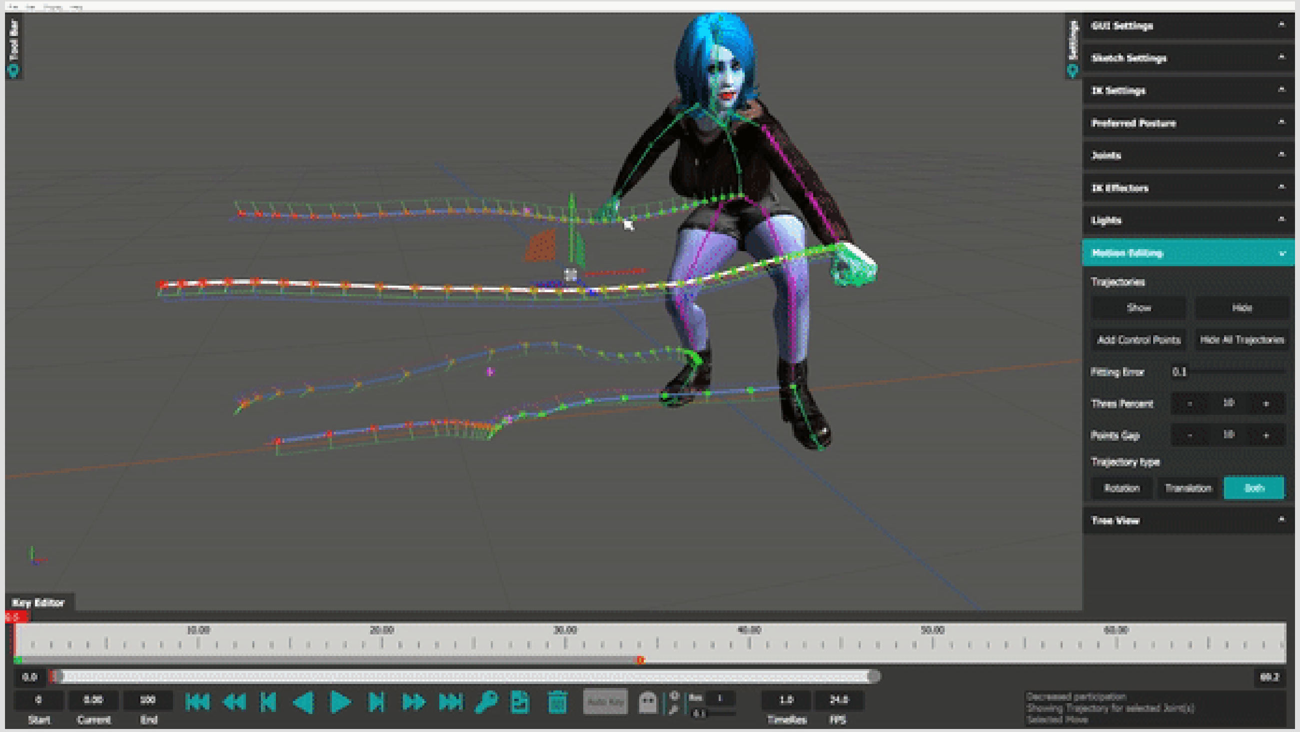
Motion Editing

- Getting the motion you ***want*** from the motion you ***have***

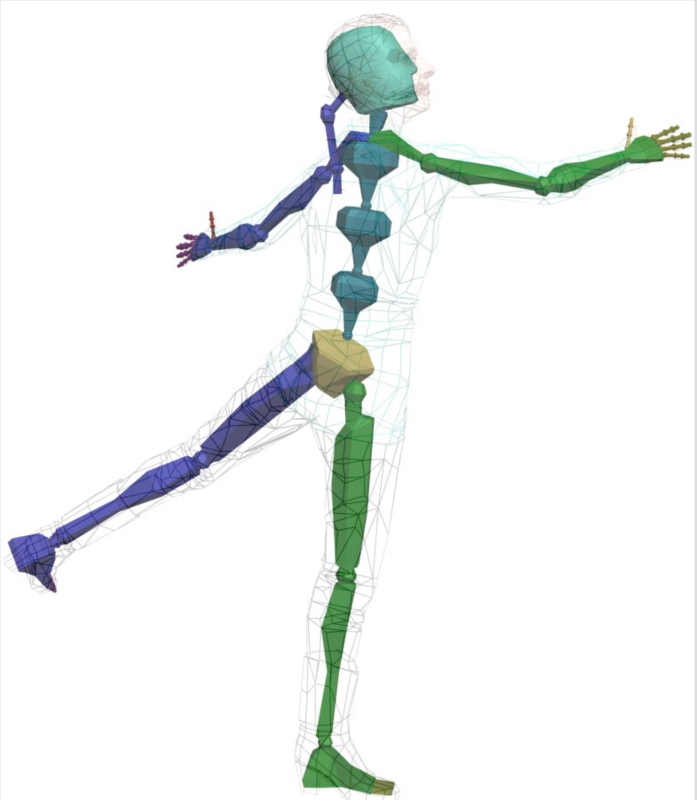
- General techniques
  - Warping
  - Splicing
  - Blending
  - Transplanting



Credit: Muni Animation

Overview

- Motion Editing
  - Warping
  - Splicing
  - Blending
  - Transplanting



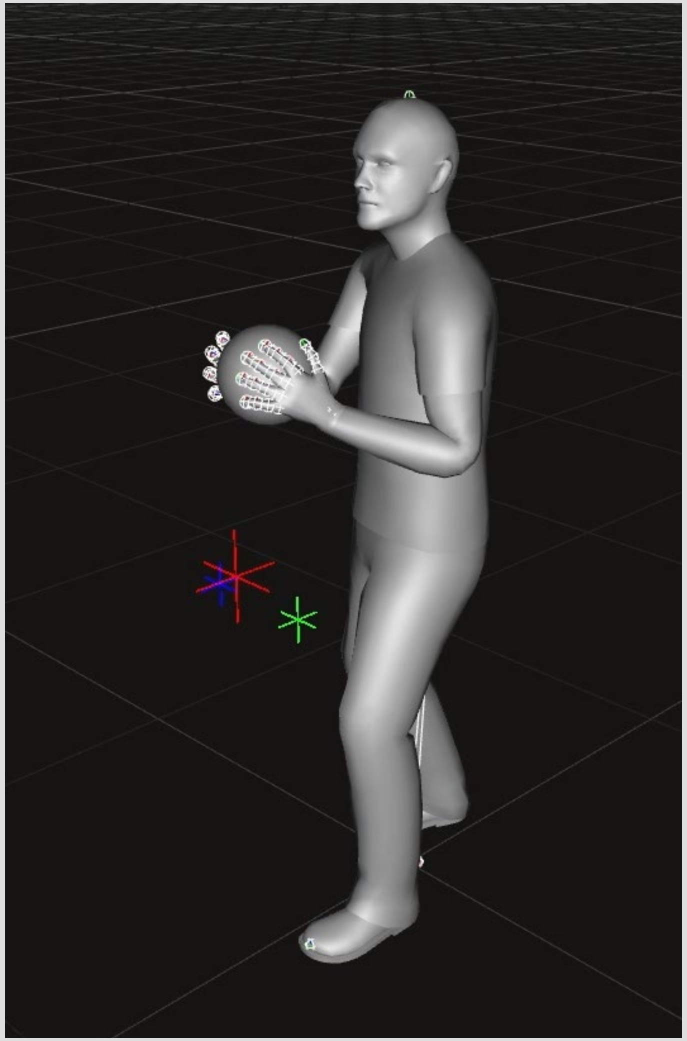
Next lecture:

- *State Machines*
- *Motion Graphs*
- *Neural State Machines*

Warping Motions

- Adapt a single motion
  - e.g., character needs to catch a ball
  - A catching motion is available
  - Problem: that motion catches where ball *isn't*

- Common solution:
  - Inverse kinematics
  - *Displacement splines*



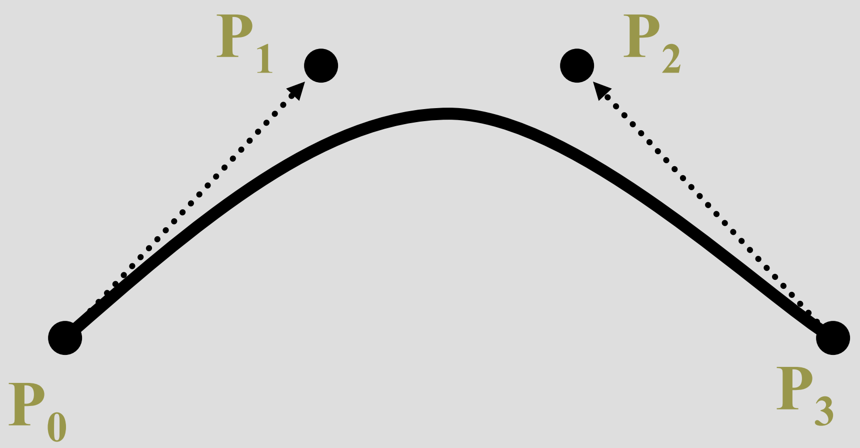
Displacement Splines

- Spline: smooth piecewise polynomial curve
- Displacement: distance away

- *Displacement spline:* smooth curve giving distance away from the original motion data
  - Smoothly moves animation away from original data to target point

Displacement Splines

- e.g., cubic Bezier curve
  - four control points P_i
  - $B(t) = (1-t)^3 P_0 + 3t(1-t)^2 P_1 + 3t^2(1-t) P_2 + t^3 P_3$
  - $B(0) = P_0$
  - $B(1) = P_1$
  - $B'(0) = 3(P_1 - P_0)$
  - $B'(1) = 3(P_3 - P_2)$

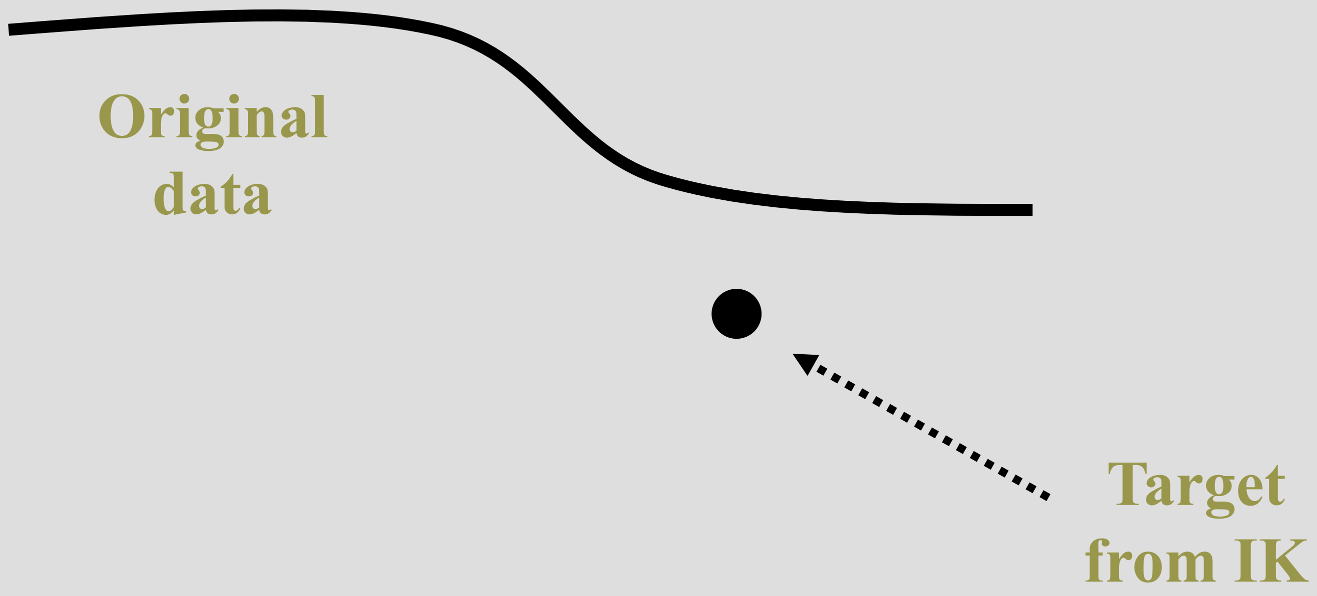


Displacement Splines

- e.g., cubic Bezier curve
  - Create longer curve via sequential cubics
  - Overlap first point of B_{i+1} with last point of B_i to make (C0) continuous curve
  - Overlap two points to make smooth C1 (velocity) continuous curve
  - Smoother curves require more advanced splines (e.g., cubic uniform B-splines)

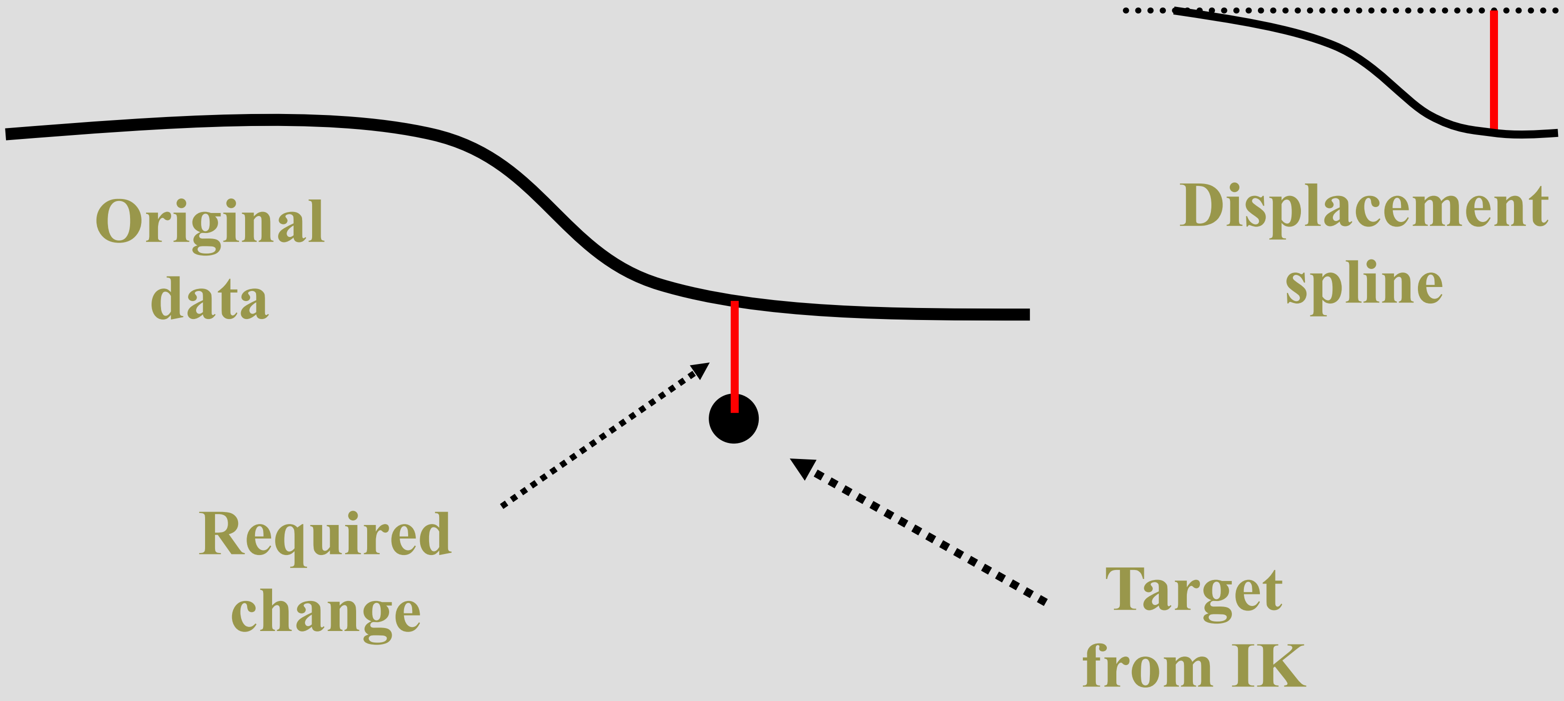
Warping Motions

- e.g., angle of elbow joint



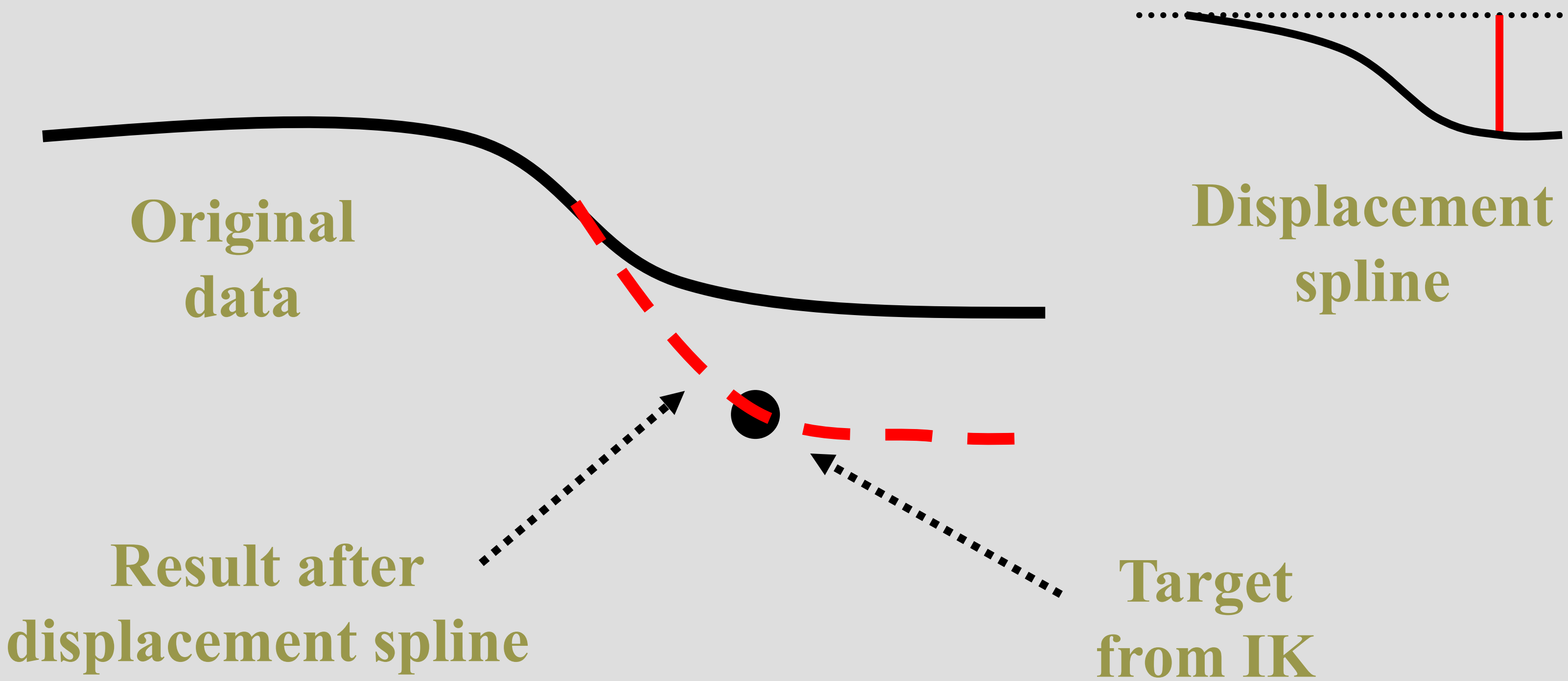
Warping Motions

- e.g., angle of elbow joint



Warping Motions

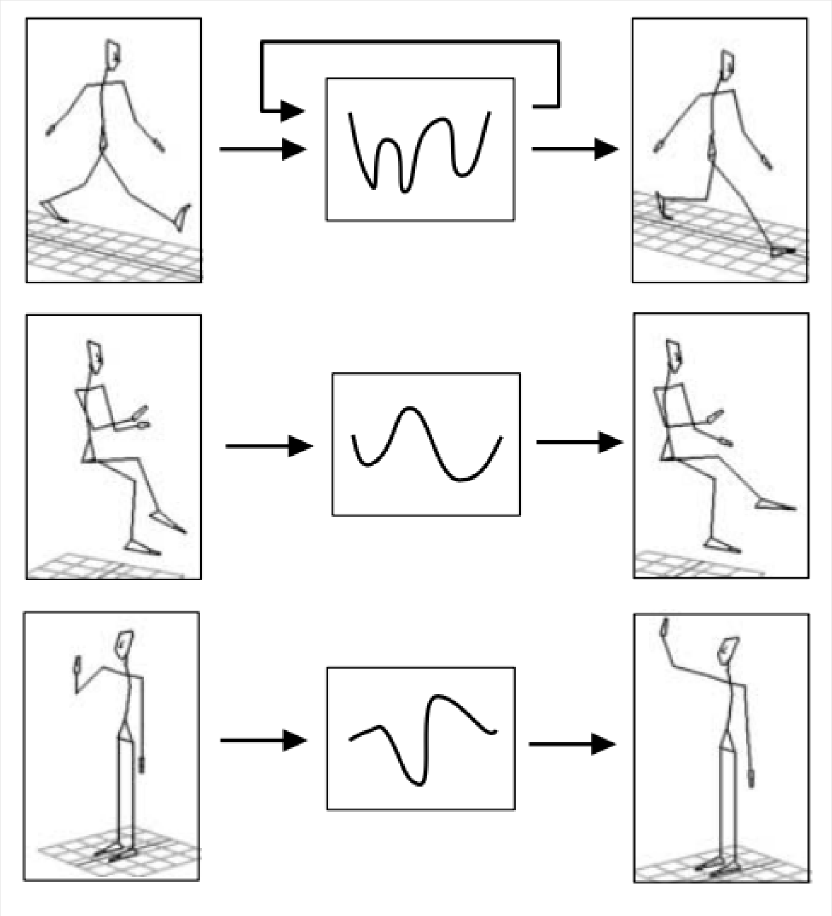
- e.g., angle of elbow joint



Warping Motions

- Also used to adjust locomotion
  - Longer steps
  - Higher jump
  - Proper turning angle
  - Duck under a beam

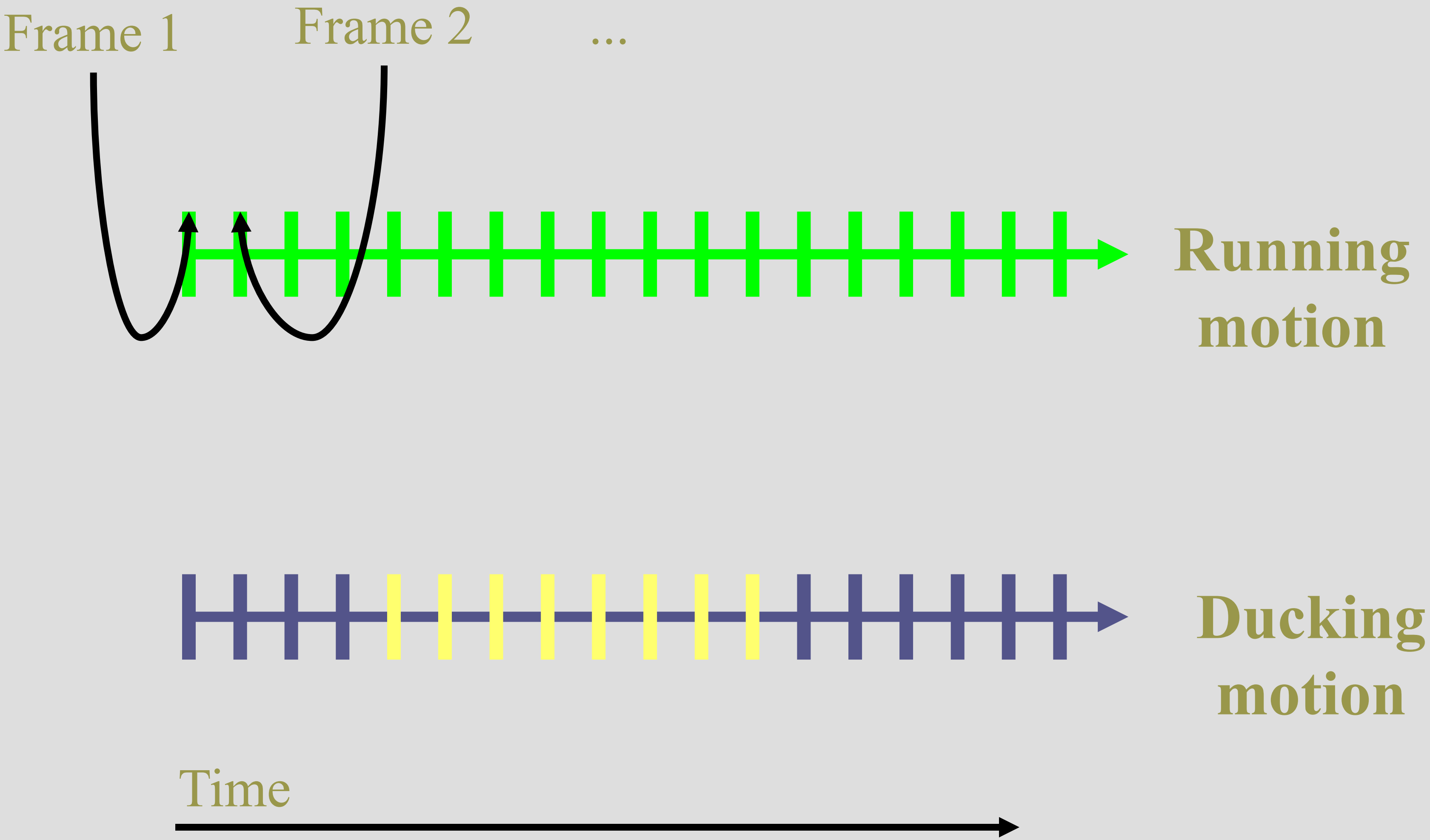
- General approach for adjusting or adding constraints to a single motion



Splicing Motions

- Have running motion
- Have ducking motion
- Want a motion where the character runs and then ducks

Example: Run + Duck

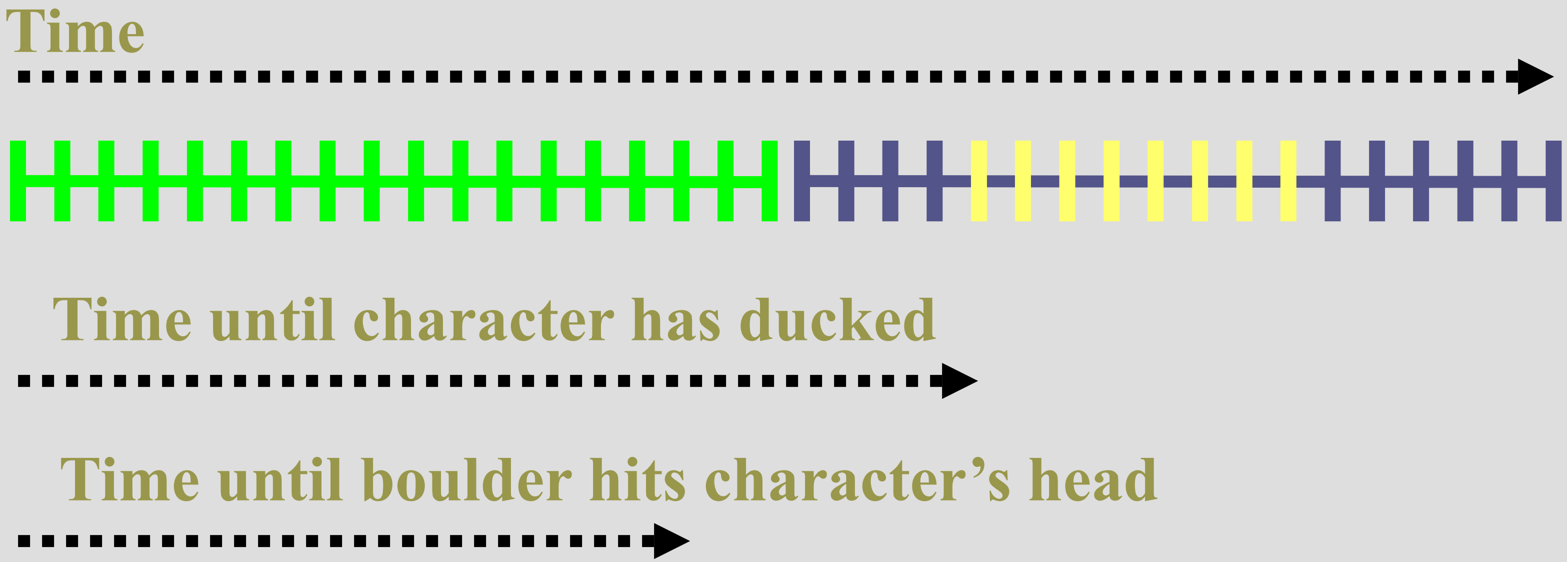


Approaches

1. Play running then ducking clips

Approaches

1. Play running then ducking clips

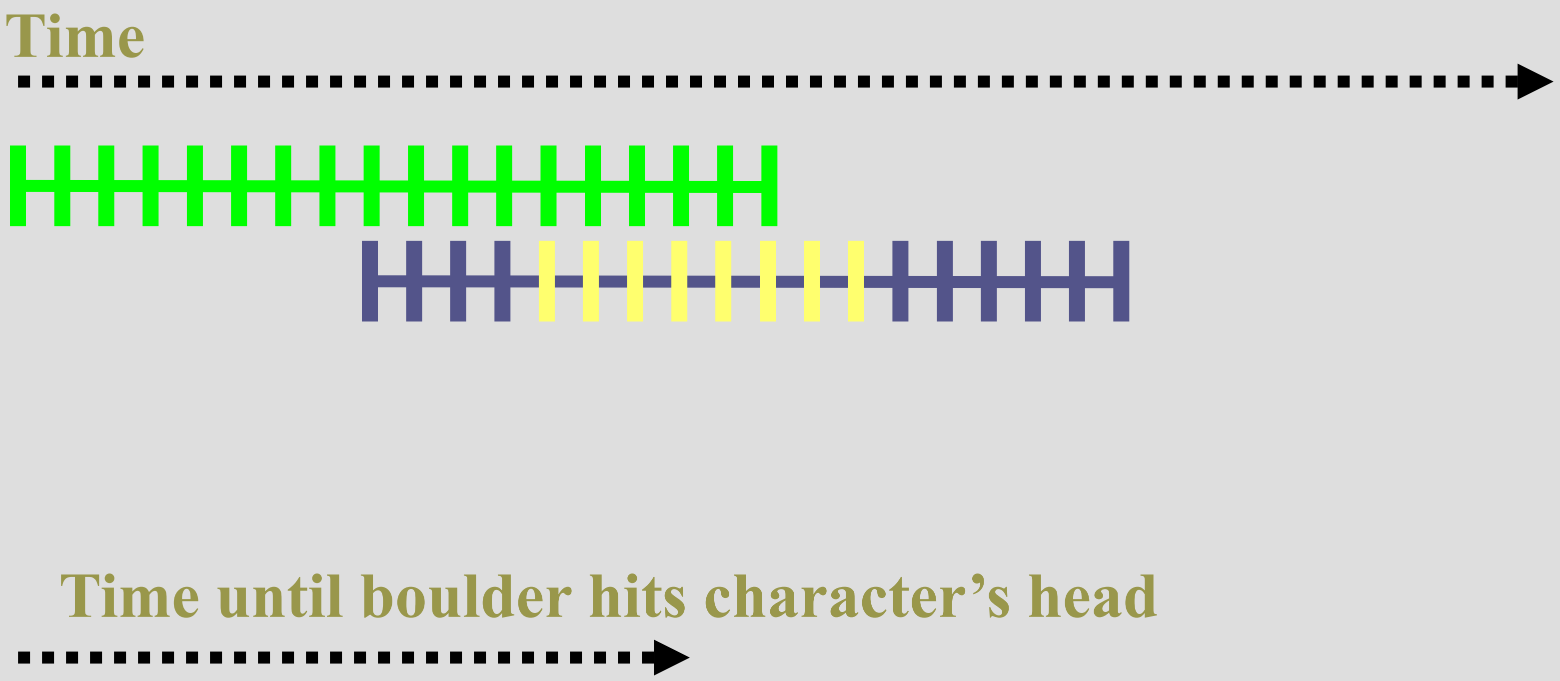


Approaches

1. Play running then ducking clips
    - Problem: no control over timing
2. Cut from running clip to ducking when needed

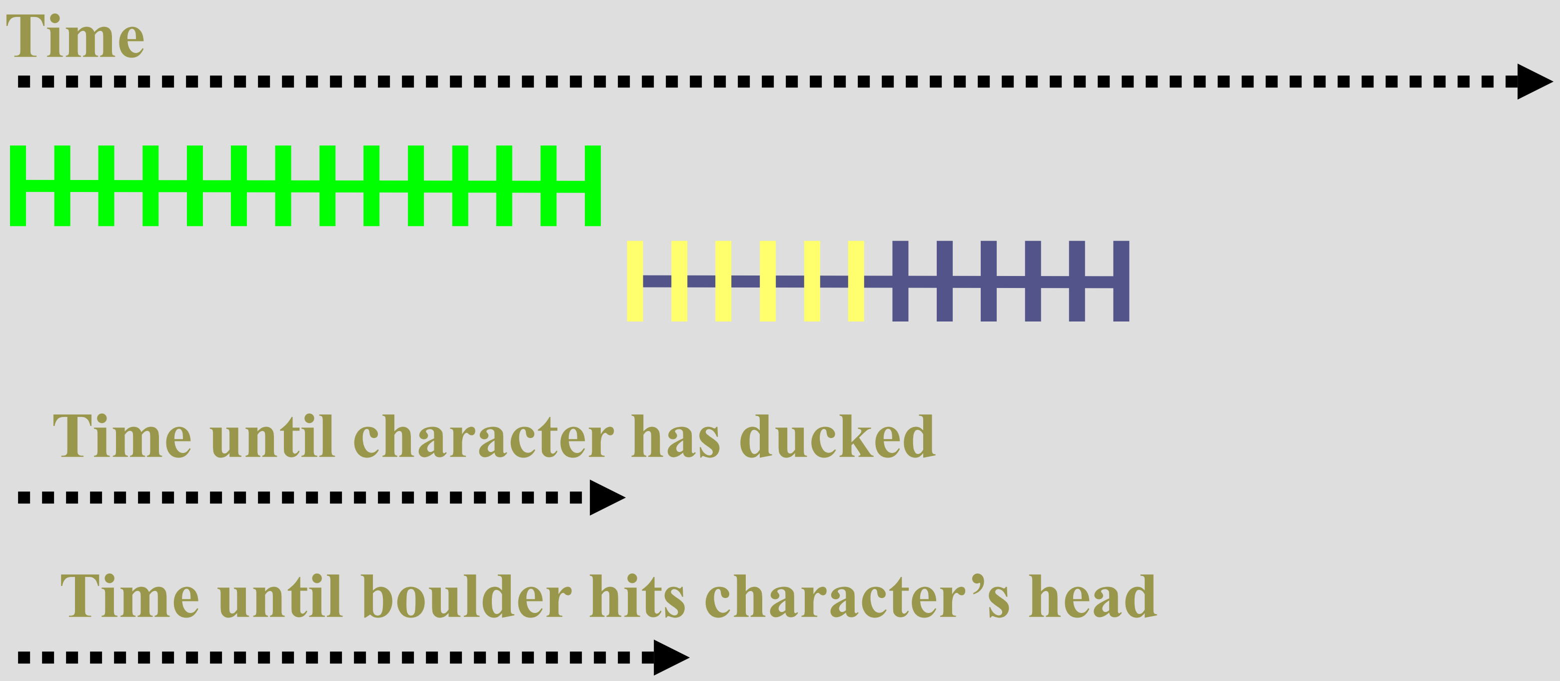
Approaches

1. Play running then ducking clips



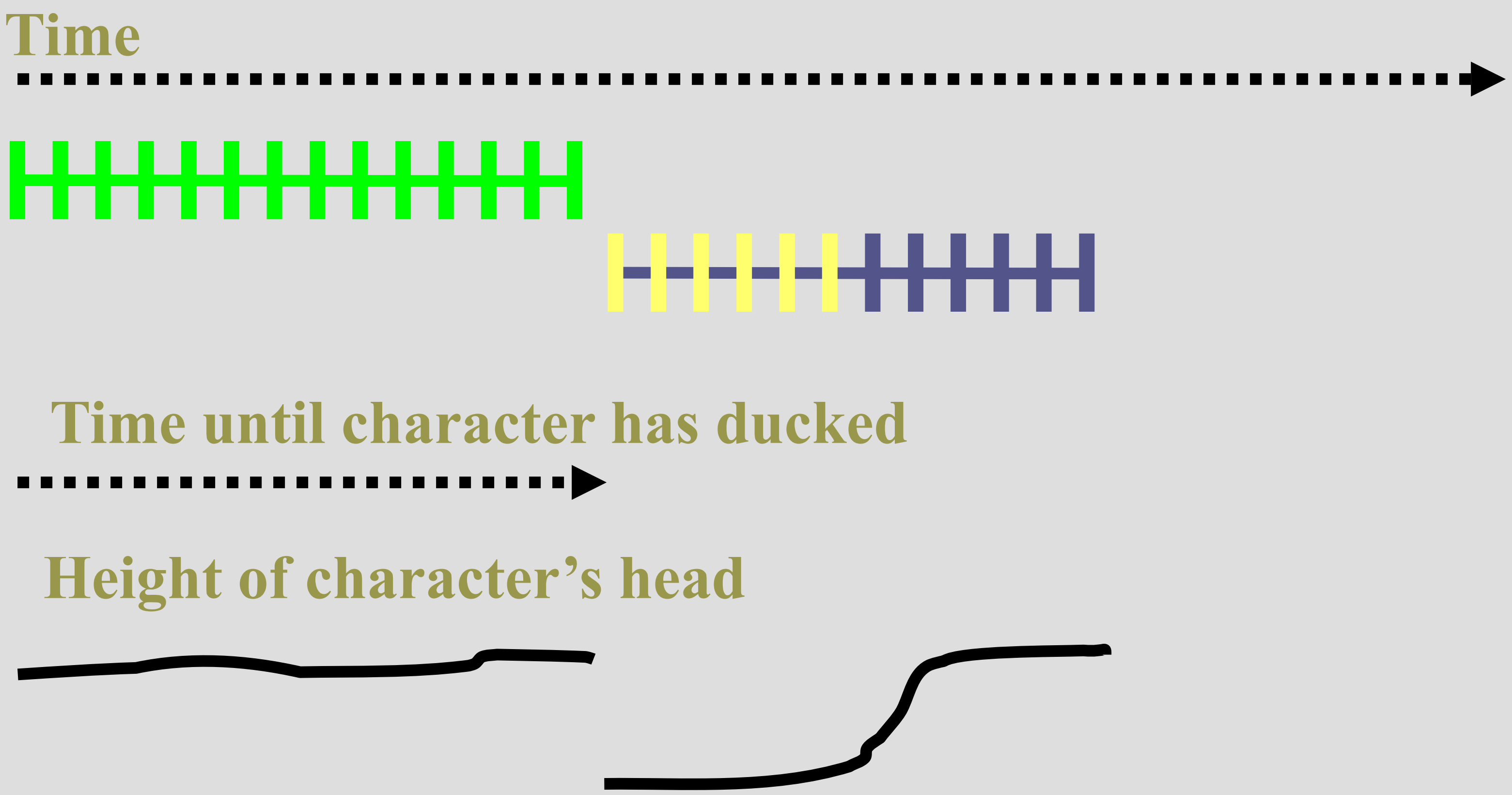
Approaches

1. Play running then ducking clips



Approaches

1. Play running then ducking clips



Approaches

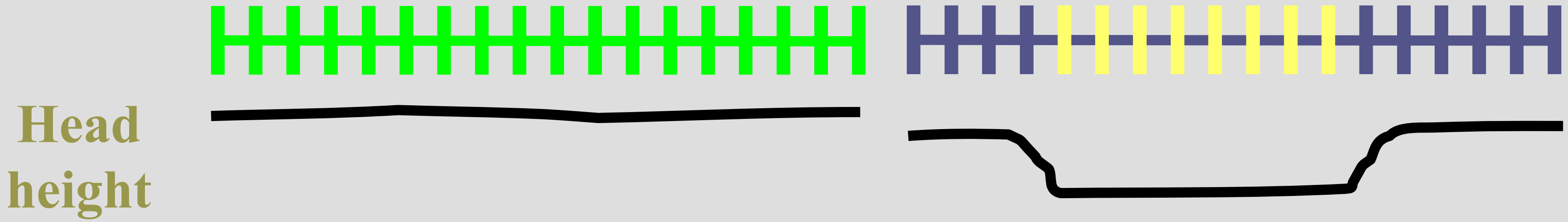
1. Play running then ducking clips
2. Cut from running clip to ducking when needed
    - Problem: bad transitions

Approaches

1. Play running then ducking clips
2. Cut from running clip to ducking when needed
   - Problem: bad transitions
   - *Aesthetic* problem (user preference)

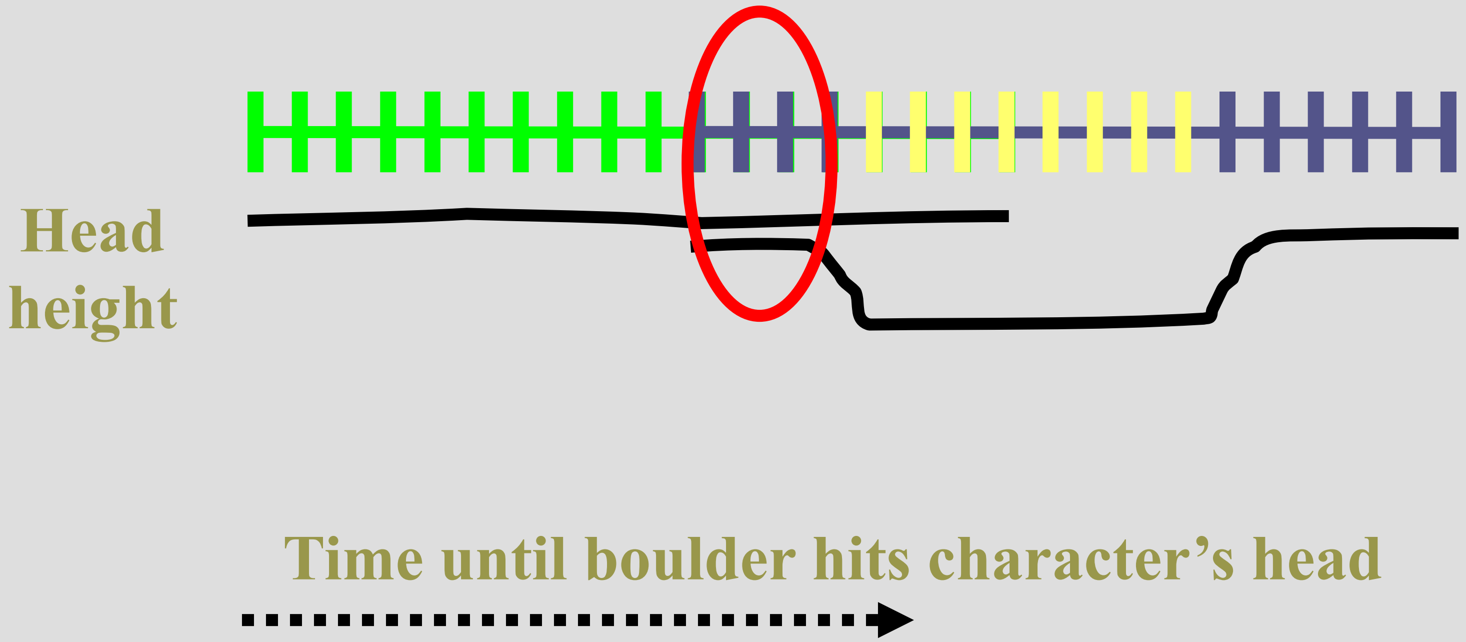
Avoiding Bad Transitions

1. Capture the aesthetics
   - *Motion similarity metric*
   - Combine motions at highly similar points



Avoiding Bad Transitions

1. Capture the aesthetics
   - *Motion similarity metric*
   - Combine motions at highly similar points

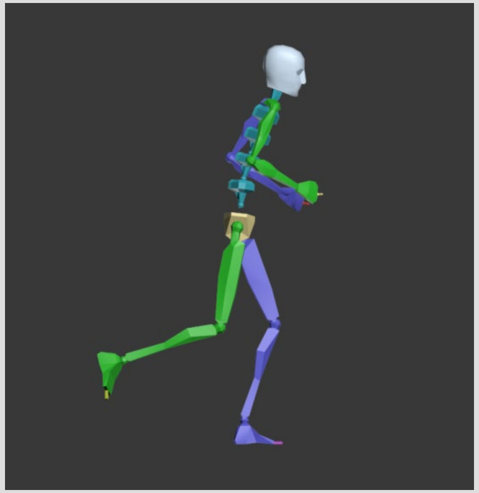
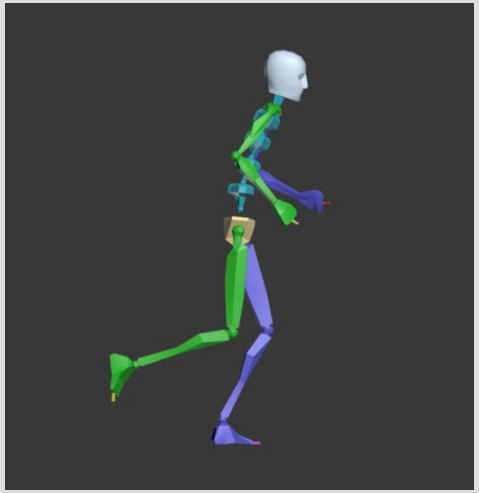
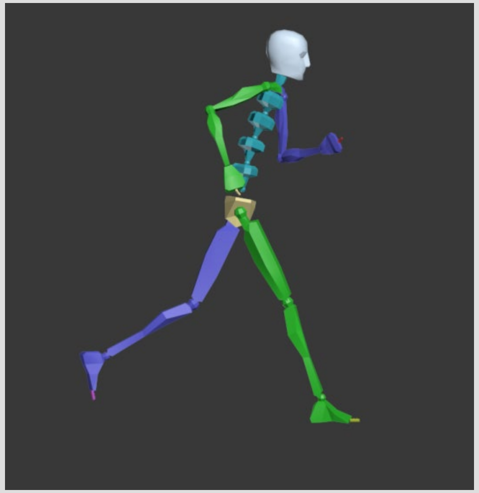


Avoiding Bad Transitions

1. Capture the aesthetics
   - *Motion similarity metric*
   - Combine motions at highly similar points
2. Smooth remaining gaps
   - *Motion smoothing algorithm*

Motion Similarity Metric

- Where are motions similar enough to move from one to the other with minimal disruption?
  - Pose similarity
  - Velocity similarity
  - Interaction similarity

Pose Similarity

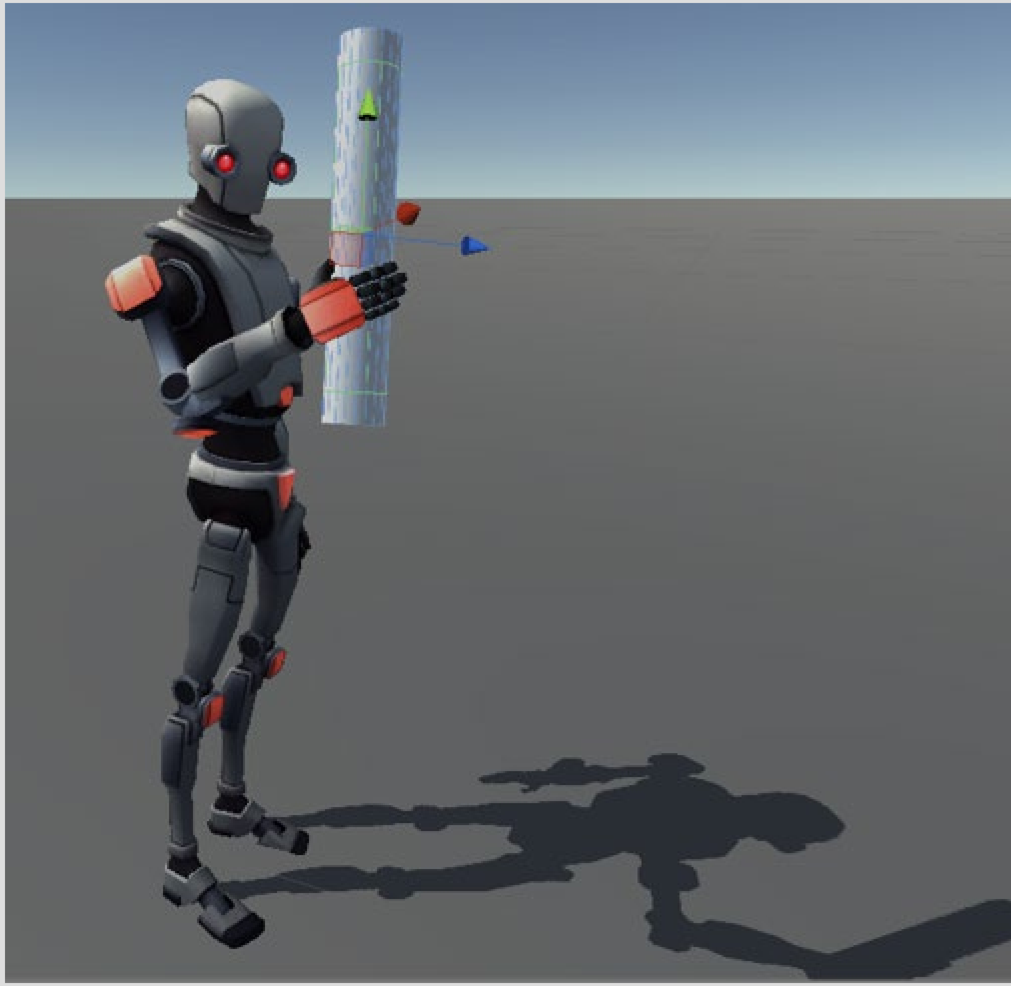
- Simplest: joint angle differences
  - Relative importance (shoulder vs. wrist)
  - e.g. $\boldsymbol{\Sigma}\ w(j)*(P(j_s) - P(j_t))^2$
- Alternatives
  - Alignment of virtual points on body
  - Visual difference of projection
  - Perceptual model (perception-based per-joint similarity weights)

Velocity Similarity

- To avoid splicing a touch and a punch
- Options:
  - Compute numerical derivative
    - $V(t) = P(t) - P(t-1)$
  - Match poses over time window

Interaction Similarity

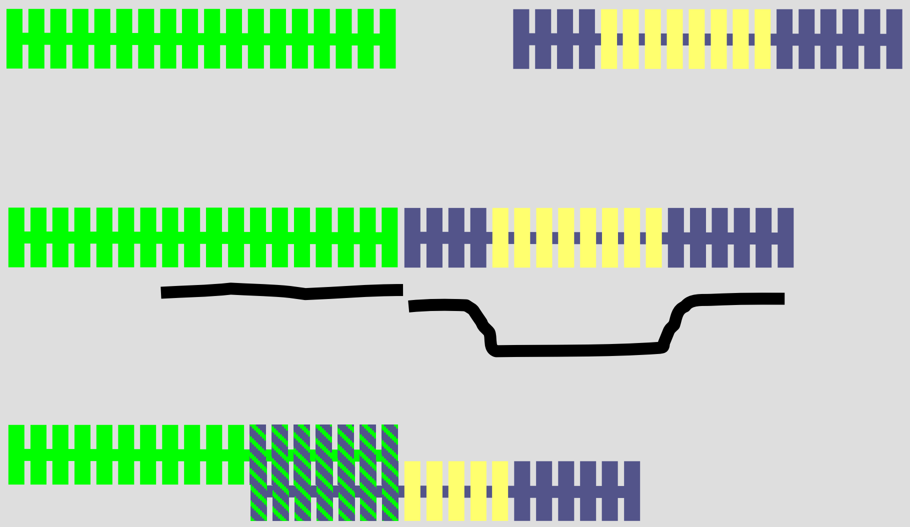
- Contact with the environment should remain crisp
  - Smoothing often loses this
  - e.g. feet sliding on/in ground
- Pose/velocity measures enforce this poorly
- May want to penalize score if motions differ in interaction with environment
  - e.g., setting foot down vs. standing still



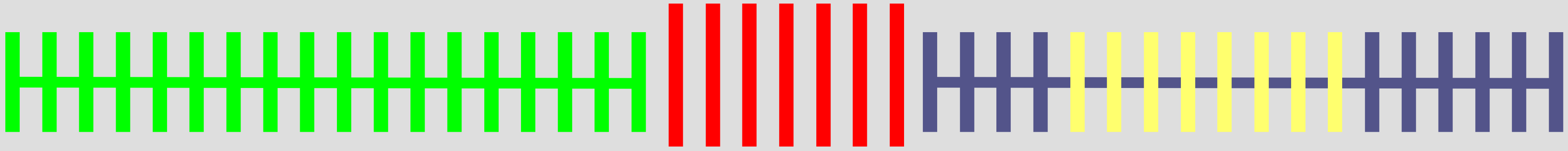
Motion Smoothing Algorithm

- Three basic choices

1. Connecting
2. Warping
3. Blending

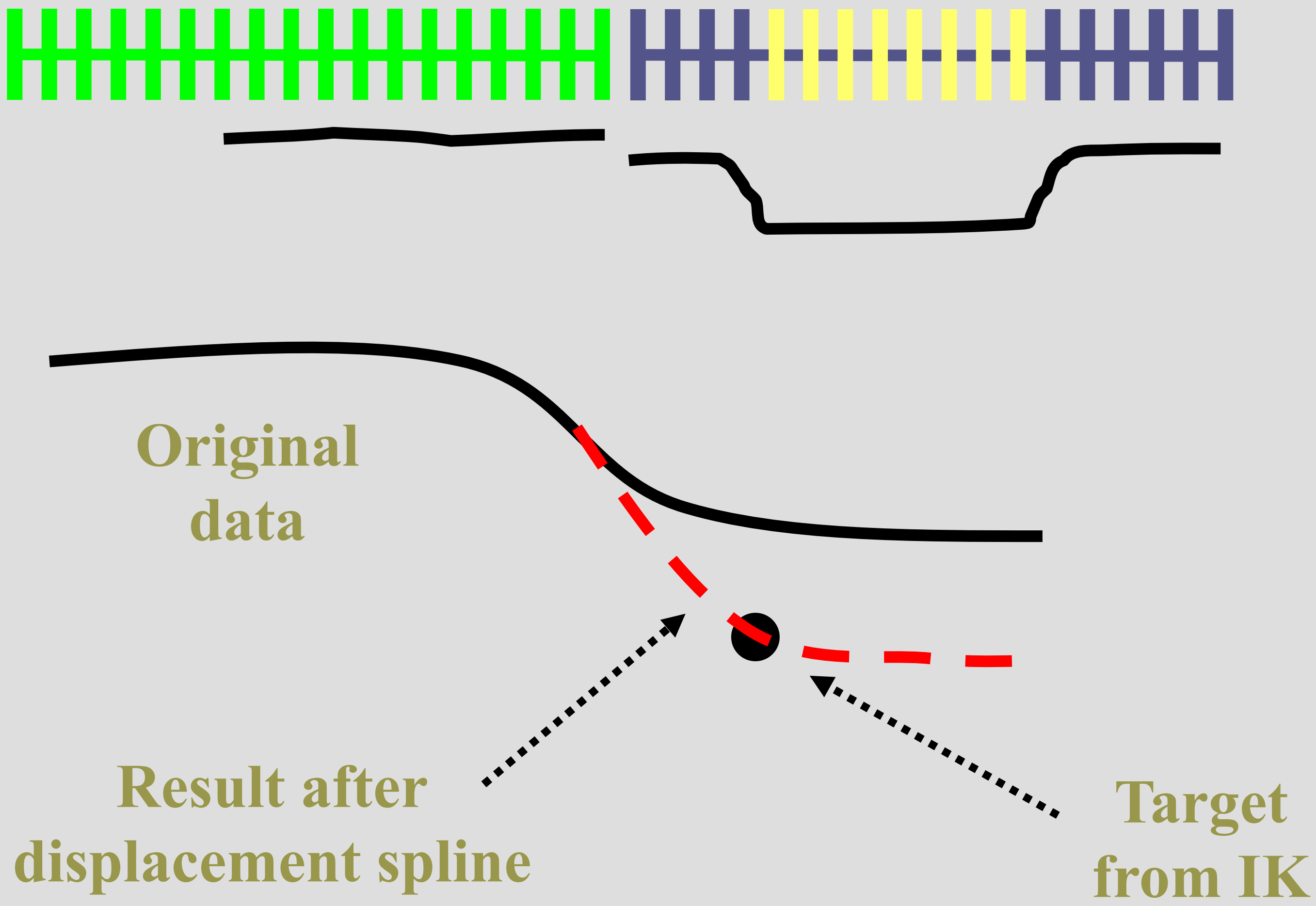


Connecting Motions



- Synthesize frames to fill the gap
  - Typically interpolate between poses
    - e.g., SLERP
  - Physics-based optimization also possible
    - Minimum-energy trajectory between poses

Warping Motions

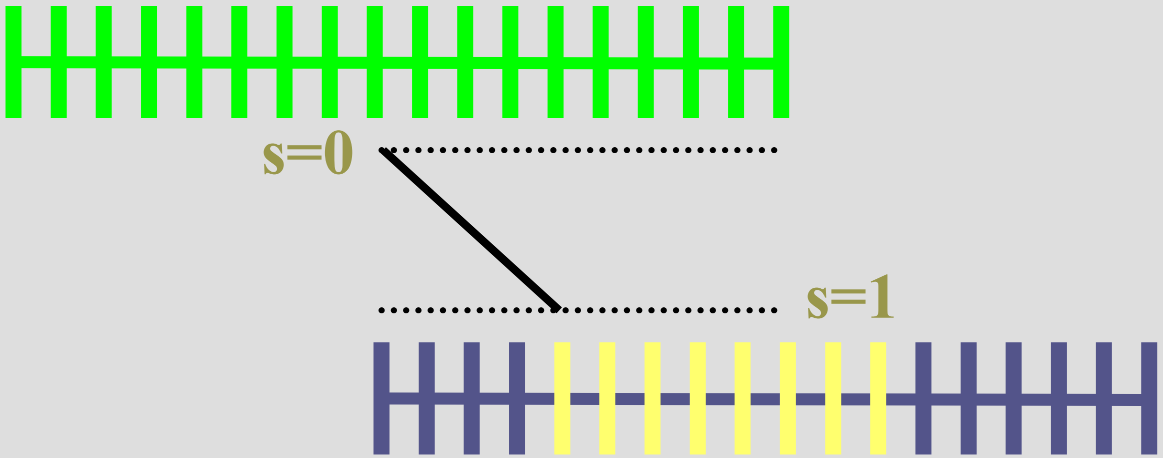


Blending Motions

- The term *animation blending* refers to any technique that allows more than one animation clip to contribute to the final pose of the character
- Combine 2 input poses to create an output pose
- Example
  - Injured walk animation
  - Regular walk animation
  - Can get various intermediate levels of injury

LERP Blending

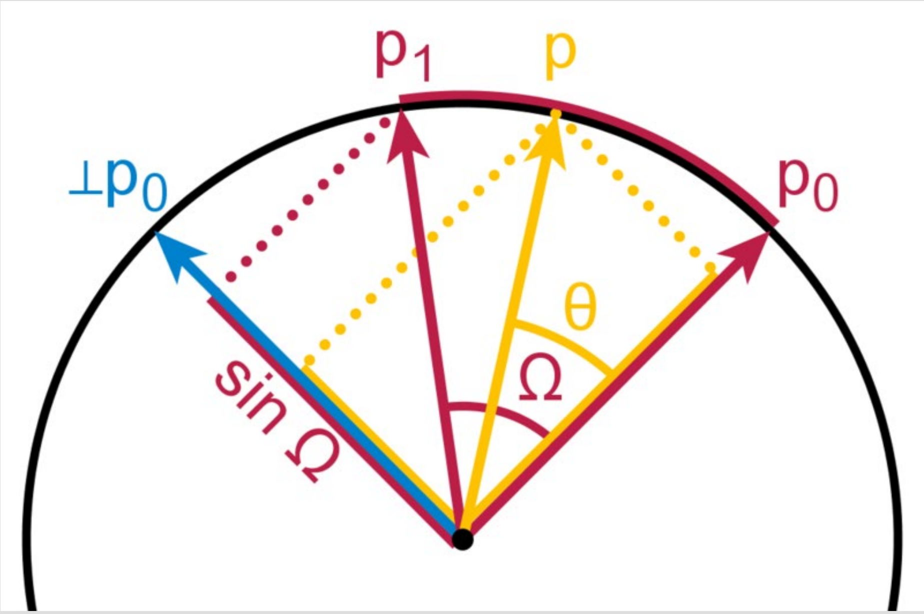
- Transition smoothly from motion A to motion B
  - Need blending algorithm, blending weights
  - $P(t) = (1-s)P_A(t) + sP_B(t)$, $0<=s<=1$



SLERP

- The linear interpolation of the rotation component is a quaternion LERP or SLERP (Spherical LERP)

$$\frac{\sin\big((1-\beta)\theta\big)}{\sin(\theta)}(q_A)_j + \frac{\sin(\beta\theta)}{\sin(\theta)}(q_B)_j$$



LERP

- The most natural looking intermediate poses are generally when each joint pose is interpolated <u>*independently*</u> of the others, in the space of that joint's immediate parent
- If we blended global poses, the results would look biomechanically <u>impossible</u>
  - **Pose blending** is generally performed on ***local poses***
- Since joints are blended independently of the interpolations of other joints
  - Interpolation can be performed in <u>parallel</u>

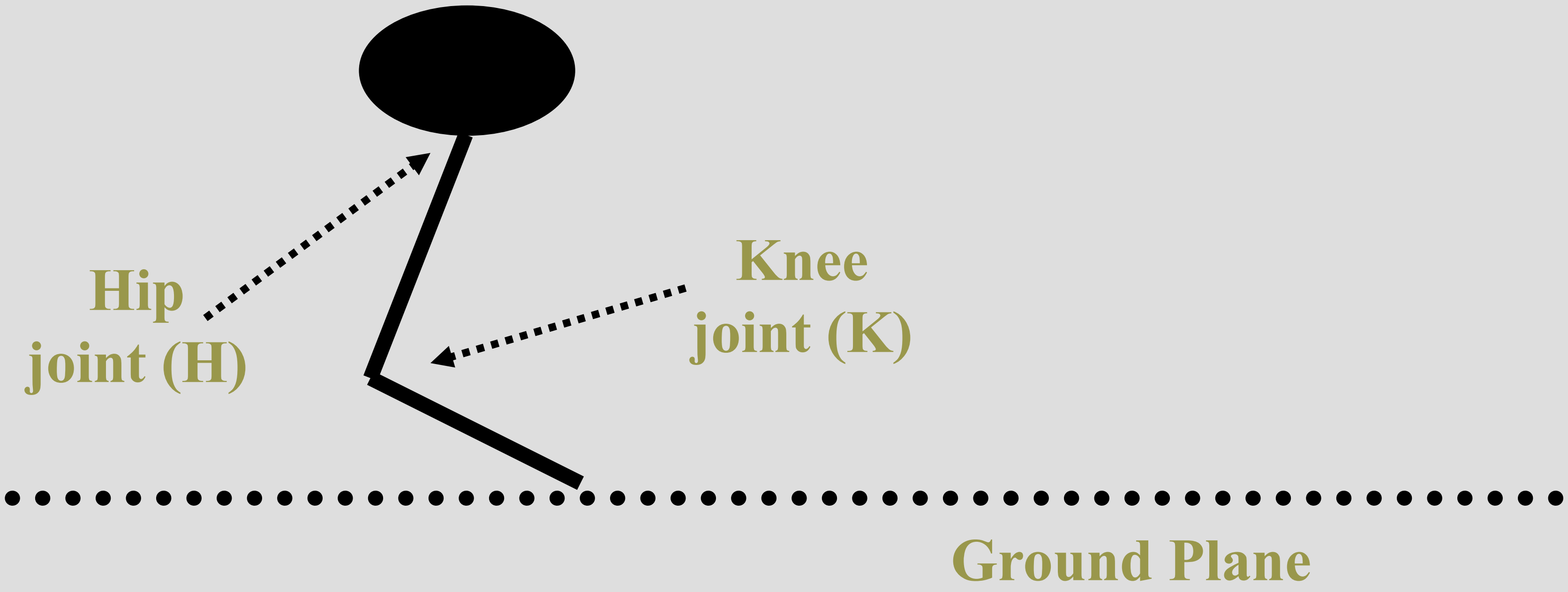
Blending Motions

- Not only for transitioning
- E.g.
  - Have a running motion
  - Have a walking motion
  - Want a ***jogging*** motion

- Problems with Blending Animation?
  - Bad environment contacts
  - Out-of-phase animation

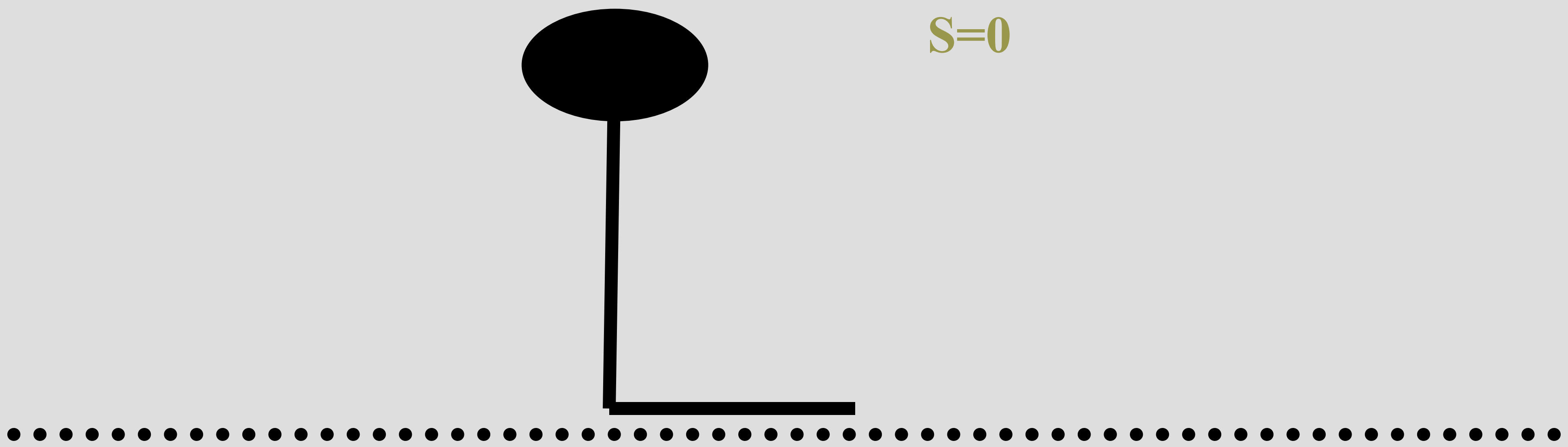
Blending Motions

1. Blend the joint angles



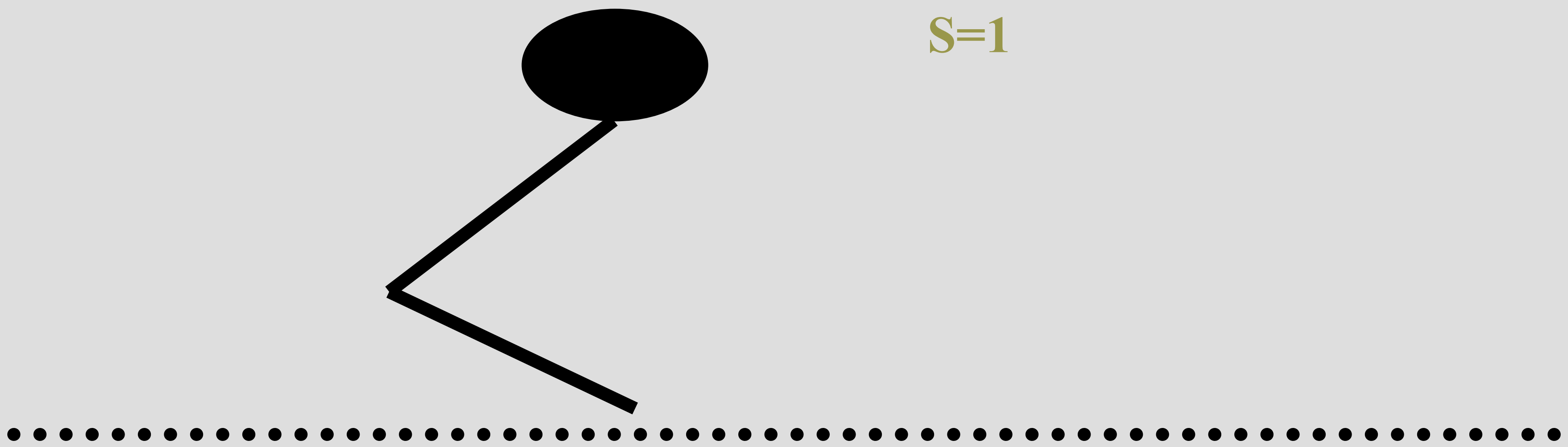
Blending Motions

1. Blend the joint angles



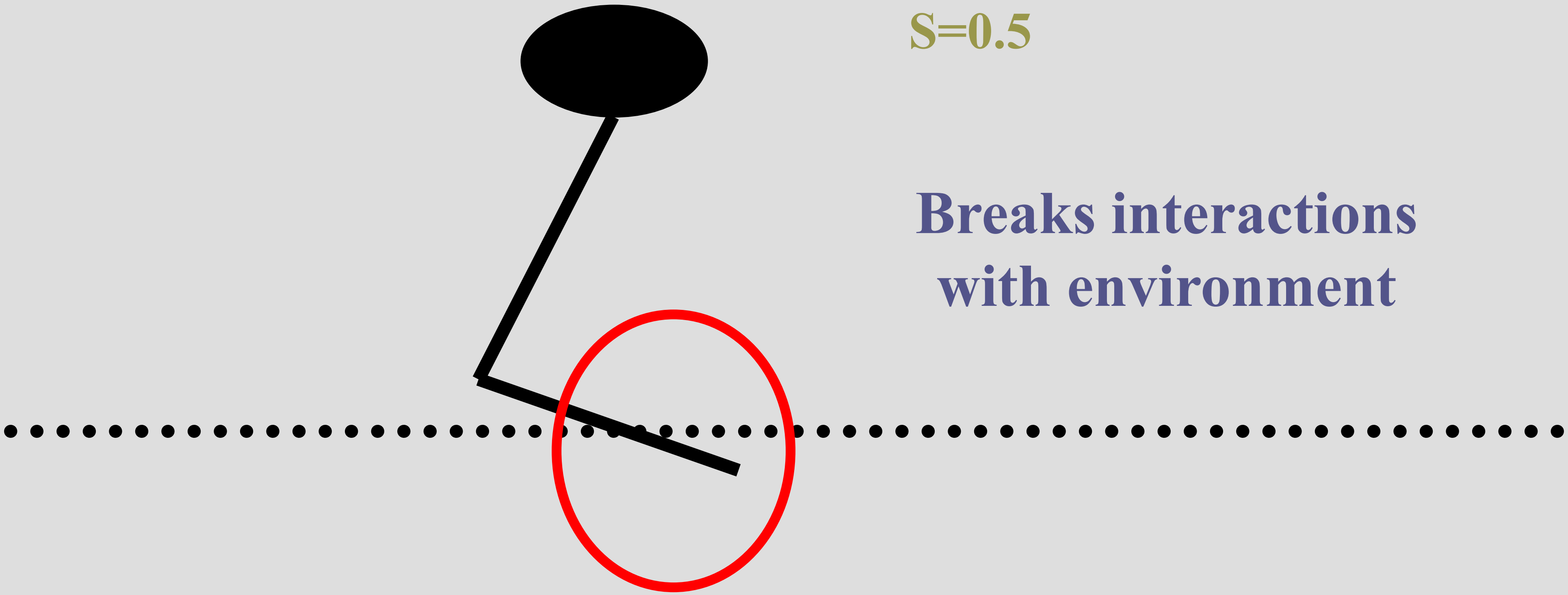
Blending Motions

1. Blend the joint angles



Blending Motions

1. Blend the joint angles
   - Problem:



S=0.5

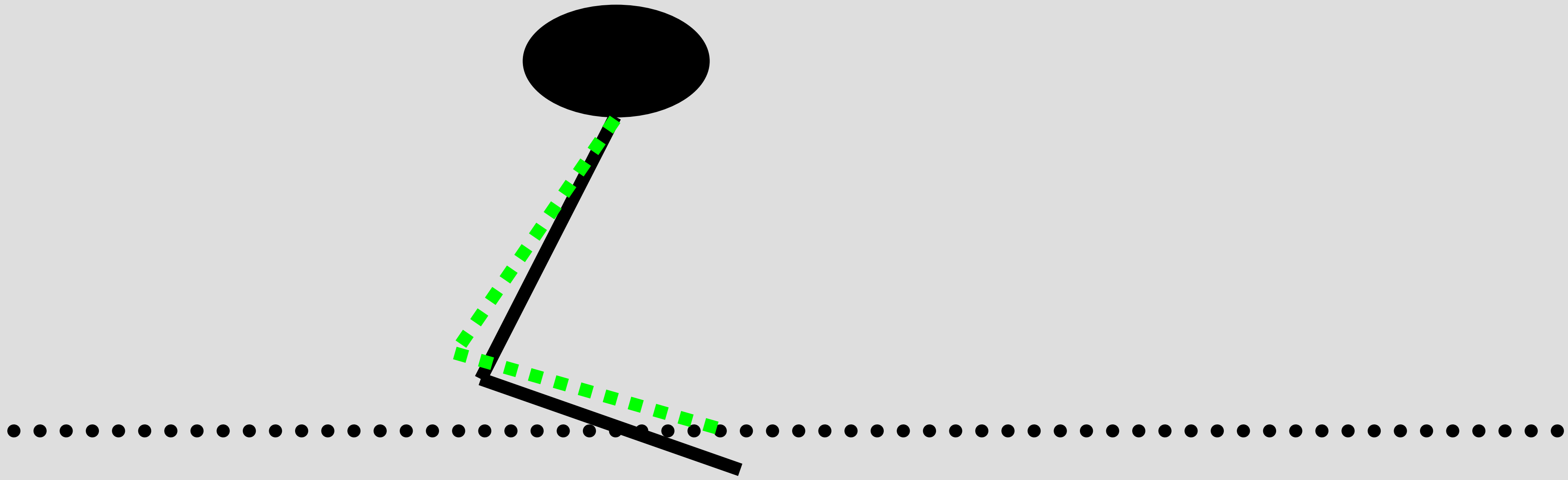
Breaks interactions with environment

Blending Motions

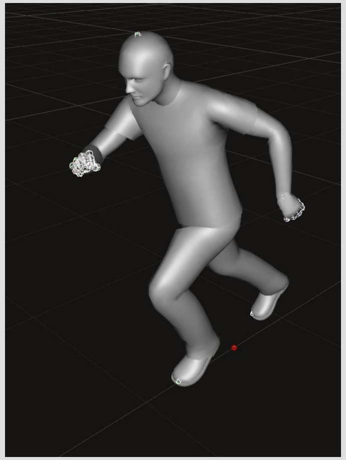
1. Blend the joint angles
   - Problem:
     - Breaks interactions with environment
     - Solution: *IK and motion warping*

Blending Motions

1. Blend the joint angles
   - Post-processing fix with IK



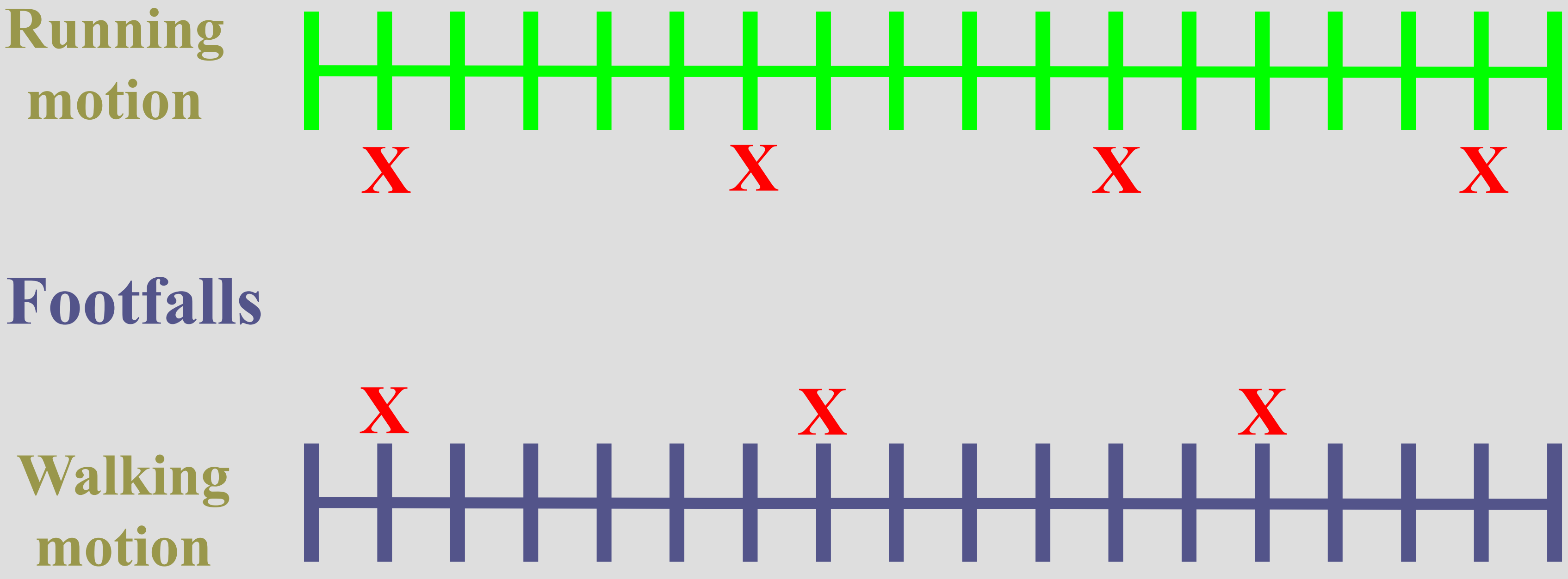
Blending Motions



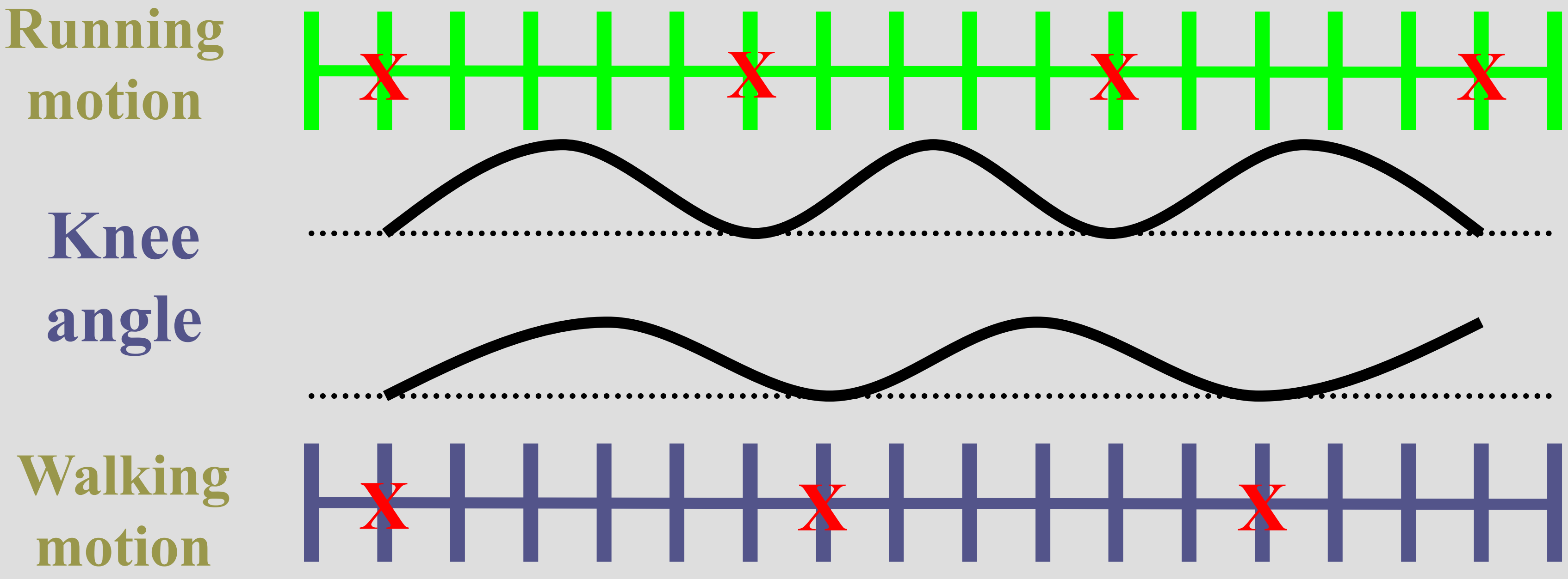
Running motion

Walking motion

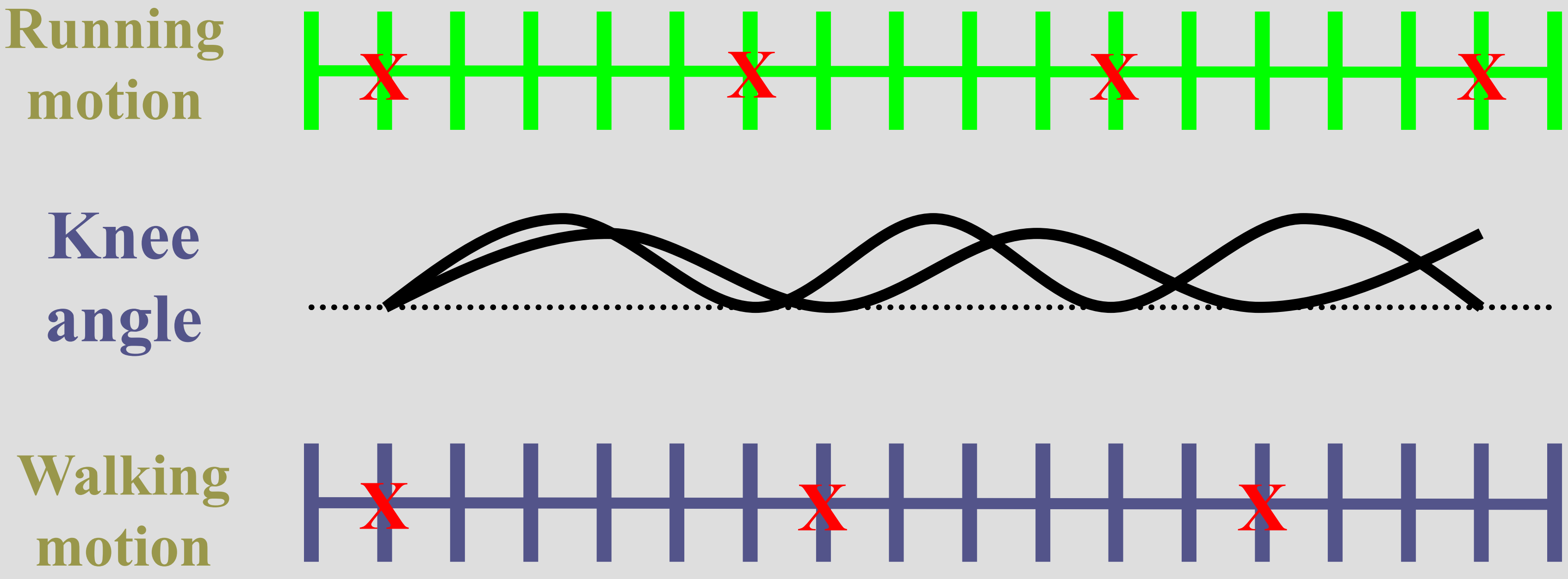
Blending Motions



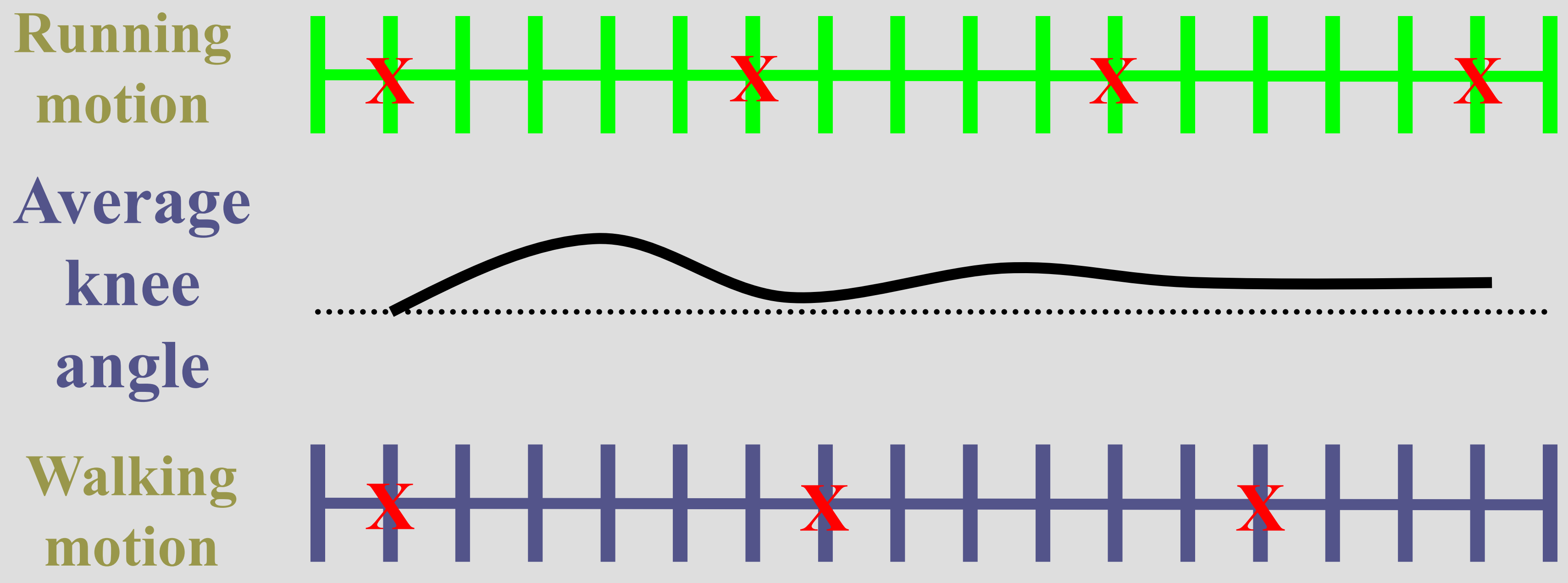
Blending Motions



Blending Motions



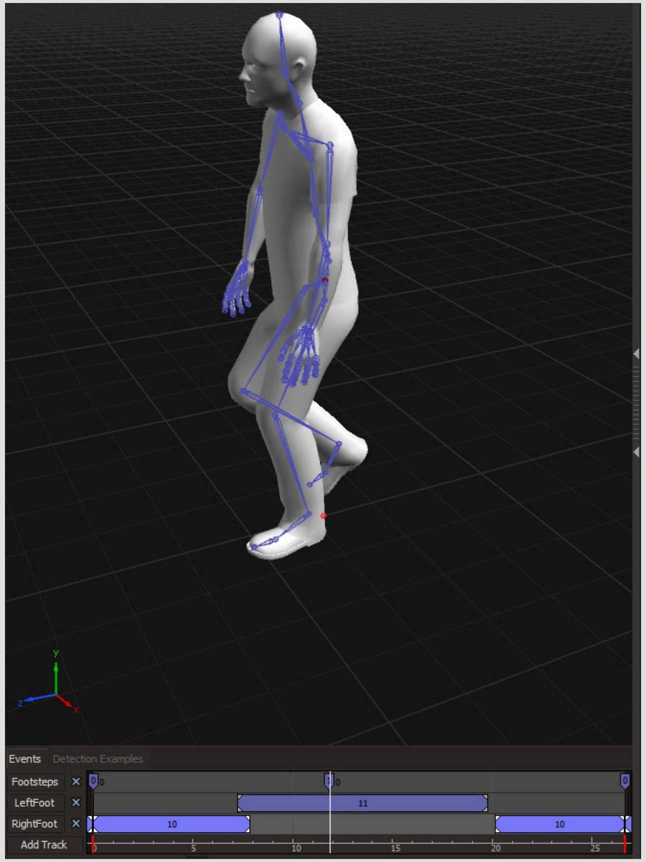
Blending Motions

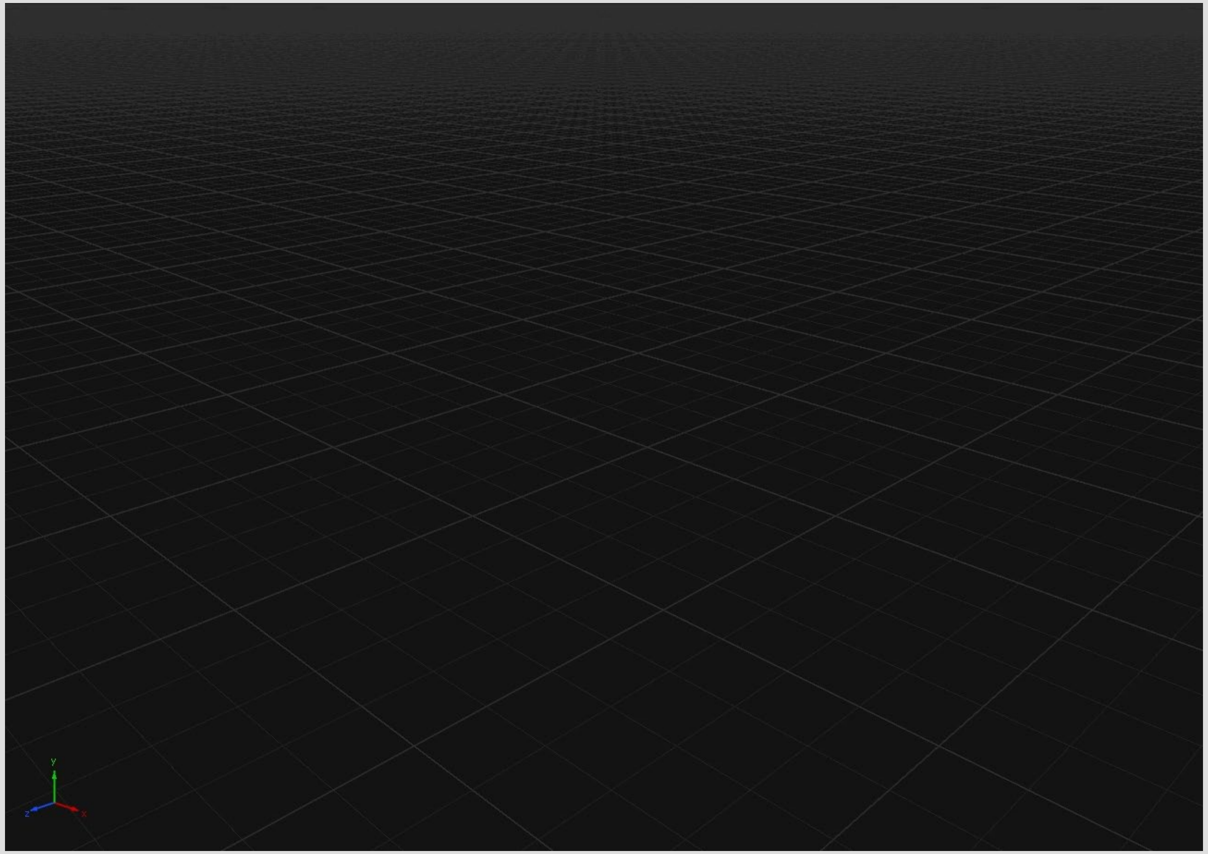


Blending Motions

1. Blend the joint angles
   - Problem: even similar motions get out of phase
   - *Time alignment algorithm*

Out of Phase Problem



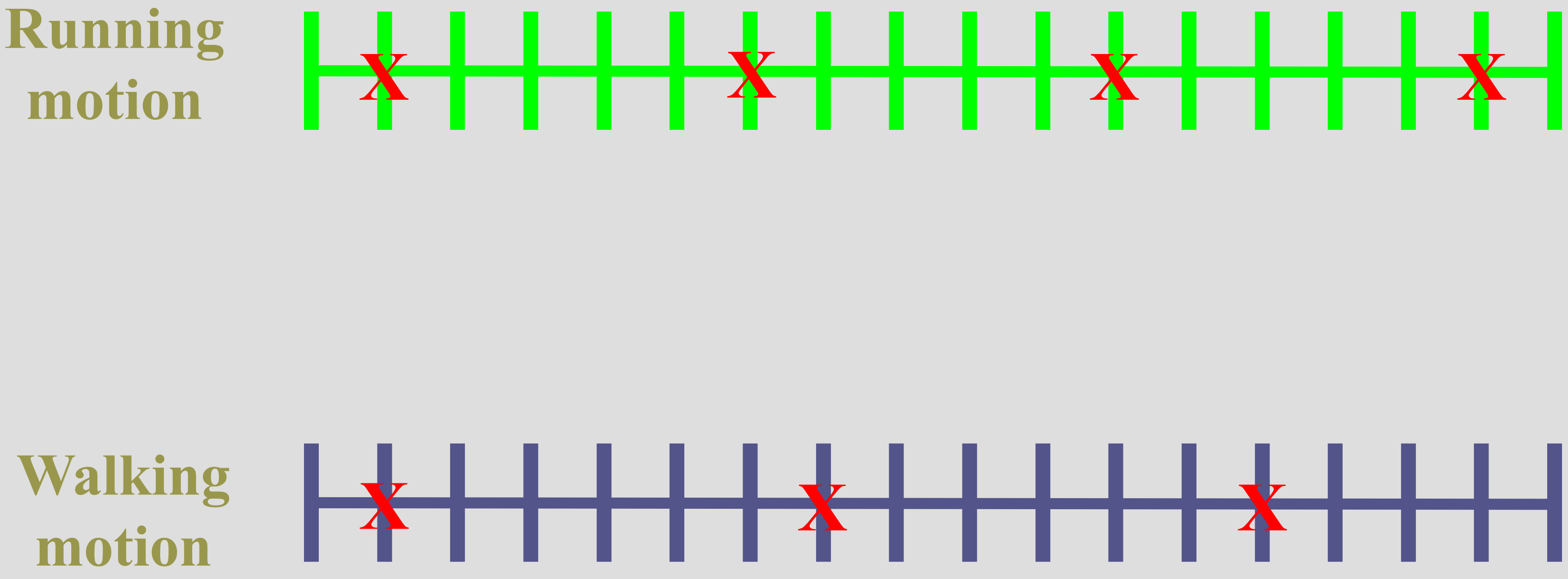


Time Alignment Algorithm

- Align logically similar parts of motion
  - e.g., footfalls with footfalls
  - Use *motion similarity metric (or do manually)*

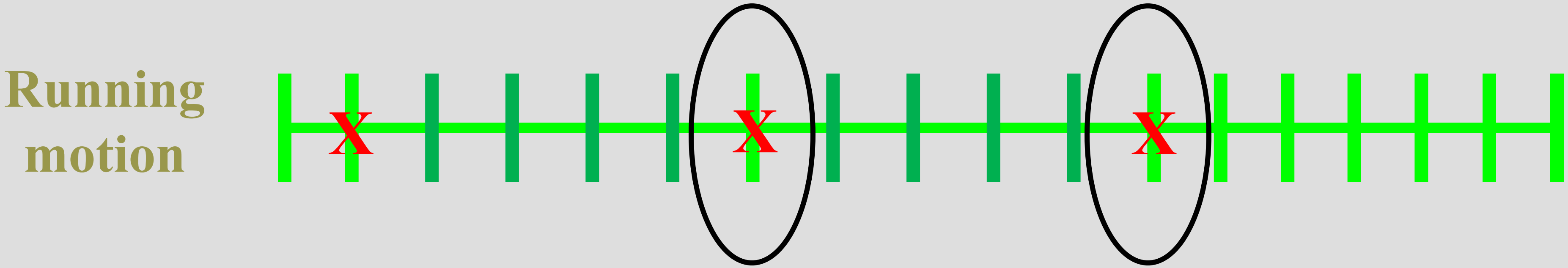
- Define mapping between frames of motions
  - Shorter motion **gets extended** to match longer motion
  - Perceptually: slowing down less noticeable than speeding-up

Aligning Motions

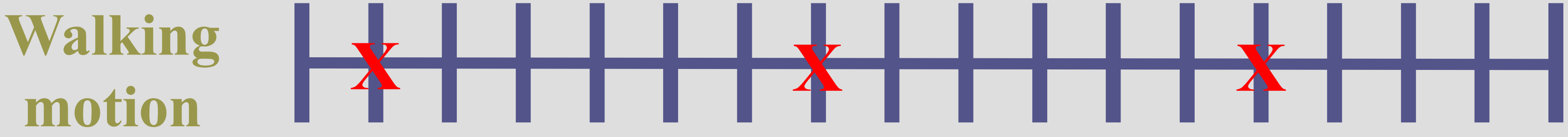


Aligning Motions

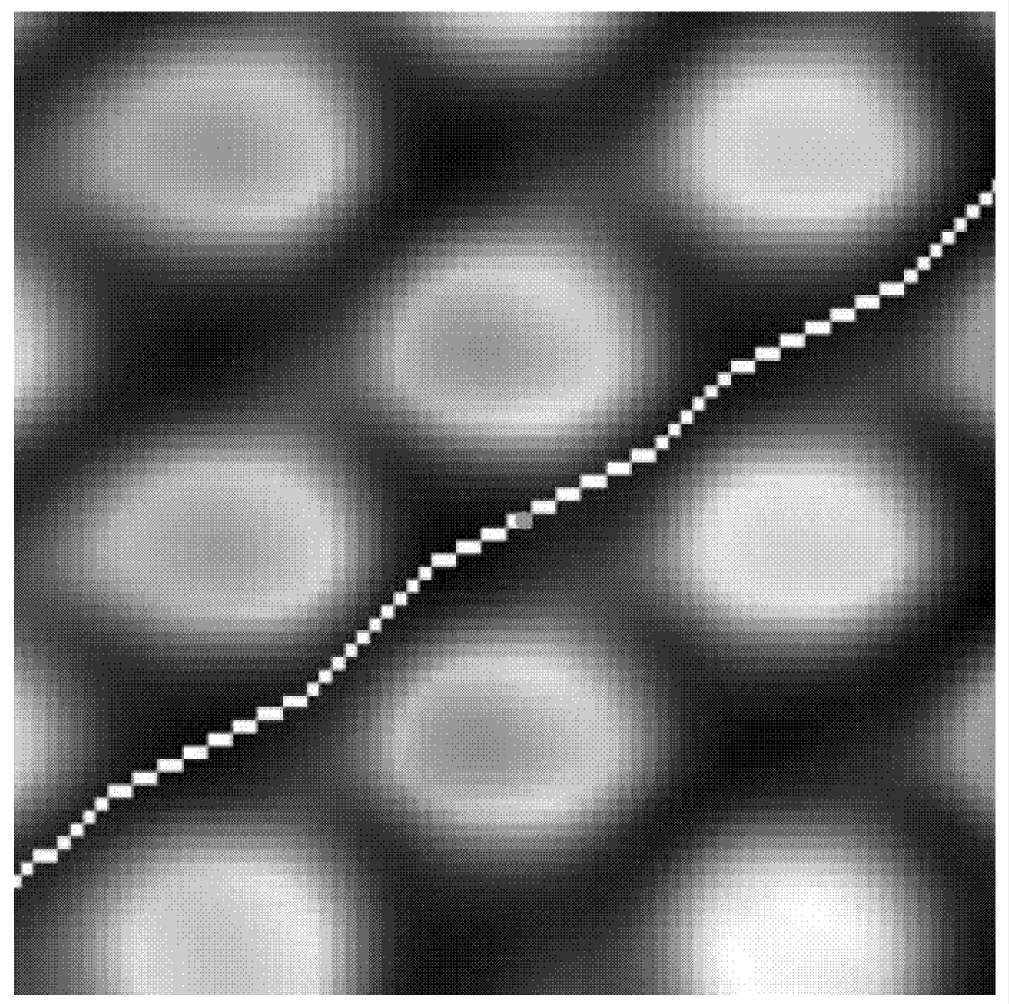
- Duplicate frames to fill gaps



- Extend motion to maximize similarity



Dynamic Timewarping



Frames of walking motion

Frames of running motion

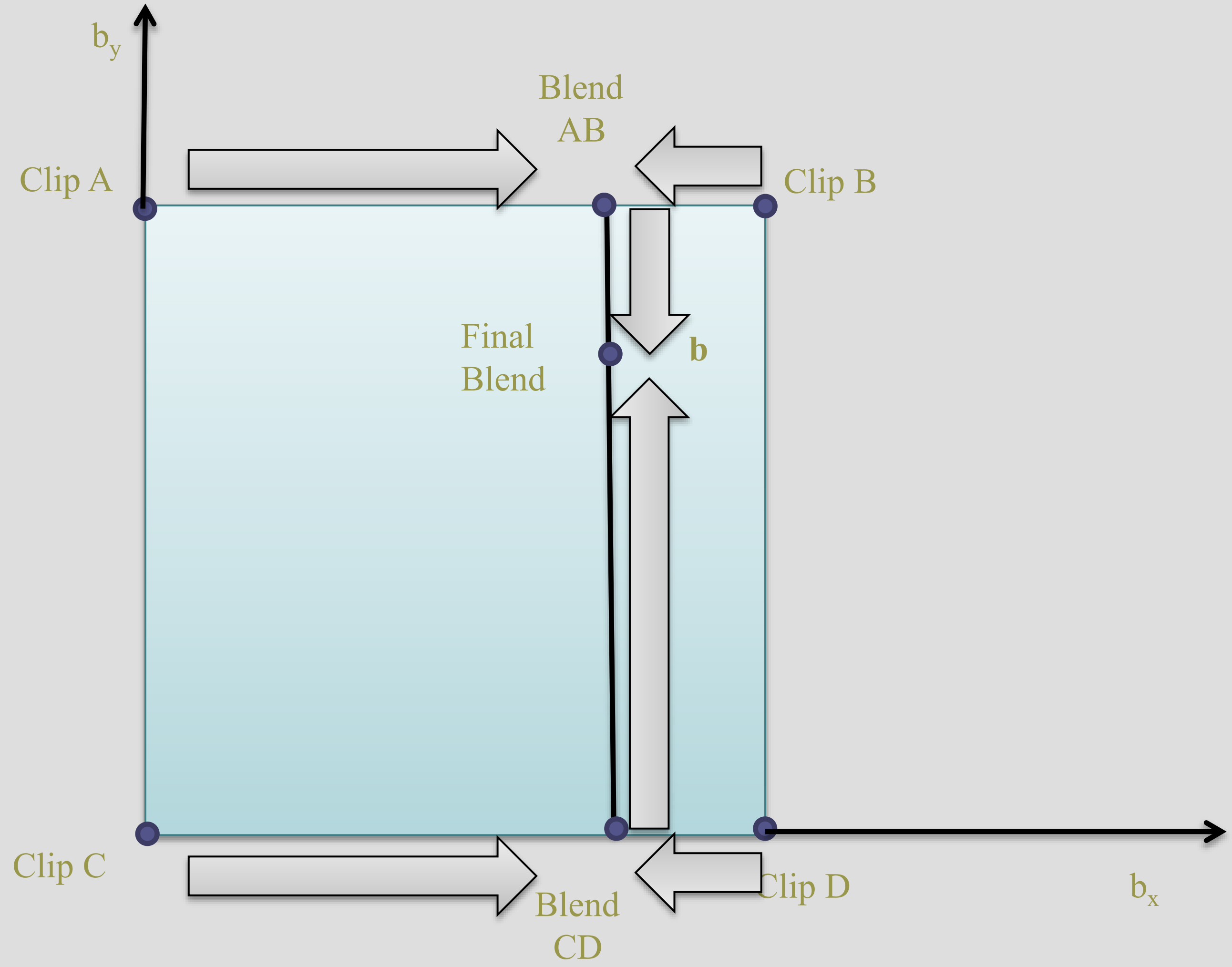
Matrix of frame-to-frame similarity (darker = more similar)

Two-Dimensional LERP Blending

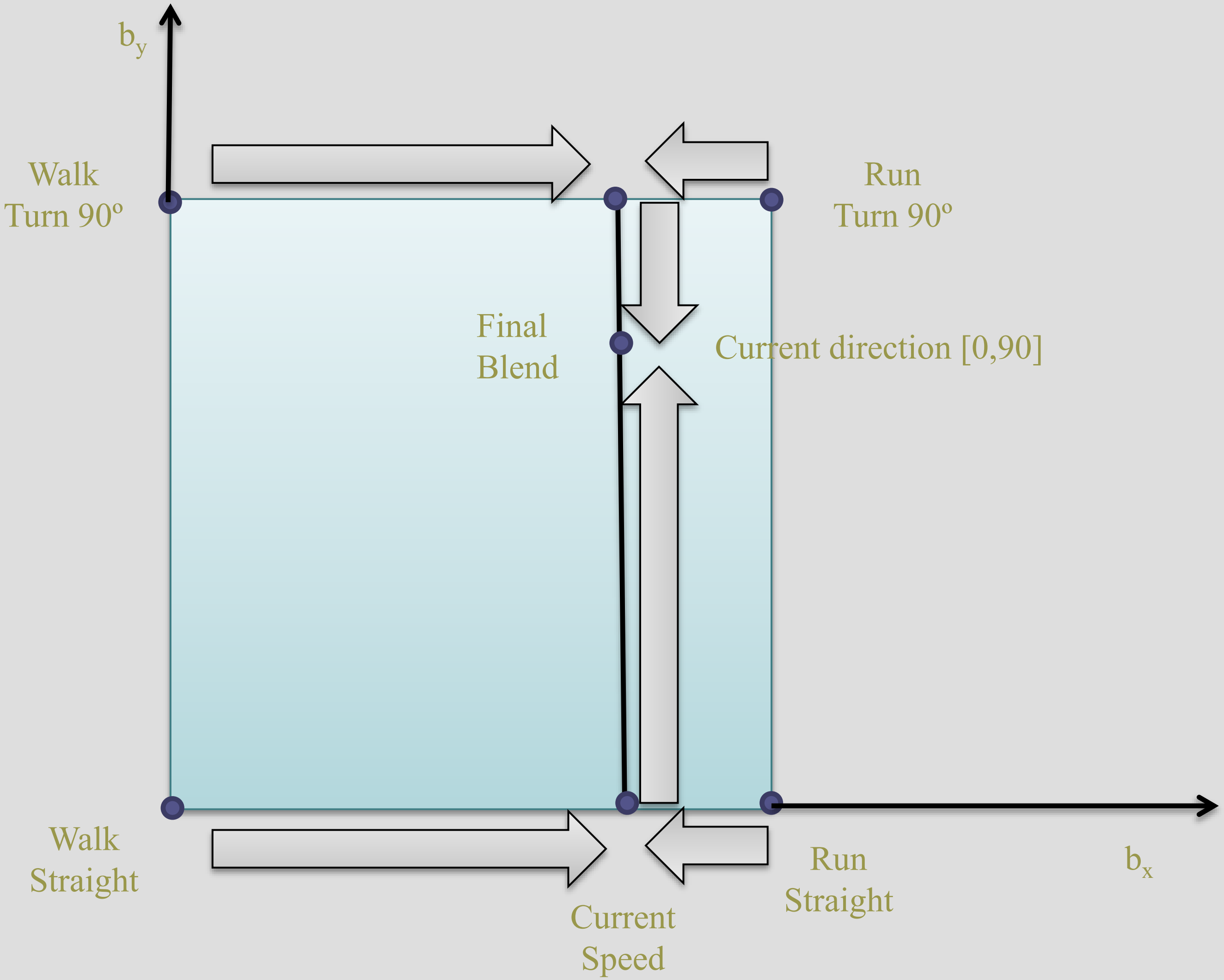
- Sometimes we would like to smoothly vary *two* aspects of a character's motion simultaneously
  - Vary pace and feet separation

- Extend 1-D LERP to 2-D in order to achieve these kinds of effects
  - Use two 1D blends
  - $\mathbf{b} = [b_x, b_y]$

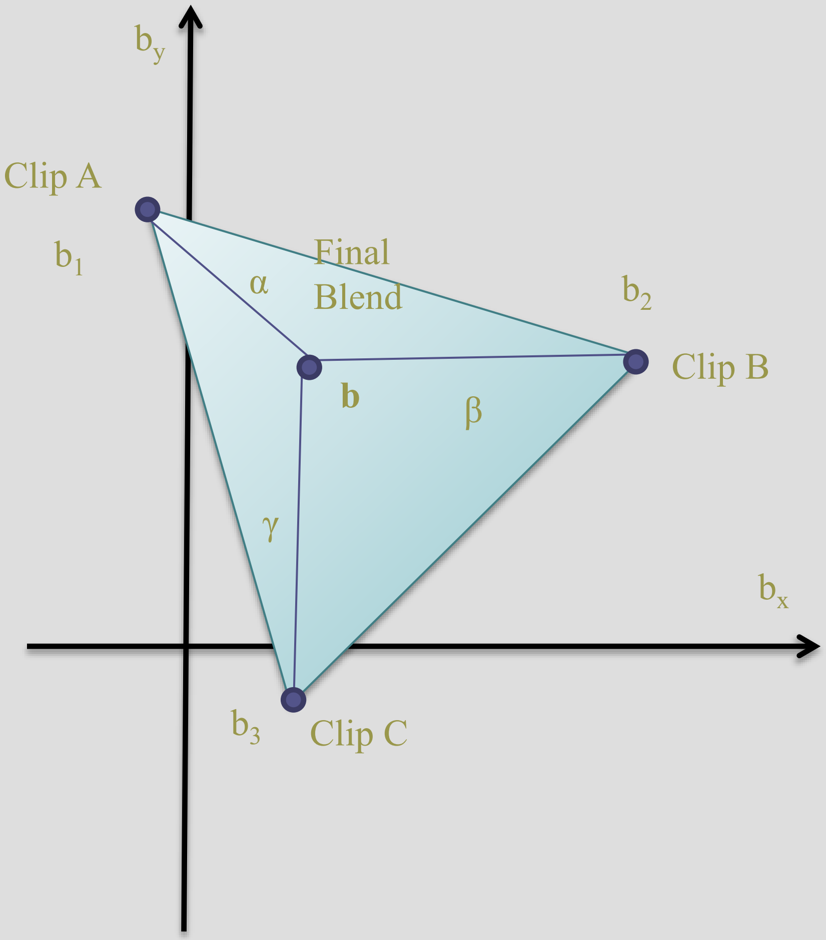
Two-Dimensional LERP Blending



Two-Dimensional LERP Blending



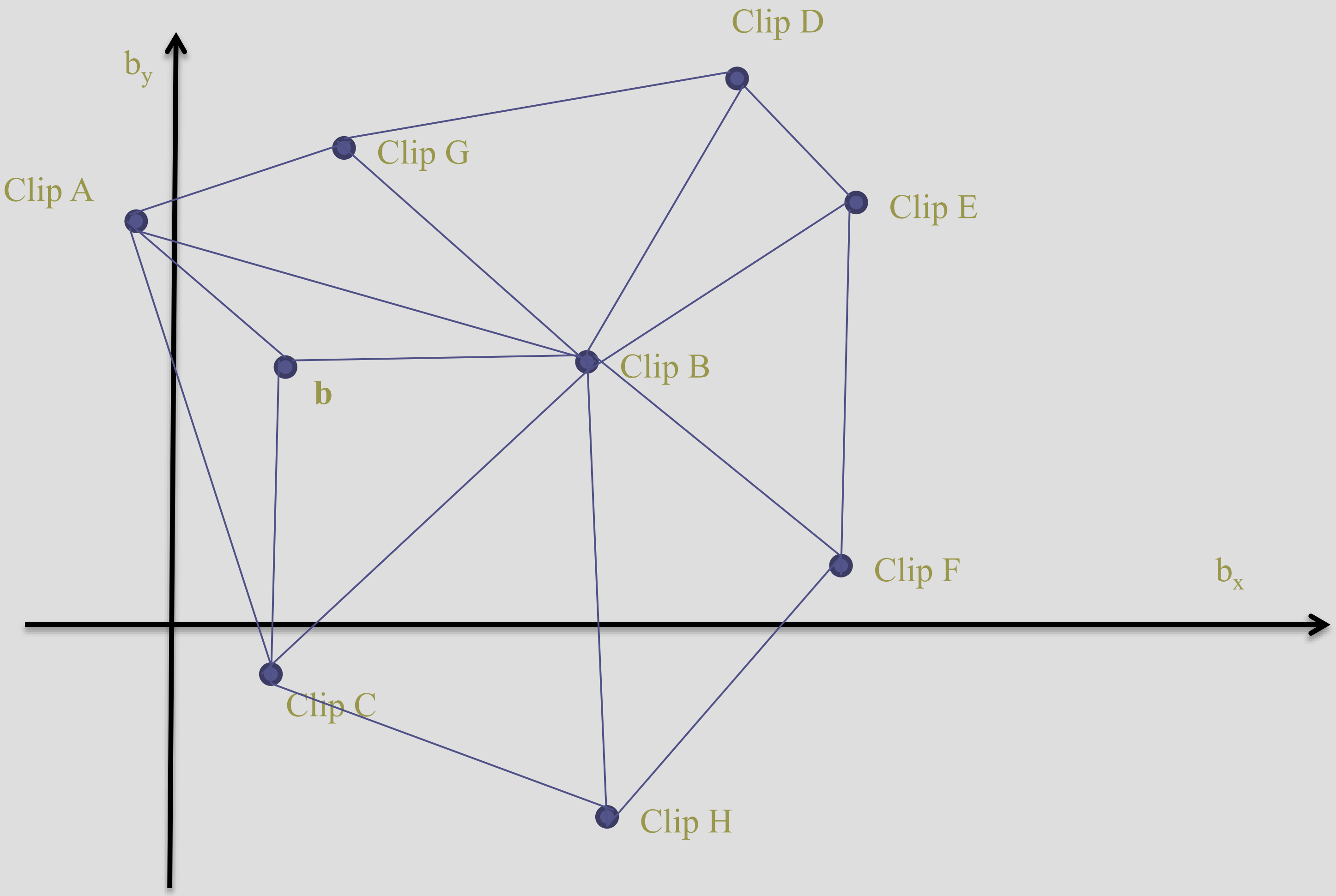
Barycentric Coordinates



- For 3-input LERP, we use 3 weights:
  - α, β, γ
  - With $\gamma = (1 - \alpha - \beta)$

$$b(t) = \alpha\, b_1(t) + \beta\, b_2(t) + (1 - \alpha - \beta)\, b_3(t)$$

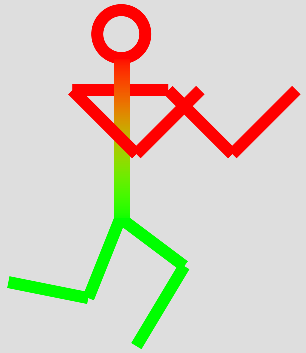
Generalized 2-D LERP



Delaunay Triangulation

Transplanting Body Parts

- A human being can control different parts of her body independently
  - Have a running motion
  - Have a catching motion
  - Want a catching-while-running motion



- Previously, we used the same blend percentage on every joint in the skeleton
  - Partial-Skeleton blending extends this idea by permitting the blend percentage to vary on a per-joint basis

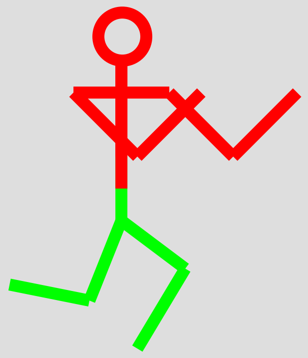
Blend Mask

- For each joint *j*, we define a separate blend percentage β_j
- The set of all blend percentages for the entire skeleton is called the *blend mask*
- It can be used to "mask" out certain joints by setting their blend percentages to zero

Transplanting Body Parts

- Have a running motion
- Have a catching motion
- Want a catching-while-running motion

- **Solution:** stick a catching torso on top of running legs



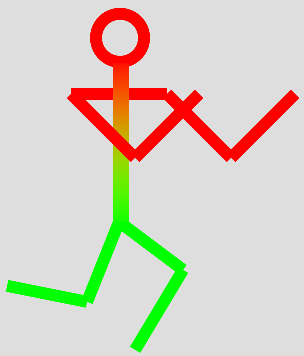
Transplant Example

Subtle example:
look at the fingers
opening and closing

Transplant Example 2

Unnatural Movement

- An abrupt change in the per-joint blend factors can cause the movements of one part of the body to appear disconnected from the rest
- Problem can be improved by **gradual blending**
  - Joints in legs get 100% of angles from running motion
  - Joints in arms get 100% of angles from catching motion
  - Joints in between blend both motions, with weights depending on whether closer to legs or arms



Unnatural Movement

- However, movements of a real human body are never totally independent
  - Persons wave while running vs. walking
  - More out of control and bouncy
- Use *additive blending*

Additive Blending

- Totally different approach
  - Use a “difference clip”

- Encodes the *changes* that need to be made to one pose in order to transform it into another pose
  - Difference from reference pose

- Input clips: source (S) and reference (R)
  - Difference (D) = S-R
  - Can be added to target clips (T): T+D

Additive Blending

- Example
  - R = character running
  - S = running in a tired manner
  - D = the changes necessary to make the character look tired
  - T = character walking
    - Add D to make the character walk while tired

Additive Blending

- We are dealing with joint poses, so we cannot simply subtract
  - Joint pose is a 4x4 matrix that transforms points from child's local space to parent space
- Matrix equivalent of subtraction is multiplication by inverse matrix
  - $D_j = S_j R_j^{-1}$
- "Adding" D to T is achieved by concatenating the difference transform and the target transform: $(S_j R_j^{-1}) T_j$

Additive vs. Partial Blending

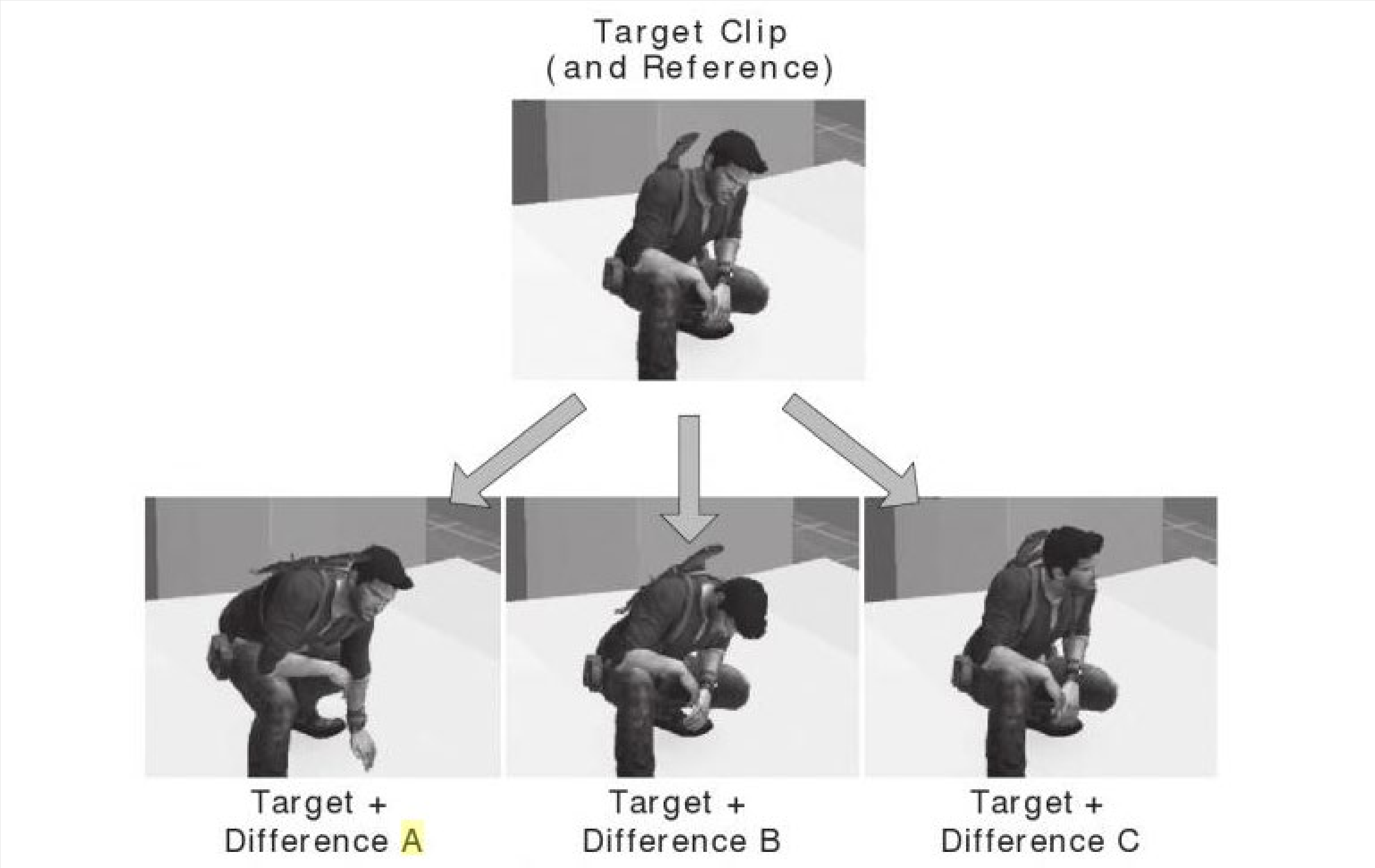
- Additive blends suffer less from the “disconnected” look of animations combined via partial blending
  - Because we are not replacing the animation for a subset of joints or interpolating 2 potentially unrelated poses
  - We are adding movement to original motion across the entire skeleton

Limitations

- Can have a tendency to over-rotate the joints in the skeleton

- Rules to avoid problems:
  - Keep hip rotations to minimum in R
  - Shoulder & elbow joints neutral in R
  - D animation for each core pose (standing, upright, crouched down, etc.)

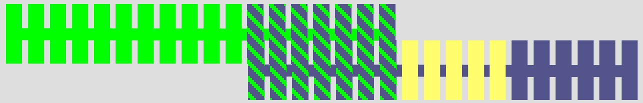
Layer Randomness to repetitive cycles



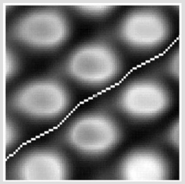
Motion Editing

- Warping

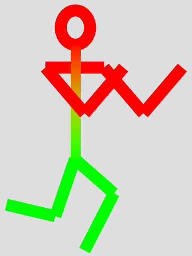
- Splicing



- Blending



- Transplanting



More Information

- Warping
  - “A hierarchical approach to interactive motion editing for human-like figures”, Lee and Shin, 1999
- Splicing
  - “Motion Graphs”, Kovar et al., 2002
- Blending
  - “Flexible Automatic Motion Blending with Registration Curves”, Kovar and Gleicher, 2003
- Transplanting
  - “Enriching a Motion Collection by Transplanting Limbs”, Ikemoto and Forsyth, 2004