Physics Engine for Soft Bodies

Brandon Payne

Thomas Marnell

Design

- Physics Engine
- 3D
- Real Time
- Soft Bodies
- Middleware

Graphics Platform

- Originally used DirectX
- Switched to XNA
  - Prebuilt Framework
  - Allowed to focus on Physics Engine

Parts of a Physics Engine

- Movement through 3D space
- Collision Detection
- Collision Reaction
  - Mesh Transformation

Client Program

- Creates Objects
  - Define a 3D Mesh
    - List of Indices and Vertices
  - Define Mass
  - Give initial position and velocity
- Handle Input
  - Reads input and sends to Physics Engine
- Display Objects
  - Based on information provided by Physics Engine

Difficulties

- Accuracy versus Efficiency
  - We want to run in Real Time
  - Complicated Meshes have many calculations
  - “Real” physics is complicated and slow
    - Model physics is quicker
- Collision Detection
  - Many points per each object to compare
    - Break object down into smaller pieces

Where the Project Stands

- Standalone Physics Engine
  - Interacts with client program
  - Uses own Object and Points Classes (accomplishes plug and play)
- Collision Detection
  - Bounding Box to Bounding Box
  - Object Bounding Sphere Intersections
  - Point Collision

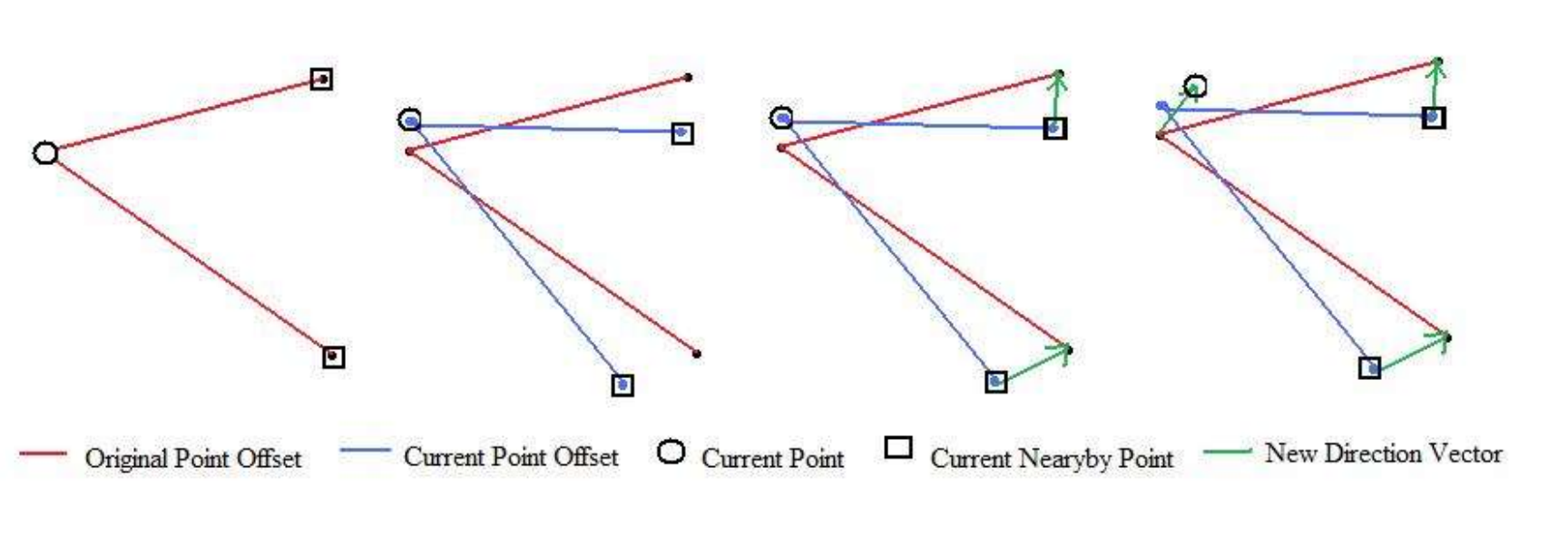
Where Project Stands Cont.

- Collision Reaction
  - Knows current velocities of objects
  - Uses idea of friction to slow objects movement
  - “Jiggle” effect

Shape Matching Algorithm

- Current Point
  - Calculate a list of nearby points
  - Find vectors that connect to nearby points
- When nearby points move
  - Find new direction vector of nearby point
    - new position – old position
  - Find new direction vector of current point
    - summation of nearby points vectors / number of nearby points

Shape Matching Sequence



Issues With Current Design

- Processing slow for large objects
- Does not use “real Physics”
  - Models physics
- Rotations

Where to Continue

- Collision Reaction
  - Attempt for even more realistic model of physics.
- Include rotation collisions

Demo!