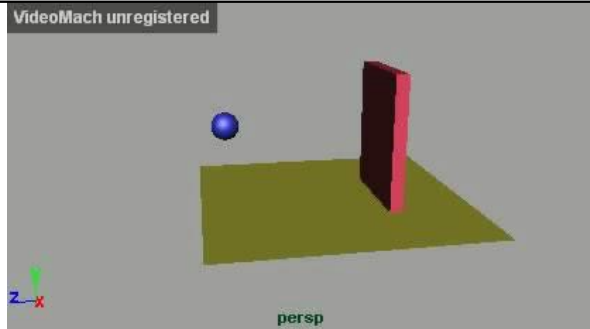
Mozgástervezés a paraméterterben

- $\mathbf{p}(t)$ elemei ált. C^2 , néha (C^1, C^0) folytonosak
- $\mathbf{p}(t)$ elemeinek a definíciója:
 - görbével direkt módon (**spline**)
 - képlettel: **script animation**
 - kulcsokból interpolációval: **keyframe animation**
 - görbével indirekt módon: **path animation**
 - mechanikai modellből az erők alapján: **physical animation**
 - mérésekből: **motion capture animation**

Let us concentrate on the definition of the vector of independent motion parameters. There are various possibilities to define time functions in them.

Parametric curves use the analogy of motion, thus they can be directly used to describe motion. The time functions can also be given by formulae (e.g. motion of a bullet from a canon). Keyframe animation requires the setting of objects at discrete points of time, called keyframes, and the inbetweening process finds curves that interpolate the discrete poses. Path animation defines the path with a motion curve also requiring that the orientation of the object changes according to the motion, i.e. the object follows its own „head” and tries to preserve a vertical direction. Physical animation does not specify motion directly, but provides the physical parameters like mass, friction, initial velocity and position etc. and solves the laws of motion to simulate motion. Finally motion capture animation measures the motion of a real-charactered (motion artist) and uses the measured data to control a virtual character.

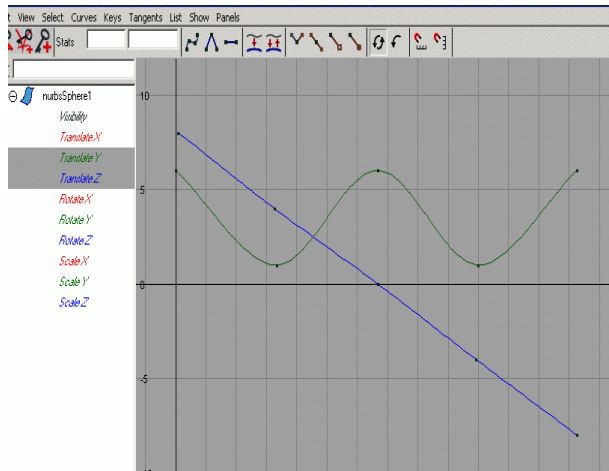
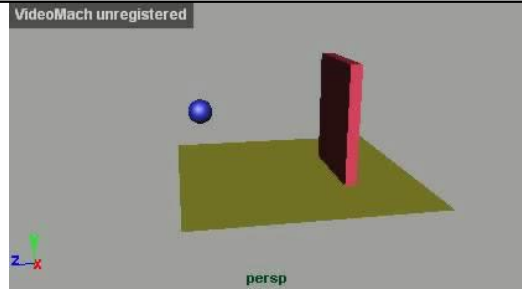
Keyframe animáció



5.key

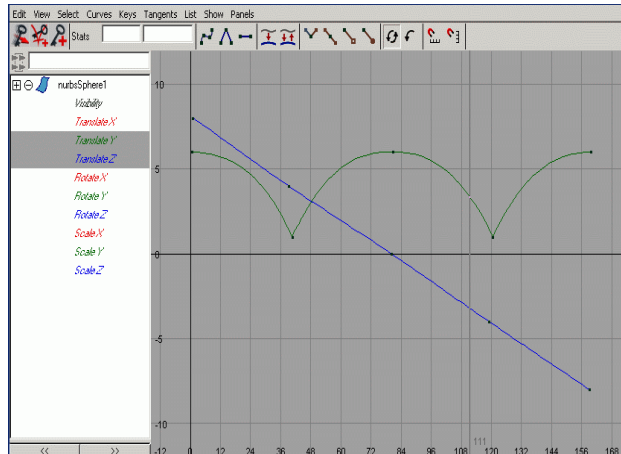
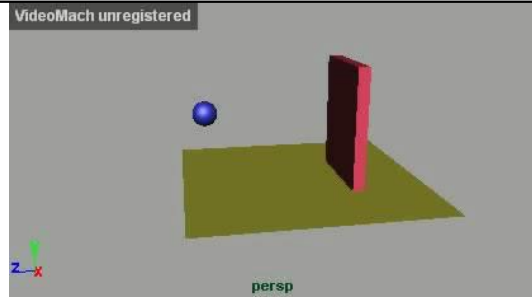
Let us define a clip by keyframe animation, where a ball bounces on the floor while a door opens and lets the ball in. At 5 points of time, the ball and the door are positioned.

Keyframe animáció görbéi



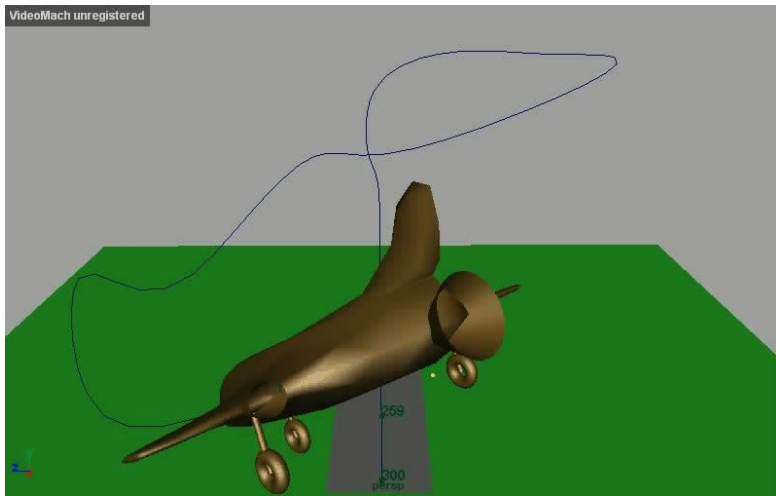
The discrete positions are black dots, i.e. Points to be interpolated on the yet unknown motion curves (y and z coordinates of the ball are shown). The interpolation is done with Catmull-Rom spline, so we get two functions interpolating the key values. The resulting animation is not realistic since it does not provide the bouncing effect.

Görbék megváltoztatása



So the splines are modified by hand. We know from physics that the y coordinate should follow a parabola.

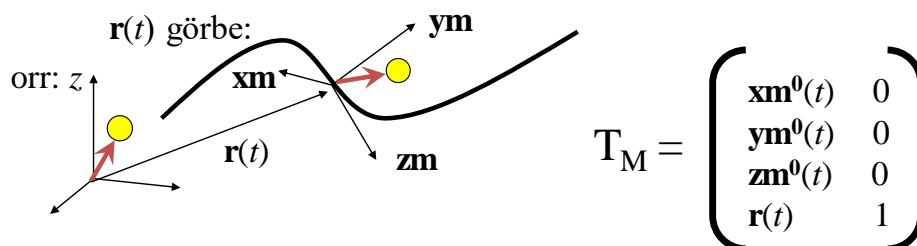
Pálya (path) animáció



$t = \text{spline paraméter vagy az ívhossz}$

For path animation a single 3D curve should be drawn, which directly defines the position and indirectly the orientation.

Pálya animáció: Transzformáció



Explicit up vektor

$$z_m = r'(t)$$

$$x_m = z_m \otimes \text{up}$$

$$y_m = z_m \otimes x_m$$

Frenet keretek:

$$z_m = r'(t)$$

$$x_m = z_m \otimes r''(t)$$

$$y_m = z_m \otimes x_m$$

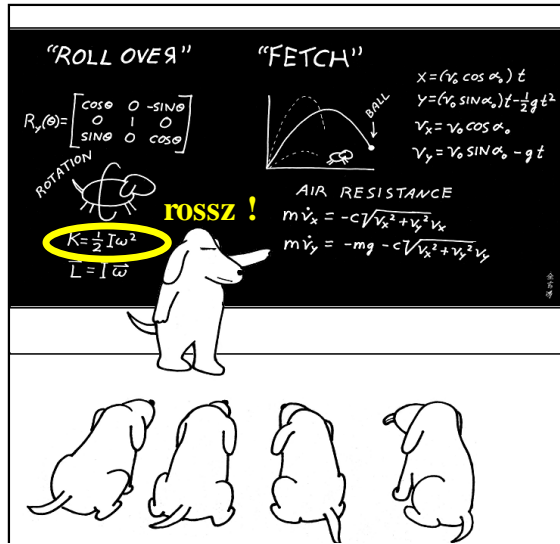
A függőleges,
amerre az erő hat

The indirect orientation definition is based on the recognition that objects like airplanes, birds, cars, people, animals etc. follow their own „nose”, meaning that their nose always point into the direction of the motion, which is the current velocity vector. The velocity is the derivative of the path. The nose direction is not enough to define the full orientation, so we also specify a „preferred vertical” direction. This can be a fixed direction or the current acceleration (this option is called Frenet frames in differential geometry).

If in modeling space, the nose direction is axis z , the vertical direction is axis y , then the transformation matrix can be directly obtained from their transformed versions.

Fizikai animáció

- Erők (pl. gravitáció, turbulencia stb.)
- Tömeg, tehetetlenségi nyomaték ($F = ma$)
- Ütközés detektálás (metszéspontszámítás)
- Ütközés válasz
 - rugók, ha közel vannak
 - impulzus megmaradás



In physical animation the forces, masses, inertia and the initial conditions are defined and motion is obtained simulating the physical laws:

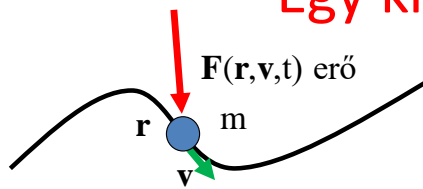
- Linear momentum is the product of the mass and the velocity of the center of mass
- The time derivative of the linear momentum is the force
- Angular momentum is the product of the inertial matrix and the angular velocity
- The time derivative of the angular momentum is the torque
- Collision happens when objects are about to penetrate into each other
- Upon collision, linear momentum and angular momentum are conserved, the kinetic energy is conserved only in case of elastic collision.

Note that in dog school the formula of the kinetic energy of rotational motion is wrong. What would be the right one?

Haladó és forgó mozgás

Pozíció:	\vec{r}	Orientáció:	\mathbf{R}
Összefűzés:	$\vec{r}_1 + \vec{r}_2$	Összefűzés:	$\mathbf{R}_1 \cdot \mathbf{R}_2$
Sebesség:	$\vec{r}(t + dt) =$ $\vec{r}(t) + \vec{v}dt$	Szögsebesség:	$\mathbf{R}(t + dt) =$ $\mathbf{R}(t)R(\vec{\omega} dt, \vec{\omega})$
Tömeg:	m	Tehetlenségi nyom:	\mathbf{I}
Lendület:	$\vec{p} = m\vec{v}$	Perdület:	$\vec{L} = \mathbf{I}\vec{\omega}$
Erő:	$\vec{F} = \frac{d\vec{p}}{dt}$	Forgató nyom:	$\vec{M} = \frac{d\vec{L}}{dt}$
Energia:	$K = \frac{1}{2}mv^2$	Energia:	$K = ?$

Egy kis mechanika



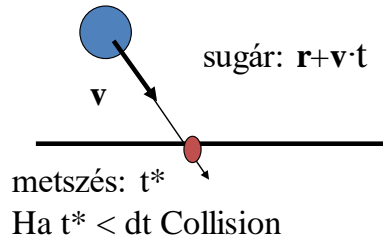
$$\frac{d\mathbf{r}}{dt} = \mathbf{v}$$

$$\frac{d\mathbf{v}}{dt} = \mathbf{F}(\mathbf{r}, \mathbf{v}, t)/m$$

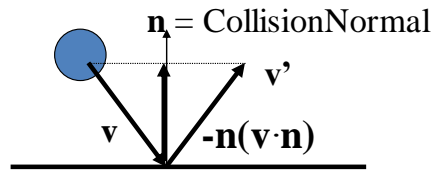
```

for( t = 0; t < T; t += dt ) {
   $\mathbf{F} = \text{Eredő erő}(\mathbf{r}, \mathbf{v})$ 
   $\mathbf{a} = \mathbf{F}/m$ 
   $\mathbf{v} += \mathbf{a} \cdot dt$ 
  if (  $\text{ÜtközésDetektál}$  )
     $\text{ÜtközésVálasz}$ 
   $\mathbf{r} += \mathbf{v} \cdot dt$ 
}
    
```

ÜtközésDetektál



ÜtközésVálasz



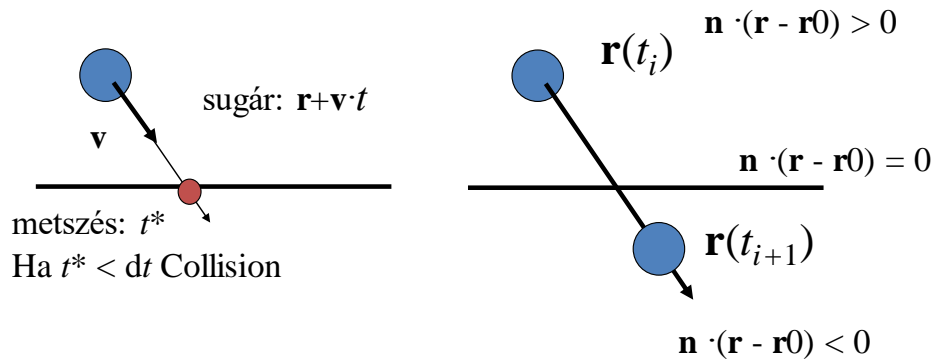
$$\mathbf{v}' = [\mathbf{v} - \mathbf{n}(\mathbf{v} \cdot \mathbf{n})] - [\mathbf{n}(\mathbf{v} \cdot \mathbf{n}) \cdot \text{bounce}]$$

For the sake of simplicity, we consider only translational motion and point like objects. The force field in nature may depend on the time (e.g. wind), position (e.g. gravity) and the velocity (e.g. air resistance), but not on higher derivatives. According to the Newton's law, the time derivative of the velocity is the force divided by the mass. According to the definition of the velocity, it is the time derivative of the position. So we have two linear differential equations that need to be solved. If the time is decomposed to small steps dt , differentials are approximated by differences, so the change of velocity will be the acceleration times dt , while the change of position the velocity time dt , since we assume that dt is small enough so acceleration and velocity are constants.

This is not the case for collision, so this should be checked and if collision happens, the modified velocity should be directly computed from the preservation laws (linear momentum and angular momentum are preserved, kinetic energy is preserved only for elastic collision).

For the motion of a point like object in interval dt when we can assume that the velocity is constant, collision detection is equivalent to ray tracing. Collision response is similar to mirror like reflection if the collision is elastic, and we should reduce the perpendicular component if it is inelastic.

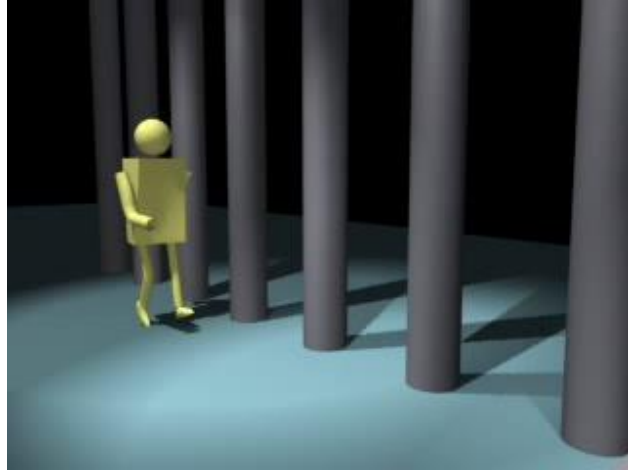
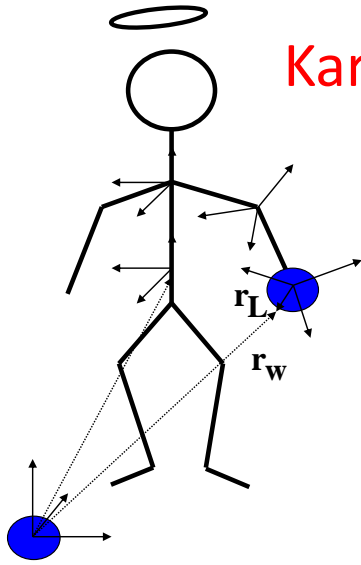
Folytonos-Diszkrét ütközés detektálás pontra és féltérre



Continuous collision detection is equivalent to ray tracing since the path of a point is assumed to be linear in dt .

Discrete collision detection checks whether the objects have penetrated into each other by containment test.

Karakter animáció

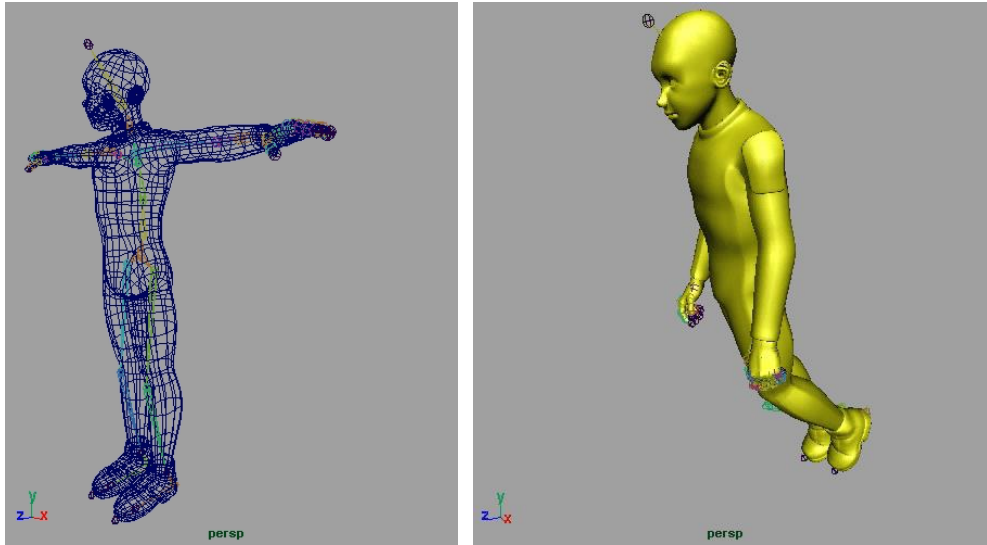


$$\mathbf{r}_w = \mathbf{r}_L \cdot \mathbf{R}_{\text{kéz}} \cdot \mathbf{T}_{\text{alkar}} \cdot \mathbf{R}_{\text{könyök}} \cdot \mathbf{T}_{\text{felkar}} \cdot \mathbf{R}_{\text{váll}} \cdot \mathbf{T}_{\text{gerinc}} \cdot \mathbf{T}_{\text{ember}}$$

homogén koordináta 4-es

In character animation, the skeleton defines the motion. Every bone in the skeleton can introduce a rotation around the joint and a translation depending on the length of the bone.

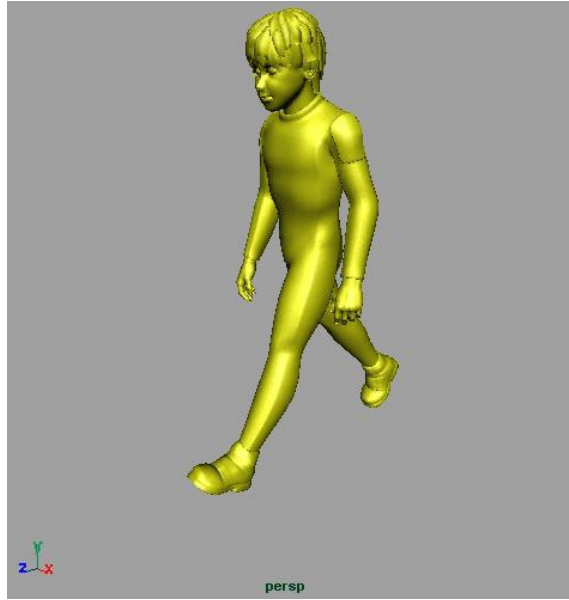
Csontváz



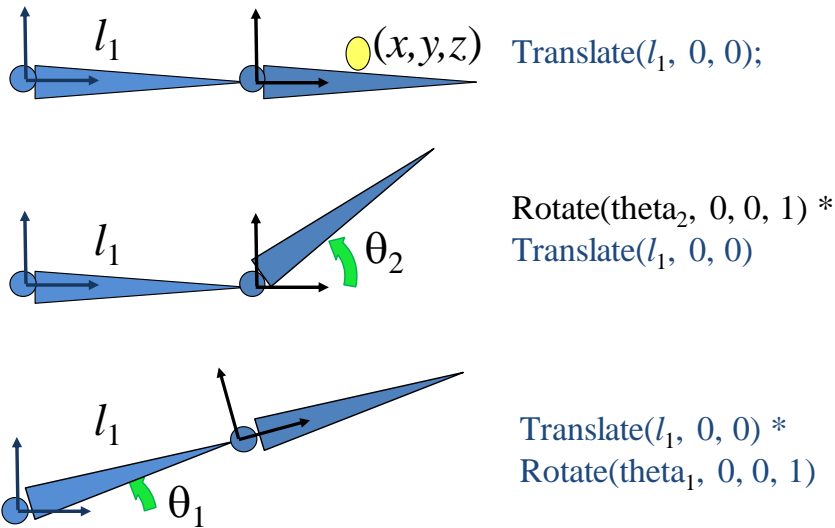
When bone animation is defined, the bones and joints connecting them are also included in the mesh representing the skin. By default, a skin vertex will be transformed by the transformation of that bone which is closest to it.

During animation, the bones are rotated in the joints and upper level bones naturally modify all other bones connected to it via joints. The skin is deformed with the resulting transformation matrices of the bones.

Séta



Tranzformáció hierarchia



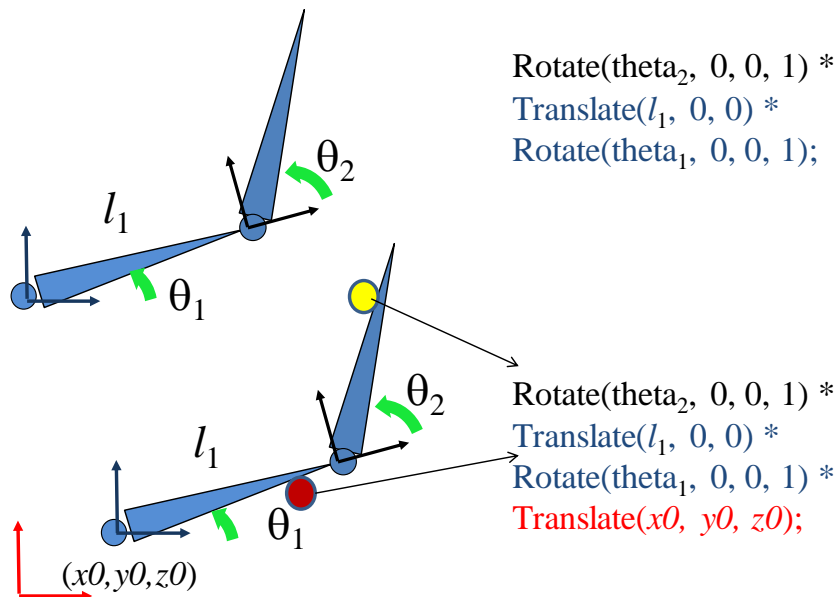
To examine how this can be done in OpenGL, let us consider a skin vertex attached to the cyan bone. When this correspondence is made, the skin vertex is expressed relative to its bone, i.e. in the coordinate system of the bone, resulting in (x,y,z) .

This point can also be expressed in the coordinate system of the parent bone (brown bone), just the transformation between the reference systems of the two bones should be executed. This is currently a translation along axis x with the length of the bone l_1 .

If the child (cyan) bone is rotated, this rotation applies to the skin in its coordinate system, thus rotation happens before applying the translation to the parent system.

If the parent bone is rotated, the skin is translated first to the parent's coordinate system, then rotation takes place.

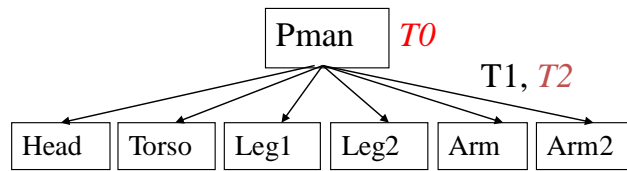
Tranzformáció hierarchia



If both bones are rotated, first child rotation, then translation to the parent's system, finally rotation of the parent are executed. If the parent is also a child of some other node, or the parent is placed in the world, then new transformations must be added on the top of the hierarchy.

Generally, a bone is a rotation transformation for its own skin and a rotation + translation to its children, which should be applied recursively on hierarchical characters.

PMan

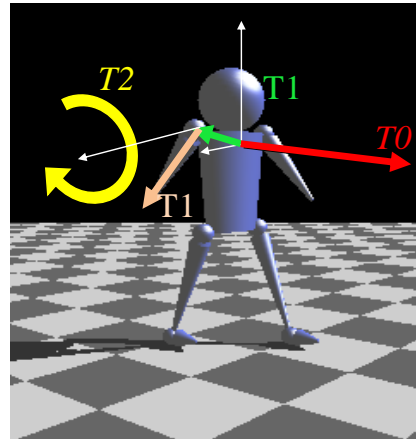


T0 = Pman előremozog
Translate(forward, up, 0)

T1 = váll pozíció
Translate(leftShoulderPos)

T2 = kar forgatás
Rotate(armAngle, armRotAxis)

...
T3 = kéz pozíció
Translate(armLength, 0, 0)



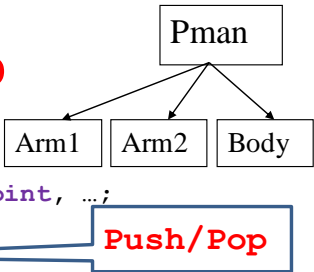
Our simple example is the primitive man, which consists of a head, torso, two independent legs and two arms. Let us consider just an arm.

Pman swings his arm while walking. The swing rotation is defined in a coordinate system where the origin is the shoulder position. Then, the position of the shoulder with respect to Pman should be defined, i.e. the swinging arm should be expressed in a coordinate system having the origin in the center of Pman. This is a translation. Finally, Pman moves forward, i.e. its center is translated in the world coordinate system. So the arm is first rotated (T2), then translated (T1), and translated again (T0). From these transformations, T0 and T2 change in time, but T1 remains constant, which defines the physiological constraints of the body:

Pman can move its complete body and can swing its arm, but cannot remove its arm from its shoulder.

Pman rajzolás és animáció

```
class Pman {
    float armAngle, dArmAngle, forward, up;
    const ... armLength, armRotAxis, rightArmJoint, ...;
public:
    void DrawArm(float dt, mat4 M) {
        M = Rotate(armAngle, armRotAxis) * M; // T2
        DrawRefArm(M);
    }
    void DrawPman( ) { // set matrices from animation parameters
        mat4 M = Translate(forward, up, 0); // T0
        DrawRefBody(M);
        DrawArm(dt, Translate(rightArmJoint) * M); // T1
        DrawLeg(dt, Translate(rightLegJoint) * M);
        ...
    }
    void Animate(float dt) { // calculate animation parameters
        armAngle += dArmAngle * dt;
        if (armAngle > 0.5 || armAngle < -0.5) dArmAngle *= -1;
        forward += 5 * dt;
    }
};
```



Only the parameters of T0 and T2 are updated.

T2 is a periodic swinging rotation, which is defined by two key frames defining the two extreme angles and the rotation angle is linearly interpolated in between according to the elapsed time.

Note that when we step on a lower hierarchy level, the transformation matrix is pushed on stack, and then restored since objects on the same level should not interfere (arms are independent, so are legs). However, the parent affects all its children.

Inverz kinematika

T0 = előremozgás (forward, up) ???

T2 = láb forgatás(ang)

Támaszkodó nem csúszkálhat

```

forward += leg * fabs(sin(angNew) - sin(angOld)) ;
up       = leg * cos(angNew) ;

```

The motion defined by constant forward moving velocity and constant angular speed in the hips and shoulders is not realistic since the leg will slip on the floor (like a break dancer) and the body will fly over the floor.

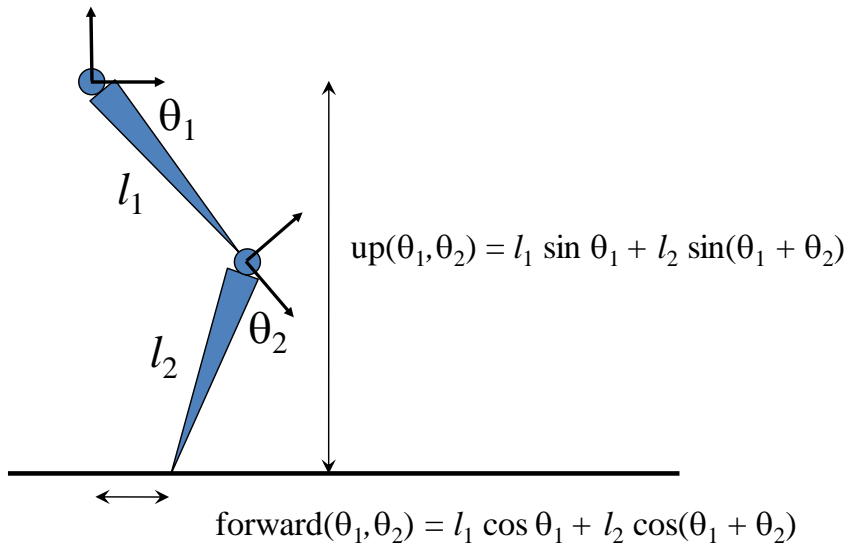
In realistic walking one leg should stand still on the floor. In an animation, the point of interest for which constraints are given is called the **end effector** (a term inherited from robotics). The end effector of the walking is the leg holding the weight of the body.

The problem is that we define the character state from top to bottom by setting a sequence of transformations. This is called **forward kinematics**. The end effector at the end of the transformation sequence will be affected by all transformations. The task is to determine the upper level transformations in a way that the resulting end effector position meets the specified constraint. Such problems are called **inverse kinematics**.

In this simple problem, we can explicitly solve the inverse kinematics problem since the relationship between the position of the character (forward and up) a hip rotation (ang) is defined by a right triangle of hypotenuse equal to the leg length. The end effector is always on the ground, so up directly specifies the distance from the floor. However, Pman walks forward, so its location along the forward direction is not constant. The actual forward

position is just relative to the leg, which means that forward is updated incrementally.

Inverz kinematika



In case of multiple joints and bones, the correspondence between the rotation angles and the relative position of the origin and the end effector may become complicated if the rotation axes are different. However, when rotation axes are parallel (more or less, this is the case for hip and knee rotations), a simple analytic expression can be elaborated.

Bőrözés

