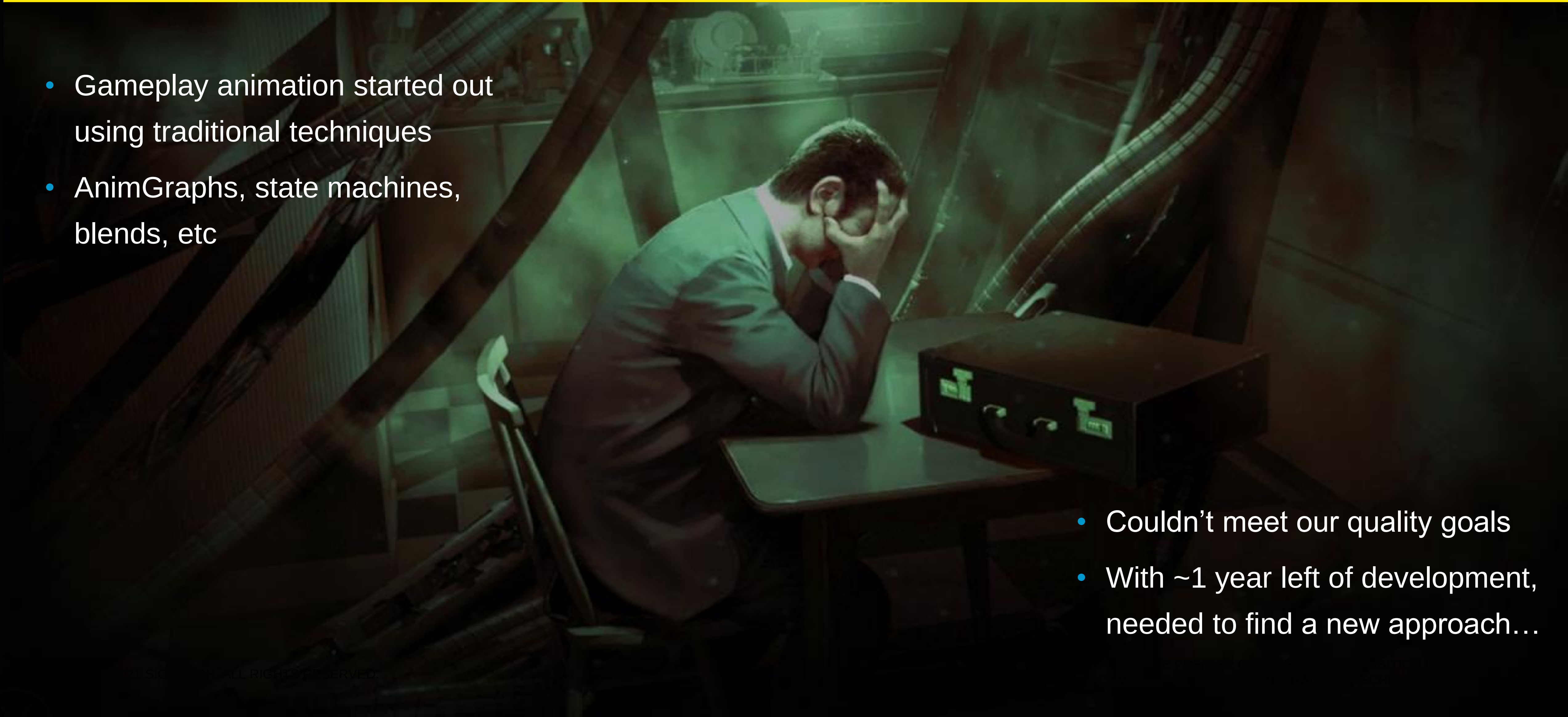
CHARACTER LOCOMOTION IN HALF-LIFE: ALYX

The Right Foot in the Wrong Place

Joe van den Heuvel
Valve

MISSING THE BAR

- Gameplay animation started out using traditional techniques
- AnimGraphs, state machines, blends, etc



- Couldn’t meet our quality goals
- With ~1 year left of development, needed to find a new approach…

Goals

What we needed from a new gameplay animation system

GOALS

SIGGRAPH 2021

QUALITY

- VR “Character Presence”
  - Character version of VR ‘Presence’
  - Easily broken by “gamey” movement
  - Both challenge and opportunity!
- Goal:
  - Reduce or remove foot slide

GOALS

MOVEMENT

- Tight, dense virtual environments
  - Frequent changes of direction
  - Small window for attacks

- Goal:
  - Character cannot leave navigation path
  - Character must stop exactly at path goal

GOALS

SIGGRAPH 2021

CONTENT

- Small gameplay animation team
- Large variety of characters
  - Could not rely on mocap

- Goal:
  - Work with limited content
  - Work for bipeds, quadrupeds… all the -peds

GOALS

SIGGRAPH 2021

PERFORMANCE

- VR Motion Sickness is a “big deal”
- Can’t let game make players feel nauseous
- Must maintain 90+ frames per second
  - 0.011 seconds for entire frame
- Goal:
  - Must be fast to calculate

Options

OPTIONS?

SIGGRAPH 2021

How are others solving these problems?

MOTION MATCHING

Pros:

- Shipped in other games
- Conceptually simple
- Debuggable

Cons:

- Content Quantity => Quality
- Not gonna mocap a headcrab

MACHINE LEARNING

Pros:

- Lots of examples of good results
- Dynamic, adapts to environment

Cons:

- Requires **lots** of example content
- Hard to debug
- Slow iteration time (training large data sets)

OPTIONS?

What we went with

SEMI-PROCEDURAL LOCOMOTION

- 2009 Thesis paper by Rune Skovbo Johansen
- Change foot steps at runtime
- http://runevision.com/thesis/

Pros:

- Can remove foot slide
- Procedural: More variety, Less content
- Works with non-humanoids

Cons:

- Multi-step animations?
- Extracted motion?
- Transitions?
- Foot Rotations?

Creating Foot Motion Data

FOOTBASE

Position (Toe) + Direction (to Heel)

Always Touching Lowest Part of Foot

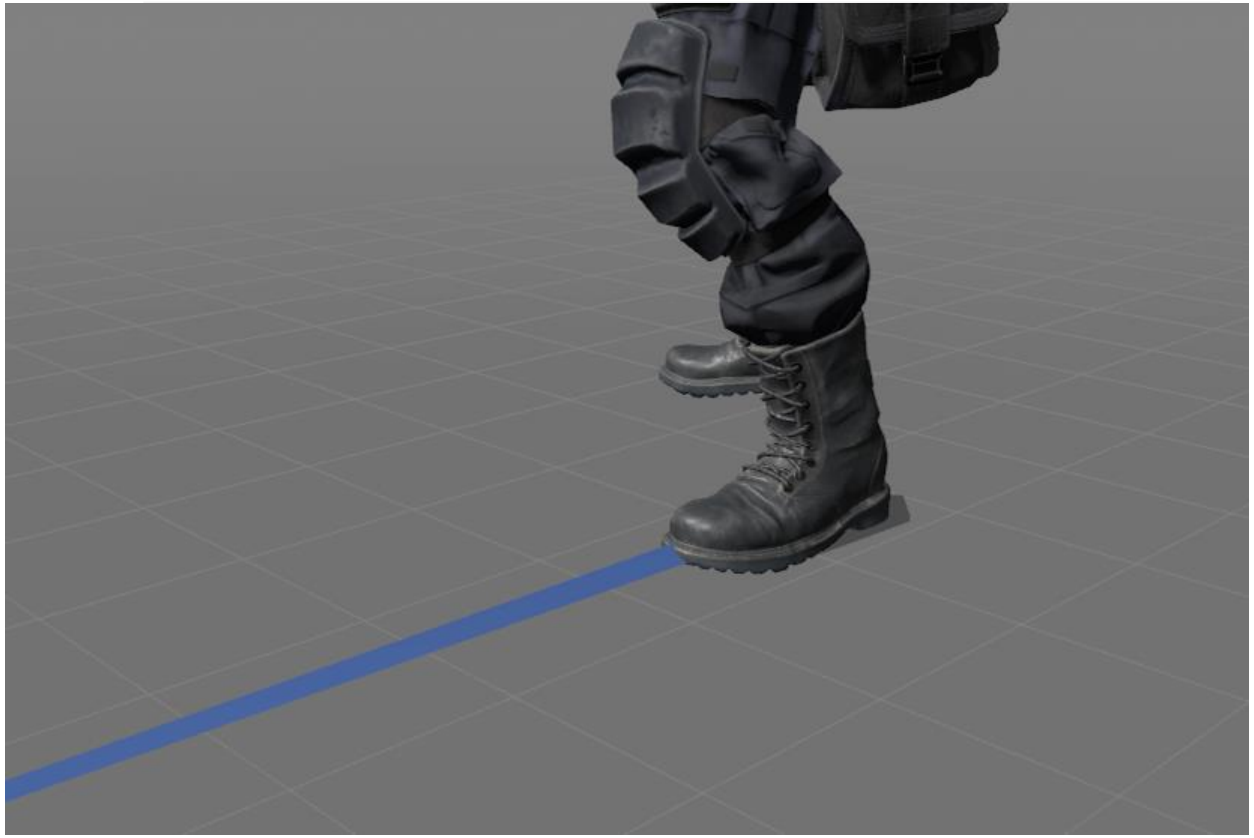
FOOTBASE

STRIDES

For each animation:

- Segment into strides
- Stride start frame: “Stance Frame”
  - Middle of time on ground
  - Foot not moving
  - Close to body

FOOTBASE

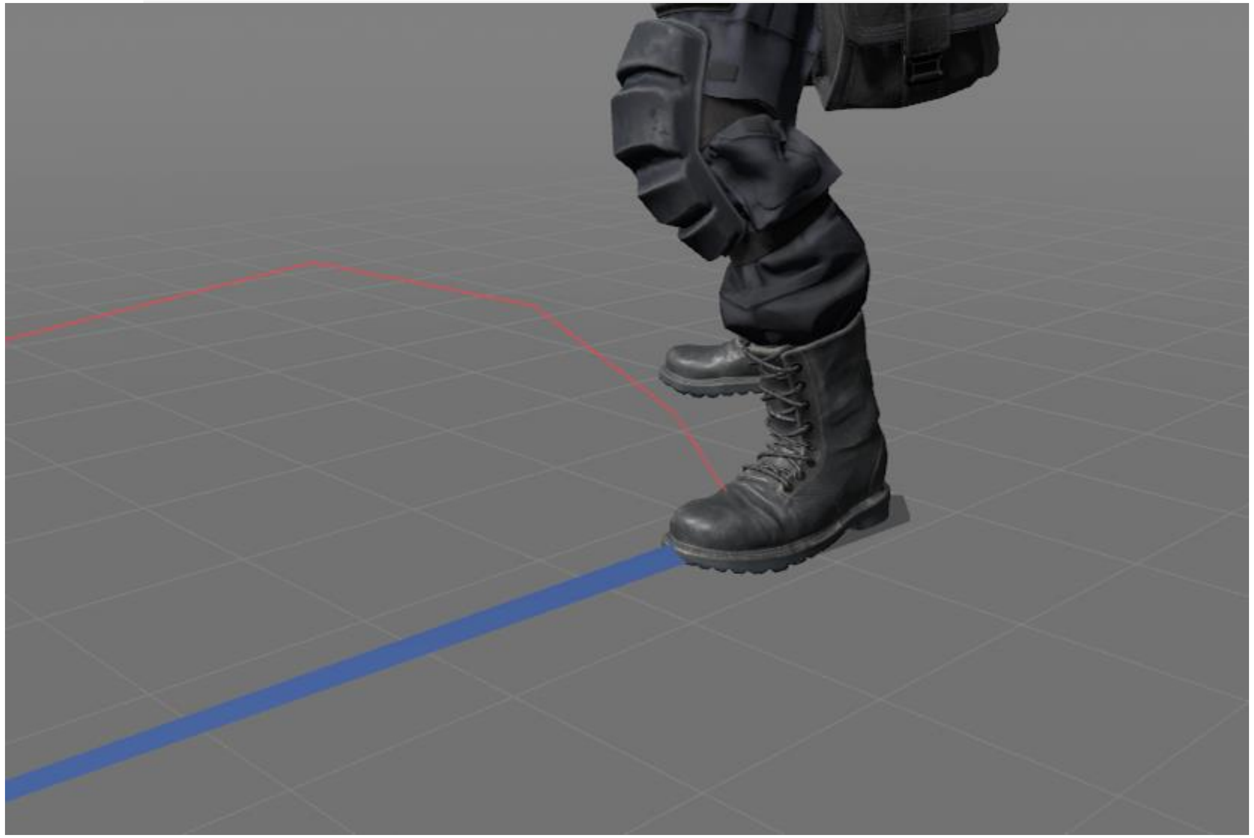


STRIDES

For each Stance Frame:

- Calculate FootBase relative to character
- “Foot Cycle”: time between steps

FOOTBASE



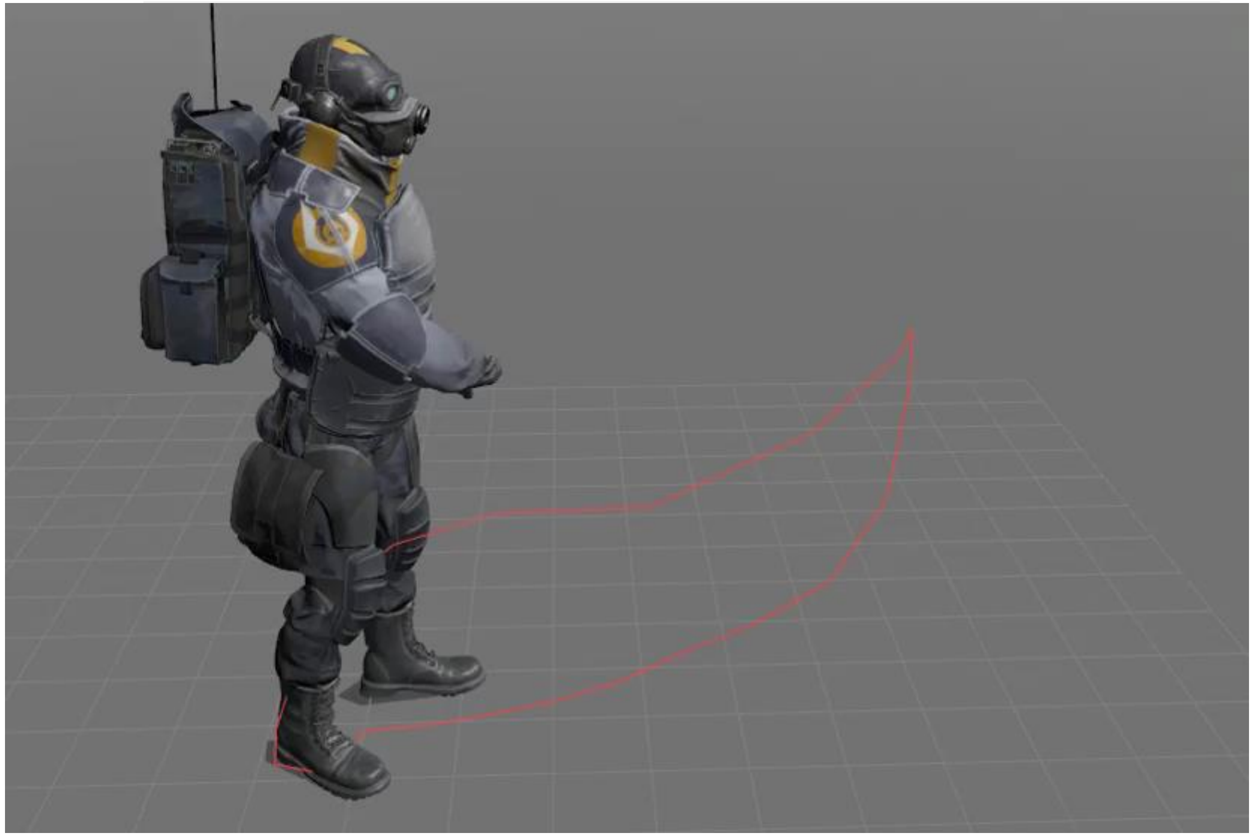
TRAJECTORIES

For each frame of each Foot Cycle:

- Calculate Footbase Position + Direction
- Project onto vector between steps (“Stride Vector”)
- Convert to offset from projected position
- Convert offset to be stride-relative

FOOTBASE

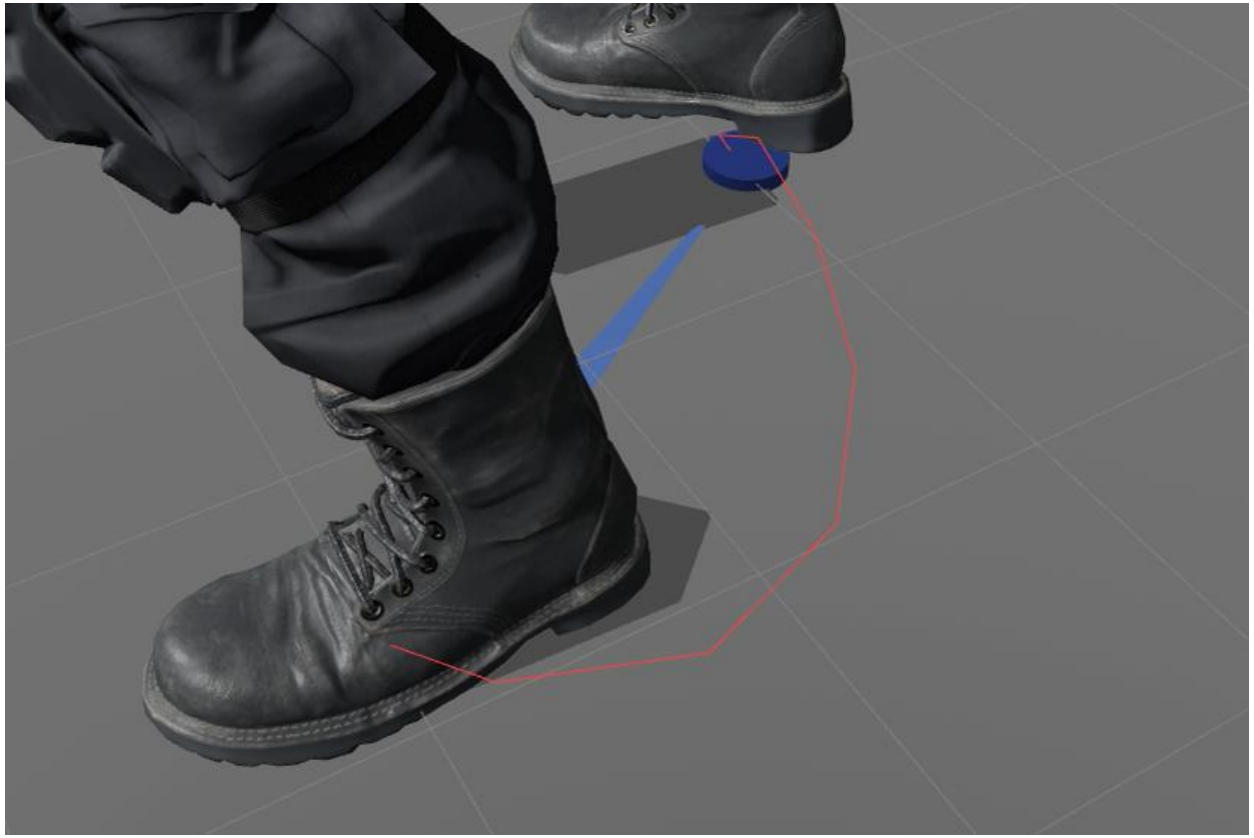
SIGGRAPH 2021



STATIONARY ANIMATIONS

- Previous Step Position == Next Step Position
- All motion contained in Trajectory Offset

FOOTBASE



ROTATION

Just like Translation:

- RotationReference = Lerp(Start, End, Cycle)
- Final Rotation = RotationReference + RotationOffset
- Must handle > 180 degree rotations!

STRIDE DATA

SIGGRAPH 2021

FOOT CYCLE DEFINITION (ONE FOR EACH STEP)

```
Float3   stancePosition;          // Starting Footbase position, in model space
Float    stanceDirection;         // Starting Footbase direction, in model space
Float    stanceCycle;             // Animation cycle (0-1) when foot cycle starts

Float3   middlePosition;          // Footbase position halfway through the step
Float3   toStrideStartPos;        // Vector from the end of the stride to the start

// Foot cycles (0-1) of when the foot lifts and lands
Float footLiftCycle, footOffCycle, footStrikeCycle, footLandCycle;
```

STRIDE DATA

FOOTBASE TRAJECTORY (ONE FOR EACH FRAME)

```
Float3   translationOffset;        // Stride-relative offset from stride vector
Float    rotationOffset;           // Stride-relative offset from stride rotation
Float    progression;              // Location of projected pos as % along stride vector
```

Stride Retargeting

STRIDE RETARGETING

RUNTIME PLAYBACK

1. Predict next step position
2. Calc FootBase from Trajectories
3. Use FootBase as IK target

STRIDE RETARGETING

SIGGRAPH 2021

Lowest point on foot must touch the FootBase

Calculate ankle position & rotation from FootBase

Use ankle as IK target for leg

STRIDE RETARGETING

INVERSE KINEMATICS

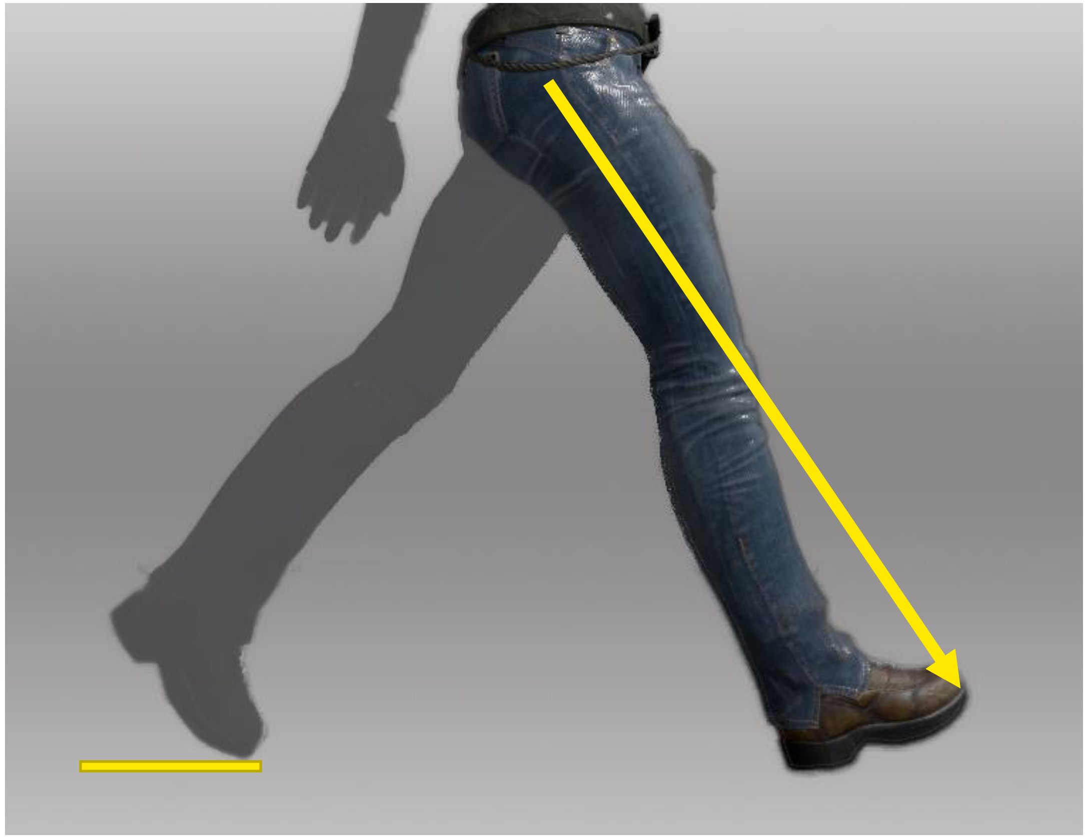
Can’t use original ankle rotation

STRIDE RETARGETING

INVERSE KINEMATICS

Bad things happen

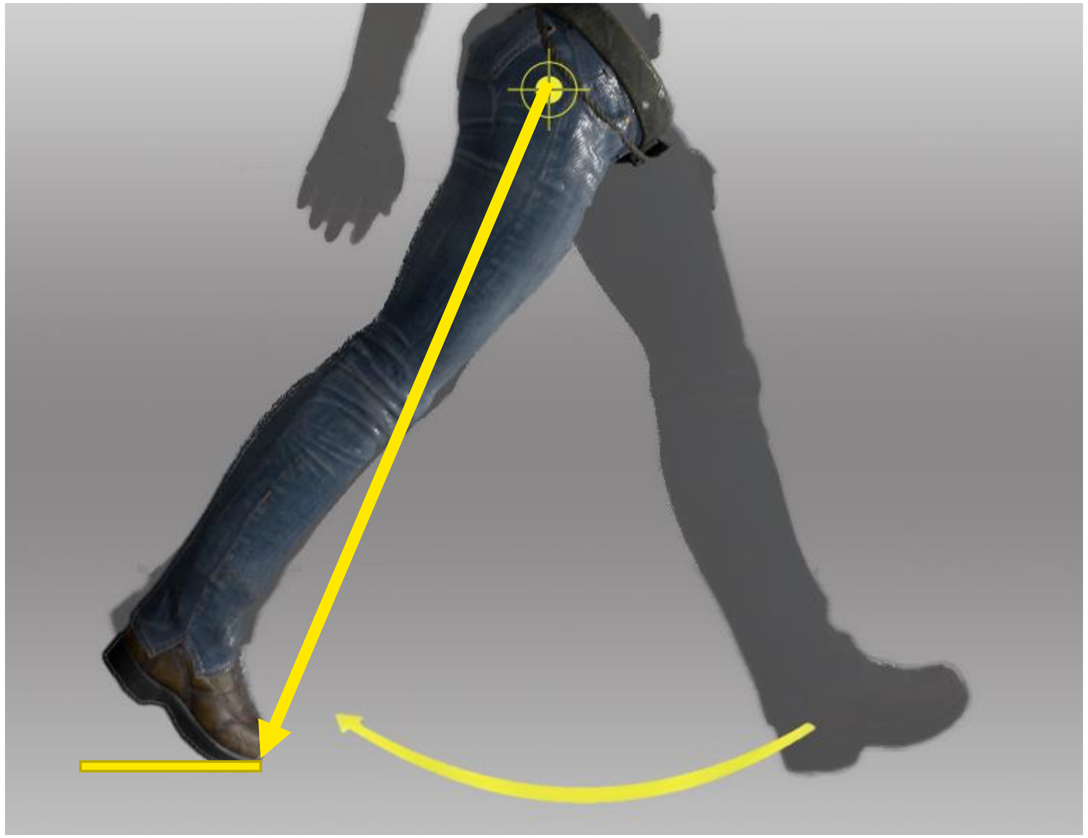
STRIDE RETARGETING



INVERSE KINEMATICS

1. **Start by rotating the leg about the hip**

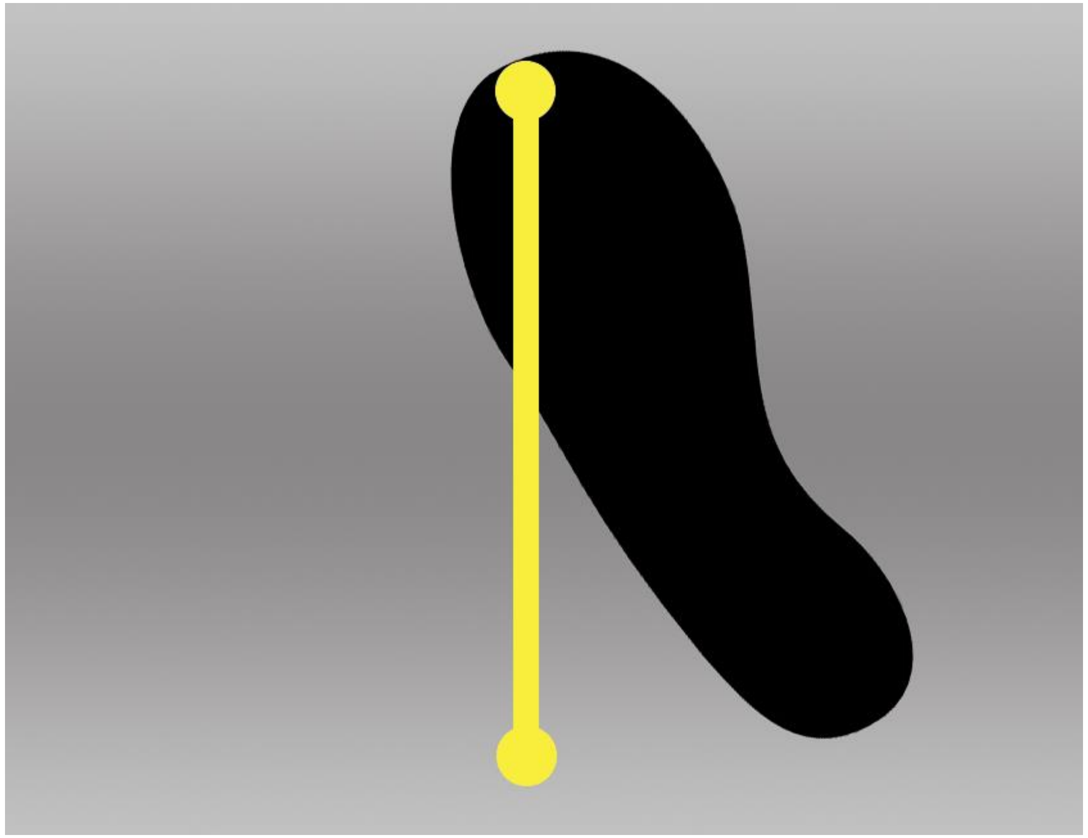
STRIDE RETARGETING



INVERSE KINEMATICS

1. **Start by rotating the leg about the hip**
   – Preserves natural foot angle

STRIDE RETARGETING



INVERSE KINEMATICS

1. Start by rotating the leg about the hip
2. **Align Horizontally**
   - Align Foot to FootBase as it gets flat
   - Otherwise, pop when lowest point changes

STRIDE RETARGETING

INVERSE KINEMATICS

1. Start by rotating the leg about the hip
2. Align Horizontally
3. **Align Vertically**
   - Align with sloped FootBase based on:
     - How flat Foot is in original anim
     - At beginning or end of stride

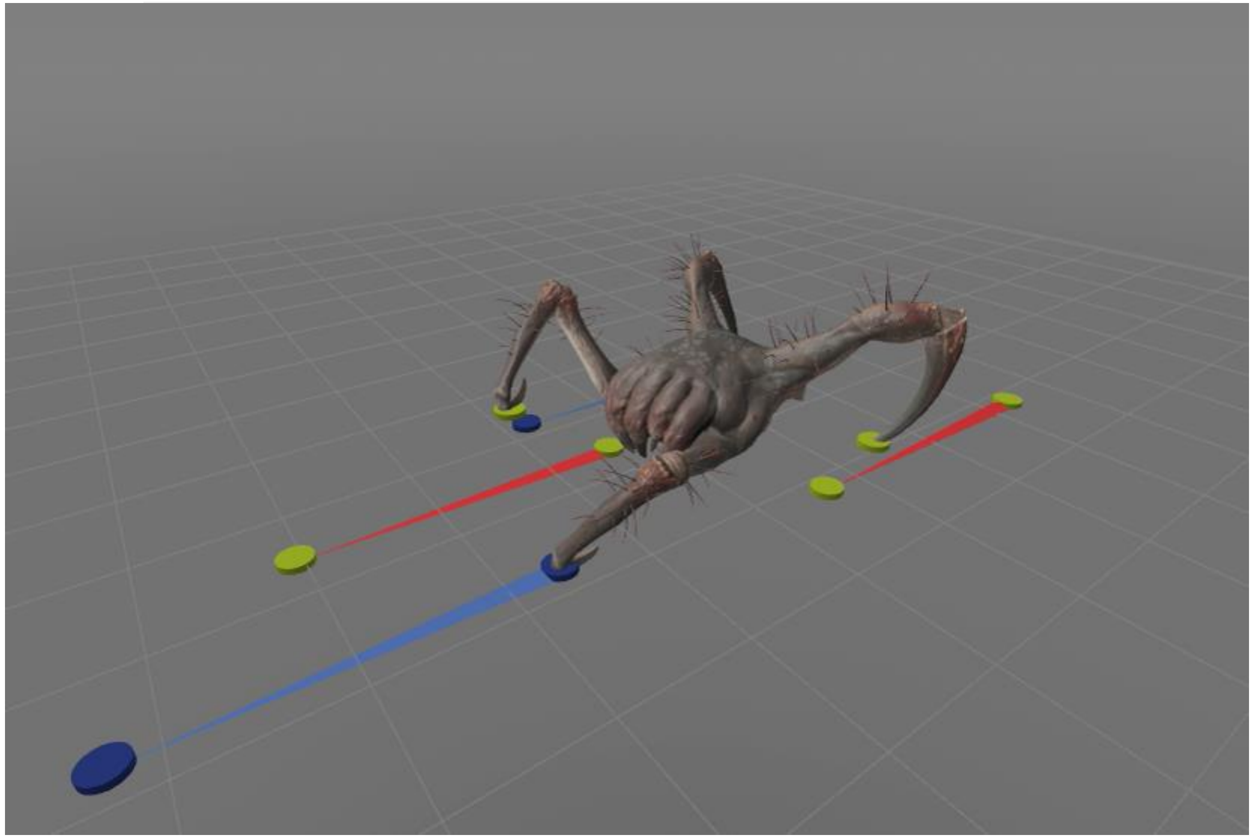
STRIDE RETARGETING

INVERSE KINEMATICS

1. Start by rotating the leg about the hip
2. Align Horizontally
3. Align Vertically
4. **Solve Leg IK for Ankle Position**
   - Second pass to enforce Ankle limits

Manipulating the Foot Motion Data

MANIPULATING THE DATA

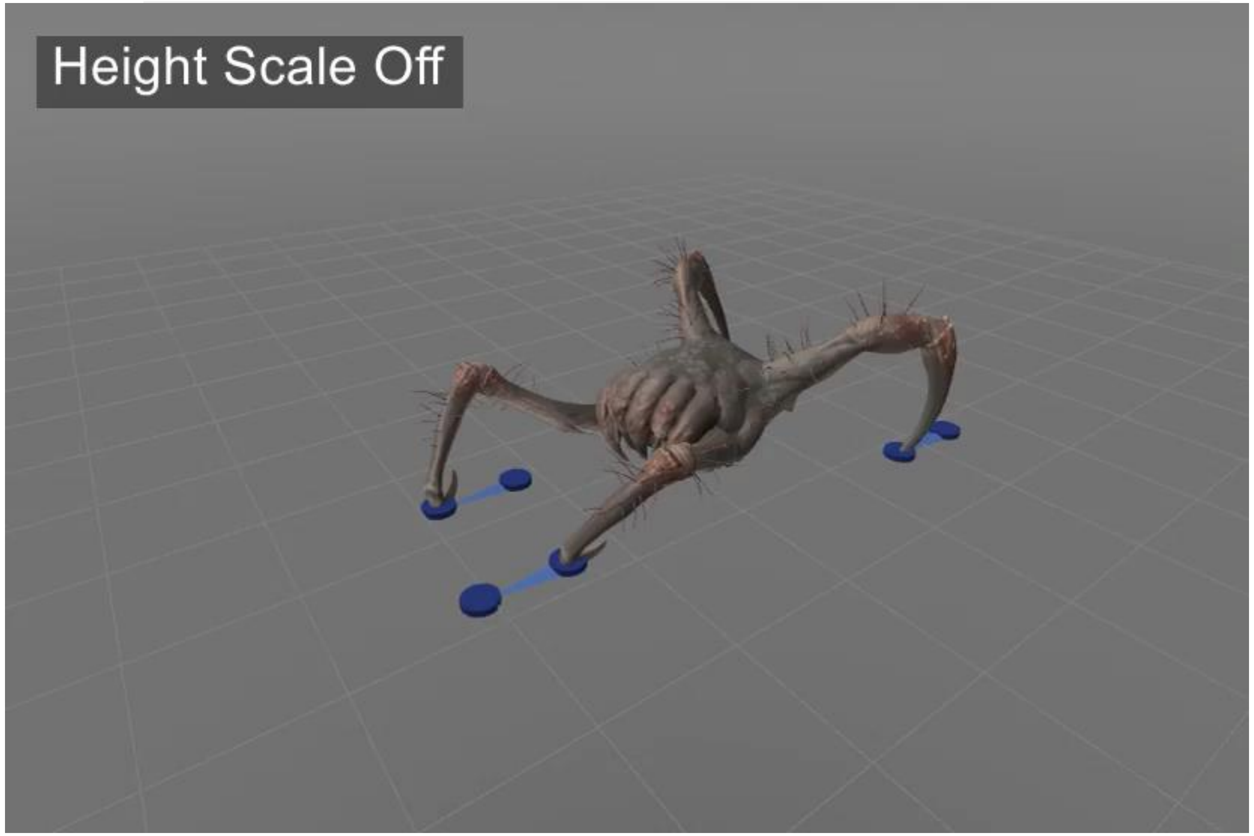


STEP LENGTH

- Scale distance traveled
- Affects foot step distance

MANIPULATING THE DATA

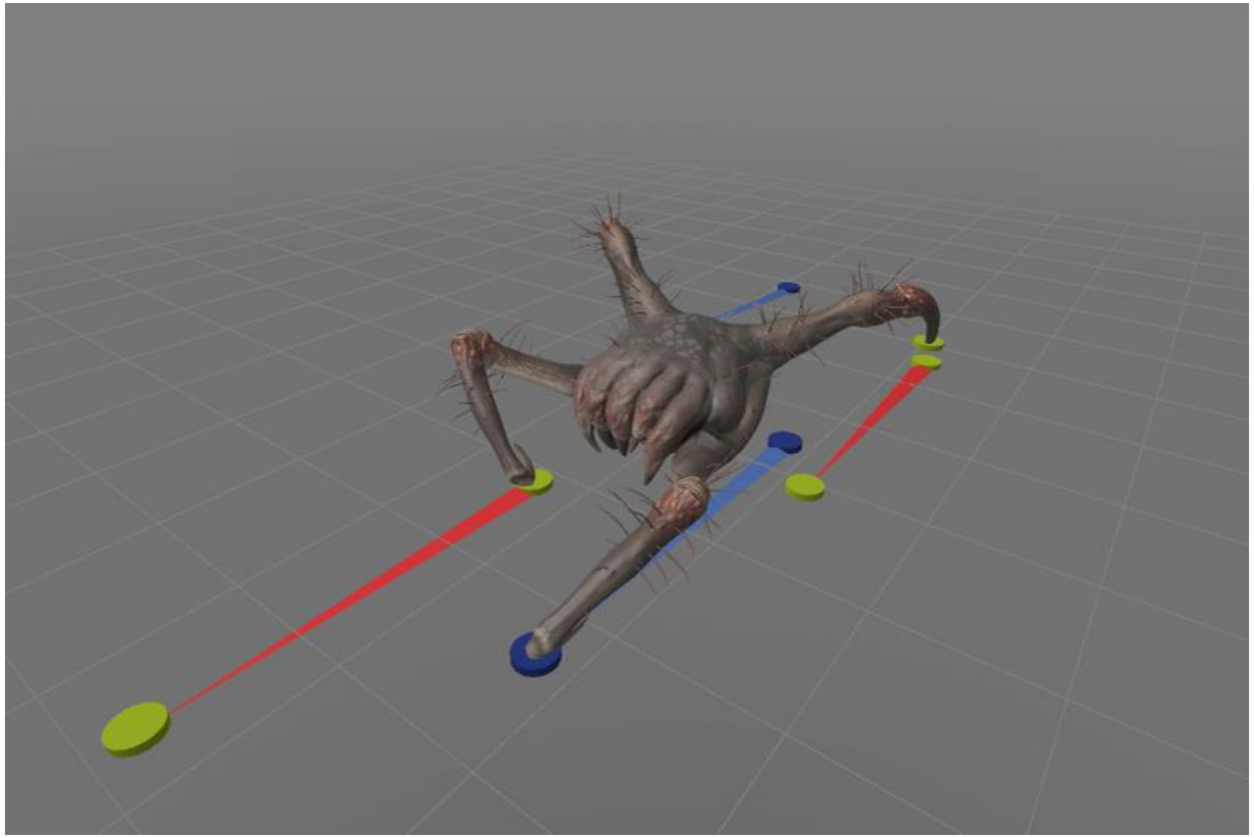
SIGGRAPH 2021



STEP HEIGHT

- Scale Trajectory Translation Offset
- Affects height/intensity of step
- Scale step height with distance for consistency

MANIPULATING THE DATA

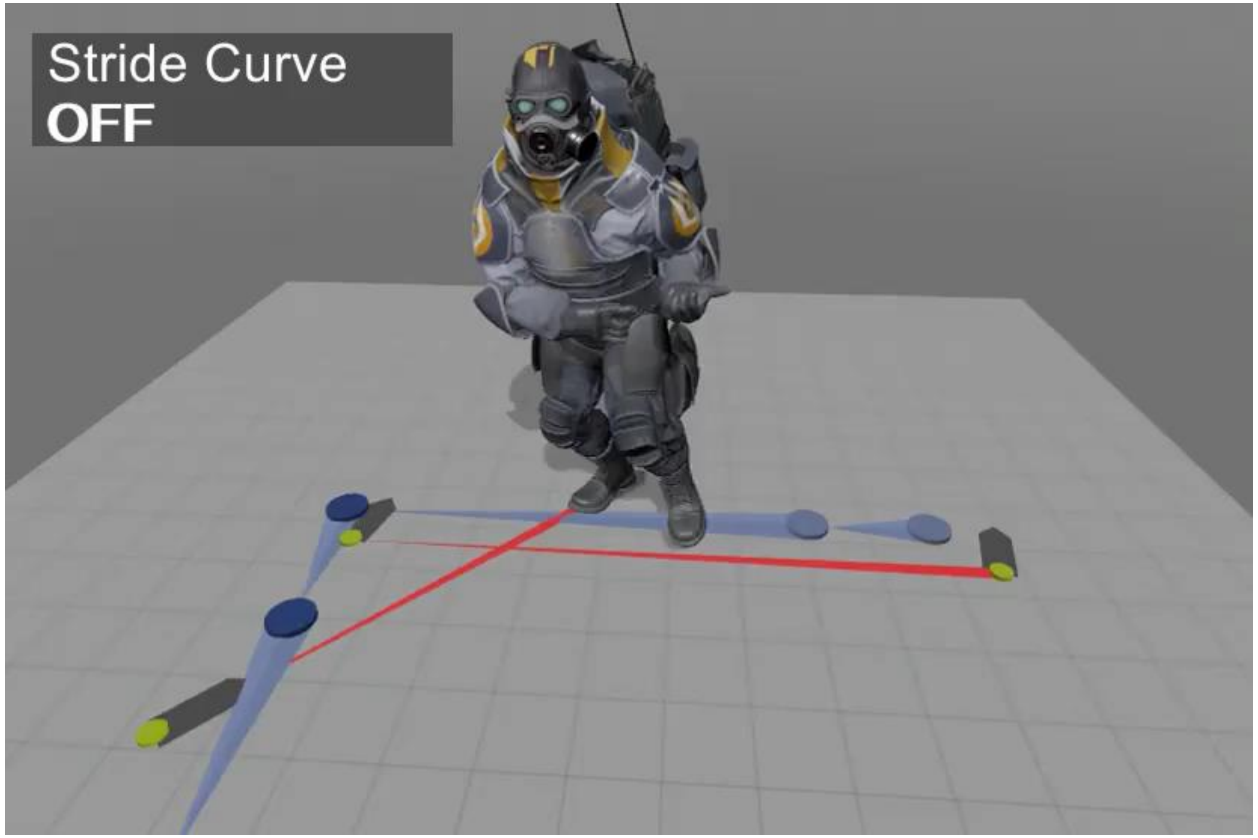


STEP LOCATION

Change steps to go to new location

- Turning
- Uneven ground / stairs

MANIPULATING THE DATA



CURVED STRIDE PATHS

- Curve path through original midpoint to prevent leg intersection
- Increase arch when walking up slopes

Transitions & Blending

All foot motion data can blend…

Except for **Progression**

Blends

Transitions

Everything Else

TRANSITIONS AND BLENDING

BLENDS

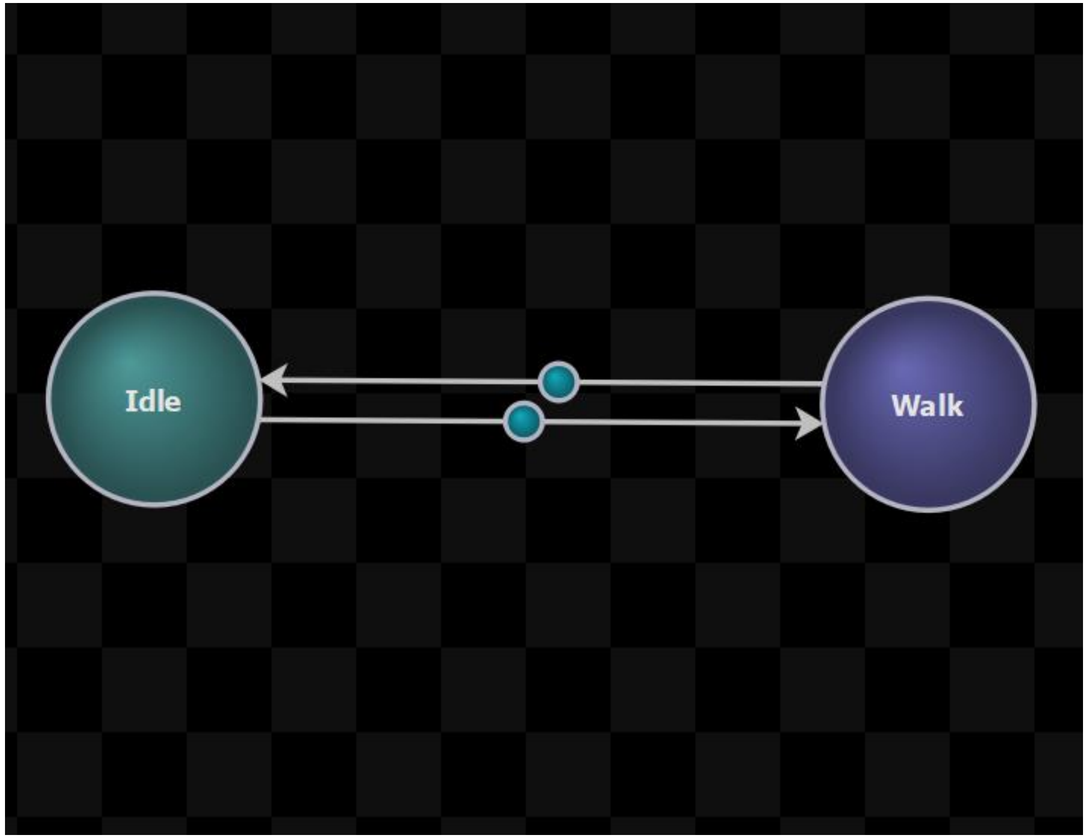
Match timing between animations

- Sync foot cycles between animations
  - Force steps to start and end at same time
- Common practice in games

Can cheat!

- Fudge numbers on foot motion data
- System will move feet to right place

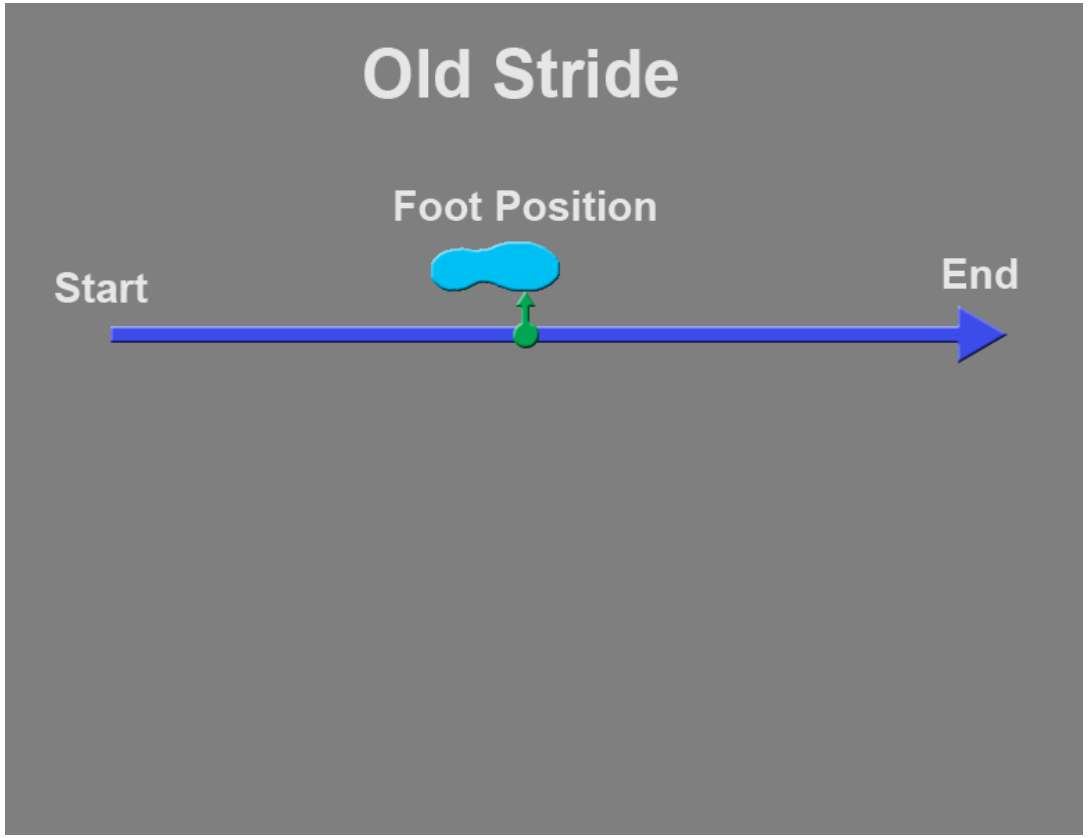
TRANSITIONS AND BLENDING



TRANSITIONS

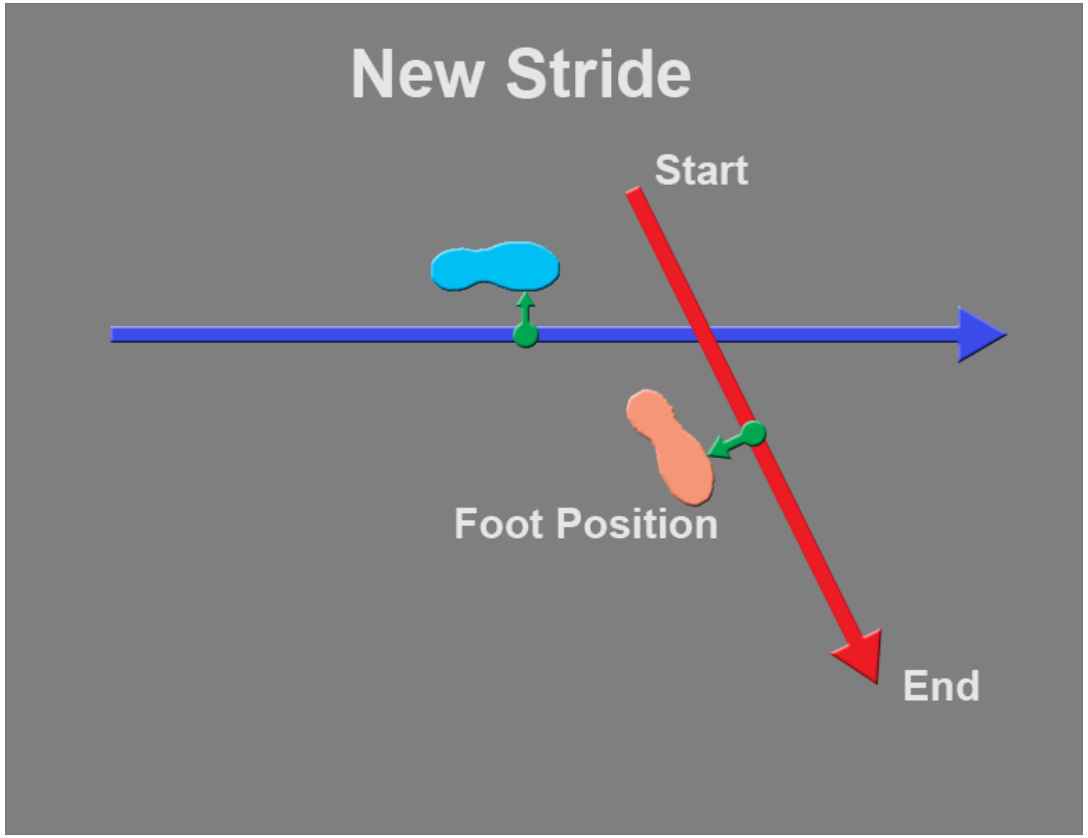
- Foot cycles are out of sync
- Can’t blend, must transition

TRANSITIONS AND BLENDING



TRANSITIONS

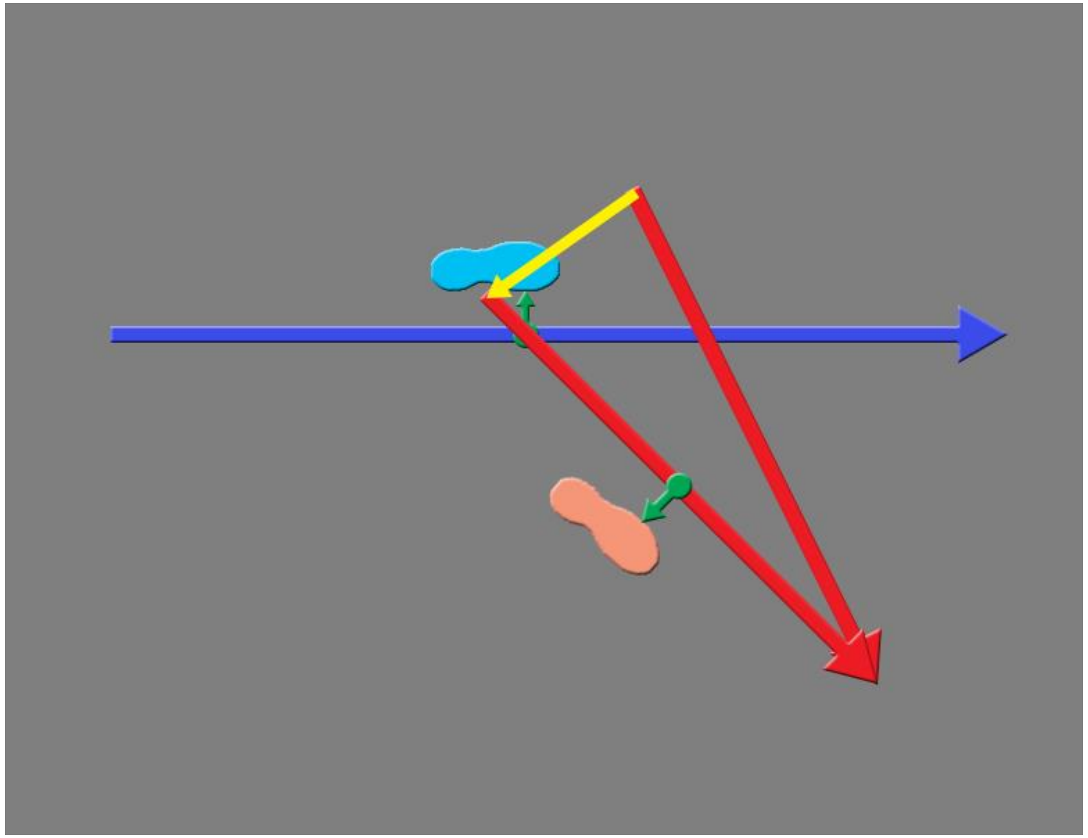
TRANSITIONS AND BLENDING



TRANSITIONS

- Can’t change End of new stride
- Can change Start

TRANSITIONS AND BLENDING



TRANSITIONS

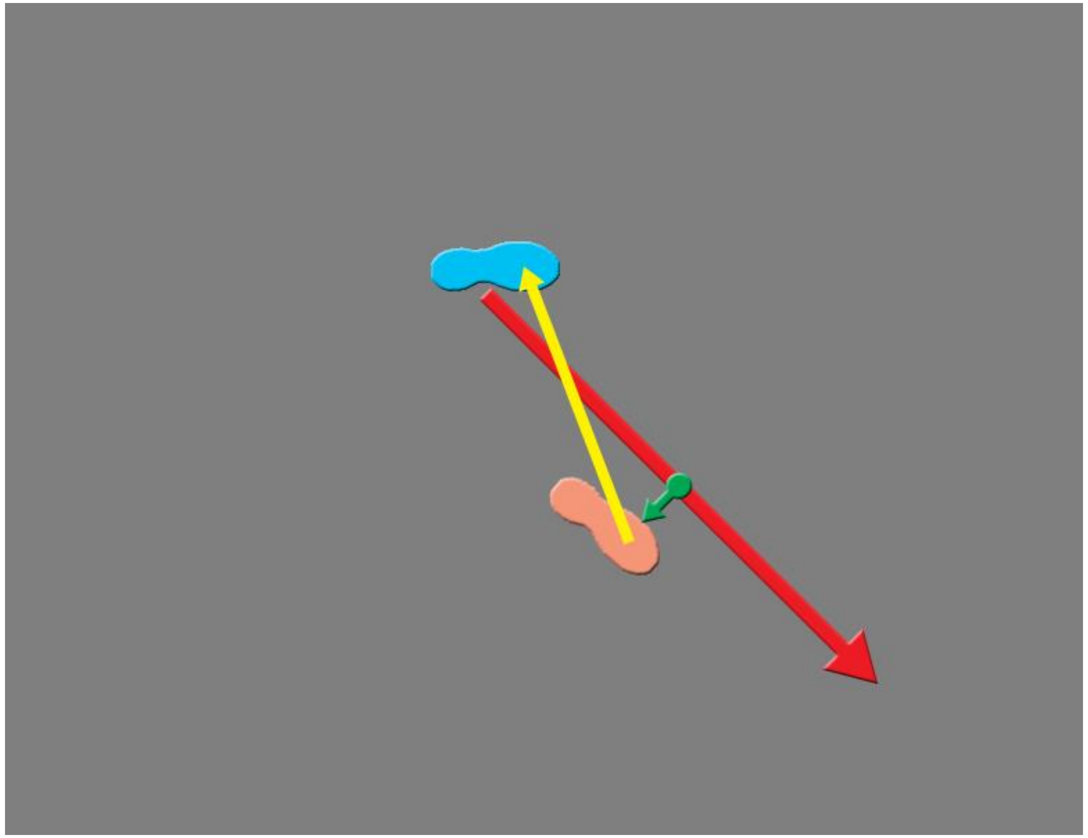
If foot is stationary on ground

- StartPos = CurrentPos

Else

- Move StartPos so reference points match
- Clamp length of stride to original length

TRANSITIONS AND BLENDING



TRANSITIONS

- Calc remaining Offset between old and new foot positions
- Blend out Offset over time

TRANSITIONS AND BLENDING

SIGGRAPH 2021

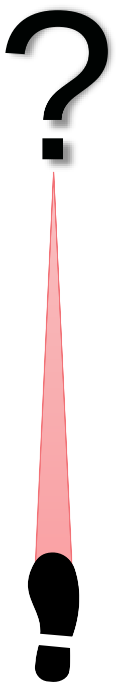
EVERYTHING ELSE

- Additive, per-bone blends, etc work
- But have to pick which source to pull foot motion from

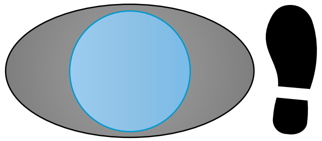
Predicting the Next Step

FOOTSTEP PREDICTION

WHERE WILL THE NEXT STEP LAND?



FOOTSTEP PREDICTION

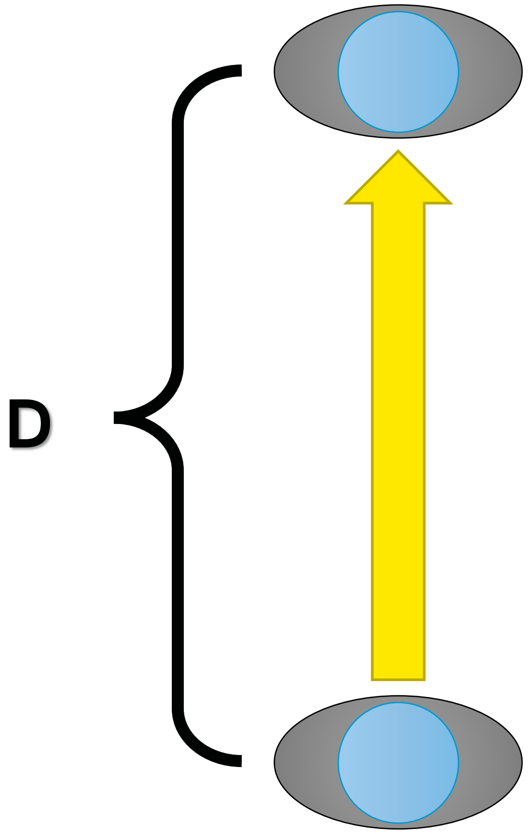


WHERE WILL THE NEXT STEP LAND?

- Foot Motion Data
  - Offset from character at each step
  - Animation frame of each step

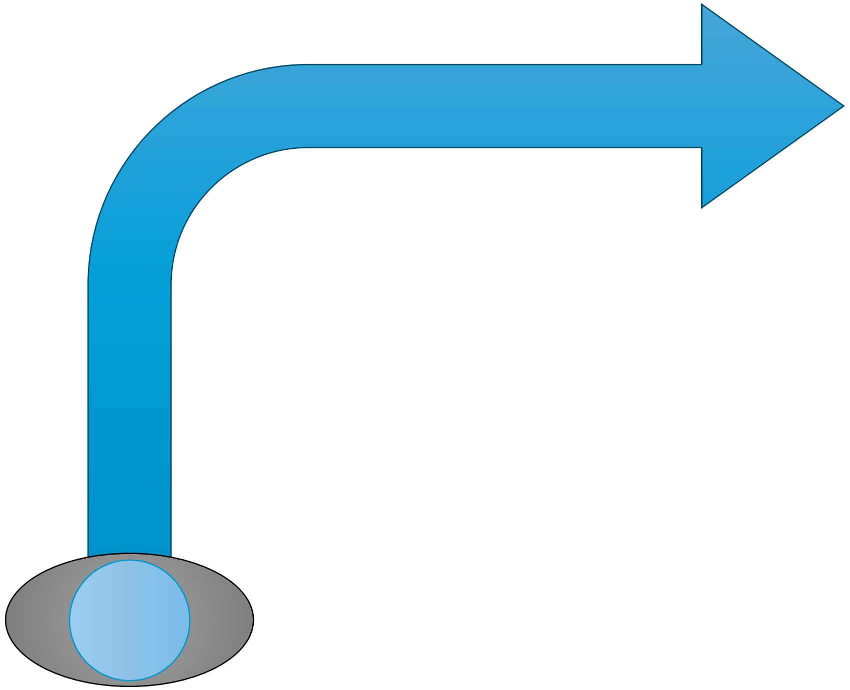
- => Figure out where character will be at that time

FOOTSTEP PREDICTION



ROOT MOTION

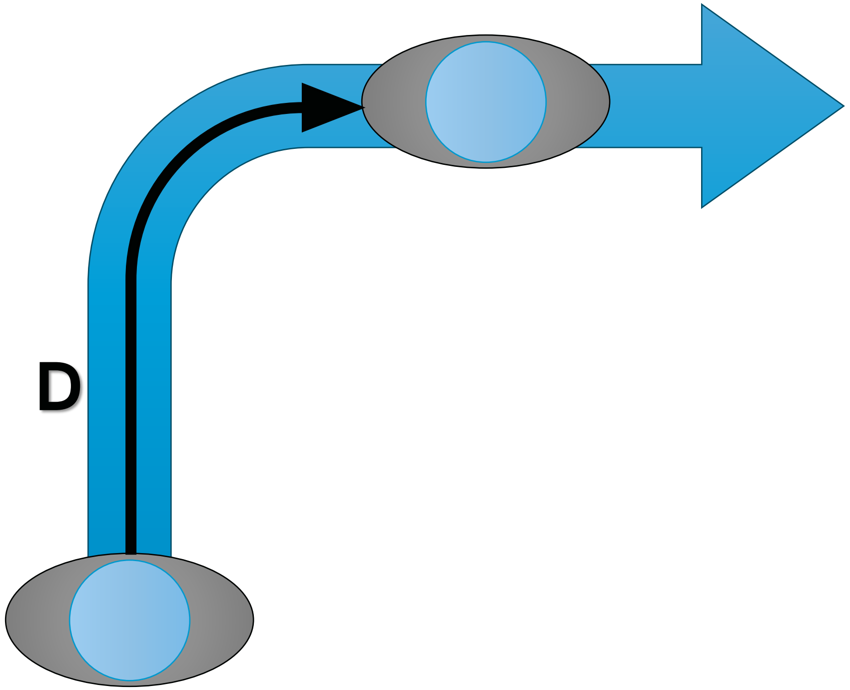
- Add up root motion from now till next step



PATHS

Given a navigation path…

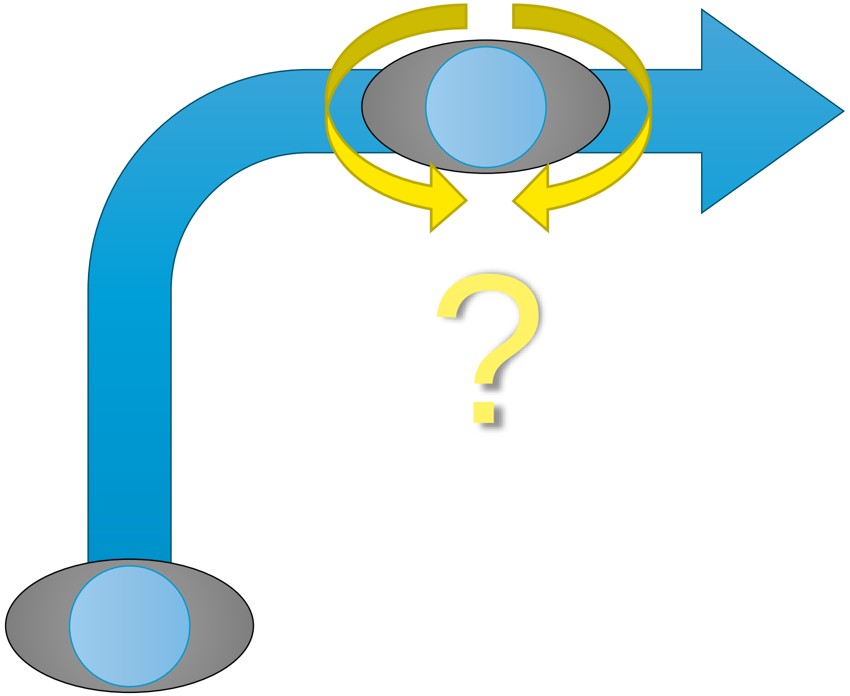
FOOTSTEP PREDICTION



PATHS

… Move along the path by same distance as animation

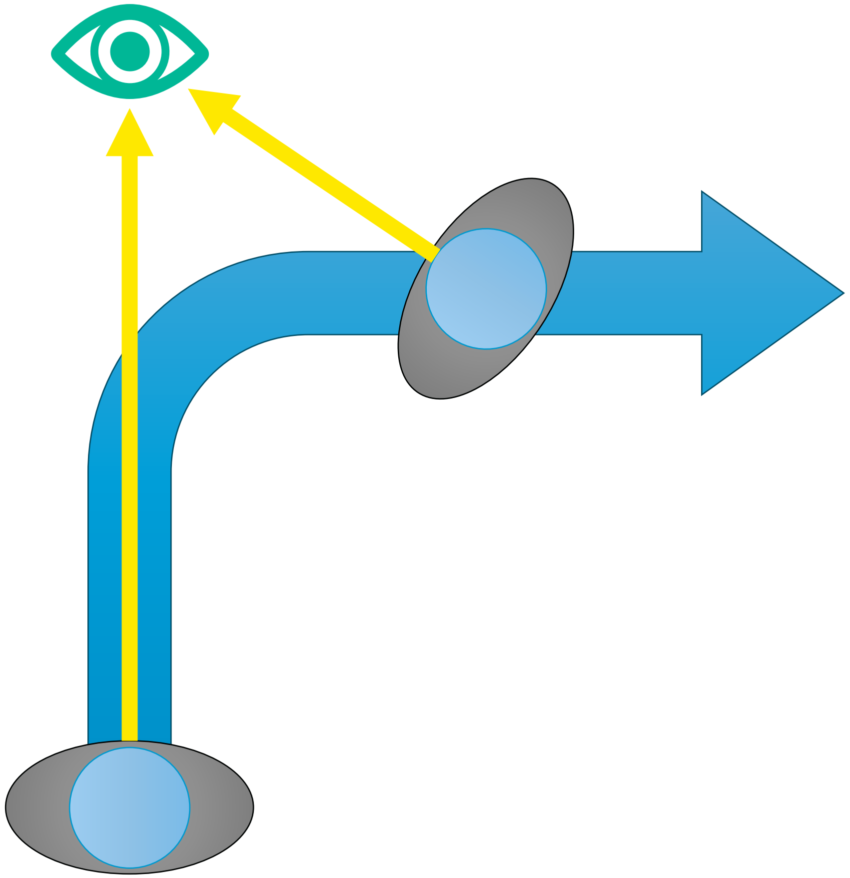
FOOTSTEP PREDICTION



ROTATION

- Need to know facing direction at future location
- 2 Modes:
  - Target Look Mode
  - Path Look Mode

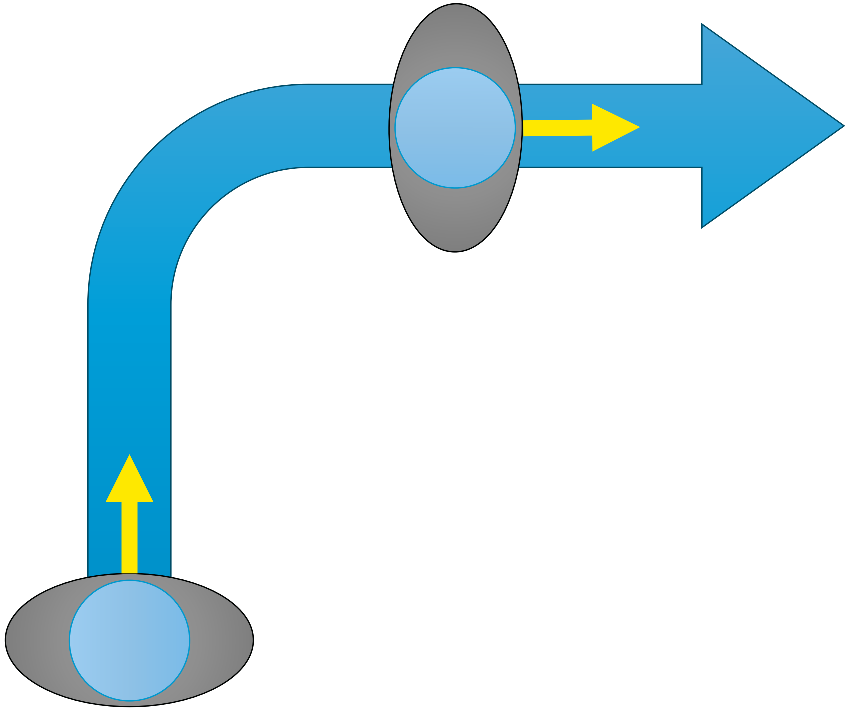
FOOTSTEP PREDICTION



ROTATION

- Target look mode
  - Always facing a target point in the world
  - Assume stationary target

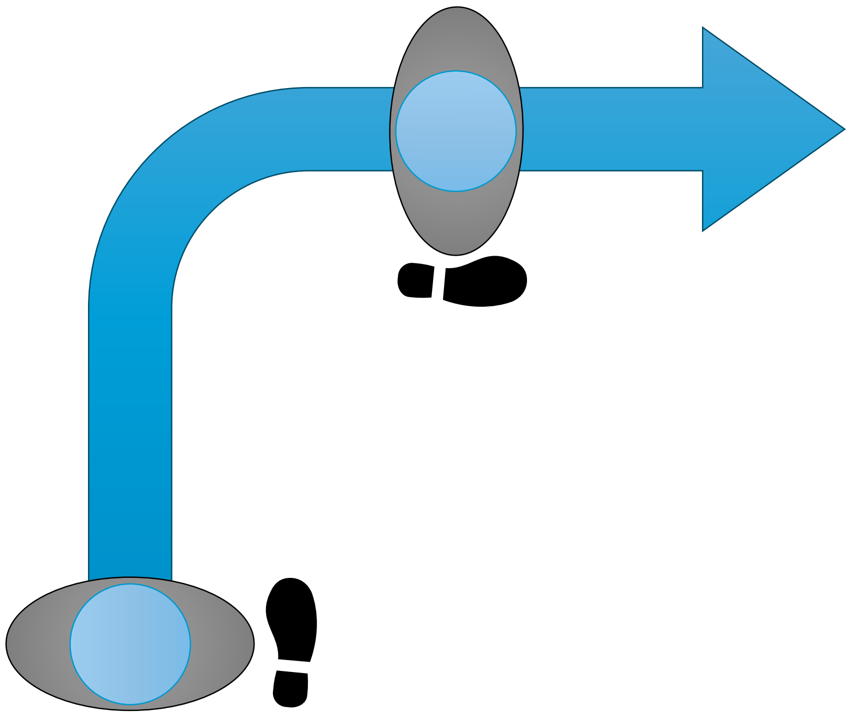
FOOTSTEP PREDICTION



ROTATION

- Path Look Mode:
  - Always facing forward along the path

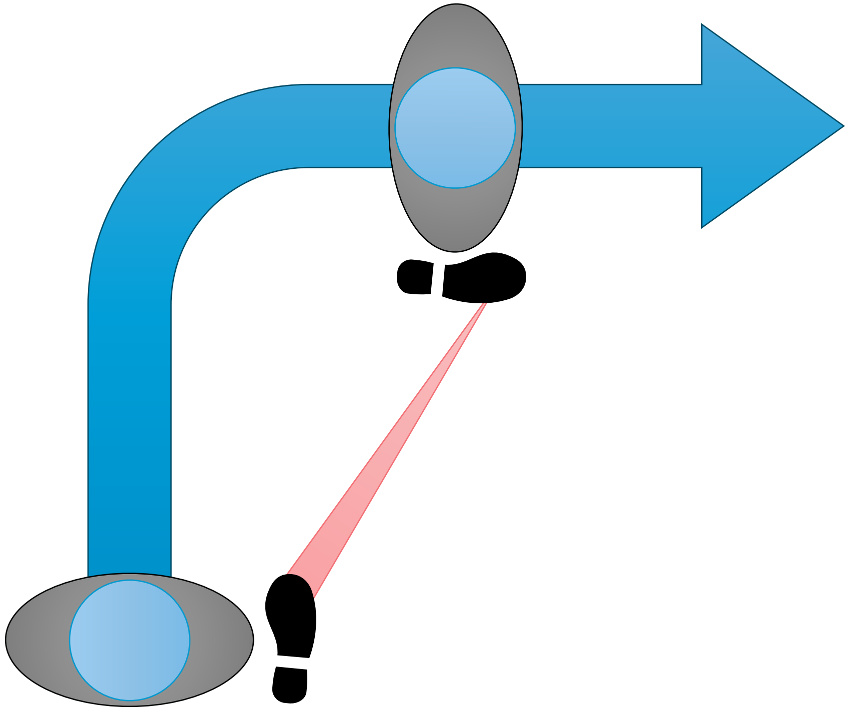
FOOTSTEP PREDICTION



PLACE FOOT

- Use FootBase offset to find foot position/rotation

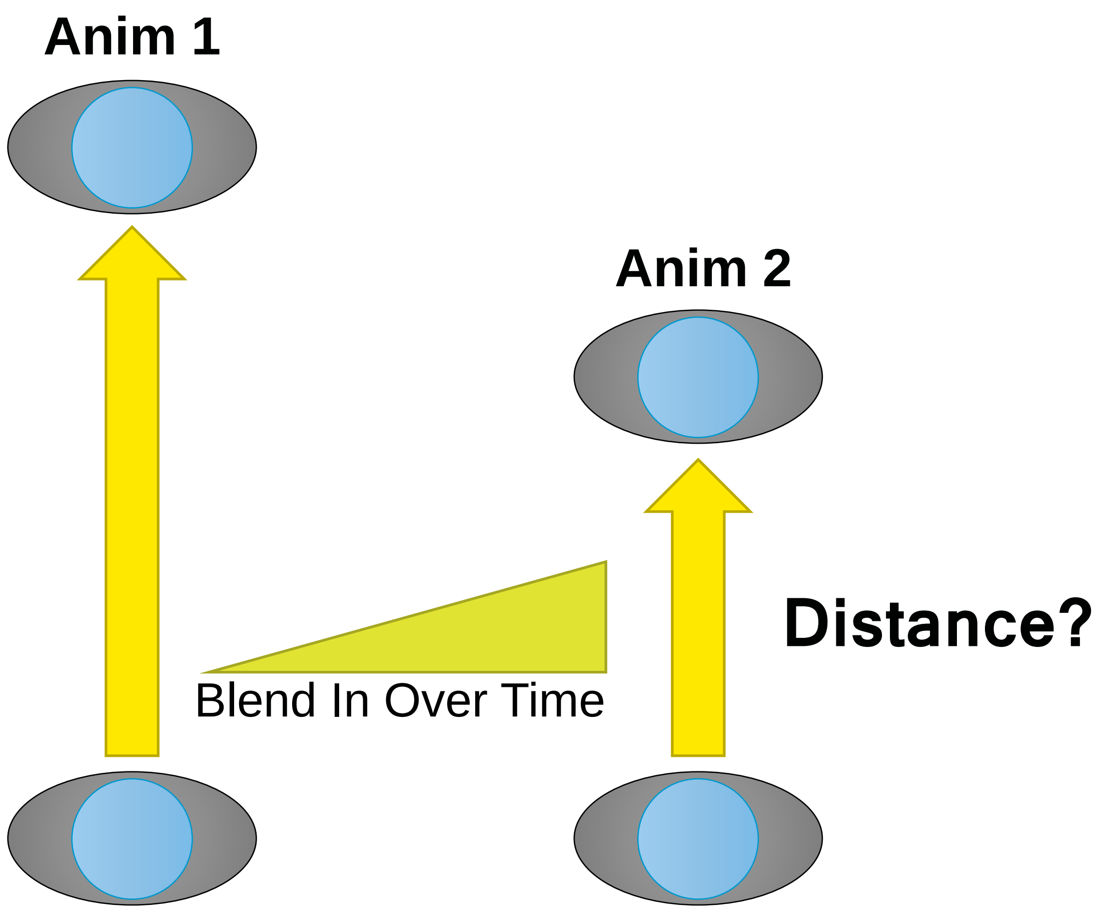
FOOTSTEP PREDICTION



PLACE FOOT

- Update predicted foot position each frame
  - Ideally, shouldn't move
  - Don't change once foot lands
  - Becomes the "Previous" position for next step

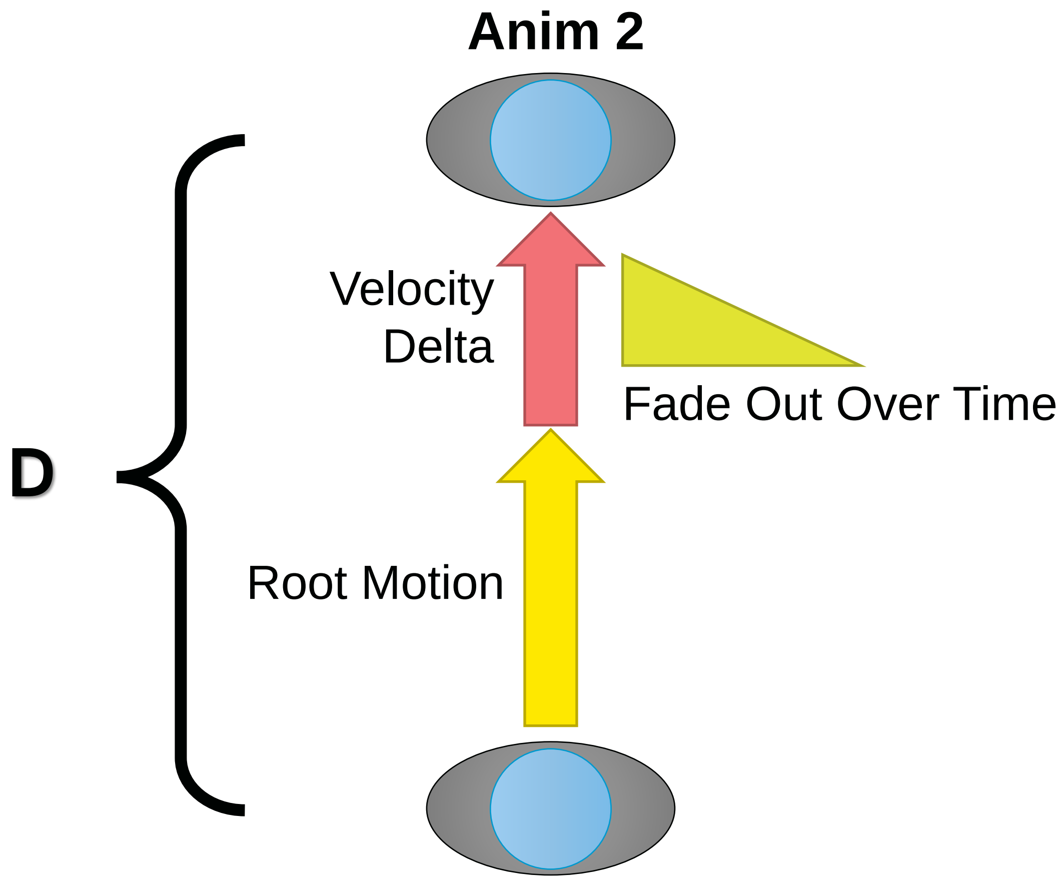
FOOTSTEP PREDICTION



TRANSITION BLENDING

- How to predict character position when cross-fading animations?

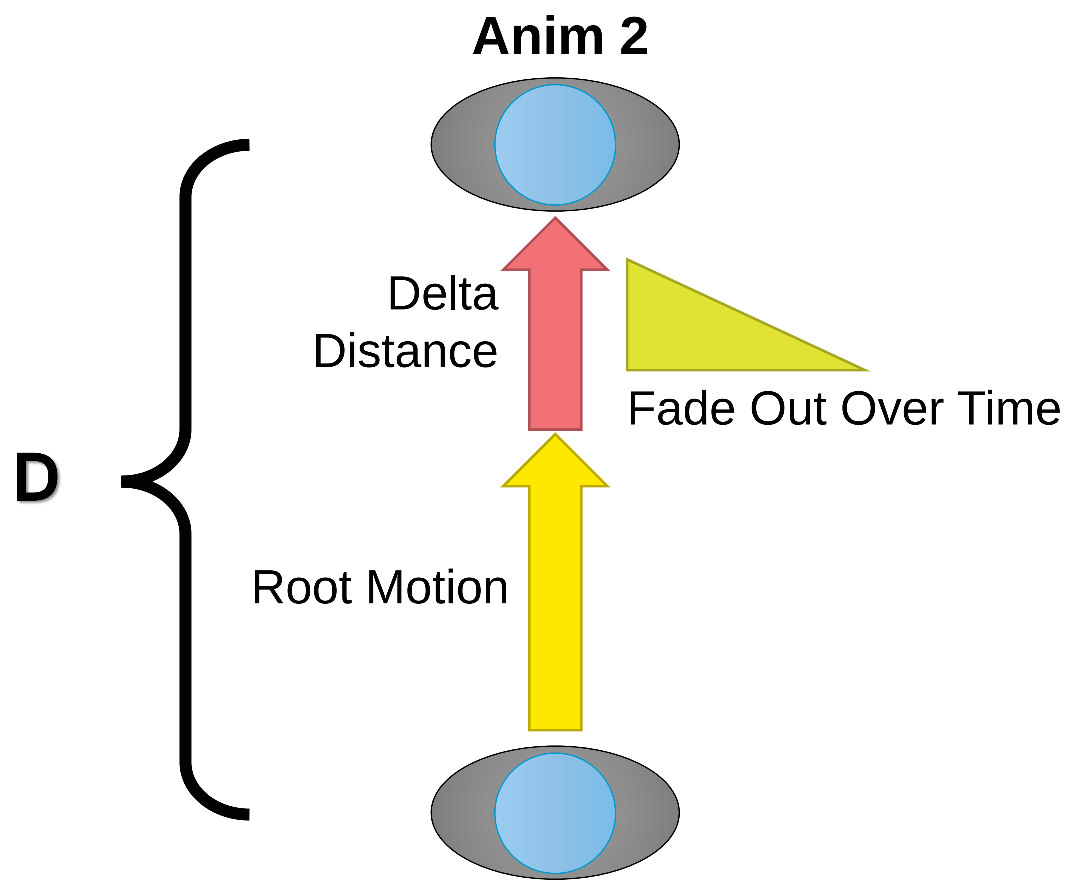
FOOTSTEP PREDICTION



TRANSITION BLENDING

- Find Velocity Delta at start of transition
- Add Velocity Delta to root motion each frame
- Reduce Velocity Delta to 0 over time

FOOTSTEP PREDICTION



FIND TOTAL MOVEMENT FROM VELOCITY DELTA

- Velocity added to Root Motion at time t:

V(t) = VelocityDelta * (1 - t / BlendTime)

- Integrate to get distance at time t:

D(t) = (K/2)*t^2 + VelocityDelta *t + C

Where K = -VelocityDelta / BlendTime

Total Delta Distance = $\int_{t0}^{t1} D(t)$

Animation Selection

ANIMATION SELECTION

- Stride Retargeting can change animations
- Extreme changes look bad
- Need to pick closest animation…

ANIMATION SELECTION

SIGGRAPH 2021

- Stride Retargeting can change animations
- Extreme changes look bad
- Need to pick closest animation…

- **Motion Matching**

MOTION MATCHING

SIGGRAPH 2021

Why Now?

- Don’t need anim for every possibility
  - Stride Retargeting fills the gaps
- Already calculating future “goal” state
- Leverage foot motion data

MOTION MATCHING OVERVIEW

Metric 1

Metric 2

Metric 3

Metric 4

- Metrics = measurements about the state of an animation. Can be about current state or desired future state
  - Eg: Current Velocity, Future Velocity

MOTION MATCHING OVERVIEW

SIGGRAPH 2021

All Animations

	Sample 1	Sample 2	Sample 3	Sample 4	···	Sample N
Metric 1	Value 1	Value 1	Value 1	Value 1	···	Value 1
Metric 2	Value 2	Value 2	Value 2	Value 2	···	Value 2
Metric 3	Value 3	Value 3	Value 3	Value 3	···	Value 3
Metric 4	Value 4	Value 4	Value 4	Value 4	···	Value 4

Sample all the animations at a fixed interval and calculate the value for each Metric

MOTION MATCHING OVERVIEW

SIGGRAPH 2021

	Goal State		All Animations				
		Sample 1	Sample 2	Sample 3	Sample 4	…	Sample N
Metric 1	Value 1	Value 1	Value 1	Value 1	Value 1	…	Value 1
Metric 2	Value 2	Value 2	Value 2	Value 2	Value 2	…	Value 2
Metric 3	Value 3	Value 3	Value 3	Value 3	Value 3	…	Value 3
Metric 4	Value 4	Value 4	Value 4	Value 4	Value 4	…	Value 4

Define a Goal State, that represents the desired state of the character.

Calculate a ‘score’ for each sample based on how close it is to the Goal State

MOTION MATCHING OVERVIEW

SIGGRAPH 2021

	Goal State		All Animations					
			Sample 1	Sample 2	Sample 3	Sample 4	…	Sample N
Metric 1	Value 1		Value 1	Value 1	Value 1	Value 1	…	Value 1
Metric 2	Value 2		Value 2	Value 2	Value 2	Value 2	…	Value 2
Metric 3	Value 3		Value 3	Value 3	Value 3	Value 3	…	Value 3
Metric 4	Value 4		Value 4	Value 4	Value 4	Value 4	…	Value 4

Score = $\sqrt{\sum((GoalValue_n - TestValue_n)^2 * MetricWeight_n)}$

MOTION MATCHING OVERVIEW

	Goal State	Current State	All Animations: Sample 1	Sample 2	Sample 3	Sample 4	…	Sample N
Metric 1	Value 1	Value 1	Value 1	Value 1	Value 1	Value 1	…	Value 1
Metric 2	Value 2	Value 2	Value 2	Value 2	Value 2	Value 2	…	Value 2
Metric 3	Value 3	Value 3	Value 3	Value 3	Value 3	Value 3	…	Value 3
Metric 4	Value 4	Value 4	Value 4	Value 4	Value 4	Value 4	…	Value 4

Sample with lowest score becomes the current state.

Repeat process, but now the sample has to score better than the current state.

Using motion matching is like raising a child…

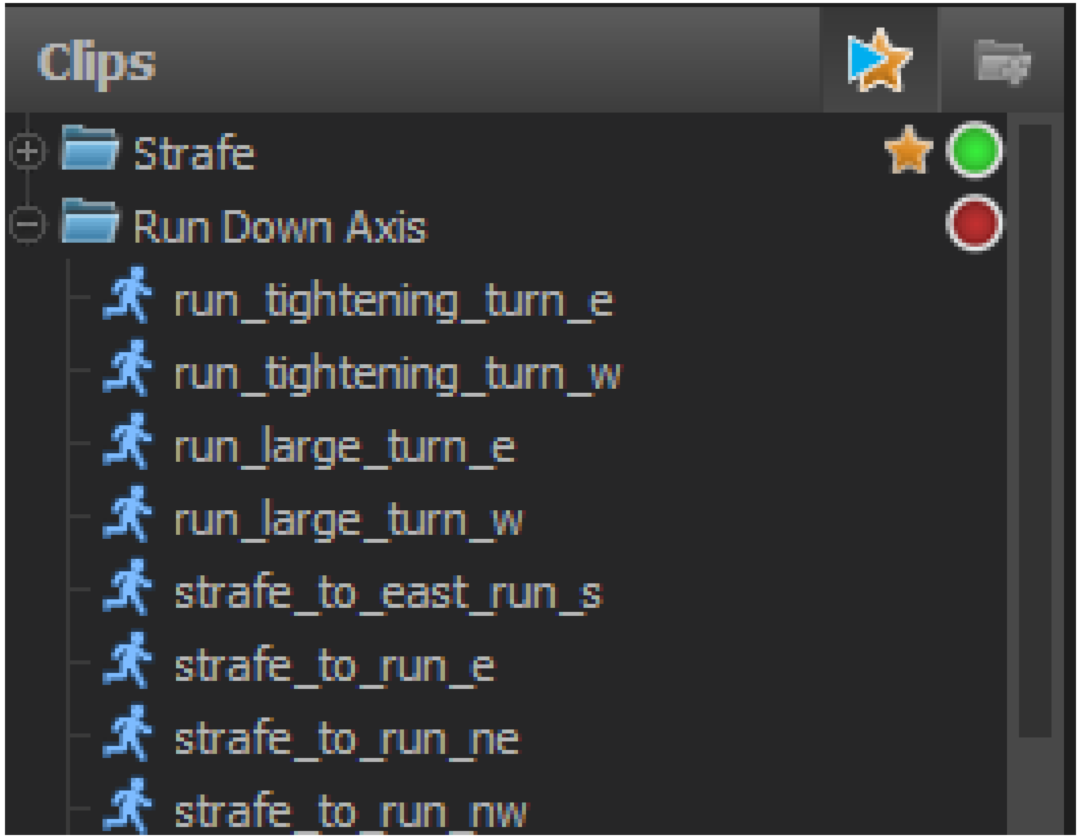
“Make Good Choices”

"Don't Make Bad Choices"

Avoiding Bad Choices

AVOIDING BAD CHOICES

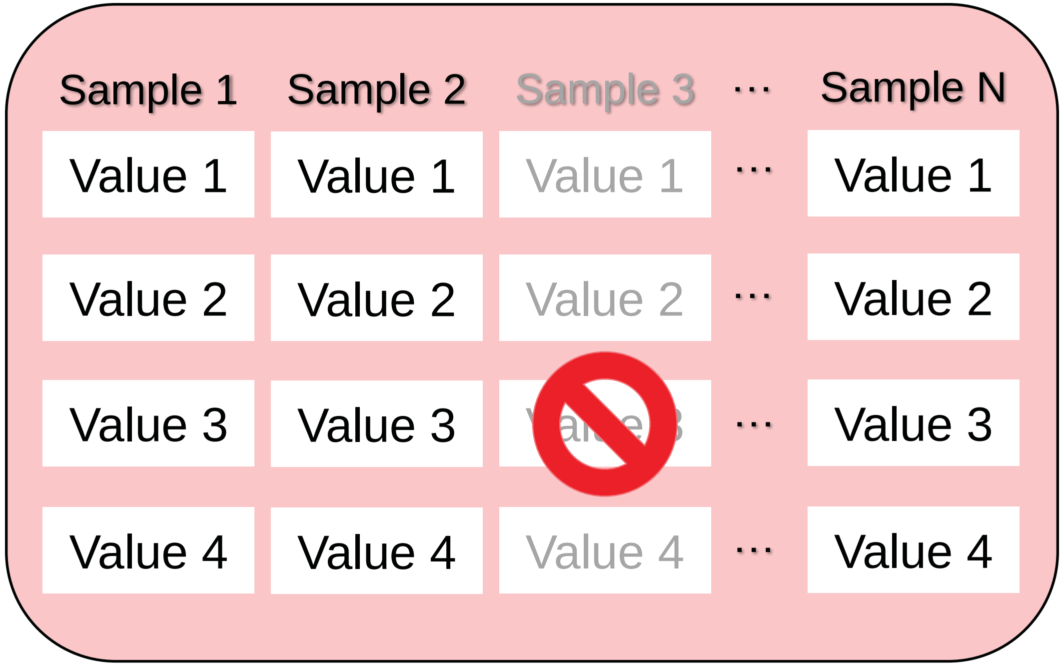
SIGGRAPH 2021



CLIP GROUPS

- Put anims in groups
- Only allowed to search in active groups
- Game logic determines active groups

AVOIDING BAD CHOICES



FILTERS

- Method to exclude certain samples
- Define valid metric range (Min/Max)
- Skip samples that fail check

LEAST BAD CHOICE

Weighted Score =

$$\sqrt{\sum((GoalValue_n - TestValue_n)^2 * \boldsymbol{MetricWeight_n})}$$

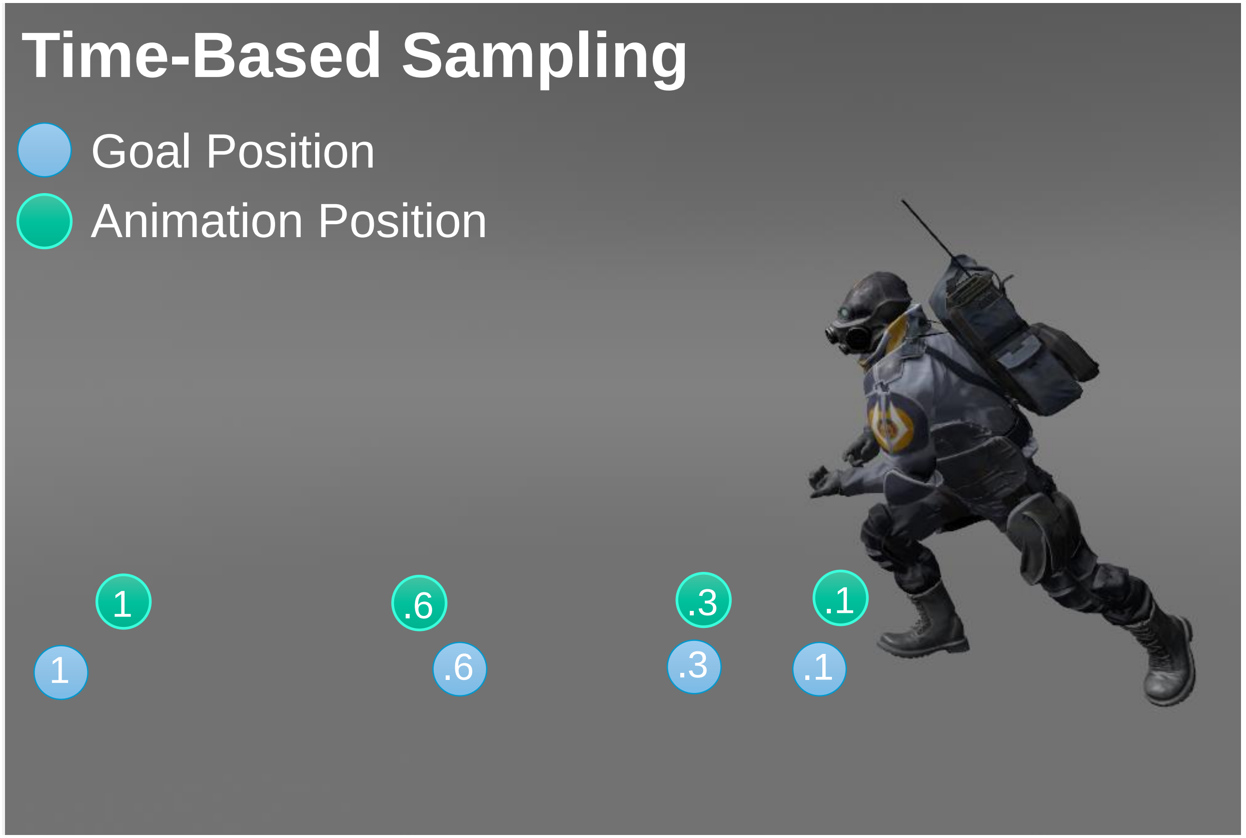
WEIGHTS

- Eventually need to pick between imperfect options
- Add weight scales score for metric
- Larger weights => More important
- Zero weight => Metric ignored
- Best results when all weights close to 1

Making Good Choices

IMPROVING ACCURACY

SIGGRAPH 2021

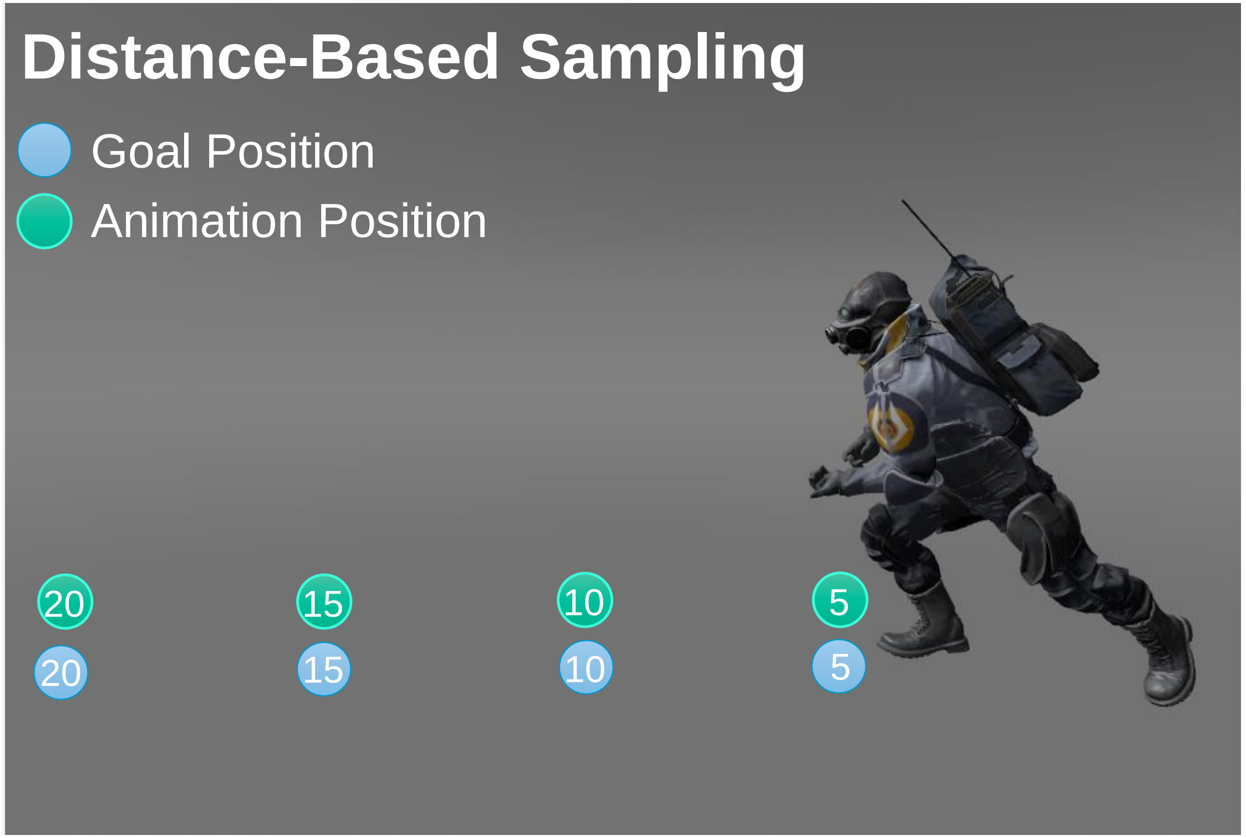


PICKING PATH SAMPLES

- Time-based path sampling
  - Use physics to estimate future position
  - Guess at Acceleration/Jerk
  - Doesn’t match all anims

IMPROVING ACCURACY

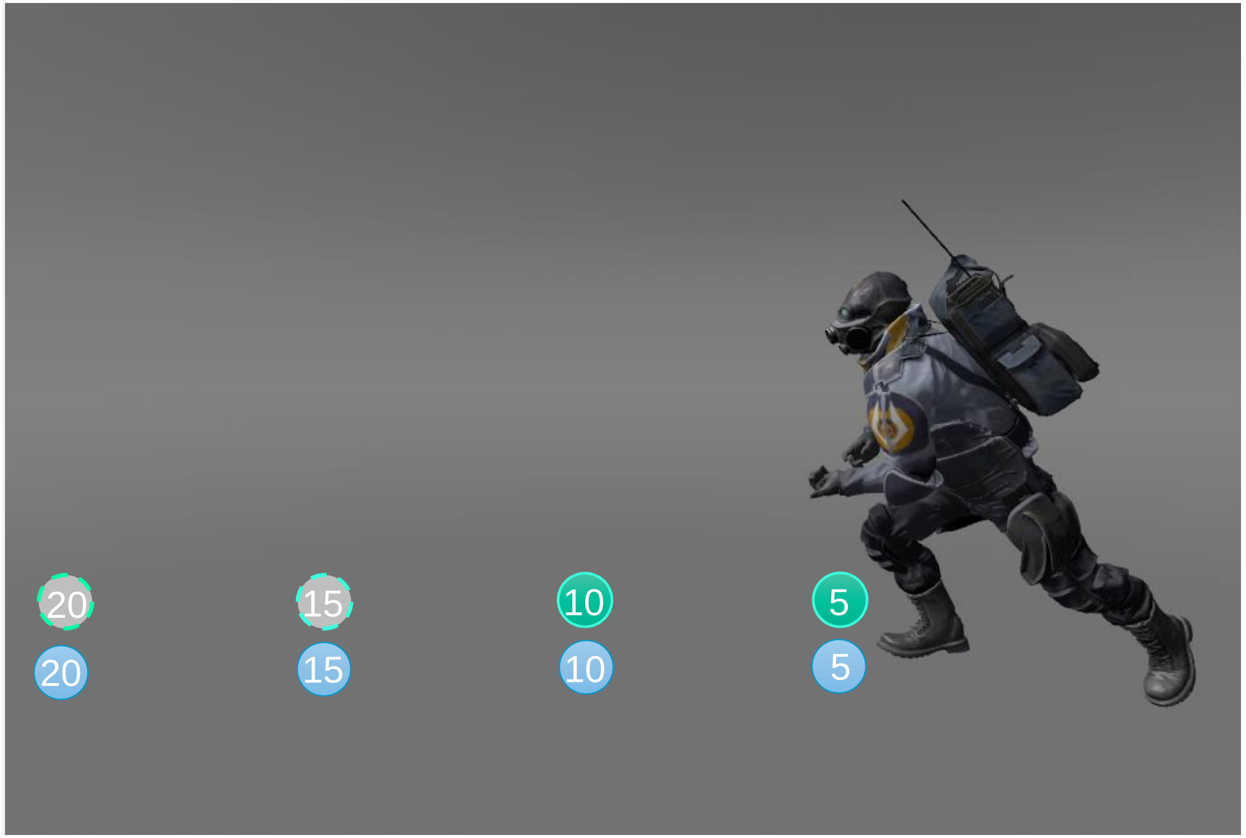
SIGGRAPH 2021



PICKING PATH SAMPLES

- Distance-based path sampling
  - Find Position after moving X distance
  - Consistent for all animations
  - Easy to find along path

IMPROVING ACCURACY



EXTRAPOLATING SAMPLES

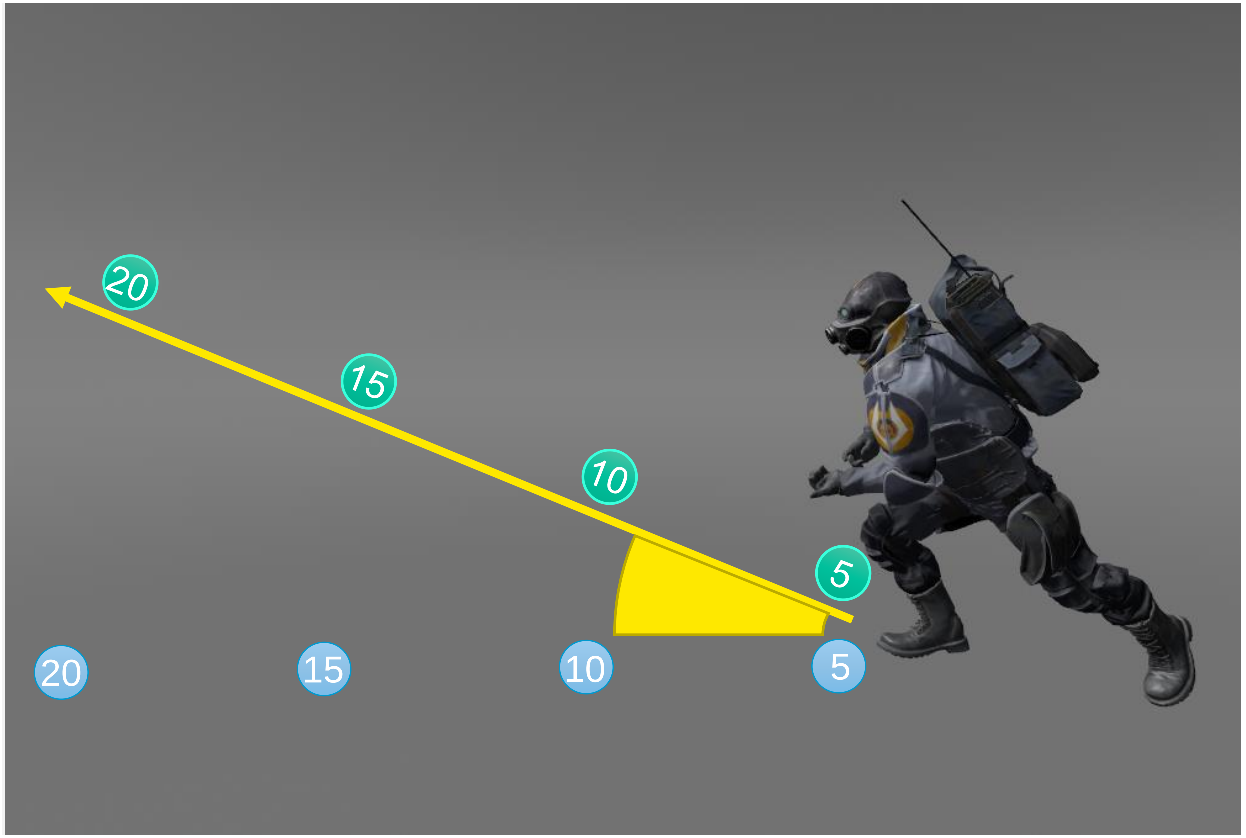
- What if Animation doesn't move far enough?
- Estimate position based on final speed

IMPROVING ACCURACY

SLOPES / UNEVEN TERRAIN

- Goal samples at different heights/distances
- Anim samples flat
- => Flatten Goal samples

IMPROVING ACCURACY

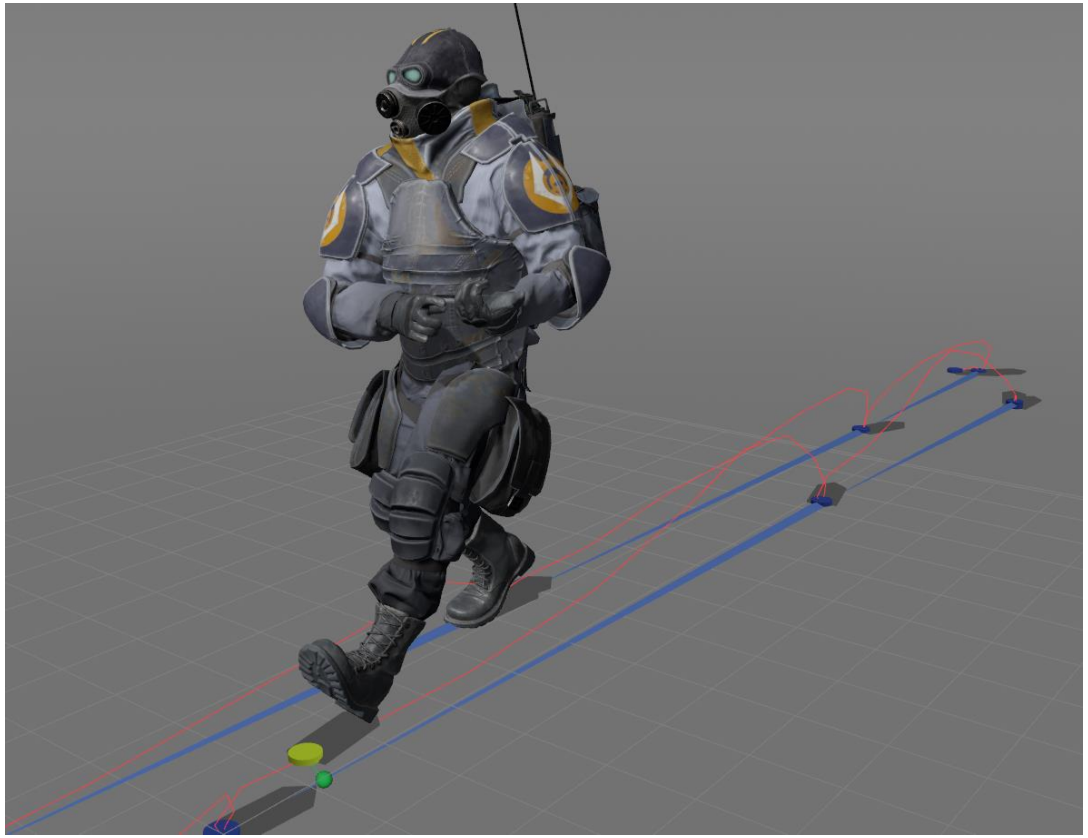


CORRECTIONS

- Match is never perfect
- Apply Correction for Position and Rotation
- Must include corrections in Score for Current Choice

Leveraging Foot Motion

LEVERAGING FOOT MOTION



FOOT POSITION METRIC

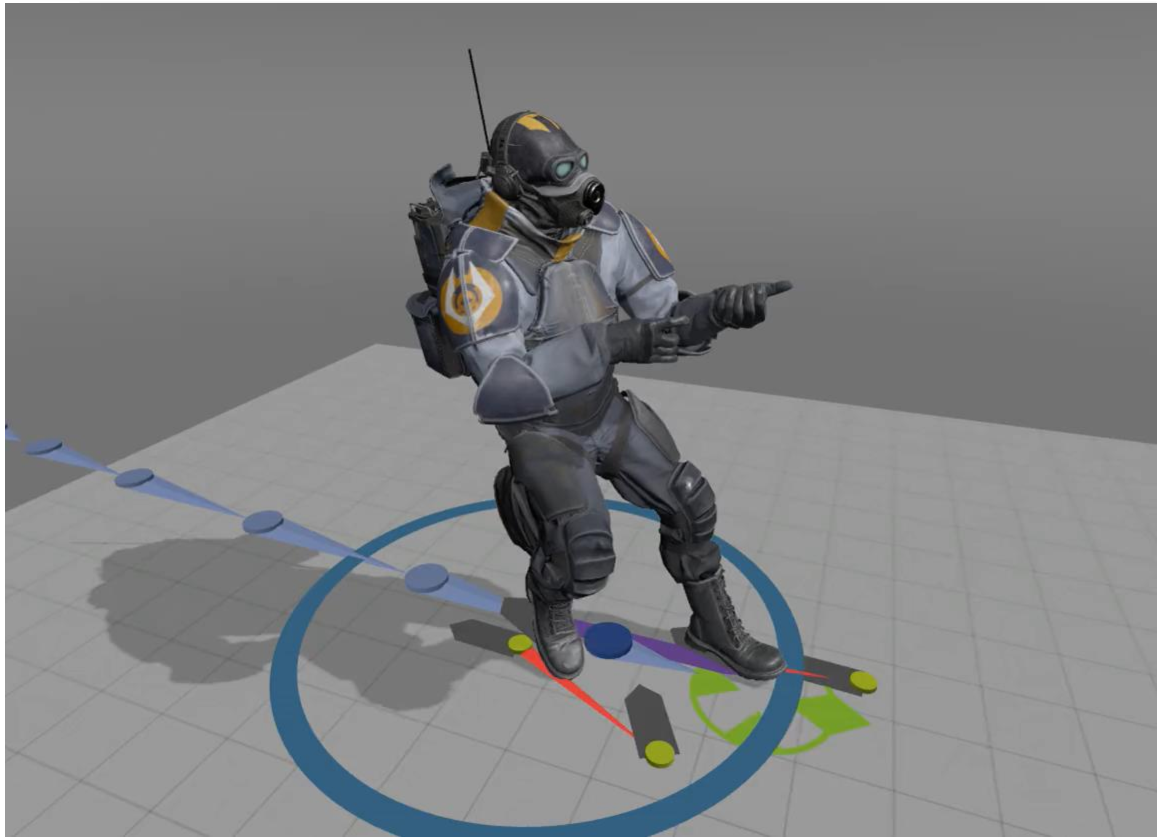
- X,Y,Z position of FootBase
- Character-relative
- More accurate than foot bone position metric

LEVERAGING FOOT MOTION

FOOT STEP PROGRESSION METRIC

- Metric for foot step progression
- Matches how far through a step samples are
- Convert Progression (0->1) to a 2D direction vector
  - So start and end match

LEVERAGING FOOT MOTION



STEPS REMAINING FILTER

- On stopping anims, Root still moves after feet stop
- Don’t want to pick these for short steps
- Add “Steps Remaining” metric, use as filter

LEVERAGING FOOT MOTION

DISTANCE REMAINING FILTER

- Need to stop exactly at end of path
- MM was picking clips that stopped just short
- Add “Distance Remaining” metric
  - Filter out samples that don’t reach goal
- Scale root motion to stop at goal

LEVERAGING FOOT MOTION

SEARCH FREQUENCY

- Can perform new search at fixed intervals
- Looked slightly better if only searching when a foot is planted

METRICS

SIGGRAPH 2021

Metrics

Weight	Category	Type
1.00	*Pose Metric*	Current Velocity
1.00	*Pose Metric*	Bone Velocity (ankle_L)
1.00	*Pose Metric*	Bone Velocity (ankle_R)
0.00	*Filter Metric*	Steps Remaining
0.00	*Goal Metric*	Time Remaining
1.00	*Pose Metric*	Foot Cycle
1.00	*Goal Metric*	Distance Remaining
1.75	*Goal Metric*	Path
1.00	*Goal Metric*	Future Facing
1.75	*Goal Metric*	Future Velocity
1.00	*Pose Metric*	Foot Position

COMBINE SOLDIER

- Metrics used by the combine soldier

ANIMATIONS

SIGGRAPH 2021

117 Animations used for motion matching

- **Strafe Mode (77)**
  - Idle*
  - Run Loop* x 8 directions
  - Short Hops*, 8 distances x 8 directions
  - Run Fwd then Bwd, Bwd then Fwd
  - Run Left then Right, Right then Left
  - Square Strafe Clockwise, Counter-Clockwise

- **Face Path Mode (70)**
  - Running turns: Left/Right x Large/Small Radius
  - Strafe then Face Path x 8 directions
  - Face Path then Strafe x 8 directions
  - Stand to Run x 8 directions
  - Run to Stand x 8 directions
  - Plant turns: 90/180 Left/Right
  - All Strafe Mode Animations

* Created before MM implementation and re-used

SIGGRAPH 2021

QUESTIONS?

LIVE Q & A ON DATE TBD