Parametric Character Animation in Crysis 2

Sven Van Soom
Senior Animation Programmer
Talk Overview

1. The Challenge of Interactivity
2. Parametric Blending
   - What is it?
   - Our Solution
3. Parametric Animation in Crysis 2
4. Beyond Parametric Animation
Part 1

The Challenge of Interactivity
Animation Blending

Transitions

Interpolations

Layering

All 3 methods use a direct mapping of Blend-Weights to Motion-Data
With simple manipulation of these Blend-Weights, we can get interactive animation-control.

What are the typical issues?
Issues with Animation Blending

1 – Blend weights can be complex to calculate
2 – Blend weights are not intuitive
3 – Blend weights can give unpredictable results

Is this always a problem?
When is it a problem?
Part 2

Parametric Blending
Parametric Blending

1. **What is it?**
   - An extension of animation blending
   - A method to create predictable blending-results

2. **How does it work?**
   - It uses the captured properties of a motion-clip directly
   - It generates the blend-weights in relation to these properties

3. **What can we use it for?**
Parametric Blending

Some Applications in Games

Parametric Interpolations

Parametric Transitions

Parametric Layering

Predictable Interactive Animation
Parametric Blending

Some Applications in Games

Parametric Interpolations
Parametric Transitions
Parametric Layering

Predictable Interactive Animation
Parametric Blending
Some Applications in Games

Parametric Interpolations
Parametric Transitions
Parametric Layering

Predictable Interactive Animation

video
Parametric Blending

Some Applications in Games

- Parametric Interpolations
- Parametric Transitions
- Parametric Layering

Predictable Interactive Animation
Animation vs Parametric Blending

The hard part is to generate Correct Blend-Weights and Natural Results!
Getting both at the same time can be an extremely difficult process
The 5 Features a Parameterizer must have!

1. Accurate Parameter Mapping
2. Artist Directed Blending
3. Continuous Control
4. Runtime Efficient
5. Memory Efficient

Conclusion: if only one of these features is missing, then it’s very hard to use it in game productions.
Virtual Example Grids

Overview

• Process
  • Offline
  • Runtime

• Issues when using a lot of parameters
• Solutions
Virtual Example Grids

The Offline Process
Virtual Example Grids

The Offline Process

Step 1: Asset Selection
Virtual Example Grids

The Offline Process

Step 2:
Parameter Extraction

- Move-speed (m/sec)
- Turn-speed (rad/sec)
- TravelDir (rad)
- SlopeAngle (rad)
- TravelDistance (meter)
- TurnDistance (radian)
- JumpHeight (radian)
- Aim-DirectionX (radian X)
- Aim-DirectionY (radian Y)
- Look-DirectionX (radian X)
- Look-DirectionY (radian Y)
Virtual Example Grids

The Offline Process

Step 3:
Setup of the Blend Space
Virtual Example Grids

The Offline Process

Step 4: Blending Annotations

Weird Issue:
Different combinations of Blend Weights, can give you a blended motion with Identical Parameters, but totally different Visual Poses.
Virtual Example Grids

The Offline Process

Advantages of Annotations

1. Artist Directed Blending
2. No “Scattered Data Interpolation” Problem
3. Continuous Control
4. Control over Performance
5. Simple, Precise and Easy to Debug
Virtual Example Grids

The Offline Process

Step 5:
Extrapolated Pseudo Examples
Virtual Example Grids

The Offline Process

Memory: 972 Bytes

Memory: 5880 Bytes
Virtual Example Grids

The Runtime Process
Virtual Example Grids

The Runtime Process

Step 1:
Parameterization:
Virtual Example Grids

The Runtime Process

Step 2: Time Warping
Virtual Example Grids

The Runtime Process

Step 3:
Pose Blending

\[
\begin{align*}
BWeight0 \times MotionDB[AnimTime, AnimID0] & + \\
BWeight1 \times MotionDB[AnimTime, AnimID1] & + \\
BWeight2 \times MotionDB[AnimTime, AnimID2] & + \\
BWeight3 \times MotionDB[AnimTime, AnimID3] & + \\
BWeight4 \times MotionDB[AnimTime, AnimID4] & + \\
\ldots
\end{align*}
\]

Final Blended Motion
Curse of Dimensionality

• Exponential Asset Explosion
  1D - 3 assets for move speed
  2D - 9 assets for move speed / turn left-right
  3D - 27 assets for speed / turn left-right / uphill-downhill
  4D - 27*8 assets for speed / turn left-right / uphill-downhill multiply by 8 move directions
  
  = 216 (for 1 parametric group)

  - This is the bare minimum for a full featured character, regardless of the blending method.
  - Our practical maximum was 34 assets per group
  - Extra- and Interpolations help a lot to reduce the asset count.

• Debugging Nightmare
  - More than 3 dimensions are hard to visualize & debug
  - Dimensionality Problem is the **Dead End** for Parametric Blending
But 3D is not enough!
- with 3D you have only 3 Parameters to control
- in a game you will need much more

What’s the Solution?
- build small Blend Spaces and combine them
- or we can layer Blend Spaces
### Combined Blend Spaces

**An example of a 4D Blend-Space**

- 1D-MoveSpeed
- 2D-Travel-Direction
- 3D-TurnSpeed
- 4D-SlopeAngle

**A combination of two 3D Blend-Spaces**

<table>
<thead>
<tr>
<th>50%</th>
<th>50%</th>
</tr>
</thead>
<tbody>
<tr>
<td>1D-MoveSpeed</td>
<td>1D-MoveSpeed</td>
</tr>
<tr>
<td>2D-Travel-Direction</td>
<td>2D-Travel-Direction</td>
</tr>
<tr>
<td>3D-TurnSpeed*2</td>
<td>3D-SlopeAngle*2</td>
</tr>
</tbody>
</table>
Layered Blending

• The Layer Model
• Types of Layered Animations
  - Overwrite Animations
  - Additive Animations
  - Combination of both Methods in one Asset
Parametric Blending used in Layers

- Parametric Weapon Aiming
Virtual Example Grids

Summary

• We used only **small** Blend Spaces (max 3D)
• With **combinations** it was possible to control 4D
• With **layering** it was possible to control up to 8D
Part 3

Parametric Animations in Crysis 2
Overview

• Main Changes Compared to Crysis 1
  • 3rd Person Player (Multiplayer)
  • AI
• Parametric Animation To Decorate AI
3rd Person Player (Multiplayer)

- Crysis 1: Decoupled Skeleton & Entity
- Crysis 2: Skeleton is Entity Driven
- Parametric Turning
Changes to AI

- Also Entity Driven (*)
  
  (*) but sometimes Animation Driven

- Parametric Turning

- Improved, Parametric Aiming/Gaze Control
AI – Beyond Direct Control

• Goal-Driven Control
  • This means we can Plan and Select better animations
Using the Goals

• Goals ≠ detailed steering information
• Pass along AI’s Goals to animation controller
  • Current Path Segment & Upcoming Corner/Stop
  • Information about future aim/look targets
  • Context
• Match against ‘decorative’ transition animations (Start/Stop/Direction-Change)
Decoration Selection

- Simple matching rules
  - No ‘priority’ calculation
  - Usable by non-programmers & Easy to Debug

<table>
<thead>
<tr>
<th>Type</th>
<th>Alertedness</th>
<th>Distance</th>
<th>Angle</th>
<th>Speed</th>
<th>Context</th>
<th>Signal</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start</td>
<td>Relaxed</td>
<td>1m (+-.1)</td>
<td>90 (+10)</td>
<td>walk</td>
<td>*</td>
<td>RLX_START</td>
</tr>
<tr>
<td>DirectionChange</td>
<td>Relaxed</td>
<td>1.5m(+-.1)</td>
<td>40 (+25)</td>
<td>run</td>
<td>*</td>
<td>RLX_JUKE</td>
</tr>
<tr>
<td>Stop</td>
<td>Combat</td>
<td>1.2m(+.05)</td>
<td>run</td>
<td></td>
<td>GoingIntoCover</td>
<td>COM_SLIDE_INTO_COVER</td>
</tr>
<tr>
<td>Stop</td>
<td>Combat</td>
<td>0.5m(+-.05)</td>
<td>run</td>
<td></td>
<td>*</td>
<td>COM_STOP</td>
</tr>
</tbody>
</table>
Voodoo

- Temporarily Animation Driven
- Covering Up Transitions
- Preparation Period for Alignment
- Straying off Path..
Straying off Path

• Path Follower is able to handle this
  • Physics
  • Hit Reactions, knock down, …
  • Avoidance

• E.g. Decorate/Smooth sudden direction changes
• Check Environment!
Part 4

Beyond Parametric Animations
Procedural Animations

Programmers design an algorithm

- Kinematic Methods
  Combination of simple IK-solvers
- Physically Based Animations
  - passive
  - active

These algorithms can generate the motion in real-time.

...what about the quality?
Combination of IK Solvers

- IK-Solvers (2B, 3B & CCD-IK) generate new poses
- Procedural Motion Warping

Typical Applications

- Fix of Blending Artifacts
- Ground Alignment
- Recoil
Physically Based Animations

• Just Ragdolls
• Ragdolls & Animation Blending
• Procedural Hit Reactions
• Animated Hit Reactions
• Inverse Dynamics
The Future

• More Power to the Animators
• Improve Procedural Control
• More Goal/Context-driven Animation

• Make “Contract” Between AI & Animation Explicit
• No Reliance on Visual Programming
• Logic in Code (C++)
Summary

1. Animation - Data is the foundation
2. Blend - Spaces and Parametric Animations
3. Annotations
4. Virtual Example Grids
5. Combined and Layered Blend Spaces
6. Simplification: Entity Driven/Less Procedural
7. Decorating AI Animation
8. Procedural Techniques
9. Future
Special thanks for the Help with this Presentation:

Ivo Herzeg

Benjamin Block, Chris Butcher, Daniele Duri, Frieder Erdmann,
Ivo Zoltan Frey, Mathias Lindner, Michelle Martin
Peter North, Pau Novau, David Ramos, Peter Söderbaum,
Alex Taube,
Karlheinz Watermeier
We’re Hiring!

www.crytek.com/career
Questions?