Physically-based animation
I – Continuous collision detection

Stephane Redon
NANO-D – INRIA

November 22, 2010 – M2R-MOSIG
Continuous collision detection

- Introduction
- Arbitrary In-Between Motions
- Interval Arithmetic
- Elementary Continuous Collision Detection
- Continuous Collision Detection for Bounding volumes
Continuous collision detection

• Introduction

• Arbitrary In-Between Motions

• Interval Arithmetic

• Elementary Continuous Collision Detection

• Continuous Collision Detection for Bounding volumes
Introduction

• Collision detection
  • Virtual reality
  • Games
  • Graphics
  • Robotics
  • CAD/CAM (virtual prototyping)
Introduction

• Videos
  • Avatar [Redon et al. VR 2004]
  • Chessman [Zhang et al. SIGGRAPH 2007]
  • Train [Zhang et al. SIGGRAPH 2007]
Introduction

• Context
  • Rigid/Articulated/Deformable models
  • Complex environments (several thousands of triangles)
“Brooks House Model”
120,000 triangles
Introduction

• Context
  • Rigid/Articulated/Deformable models
  • Complex environments (several thousands of triangles)

• Discrete collision detection
Introduction

• Context
  • Rigid/Articulated/Deformable models
  • Complex environments (several thousands of triangles)

• Problem: Continuous collision detection
  • Discrete methods check at fixed time instants only
  • Some collisions can be missed with thin or fast objects
  • Precise contact information is needed for dynamics
Continuous Collision Detection

Initial position

Final position
Continuous Collision Detection

Initial position

Final position
Valid portion of the trajectory

Continuous Collision Detection
Motion interpolation
Continuous Collision Detection
Initial position
Final position
Interpolation and collision detection
Continuous Collision Detection

Repositioning at the time of first contact
Continuous collision detection

• Introduction

• Arbitrary In-Between Motions

• Interval Arithmetic

• Elementary Continuous Collision Detection

• Continuous Collision Detection for Bounding volumes
Arbitrary in-between Motions

• The actual motion of an object is not known
  • user actions are sent through a sampling interface
  • dynamics laws are discretized

• We use an arbitrary in-between motion which
  • interpolates successive available configurations
  • satisfies the applications constraints (object rigidity…)
  • is simple enough to perform all the stages of the collision detection pipeline efficiently
Arbitrary in-between Motions

a, c) Actual trajectory      b, d) Arbitrary in-between motions
Arbitrary in-between Motions

- Arbitrary in-between motion for a rigid body

\[
S(t) = P^{-1}V(t)P \quad \text{for} \quad t \in [0, 1]
\]

\[
V(t) = \begin{pmatrix}
\cos(\omega.t) & -\sin(\omega.t) & 0 & 0 \\
\sin(\omega.t) & \cos(\omega.t) & 0 & 0 \\
0 & 0 & 1 & s.t \\
0 & 0 & 0 & 1
\end{pmatrix}
\]
Arbitrary in-between Motions

Problems for long motions

B: Articulated-body motion
Continuous collision detection

• Introduction
• Arbitrary In-Between Motions
• Interval Arithmetic
• Elementary Continuous Collision Detection
• Continuous Collision Detection for Bounding volumes
Interval Arithmetic

- Compute with intervals instead of real numbers
  - Simple interval \( I = [a, b] = \{ x \in \mathbb{R}, a \leq x \leq b \} \)
Interval Arithmetic

• Compute with intervals instead of real numbers
  • Simple interval \( I = [a,b] = \{ x \in \mathbb{R}, \ a \leq x \leq b \} \)
  • Vector interval

\[
I_n = [a_1,b_1] \times \ldots \times [a_n,b_n]
= \{ x = (x_1,\ldots,x_n) \in \mathbb{R}^n, \ a_i \leq x_i \leq b_i \quad \forall i, \ 1 \leq i \leq n \} \]
Interval Arithmetic

- Compute with intervals instead of real numbers
- Basic operations on intervals

\[
\begin{align*}
[a,b] + [c,d] &= [a + c, b + d] \\
[a,b] - [c,d] &= [a - d, b - c] \\
[a,b] \times [c,d] &= [\min(ac, ad, bc, bd), \max(ac, ad, bc, bd)] \\
1/[a,b] &= [1/b, 1/a] \quad \text{if } a > 0 \text{ or } b < 0 \\
[a,b] / [c,d] &= [a,b] \times (1/[c,d]) \quad \text{if } c > 0 \text{ or } d < 0 \\
[a,b] \leq [c,d] &= b \leq c
\end{align*}
\]
Interval Arithmetic – Example

• Obtain bounds for \( f(t) = 2\cos(t) + \sin(t) \) over \([0, \pi/2]\).
  • \( t \) in \([0, \pi/2]\) \(\rightarrow\) \( \cos(t) \) in \([0,1]\)
    \(\rightarrow\) \( 2\cos(t) \) in \([0,2]\).

• Similarly, \( \sin(t) \) in \([0,1]\).

• So \( f(t) \) in \([0,2] + [0,1] = [0,3]\).

  Note: The tight bound is actually \([1,2]\)
Continuous collision detection

• Introduction
• Arbitrary In-Between Motions
• Interval Arithmetic
• Elementary Continuous Collision Detection
• Continuous Collision Detection for Bounding volumes
Elementary CCD

• Three possible elementary contacts
  • Edge / Edge
  • Vertex / Face
  • Face / Vertex
Elementary CCD

- Three possible elementary contacts
  - Edge / Edge
  - Vertex / Face
  - Face / Vertex

\[ a(t)b(t) \land c(t)d(t) \]

\[ a(t)c(t) \cdot (a(t)b(t) \land c(t)d(t)) = 0 \]
Three possible elementary contacts

- Edge / Edge
- Vertex / Face
- Face / Vertex

\[ a(t)b(t) \cdot (b(t)c(t) \land b(t)d(t)) = 0 \]
Elementary CCD

- Interval arithmetic is used to determine the roots

\[
\begin{align*}
\mathbf{a}(t)\mathbf{c}(t) \cdot (\mathbf{a}(t)\mathbf{b}(t) \land \mathbf{c}(t)\mathbf{d}(t)) &= 0 \\
\mathbf{a}(t)\mathbf{b}(t) \cdot (\mathbf{b}(t)\mathbf{c}(t) \land \mathbf{b}(t)\mathbf{d}(t)) &= 0
\end{align*}
\]

Time-dependent equations solved by interval arithmetic

B: Pseudo-code for elementary CCD
Continuous collision detection

• Introduction
• Arbitrary In-Between Motions
• Interval Arithmetic
• Elementary Continuous Collision Detection
• Continuous Collision Detection for Bounding volumes
Bounding Volumes
Bounding Volumes
CCD for Bounding Volumes

Hierarchy of oriented bounding boxes
CCD for Bounding Volumes

Separating axis test for two oriented bounding boxes
CCD for Bounding Volumes

- Continuous overlap test over \([t_0, t_1]\)

\[
|\mathbf{a} \cdot \mathbf{T}_A \mathbf{T}_B| > \sum_{i=1}^{3} a_i |\mathbf{a} \cdot \mathbf{e}_i| + \sum_{i=1}^{3} b_i |\mathbf{a} \cdot \mathbf{f}_i|
\]

\([l_1, l_2] \quad \in [r_1, r_2]

The boxes are separated over \([t_0, t_1]\) if \(l_1 > r_2\)