Computer Animation and Visualization

Lecture 6.

Motion Synthesis by Optimization / Motion Editing

Taku Komura
Overview

- Motion synthesis by Optimization
  - Using forward / inverse dynamics
- Motion Editing
  - Motion Warping
  - Motion Blending
  - Dynamic Time Warping
- CMU motion database
From the last lectures...

- Equation of motion \[ F = M\ddot{x} \]
- Forward dynamics
- Inverse dynamics
From the last lectures...

- Equation of motion
  \[ F = M \ddot{x} \]

- Forward dynamics
  - Computing the motion from force.
  - Parameters: force, torque

- Inverse dynamics
From the last lectures...

- **Equation of motion**
  
  \[ F = M \ddot{x} \]

- **Forward dynamics**
  - Computing the motion from force. \( \ddot{x} = F/M \)
  - Parameters: force, torque
    » Require us to set the force/torque well to control the character in the way we wish

- **Inverse dynamics**
From the last lectures...

- Equation of motion
  \[ F = M\ddot{x} \]

- Forward dynamics
  - Computing the motion from force.
  - Parameters: force, torque
    » Require us to set the force/torque well to control the character in the way we wish

- Inverse dynamics
  - Computing the force from motion
  - Parameters: generalized coordinates
From the last lectures...

- Equation of motion
  \[ F = M \ddot{x} \]

- Forward dynamics
  - Computing the motion from force.
  - Parameters: force, torque
    » Require us to set the force/torque well to control the character in the way we wish

- Inverse dynamics
  - Computing the force from motion
  - Parameters: generalized coordinates
    » Require us to make the trajectories follow the laws of dynamics to make the motion realistic
From the last lectures...

- **Equation of motion**
  \[ F = M\ddot{x} \]

- **Forward dynamics**
  - Computing the motion from force.
  - Parameters: force, torque
    » Require us to set the force/torque well to control the character in the way we wish

- **Inverse dynamics**
  - Computing the force from motion
  - Parameters: generalized coordinates
    » Require us to make the trajectories follow the laws of dynamics to make the motion realistic
Summary of the ISSUE

- Equation of motion
  \[ F = M\ddot{x} \]

- Forward dynamics
  - Need to set the force/torque right

- Inverse dynamics
  - Need to set the motion right
Summary of the ISSUE

- Equation of motion
  \[ F = M \ddot{x} \]

- Forward dynamics
  - Need to set the force/torque right

- Inverse dynamics
  - Need to set the motion right

Use Optimization
What is Optimization?

• Given an objective function $f(x)$, find the $x$ that minimizes / maximizes $f(x)$
What is Optimization?

- Solution is found by iteratively computing the gradient at the current point, and moving in that direction.

\[ \frac{\partial f}{\partial x} \]

gradient direction

Optimal solution

parameter space of motion
What is Optimization? with Constraints

- Given an objective function $f(x)$, find the $x$ that minimizes / maximizes $f(x)$
  - Subject to constraints $g(x) = 0$

Optimal solution $g(x) = 0$

Parameter space of the motion
What are the parameters $X$?

• Given some function $f(x)$, find the $x$ that minimizes $f(x)$
  • Subject to constraints $g(x) = 0$

• Forward dynamics: Torque at the joints
• Inverse dynamics: Generalized coordinates
Background

Taking into dynamics for character animation
• Good but the control is difficult
  Torque at the joints,
  Gains, desired joint angles

\[- F = a (q - q_d) + c (q' - q'_d) \]

\(a, c\) : constants,
\(q, q'\) : current state/velocity
\(q_d, q'_d\) : desired state/velocity
Background

Taking into dynamics for character animation
- Good but the control is difficult
  Torque at the joints,
  Gains, desired joint angles

\[-F = a(q - q_d) + c(q' - q'_d)\]

- $a, c$ : constants,
- $q, q'$ : current state/velocity
- $q_d, q'_d$ : desired state/velocity

Use Optimization
Motion Synthesis by Optimization

Why?

We want the characters to satisfy the laws of dynamics
- Newton’s law, preservation of momentum

We want characters not to spend too much energy
- Humans move efficiently

We also want them to satisfy constraints
- passing some specific postures, feet on the ground etc.
What is Optimization?

- Given an objective function \( f(x) \), find the \( x \) that minimizes / maximizes \( f(x) \)
What is Optimization?

- Solution is found by iteratively computing the gradient at the current point, and moving in that direction.
What is Optimization?
with Constraints
• Given an objective function $f(x)$, find the $x$ that minimizes / maximizes $f(x)$
  • Subject to constraints $g(x) = 0$

Optimal solution $g(x) = 0$
f(x)

parameter space of motion
Issue 1: High Dimensionality

- The body has many degrees of freedom
- There are many frames in one motion
- Even we simplify the control (reducing the number of keyframes), the problem can still be very high dimensional

i.e. motion

\[
\begin{align*}
q_1 &= (q_1, q_2, q_3, q_4, q_5, q_6, q_7, \ldots, q_m) \\
q_2 &= (q_1, q_2, q_3, q_4, q_5, q_6, q_7, \ldots, q_m) \\
& \vdots \\
q_n &= (q_1, q_2, q_3, q_4, q_5, q_6, q_7, \ldots, q_m)
\end{align*}
\]

- \( m \times n \) parameters

\[
\begin{align*}
F_1 &= (\tau_1, \tau_2, \tau_3, \ldots, \tau_{m-6}) \\
F_2 &= (\tau_1, \tau_2, \tau_3, \ldots, \tau_{m-6}) \\
& \vdots \\
F_n &= (\tau_1, \tau_2, \tau_3, \ldots, \tau_{m-6})
\end{align*}
\]

- \((m-6) \times n\) parameters
Issue 2: Local Extrema

- Functions with many parameters can have lots of local extrema
- An ordinary optimization method can get trapped in local extrema
Sampling a lot of points around the current point to decide the direction to further search
Optimization in Forward Dynamics

• Compute the trajectories of torque / PD control parameters such that the character performs the best action
• Need to specify the criteria that evaluates the motion
  • Height of the jump
  • How stable the character can walk
  • How similar the motion is to the motion capture data

http://www.youtube.com/watch?v=9pGH6-QY-sQ&feature=player_embedded
Optimization in Inverse Dynamics

• Compute the trajectories of the body such that the motion follows the laws of physics
Optimization in Inverse Dynamics

• Compute the trajectories of the body such that the motion follows the laws of physics
  – What kind of motion is following laws of physics?
  – What kind of motion is not following laws of physics?
Optimizing the Motion: Spacetime Constraints

• Compute the trajectories of the body such that the momentum of the whole body is preserved while the body is flying in the air
  • When objects / bodies fly in the air, the linear / angular momentum is preserved

\[ C_l = \sum_{i: \text{body}} M_i \dot{x}_i = \text{constant} \]
Optimizing the Motion: Spacetime Constraints

• Compute the trajectories of the body such that the momentum of the whole body is preserved while the body is flying in the air
  • When objects / bodies fly in the air, the linear / angular momentum is preserved

\[ C_a = \sum_{i:\text{body}} (x_i - x_c) \times M_i (\dot{x}_i - \dot{x}_c) = \text{constant} \]
Optimizing the Motion: Spacetime Constraints

• Minimum energy
  – We wish to not move our body too much
  – We can minimize the torque at the joints

\[ E = \sum_{i:\text{joints}} T_i^2 \]
Optimizing the Motion: Spacetime Constraints

• Compute the motion \( (q) \) such that the body uses minimum energy during the motion, while following the laws of physics,

\[ \text{Min } E \text{ s.t. } C_a = a, C_l = b \]
Optimization in Inverse Dynamics

• Compute the trajectories of the body such that the amount of torque exerted by joints is minimal
  • Fang, Pollard ‘03
Discussions

• Optimization in Forward Dynamics
  • Finding a good initial motion is not easy
  • Need to keep the balance
• Optimization in Inverse Dynamics
  • Easier to start as the motion does not have to satisfy dynamic rules
  • Need a good objective function that well describes the motion
• In both cases, the cost of computation is very high
  • Due to the large number of variables
Overview

- Motion synthesis by Optimization
  - Using forward / inverse dynamics
- **Motion Editing**
  - Motion Warping
  - Motion Blending
  - Dynamic Time Warping
- CMU motion database
Motion Editing : Background

• We learnt about motion capture
• It can provide realistic movements for characters
• However,…
Animators are not so happy with the raw motion capture data

• There might be obstacles in the scene – the character needs to avoid them while walking
• The character’s body is different from those of humans
  · The hand not reaching the right position
  · Need to “retarget” the motion to the character size while maintaining constraints
• The style of motions are not stylised
  · Want to exaggerate some expressions
Editing motions

- Need an efficient method to edit the trajectories of the body
- Whilst satisfying constraints
- Must be interactive – the tools are used by animators
Motion Warping

- Edit the captured motion a little bit so that it satisfies the requirements
  - Effective for changing the location the hand or the foot passes
Basic idea

- Adding offset to the data so the constraint is satisfied

Warped Motion = original motion + offset

Offset can be a simple 1D motion
Motion Warping

- Add an offset to the trajectories to edit the motion
  - Determine the duration the motion should be edited
  - Insert a keyframe so that the motion satisfies the requirements
Motion Warping

- Very simple
- Good for avoiding obstacles

http://www.youtube.com/watch?v=BzfxGwO7swg
http://www.youtube.com/watch?v=Ro30uWsWl_s
How to satisfy constraints?

• We warp the motion in the level of generalized coordinates (root position, root orientation and joint angles)
  • No information about contacts
The supporting foot/feet slides when the motion is simply blended

http://www.youtube.com/watch?v=KumgRt895EU
Inverse Kinematics

• Editing the postures/motions by specifying the 3D location of the avatars
• We can use IK to solve the problem of foot sliding
• Edit the postures, frame by frame, and drag the foot to the original position
Motion Blending

• Given two different motions A and B
  Blended Motion = A * (1-s) + B *s  (0<s<1)
• How can we use this?
  • Generate a motion in between
  • Gradually shift from motion A to motion B
  • Concatenating two motions – blending the two ends by gradually shifting s from 0 to 1
Motion Blending: Blending Multiple Motions

Given example motions $M_1, M_2, ..., M_n$ we blend them by weights $W = (w_1, w_2, ..., w_n)$ such that

$$w_1 + w_2 + ... + w_n = 1$$

And then blend the motion by

$$w_1 M_1 + w_2 M_2 + ... + w_n M_n$$
Motion Blending:
Blending Multiple Motions

How do we compute the weights?

Say we have a target hand position $p$.

And the position of the hand in each motion is $(p_1,p_2,...,p_n)$.

The weights $(w_1,w_2,...,w_n)$ can then be computed by

$$w_i = b_i / (b_1 + b_2 + ... + b_n)$$

where

$$b_i = 1 / |p-p_i| - 1 / |p-p_k|$$

$k$ is the farthest sample
Motion Blending: Blending Multiple Motions

Some issues:

There can be issues when the hand target position is far from the samples

Because the linear blending in the generalized coordinates do not result in a linear blend in the task space (Cartesian space, hand position)
Motion Blending: 
Solution:

- Increase the number of samples by linear blending (producing new examples)

- We can use these for new samples

- In the case below,

  \[ W_j = \text{normalize}(W(1) + W(2) + W(3) + W(4)) /4) \]

  as we can interpolate samples in tetrahedra

And we compute the corresponding hand position \( p_j \) accurately
(not by linearly blending \( p(i) \) )
Motion Blending: Solution (2):

- Find the k-nearest neighbours, and then blend them

- As the samples are much denser and as the target is much closer to the samples, a linear blend in weights will result in a linear blend of the hand position

http://www.youtube.com/watch?v=qtNsF-nTp4A (around 0:50)
Another problem when blending motions

• The durations of motions are different
• The timing of some events are different
  • Suppose we blend two walking motions
  • The timing of steps are different
  • Need to synchronize the motions
Dynamic Time Warping (DTW)

- Define difference of postures at every frame in motion 1 and motion 2
  \[ D(q_1, q_2) = d(q_1, q_2) + \alpha d(\dot{q}_1, \dot{q}_2) \]
  like Euclidean distance of generalized coordinates

- Create a similarity matrix of all frames

http://www.youtube.com/watch?v=EL7ARaH5jHU
Dynamic Time Warping (DTW)

- Define difference of postures at every frame in motion 1 and motion 2
- Create a similarity matrix of all frames
- Find the shortest path from the left bottom to the right top in this matrix (assume the matrix is a map) - this gives

http://www.youtube.com/watch?v=EL7ARaH5jHU
Dynamic Time Warping (DTW)

- Define difference of postures at every frame in motion 1 and motion 2.
- Create a similarity matrix of all frames.
- Find the shortest path from the left bottom to the right top in this matrix (assume the matrix is a map) - this gives

http://www.youtube.com/watch?v=EL7ARaH5jHU
Dynamic Time Warping (DTW): Notes

- Only movements to the right and up are allowed (A directed graph)
Carnegie Mellon University
Motion Database

- Various motion data in different topics
- Locomotion
- Physical activity and sports
- Human interactions
- Can search motion by topics
- Can preview the animation at the website
Readings / References

• Witkin and Popovic “Motion Warping”, SIGGRAPH 95
• Witkin and Kass, “Spacetime Constraints”, SIGGRAPH 88
• , SIGGRAPH 2003
• Wang et al., Optimizing Walking Controllers for Uncertain Inputs and Environments, SIGGRAPH 2010
• CMU Motion Database