Development of Games

Lecture 7

Introduction to animation

Basics of Motion Generation

- let X_i = configuration of O_i at $t_k = t_0$, $\forall i$
- END = false
- while (not END) do
- display O_i , $\forall i$
- $t_k = t_k + \Delta t$
- generate X_i at t_k , $\forall i$
- END = *function*(motion generation)

Methods of Motion Generation

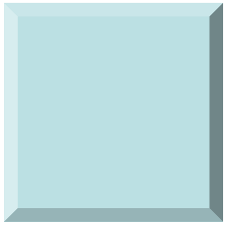
- Traditional Principles (Keyframing)
- Performance Capture (Motion Capture)
- Modeling/Simulation (Physics, Behaviors)
- Automatic Discovery (High-Level Control)

Keyframing (II)

- Advantages
  - Relatively easy to use
  - Providing low-level control
- Problems
  - Tedious and slow
  - Requiring the animator to understand the intimate details about the animated objects and the creativity to express their behavior in key-frames

Sprites

- It is simplest method of simulation motion in simple 2D games



Motion Interpolation

- Interpolate using mathematical functions:
  - Linear
  - Hermite
  - Bezier
  - … and many others (see Appendices of Richard Parent's online book)
- Forward & inverse kinematics for articulation
- Specifying & representing deformation

Basic Terminologies

- *Kinematics*: **study of motion independent of underlying forces**
- *Degrees of freedom (DoF)*: **the number of independent position variables needed to specify motions**
- *State Vector*: **vector space of all possible configurations of an articulated figure. In general, the dimensions of state vector is equal to the DoF of the articulated figure.**

Forward vs. Inverse Kinematics

- *Forward kinematics*: **motion of all joints is explicitly specified**
- *Inverse kinematics*: **given the position of the end effector, find the position and orientation of all joints in a hierarchy of linkages; also called "goal-directed motion". (See an in-class example.)**

Forward Kinematics

- As DoF increases, there are more transformation to control and thus become more complicated to control the motion.
- Motion capture can simplify the process for well-defined motions and pre-determined tasks.

Inverse Kinematics

- As DoF increases, the solution to the problem may become *undefined* and the system is said to be *redundant*. By adding more constraints reduces the dimensions of the solution.
- It's simple to use, when it works. But, it gives less control.
- Some common problems:
  - Existence of solutions
  - Multiple solutions
  - Methods used

Modeling Deformation

- Geometric-based Techniques
  – Global & local deformation (Barr’84)
  – FFD (Sederberg & Parry’86) and variants
  – … others
- Physically-based Techniques
  – particle systems
  – BEM
  – FEM & FEA
- Variational Techniques
  – Variational surface modeling (Welch & Witkin’92)
  – dynamic-NURBS (Terzopoulos & Qin’94)

Geometric Proximity Queries

- Given two object, how would you check:
-
  - ***If they intersect with each other while moving?***
  - ***If they do not interpenetrate each other, how far are they apart?***
  - ***If they overlap, how much is the amount of penetration***

Collision Detection

- Update configurations by matrices

- Check for edge-edge intersection in 2D
- (Check for edge-face intersection in 3D)

- Check every point of A inside of B &
- every point of B inside of A

- Check for pair-wise edge-edge intersections

- *Imagine larger input size: N = 1000+ ……*

Classes of Objects & Problems

- **2D vs. 3D**
- **Convex vs. Non-Convex**
- **Polygonal vs. Non-Polygonal**
- **Open surfaces vs. Closed volumes**
- **Geometric vs. Volumetric**
- **Rigid vs. Non-rigid (deformable/flexible)**
- **Pairwise vs. Multiple (N-Body)**
- **CSG vs. B-Rep**
- **Static vs. Dynamic**
- ***And so on… This may include other geometric representation schemata, etc.***

Some Possible Approaches

- **Geometric methods**
- **Algebraic Techniques**
- **Hierarchical Bounding Volumes**
- **Spatial Partitioning**
- **Others (e.g. optimization)**

Test of intersection

- 3D object is incapsulated in sphere or box (Axis-Aligned Bounding Boxes or Object-Oriented Boxes)
  - Sphere-sphere intersection
  - Sphere-ray intersection
  - Sphere-plane intersection
  - AABB-AABB intersection
  - AABB-ray intersection
  - AABB-plane intersection
  - OOB-OOB intersection
  - OOB-ray intersection
  - OOB-plane intersection

Motion Capture (I)

- Use special sensors (trackers) to record the motion of a performer

- Recorded data is then used to generate motion for an animated character (figure)

Motion Capture (II)

- Advantages
  - Ease of generating realistic motions
- Problems
  - Not easy to accurately measure motions
  - Difficult to “scale” or “adjust” the recorded motions to fit the size of the animated characters
  - Limited capturing technology & devices
    - Sensor noise due to magnetic/metal trackers
    - Restricted motion due to wires & cables
    - Limited working volume

Physically-based Simulation (I)

- Use the laws of physics (or a good approximation) to generate motions
- Primary vs. secondary actions
- Active vs. passive systems
- Dynamic vs. static simulation

Physically-based Simulation (II)

- Advantages
  - Relatively easy to generate a family of similar motions
  - Can be used for describing realistic, complex animation, e.g. deformation
  - Can generate reproducible motions
- Problems
  - Challenging to build a simulator, as it requires in-depth understanding of physics & mathematics
  - Less low-level control by the user

High-Level Control (I)

- Task level description using AI techniques:
  - Collision avoidance
  - Motion planning
  - Rule-based reasoning
  - Genetic algorithms
  - … etc.

High-Level Control (II)

- Advantages
  - Very easy to specify/generate motions
  - Can reproduce realistic motions
- Problems
  - Need to specify all possible “rules”
  - The intelligence of the system is limited by its input or training
  - May not be reusable across different applications/domains