Motion:

Character Motions is a series of poses over time

recall: poses denoted Θ

Keyframed Motion: $\langle t_0, \Theta_0 \rangle, \langle t_1, \Theta_1 \rangle, \ldots \langle t_n, \Theta_2 \rangle$

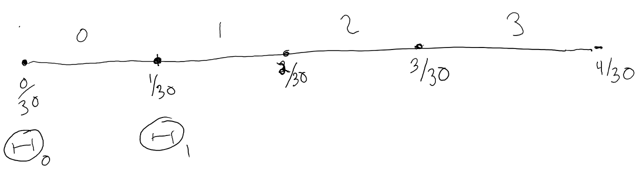
* times may not be uniformly spaced
* relies on artist typically to create each pose Θ_i
* use splines to interpolate poses

Fixed framerate Motion: $[\Theta_0, \Theta_1, \Theta_2, \ldots \ldots \Theta_n]$

* don't store time because the time between each key is known
* motion capture produced fixed framerate motion

$\boxed{\text{EX}}$ Motion Capture systems usu. capture poses at either 24, 30, 120 fps.
$\Rightarrow$ If the fps = 24, the time between each frame is $\frac{1}{24}$ s

$\boxed{\text{EX}}$ Suppose we have a 30 fps motion that is 2s long.
What is the pose at time Θ_3 s? Θ_4



Step 1: Find segment containing time
$\rightarrow$ Each segment corresponds to $\frac{1}{30}$ seconds
$\Rightarrow \Delta t = 1/30$
$\Rightarrow$ segment $= \text{floor}(0.1/\Delta t) = \text{floor}(3) = 3$

Step 2: Compute normalized time, u
StartTime for segment = segment ID $* \Delta t = 3\left(\frac{1}{30}\right) = \frac{3}{30} = 0.1$
Segment end time = (segment + 1) $* \Delta t = 4\left(\frac{1}{30}\right) = 4/30$

$$\text{normalizedTime} = \frac{\text{time} - \text{segmentStartTime}}{\text{segmentEndTime} - \text{segmentStartTime}} = \frac{0.1 - 0.1}{\Delta t} = 0$$

SegmentEndTime − SegmentStartTime Δt

Step 3: Interpolate

Θ = Interpolate (Θ_3, Θ_4, 0)

EX How can I play a motion twice as fast?

Approach 1: Resample a motion to have a duration, or change fps

Approach 2: During playback, you can use a "time scale" to play at different speeds w/out changing the motion, e.g.

```
update()
  time = elapsedTime();
  Θ = motion.getValue(time);
```

play at recorded speed

```
update()
  time = elapsedTime() * timeScale;
  Θ = motion.getValue(time);
```

scaled time

Motion Editing:

In practice, we have motion clips for walk, stand, anything we want our character to do

Problem: We can't create motion clips for every possibility

→ too labor intensive

→ often impossible: needs special equipment, full knowledge about environments / context where motions would be used

Soln: (1) Generate new motions from existing ones

ex. Greeting motion + sad motion = sad greeting

ex. blending between motion clips to create transitions

(2) Adapt motion clips to new settings

ex. a walk motion can be modified for uneven terrain

ex. holding a cup; opening a door

Approach: To edit a motion, we only need to edit its keys

Editing poses:

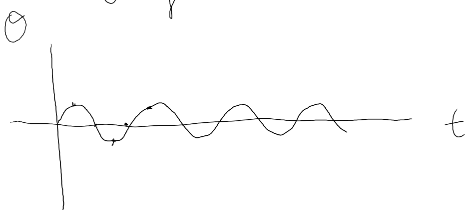
· Freezing a joint. (setting a constant value for a joint)

Technique 1: Freezing a joint. (setting a constant value for a joint)

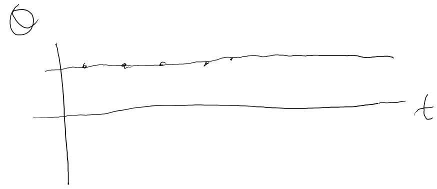
[EX] Zombie Arms.

→ replace the rotation curve for the shoulders to have a constant value

Original Curve



Set curve to a constant



Technique 2: Splicing

→ Copy upper body joints from one motion & paste them onto an-other motion

[EX] Splicing a "drink water" motion onto the upper body of a walk motion

Technique 3: Blending

Idea: Combining multiple motions together

Application 1: transitioning between motions

Application 2: combining aspects of motion

Notation: Motion is a seqn of poses

$$M = [\Theta_0, \Theta_1, \ldots, \Theta_m] \leftarrow$$ assume fixed frame rate

$$M(i) = \Theta_i \leftarrow$$ the i^{th} pose motion

[EX] If we interpolate between poses,

$$\Theta = \Theta_i (1-\alpha) + \Theta_j \alpha \quad , \alpha \in [0,1]$$

... the above means we

$\Theta = \Theta_i (1-\alpha) + \ldots \alpha$

Implementation-wise, the above means we interpolate each pair of quantities in Θ_i & Θ,

→ use slerp for quaternions

→ use lerp for vectors

In hw, Θ consist of a root pos & quaternions for ea joint (atk::Pose)

$\boxed{\text{EX}}$ To blend a motion, we do

$$M = M_1(1-\alpha) + M_2\alpha, \quad \alpha \in [0,1]$$

which means we blend pairs of keys between M_1 & M_2 w

Blending Caveats

Problem 1: What if M_1 & M_2 have a different # keys?

Problem 2: What if M_1 & M_2 are "very different"

→ blending walking w/ dance? (not possible)

→ blending chicken walk with a fast walk?

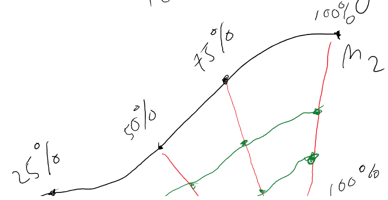
Problem 3: What if the motions are similar, but go in different directions?

Solving Problem #1: Different durations / # keys

Several Approaches:

Approach #1: "Time Align" the motions

Idea: We use percentages along each motion to match for blending.



This produces a motion whose dur is duration(M_1) < d < duration g

ch

sing α

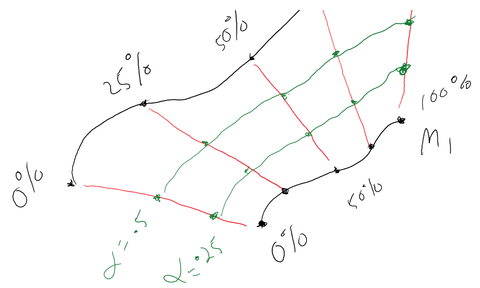
t

n key

ation

(M_2)

15



$M_1(1-\alpha) + M_2\alpha$

<u>Algorithm</u>: let duration = duration $(M_1)(1-\alpha)$ + duration$(M_2)\alpha$

$\Delta t = 1/fps$

for ($t = 0$; $t \le$ duration; $t \mathrel{+}= \Delta t$)

$$t_1 = \left(\frac{t}{\text{duration}}\right) \text{duration}(M_1)$$

$$t_2 = \left(\frac{t}{\text{duration}}\right) \text{duration}(M_2)$$

$$\Theta_1 = M_1(t_1)$$

$$\Theta_2 = M_2(t_2)$$

$$\Theta = \Theta_1(1-\alpha) + \Theta_2\alpha$$

<u>Approach #2</u>: Clamp shorter motion (ex. drink water & sta

→ blend shorter motion & keep remaining frames in lo

motion

<u>Approach #3</u>: Loop shorter motion (ex. wave hand & wa

<u>Problem#2</u>: What if motion are too different?

Some motions are too different to blend.

Classes (jump, walk, run) of motion can be blending by

matching foot contacts.

<u>Foot contact</u>: series of frames where 1 or 2 feet are in cont

w/ the floor

x

nd)

onger

ek)

act

EX Walking has a distinct foot contact patterns

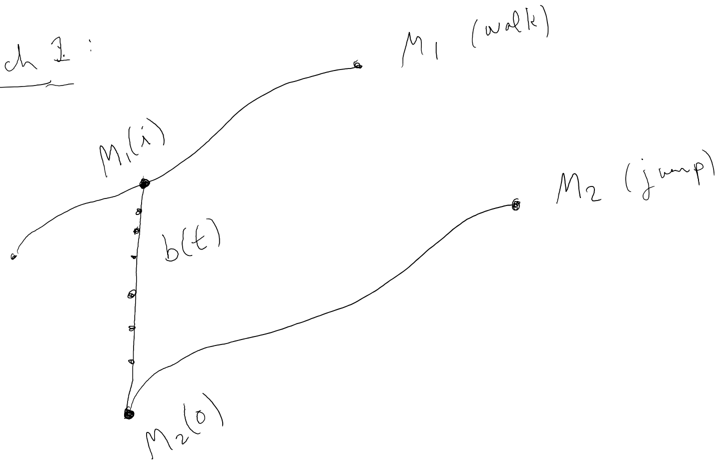
LF $\underbrace{\quad}_{\text{phase}}$ LF/RF $\underbrace{\quad}_{\text{phase}}$ RF $\underbrace{\quad}_{\text{phase}}$ LF/RF

To blend motions w/ the same foot contacts, time align & bl
each phase

Problem #3: Similar motions but different directions?
To solve, we would align the heading & position of the
second motion to match the first.

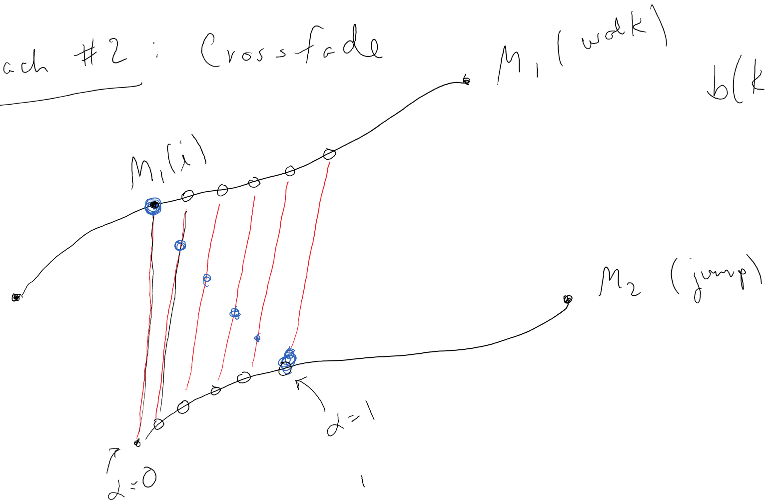
Using blending to generate transitions:
ex. from walk to jump

Approach 1:



Let $b(t)$ be a linear ble
from $M_1(i)$ to $M_2(0)$

Approach #2: Crossfade



$b(k) = M_1(i+k)(1-\alpha) + M_2(k$

where α starts with v
of 0 and ends wit
value of 1

Alignin Motions:

end

nd

) α

alue

h

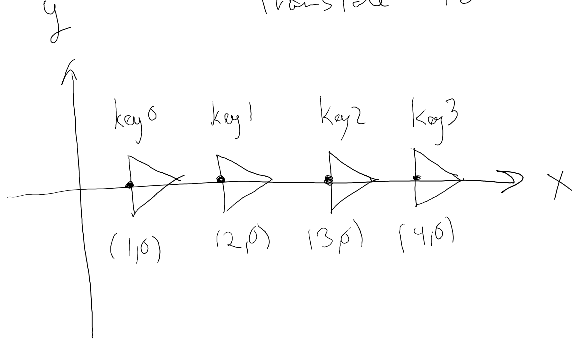
Aligning Motions:

Goal: Translate & rotate a motion to a new position & orientation

Idea: Only need to change the keys for the root joint

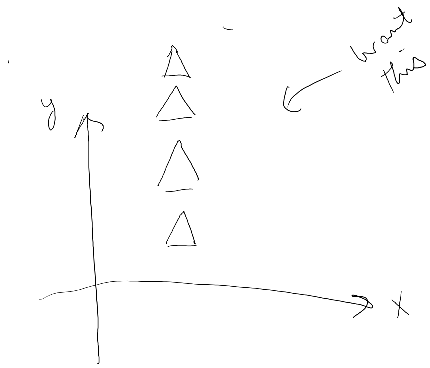
Approach: Translate back to the origin
Rotate
Translate to the desired position

EX



key	pos	fwd	ori
0	(1,0)	(1,0)	I
1	(2,0)	(1,0)	I
2	(3,0)	(1,0)	I
3	(4,0)	(1,0)	I

Rotate the motion 90° & start at the point (1,1)



Step 1: Define a transform that moves the first key to the origin.

$$T_1 = \left(\begin{array}{c|c} I & \begin{pmatrix} -1 \\ 0 \\ 0 \end{pmatrix} \\ \hline 0 & 1 \end{array}\right)$$

Step 2: Define a transform that moves the first frame to the desired rotation & translation

$$T_{desired} = \left(\begin{array}{c|c} R_z(90) & \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} \\ \hline 0 & 1 \end{array}\right)$$

Step 3: Apply to two transforms to the whole motion clip,

Step 5 Apply the new transform root pos & rot in ...
& save the new transform

For each key in our motion M

$$T_{orig} = \left(\begin{array}{c|c} R^{0}_{root} & d^{0}_{root} \\ \hline 0 & 1 \end{array} \right)$$ ← root transform for key i

$$T_{new} = T_{desired} T_1 T_{orig}$$

Save R_{new} & d_{new} to key i

NOTE: In the previous example, we were given our rotational offset. What if we are instead given a desired orientation?

$R_z(90)$ in our example

$$\Delta R \, R_{current} = R_{desired}$$

$$\rightarrow \quad \Delta R = (R_{current})^{-1} R_{desired}$$

Subtle Point: Typically, we don't want to align the whole rotation, just the headings

0