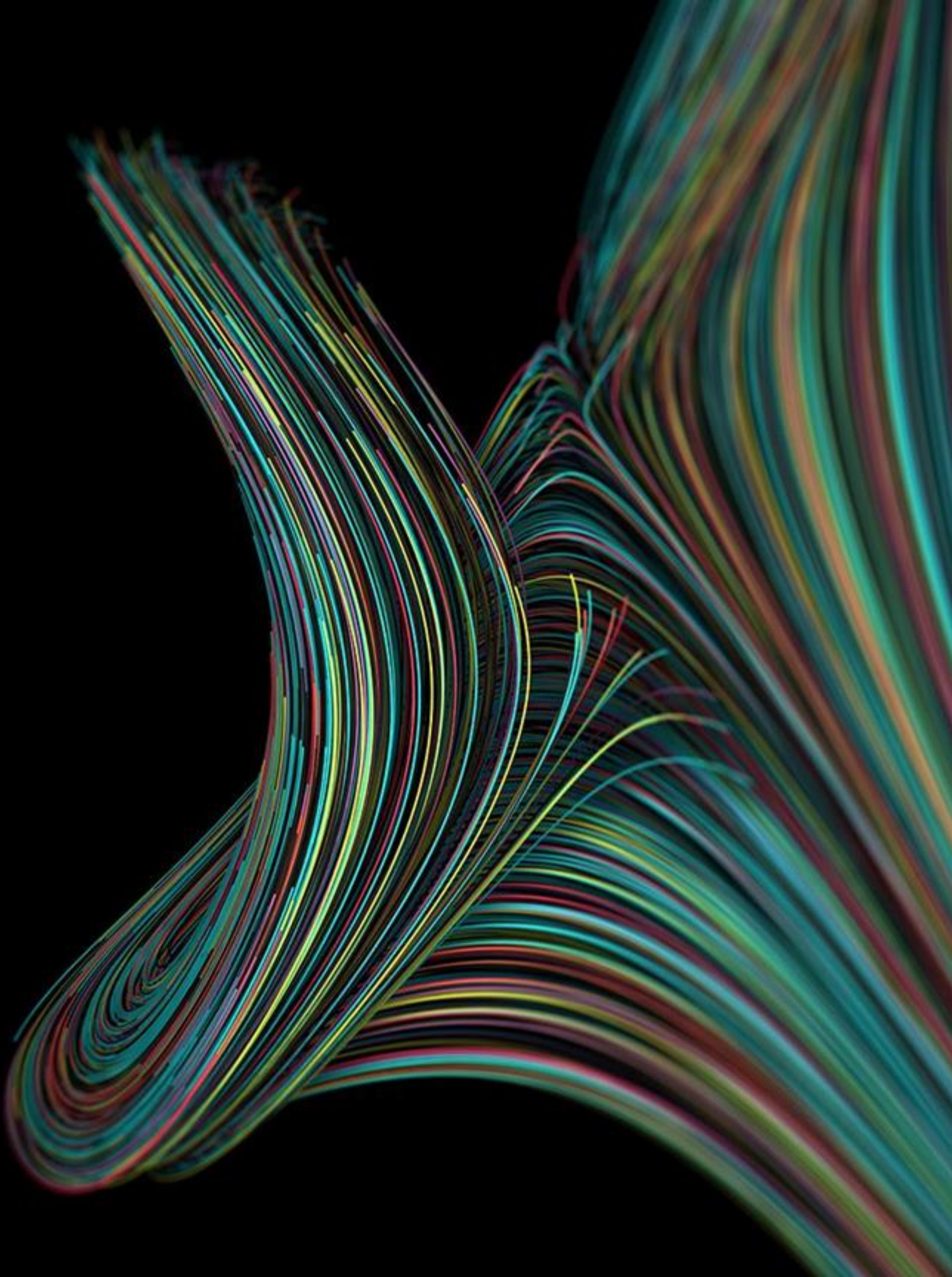
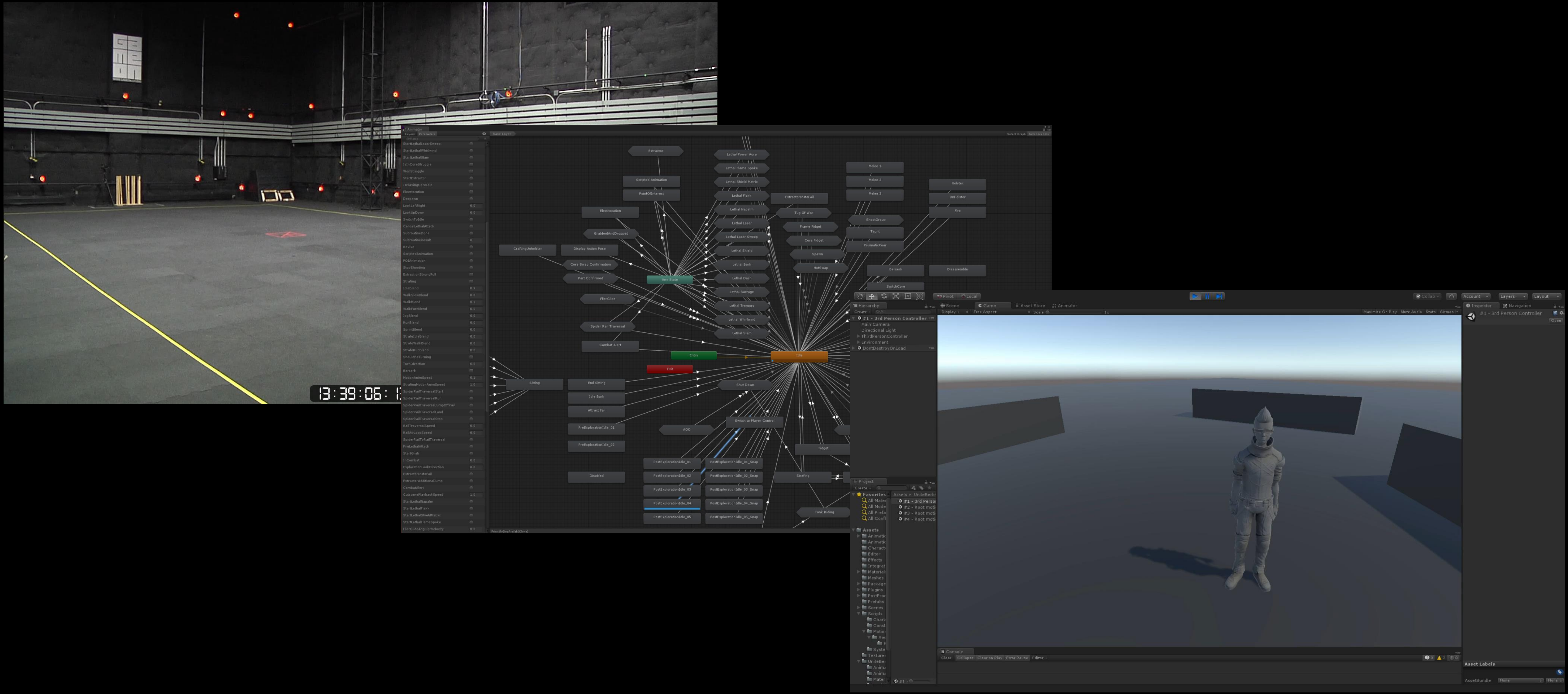
Machine Learning for Motion Synthesis and Character Control in Games

Michael Buttner, Unity Labs
Principal Research Engineer
michaelbu@unity3d.com

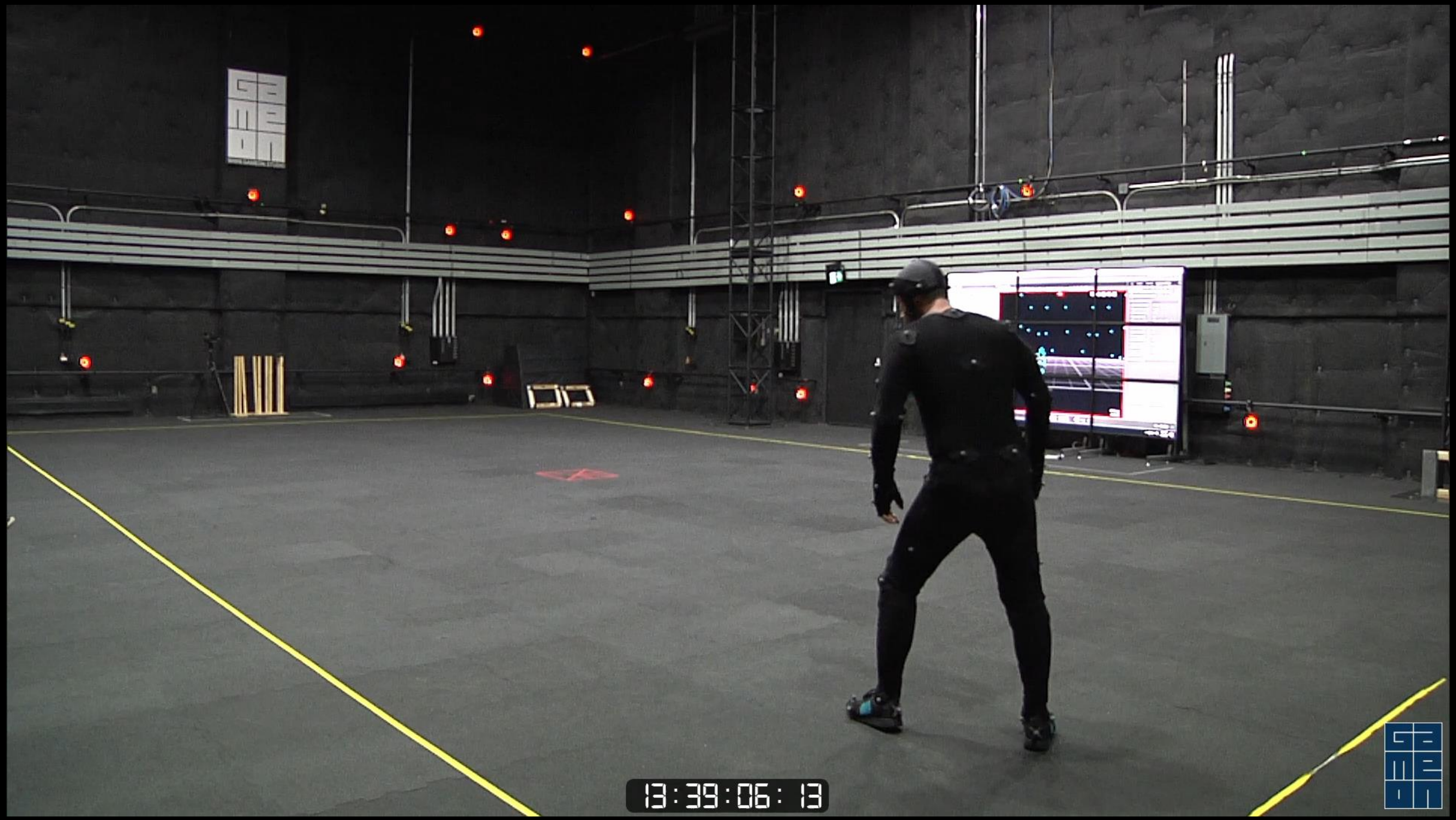


Generative Art – Made with Unity

unity



unity



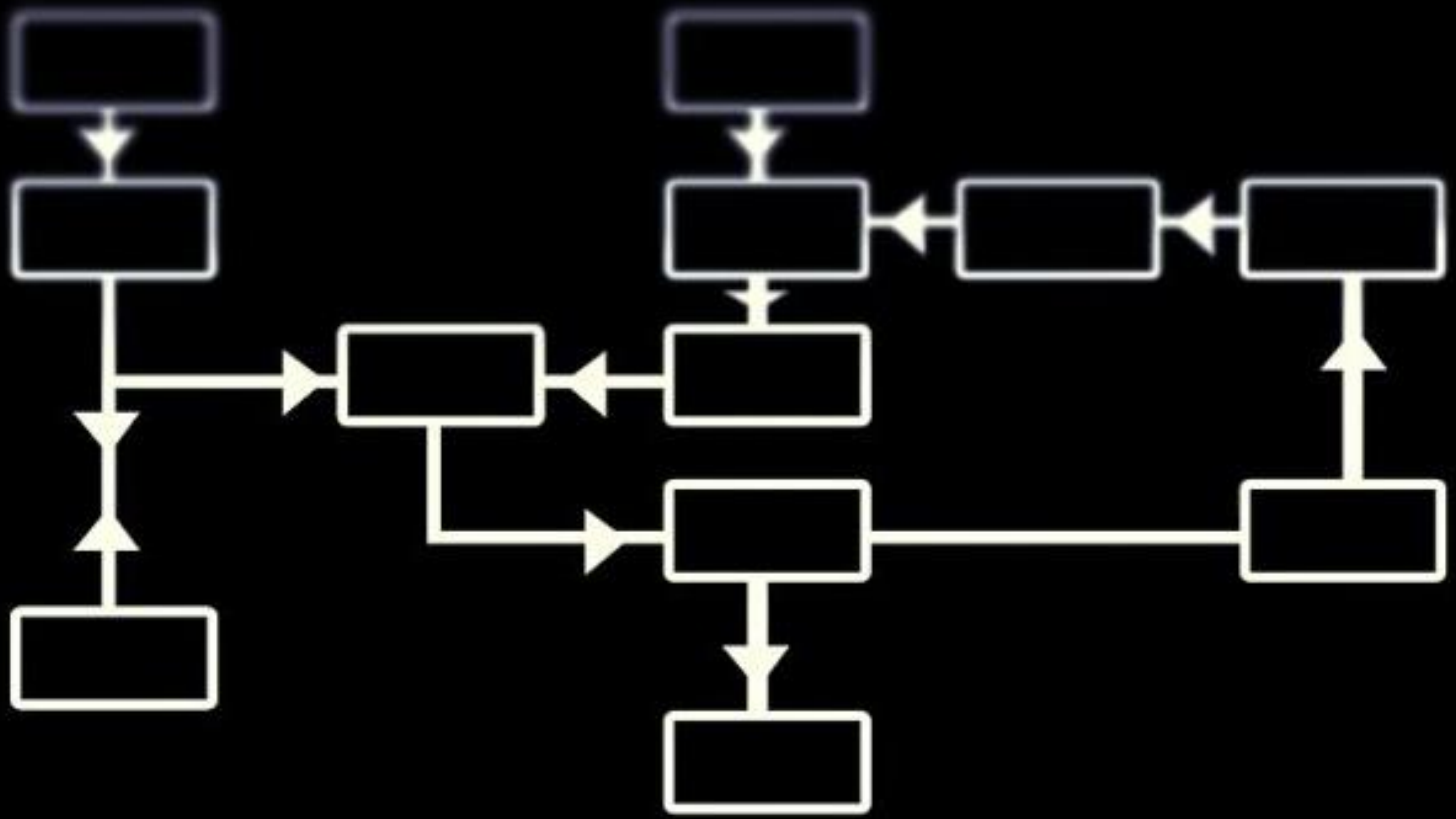
unity


unity
Kinematica
Experimental
unity

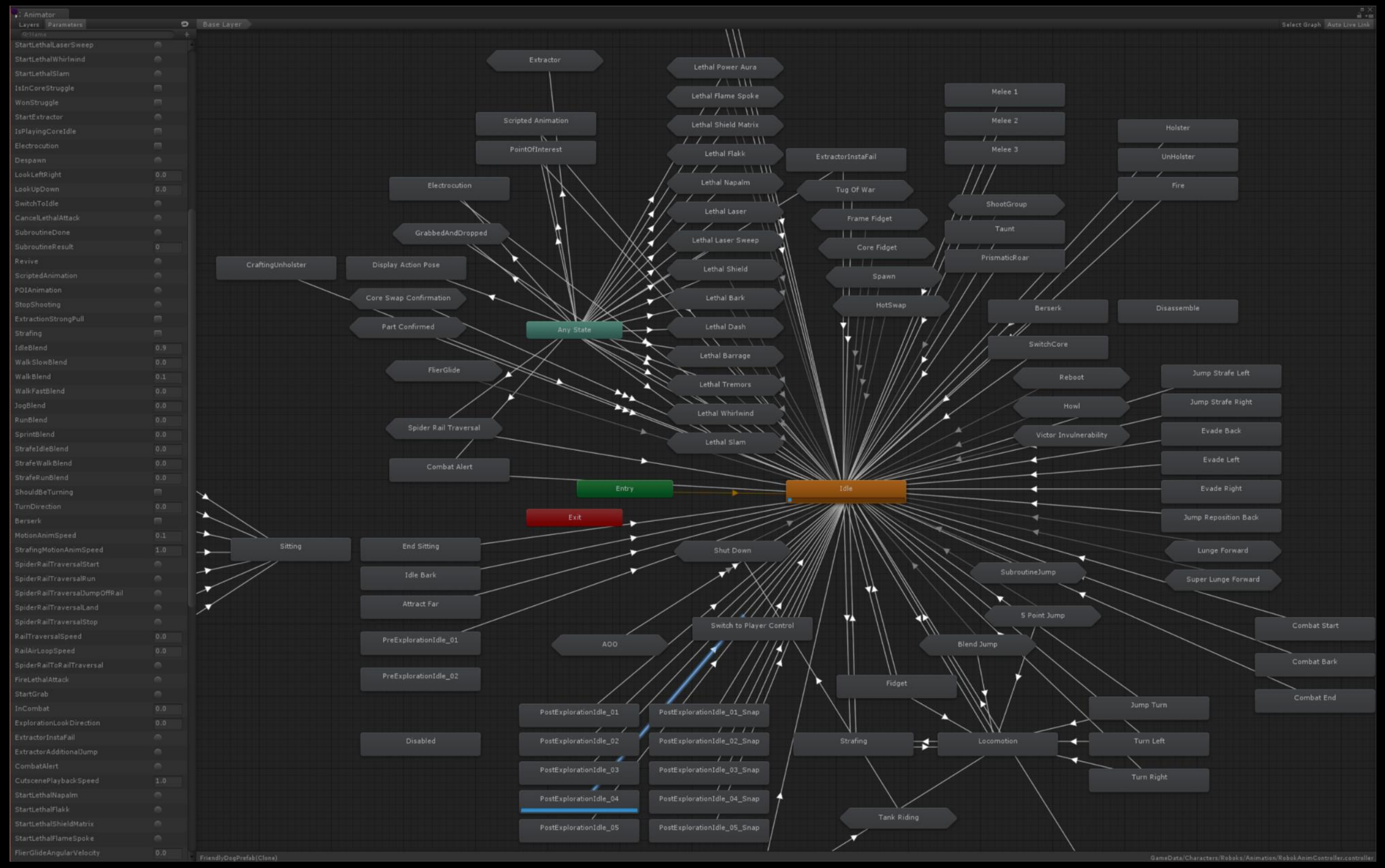

PlayStation
PAL
MISSION: IMPOSSIBLE
ASSASSIN'S
CREED
UNITY
FARCRY
PRIMAL
FOR
HONOR
DEMONWARE
AMIGA

Game State | Animation Graphs

Game Logic



Animation Logic




Pivot
Local
Collab
Account
Layers
Empty
Game
Display 1
Free Aspect
Scale
1x
Maximize On Play
Mute Audio
Stats
Gizmos


Scene
Game
Animator
Pivot
Local
Collab
Account
Layers
Mickey
Layers
Parameters
Base Layer
IK
Base Layer
Auto Live Link
Airborne
Any State
Entry
Grounded
Exit
Crouching
Siggraph2018/Animator/ThirdPersonAnimatorController.controller

ThirdPersonCharacter.cs

D:\Workspace\ECSAnimationPlayer\Samples\Assets\Siggraph2018\Scripts\ThirdPersonCharacter.cs

```
using UnityEngine;

namespace ThirdPerson
{
    [RequireComponent(typeof(Rigidbody))]
    [RequireComponent(typeof(CapsuleCollider))]
    [RequireComponent(typeof(Animator))]
    public class ThirdPersonCharacter : MonoBehaviour
    {
        void Start()
        {
            animator = GetComponent<Animator>();
            rigidbody = GetComponent<Rigidbody>();

            rigidbody.constraints = RigidbodyConstraints.FreezeRotationX | RigidbodyConstraints.FreezeRotationY | Rigidbody
            originalGroundCheckDistance = groundCheckDistance;
        }

        private void FixedUpdate()
        {
            //
            // Read current gamepad state
            //

            float h = Input.GetAxis("Left Analog Horizontal");
            float v = -Input.GetAxis("Left Analog Vertical");
```

Animation Graphs

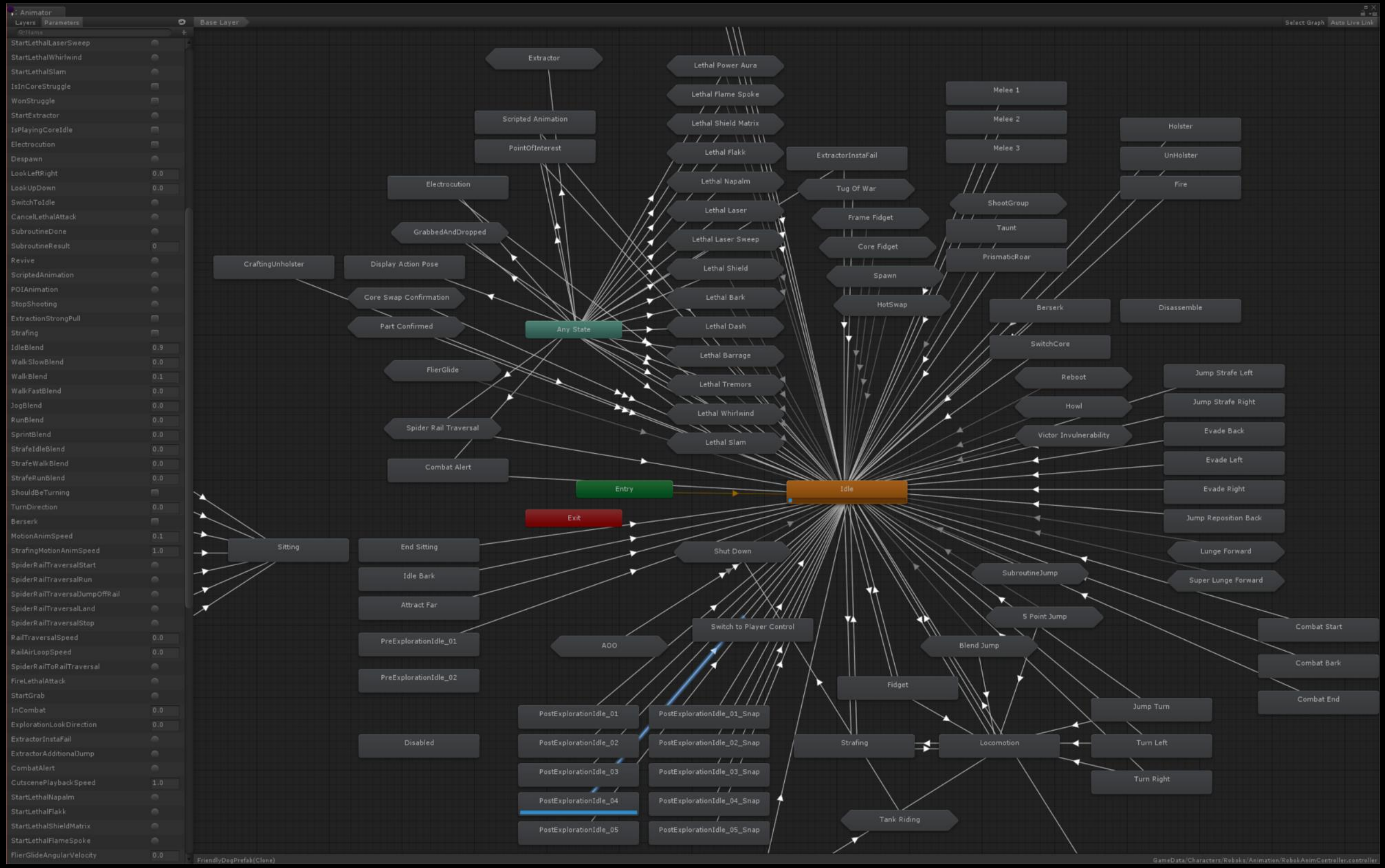
Provide structure for animation clips

Allow animations to be “addressable”

Allow related animation to be grouped

Transitions have to be explicitely spelled out

Combinatorical explosion




CHINESE NEW YEAR
0
Warp 36,375
Averages FI 89,502 WORLD 82,796

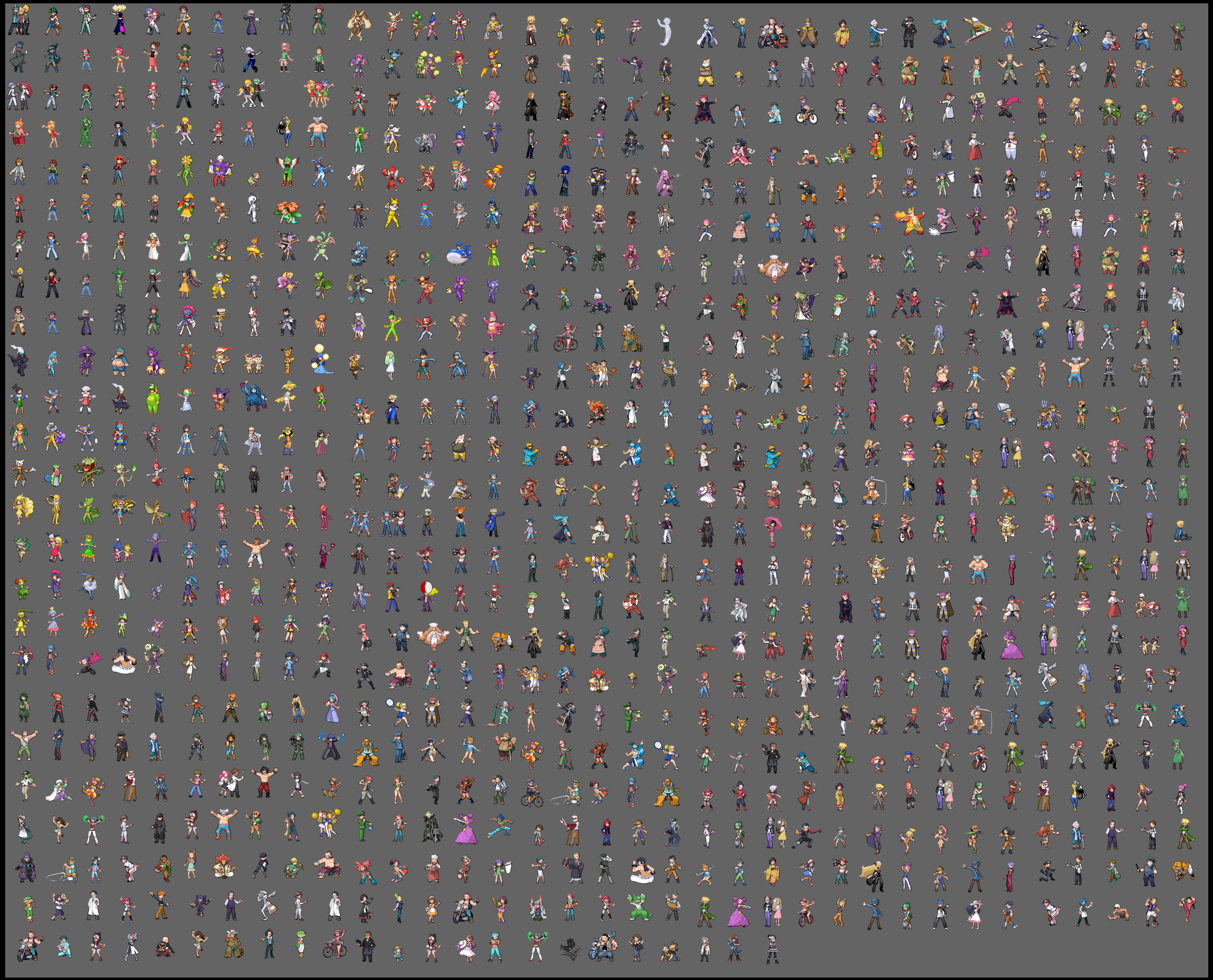
Unstructured Animations

If we would combine all animation clips into a single unstructed library…

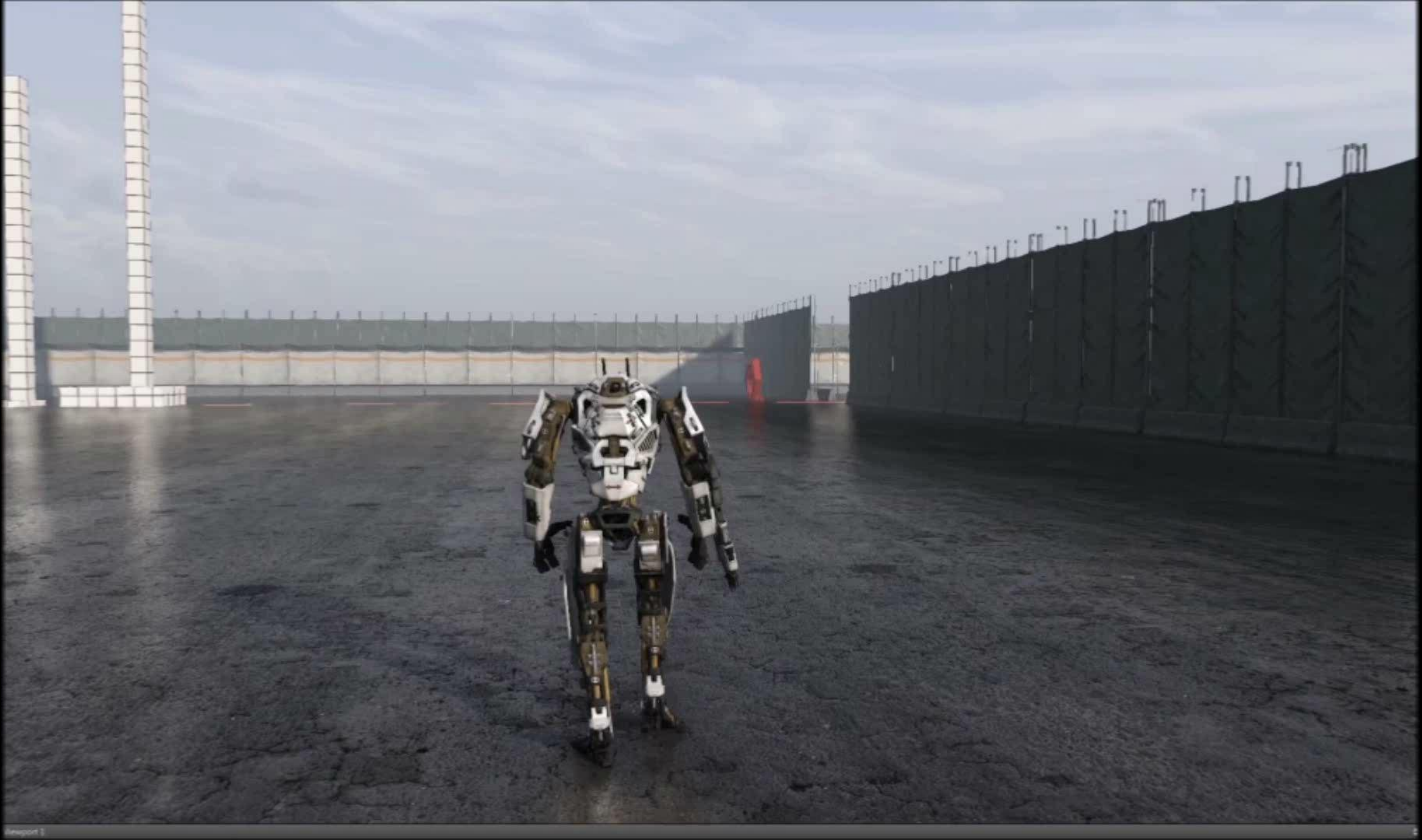
How can we infer information from the gameplay state directly?

Can we extract relevant information from the physical properties of the animations themselves?

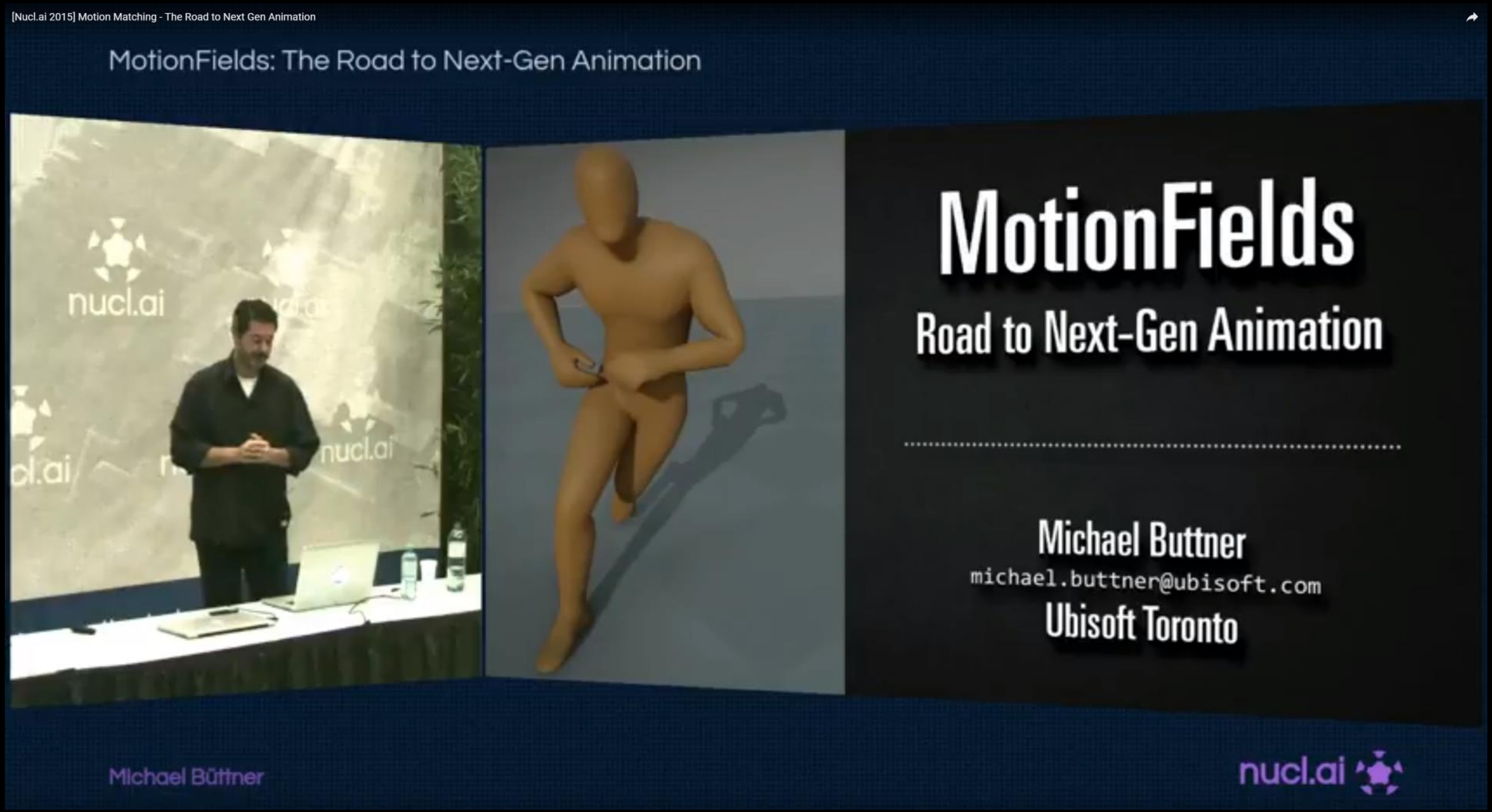
How can we achieve the same level of versatility as animation graphs?



unity



Motion Matching



Motion Matching [Michael Buttner, 2015]

unity

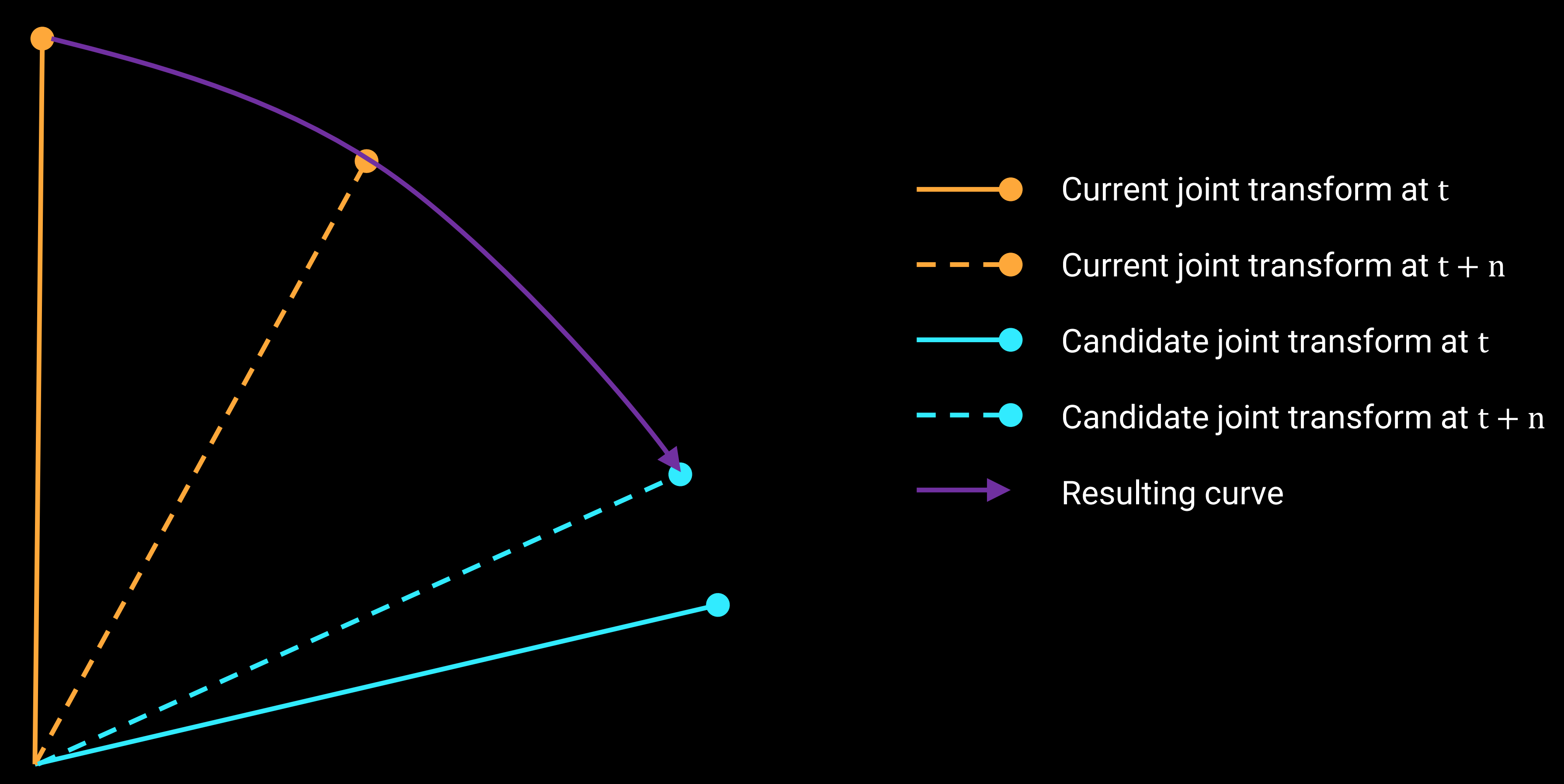

GA
ME
ON
13:39:06:13
GA
ME
ON

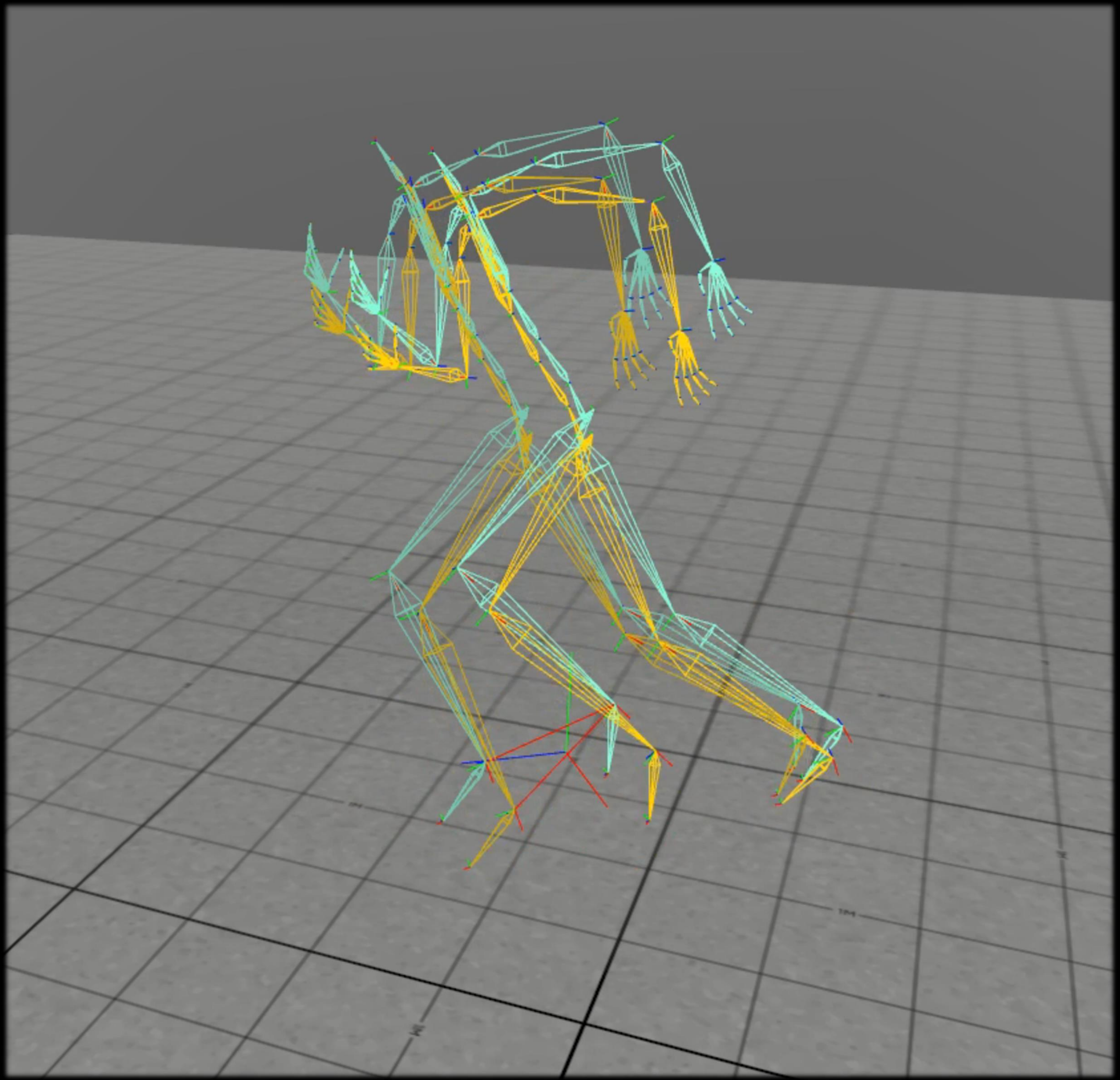

Pivot
Local
Collab
Account
Layers
Default
Scene
Game
Asset Store
Shaded
2D
Gizmos
All
Persp
Inspector
GameObject
Static
Tag
Untagged
Layer
Default
Transform
Position
X 0
Y 0
Z 0
Rotation
X 0
Y 0
Z 0
Scale
X 1
Y 1
Z 1
Fragment (Script)
Script
Fragment
Resource Name
Binary.bin
Display Root Trajectory
Display Joint Trajectories
Display Relative
Display Fragment
Display Matching Fragmen
Policy Index
0
Timeline
85.00
90.00
95.00
100.00
Offset
0
Add Component
Auto Generate Lighting On

Geometric Pose Comparison

- Minimum squared distance between joint positions
- Using only “relevant” joints

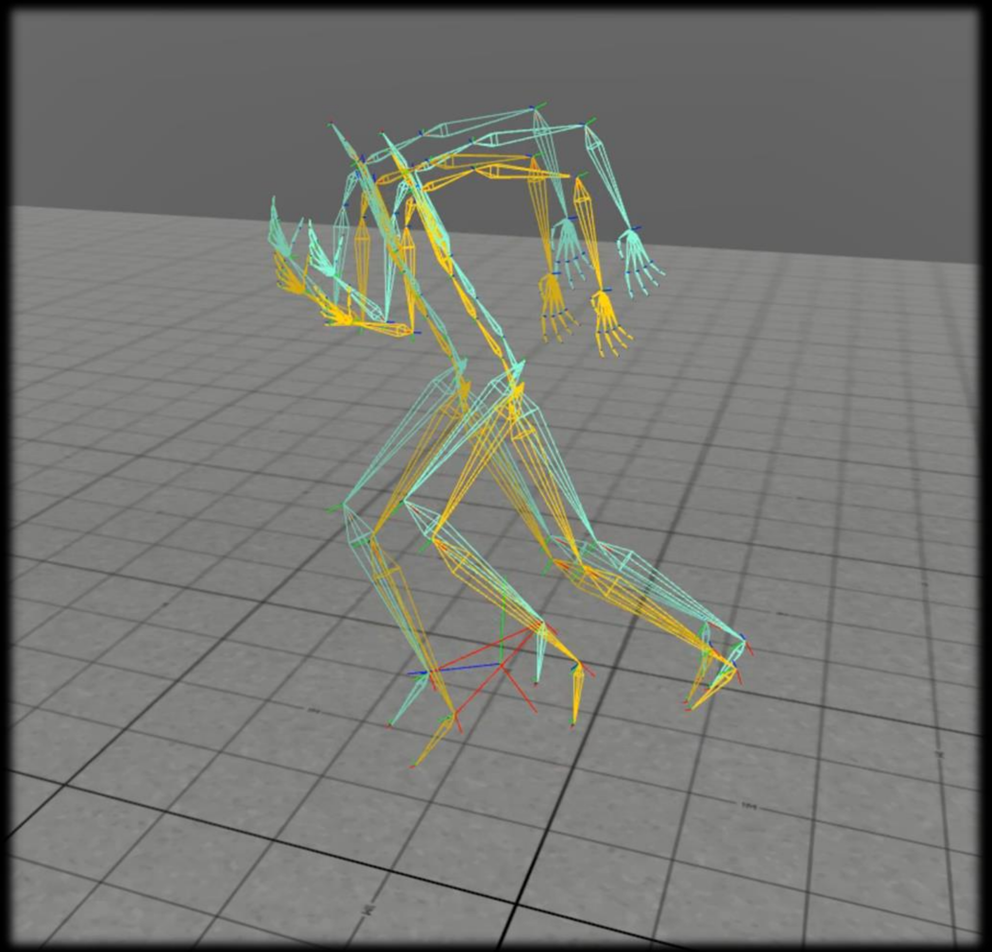
- Minimum local joint rotation delta is flawed





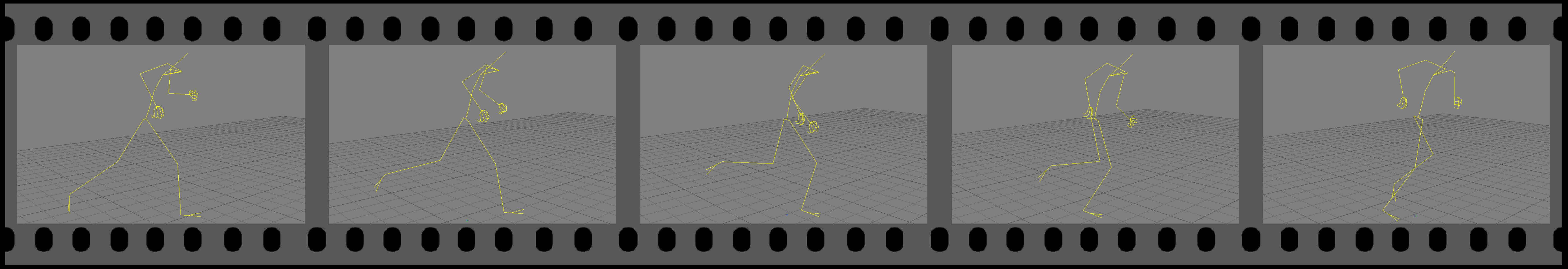
unity

Motion Matching Algorithm



Query -> $F(y)$

$\arg\min x\, s(F(x), F(y))$

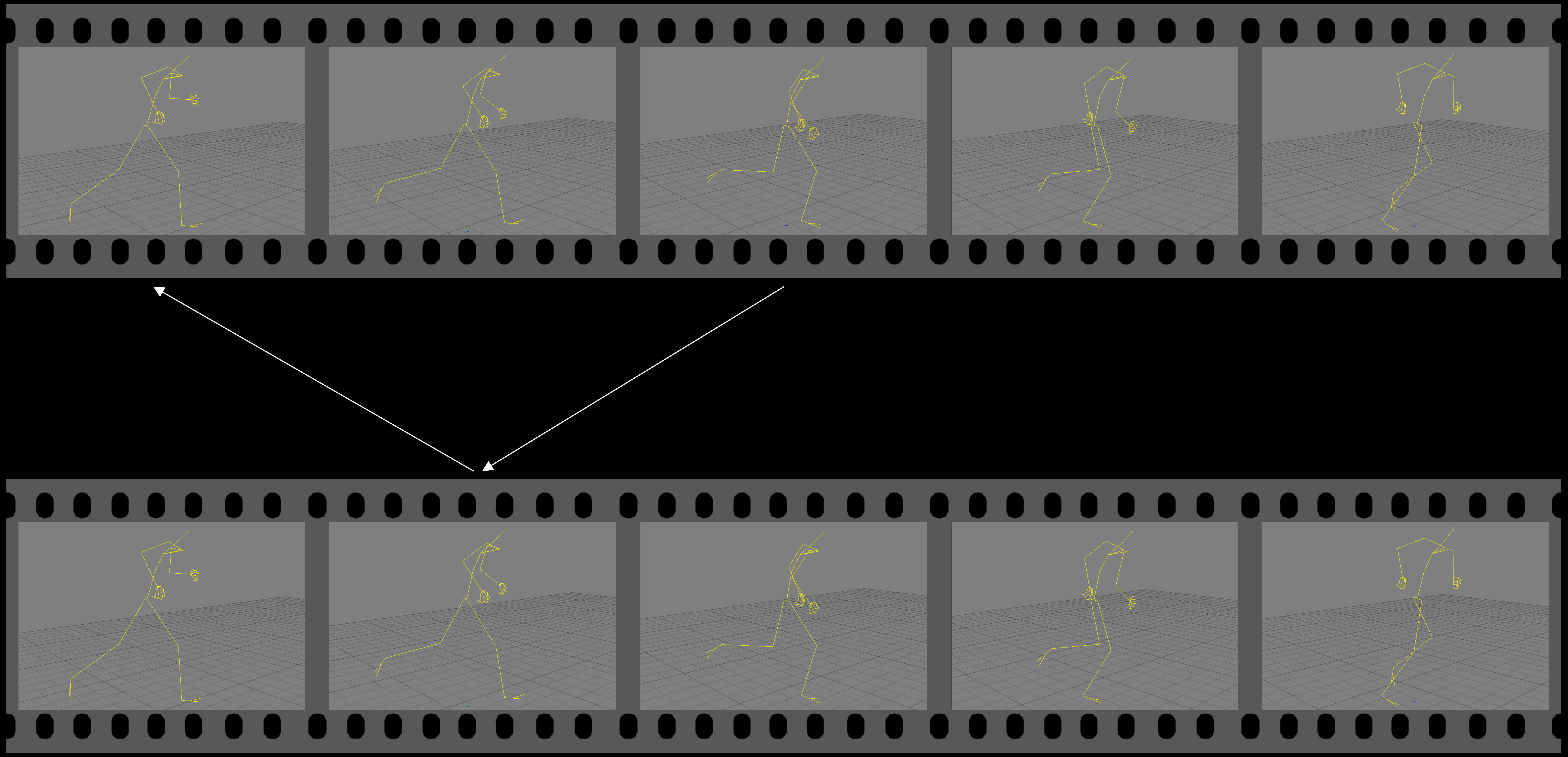


Motion Database

unity


Pivot
Local
Collab
Account
Layers
Layout
Scene
Game
Asset Store
Display 1
Free Aspect
Scale
1x
Maximize On Play
Mute Audio
VSync
Stats
Gizmos
Inspector
Navigation
Linear Error Multiplie 1
Linear Velocity Error 1.5
Angular Error Multipli 1.4
Time Between Jumps 0
Animator
Motion Synthesizer (Script)
Script MotionSynthesizer
Resource Name Animations/Binary.bin
Root Root (Transform)
Blend Duration 0.3
Record & Rewind
Enable Recording
Recorder State Record
Debug setting
Display Trajectory
Display Current
Current Time Offset 0
Display Candidate
Candidate Time Offs 0
Display Pose Error
Display Policy Error
Debug Policy
Candidate Index 0
Actor Controller (Script)
Is Enabled
Use Transform
Actor Properties
Width (as radius) 0.35
Mass 2
Gravity
Is Gravity Enabled
Dampen Factor 1
Skin Width 0.01
Grounding
Use Grounding Layer
Grounding Layers Default
Grounding Radius 0.1
Grounding Ray 0.5 0.75
Force Grounding 0.3
Collisions
Is Collision Enabled
Stop Rotations
Allow push back
Collision Layers Mixed...
Steps
Step Height 0.3
Step Up Speed 0.75
Step Down Speed 0.75
Slopes
Max Slope 45
Is Sliding Enabled
Limits
Auto Generate Lighting Off

Back-in-time Problem



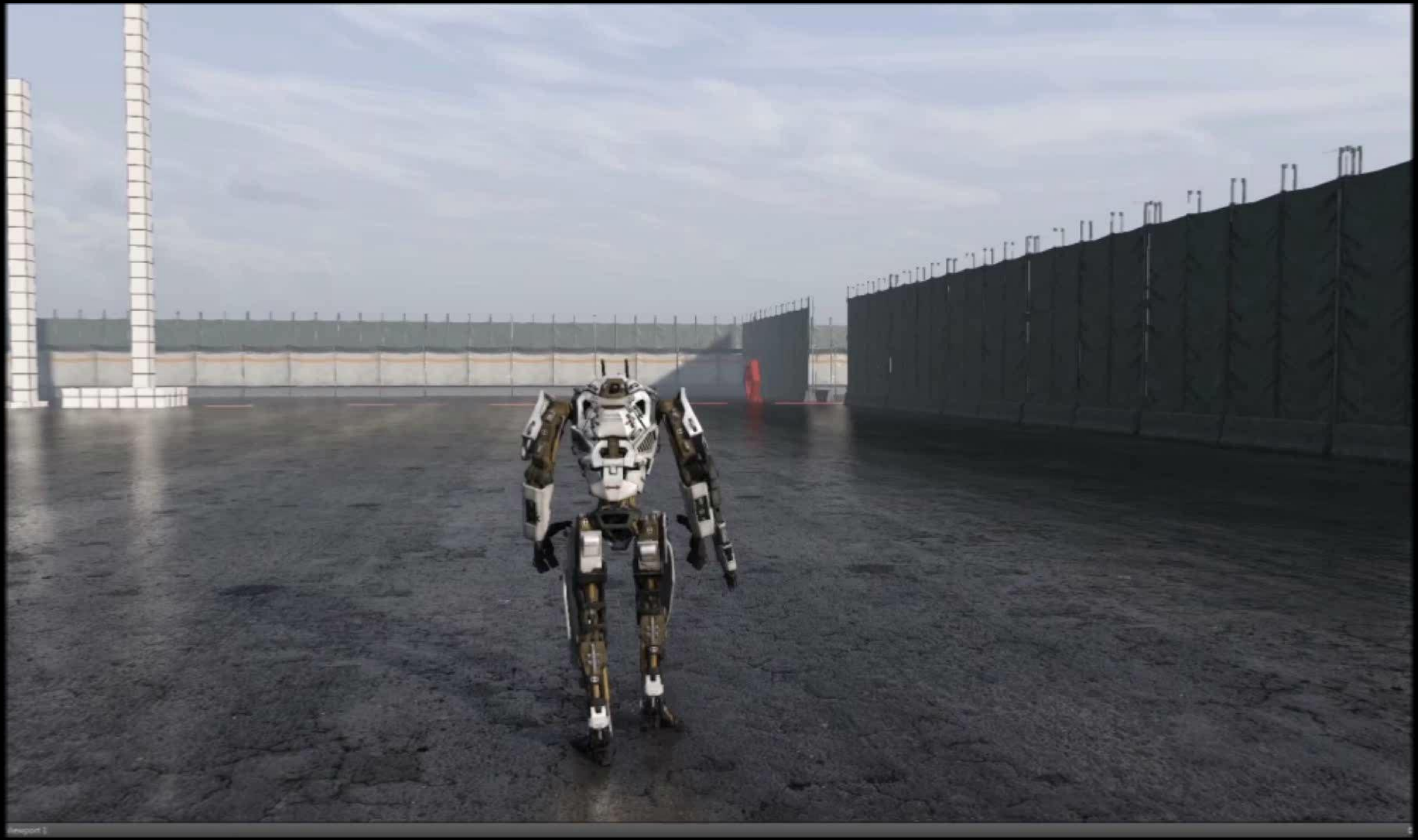
Motion Matching

Pros

- Preserves high quality result
- Does not rely on phases
- Relatively easy to implement

Cons

- Prediction must match data
- Construction of cost functions
- Requires a lot of tweaking
- Doesn’t scale well
- Duplicate data problem
- “Back-in-time” problem



unity

Motion Synthesis Research

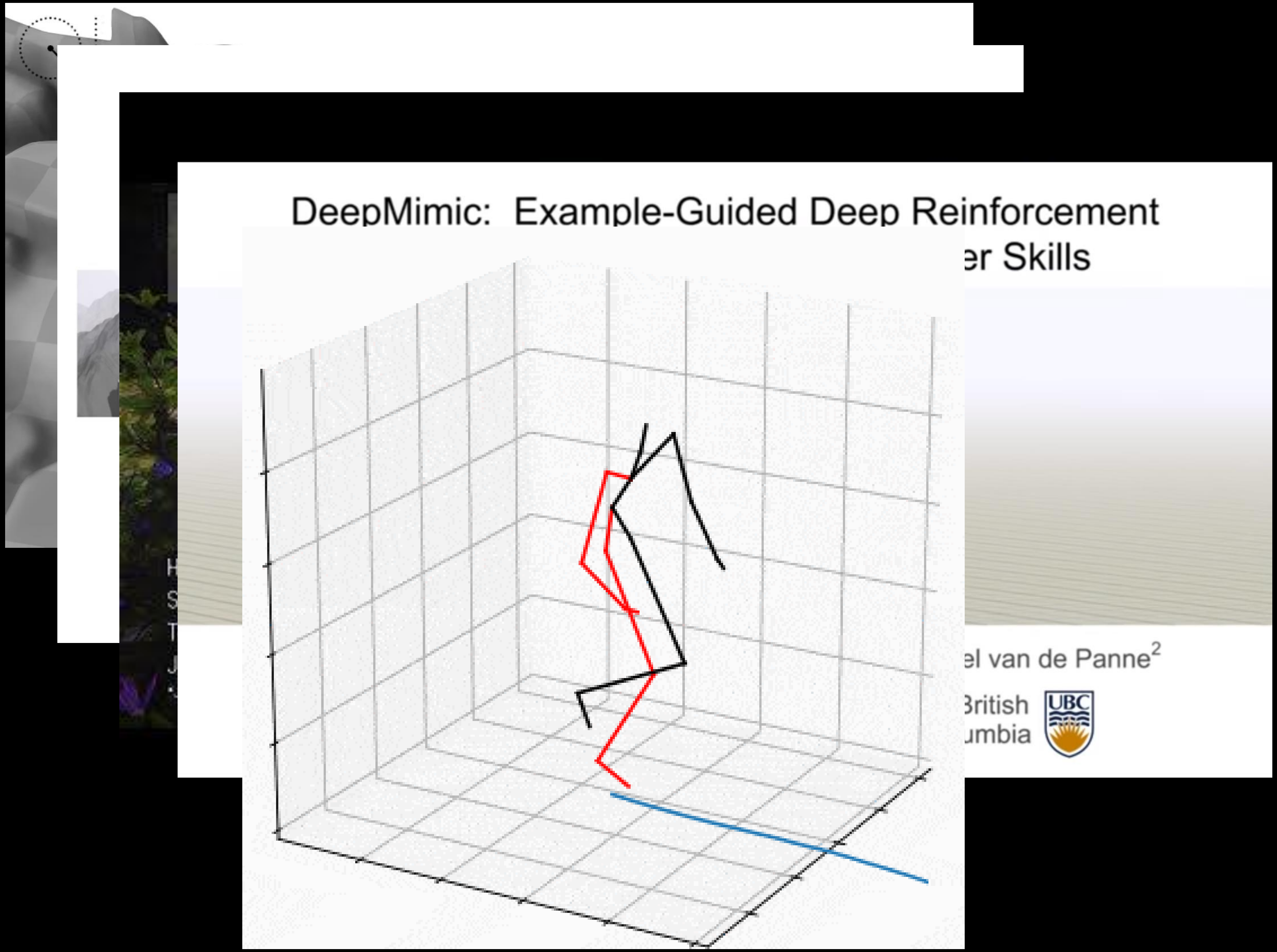
Phase Function NN [Daniel Holden, 2017]

Deep Loco [Peng, 2017]

Mode Adaptive NN [Sebastian Starke, 2018]

Deep Mimic [Peng, 2018]

QuaterNet [Pavvlo, 2018]



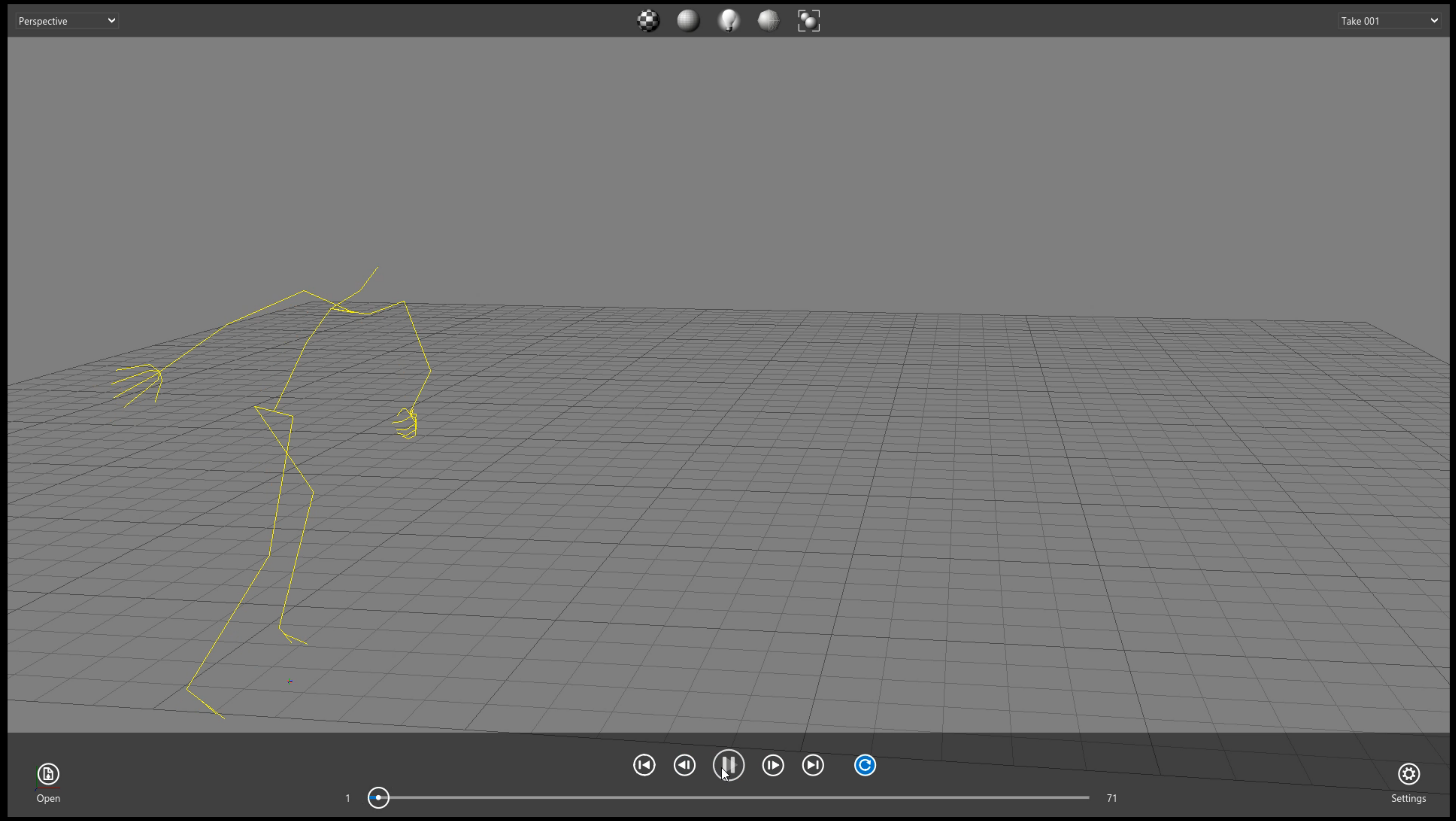
unity

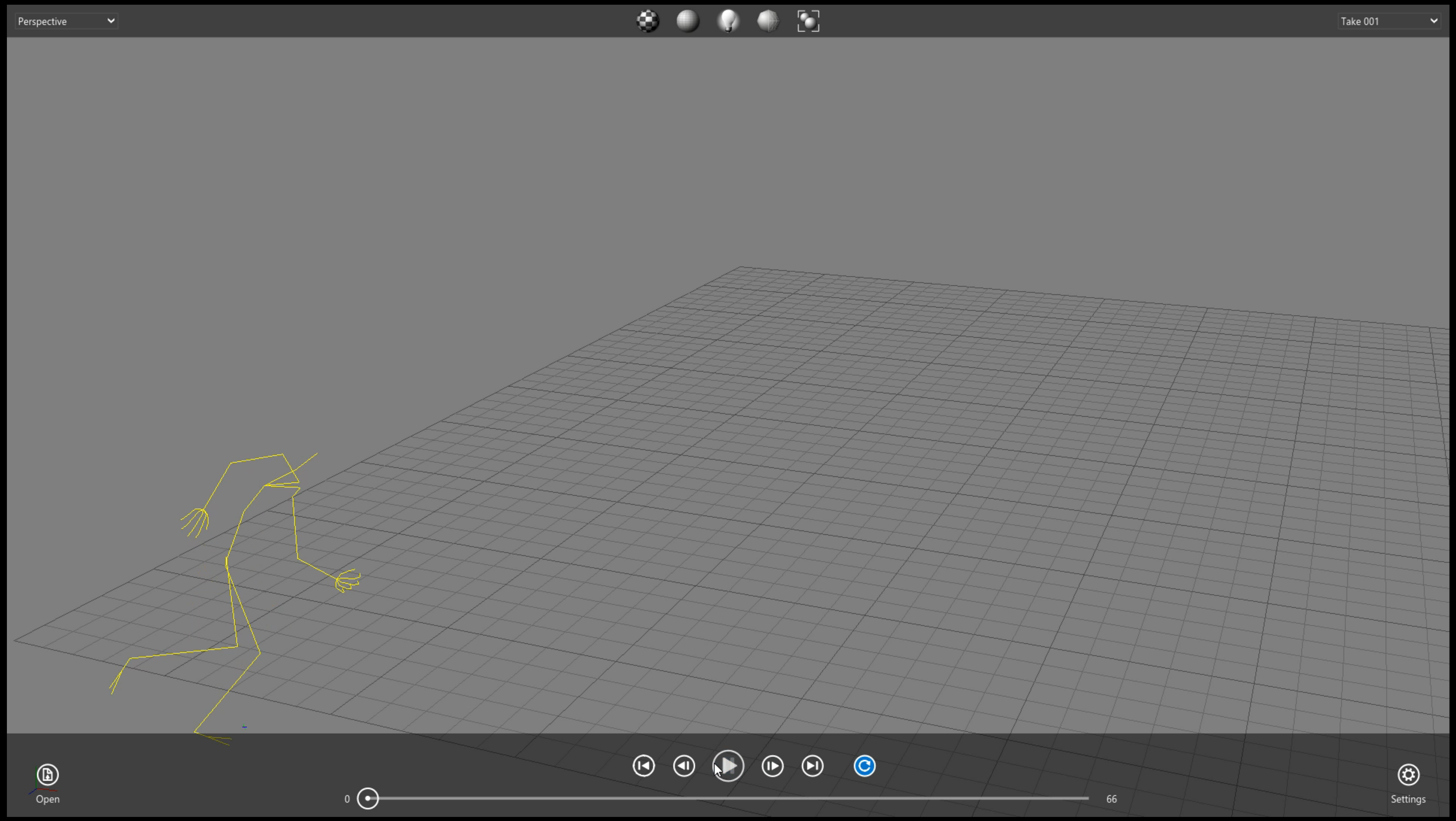
The 4 No-No's

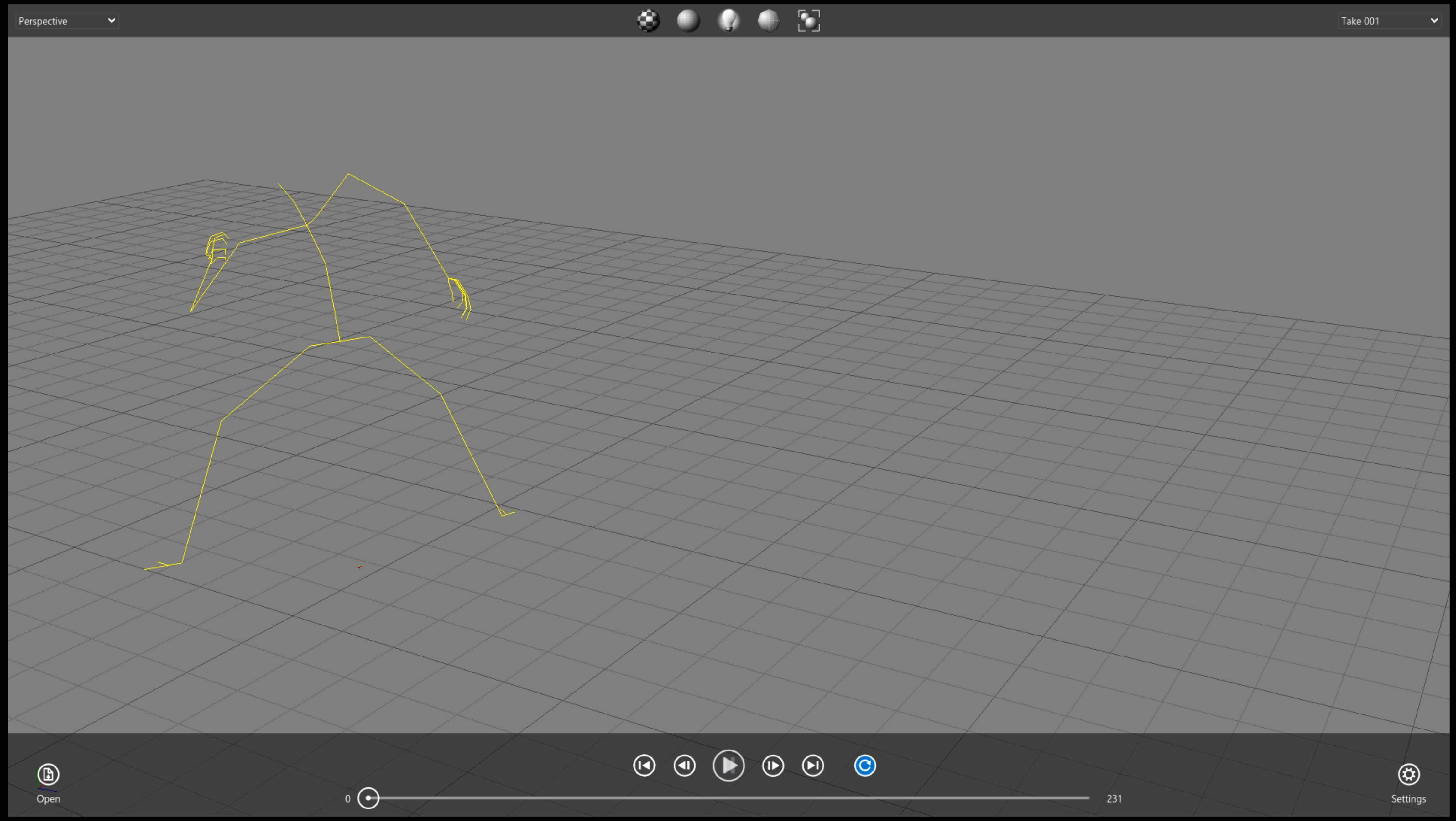
- Locomotion & Cyclic motions
- Phase as temporal progression
- Pose merging
- Fitting animation along predicted trajectory

Locomotion & Cyclic motions

Most motion synthesis research emphasizes cyclic motions in general and locomotion in particular







unity

Phase as temporal progression

Most motion synthesis research uses the concept of a “phase”; scalar variable in the range 0 to 2π representing the point in time of the current pose in the locomotion cycle

Not true → Given a pose that corresponds to Θ all poses that corresponds to $\Theta + \Delta t$ are “similar”

“Phase” can have an arbitrary meaning (footcontact, entire “action” like for example cartwheel) – not a general concept

```
while not stop:
    direction = spline.interpolate(travel_distance, 'tangent')
    facing_direction = spline.interpolate(travel_distance, 'direction')
    avg_speed = spline.interpolate(travel_distance, 'amplitude').squeeze()
    freq = spline.interpolate(travel_distance, 'frequency').squeeze()

    rot, displacement, height = next_frame(current_phase, avg_speed, direction, facing_direction)

    current_phase = (current_phase + freq) % (2*np.pi)
    travel_distance += avg_speed
    next_distance = travel_distance + displacement

    if next_distance < spline.length():
        rotations.append(rot)
        positions.append((next_distance, height))
    else:
        # End of spline reached
        stop = True
```

unity

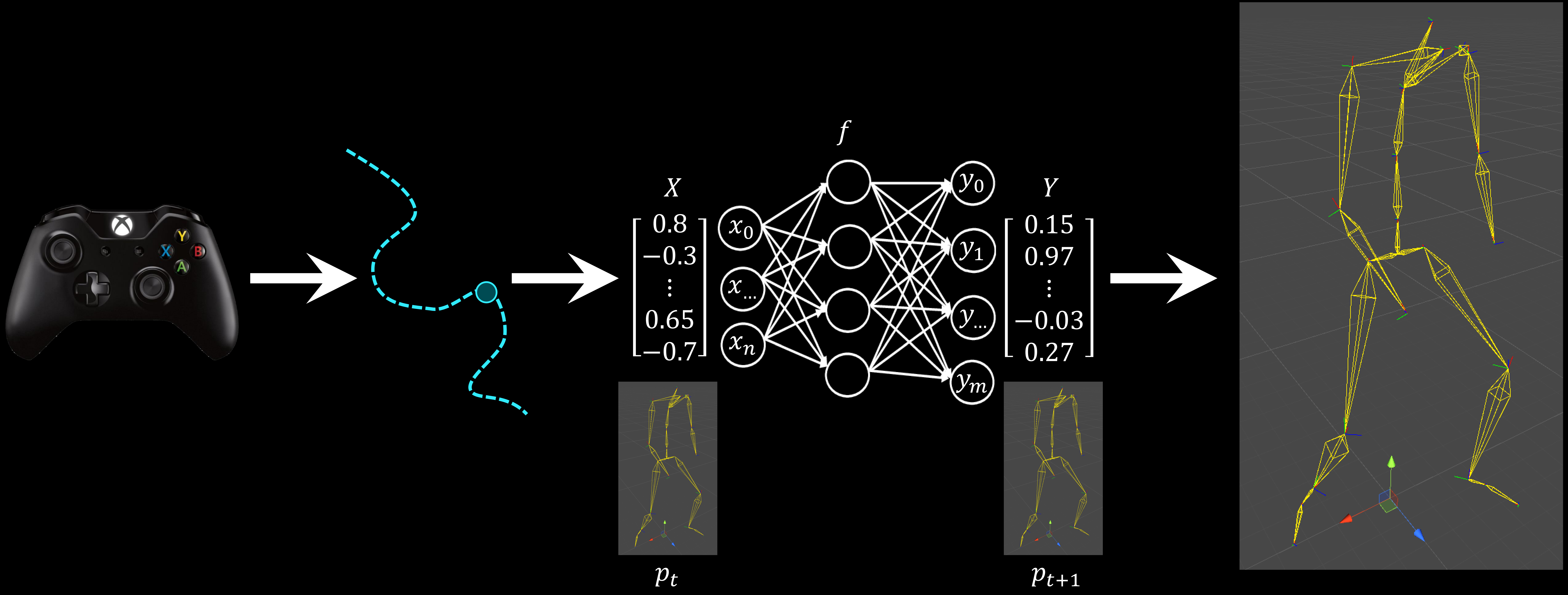

Perspective
Take 001
Open
401
721
Settings


Perspective
Take 001
Open
401
721
Settings


Perspective
Take 001
Open
35
721
Settings

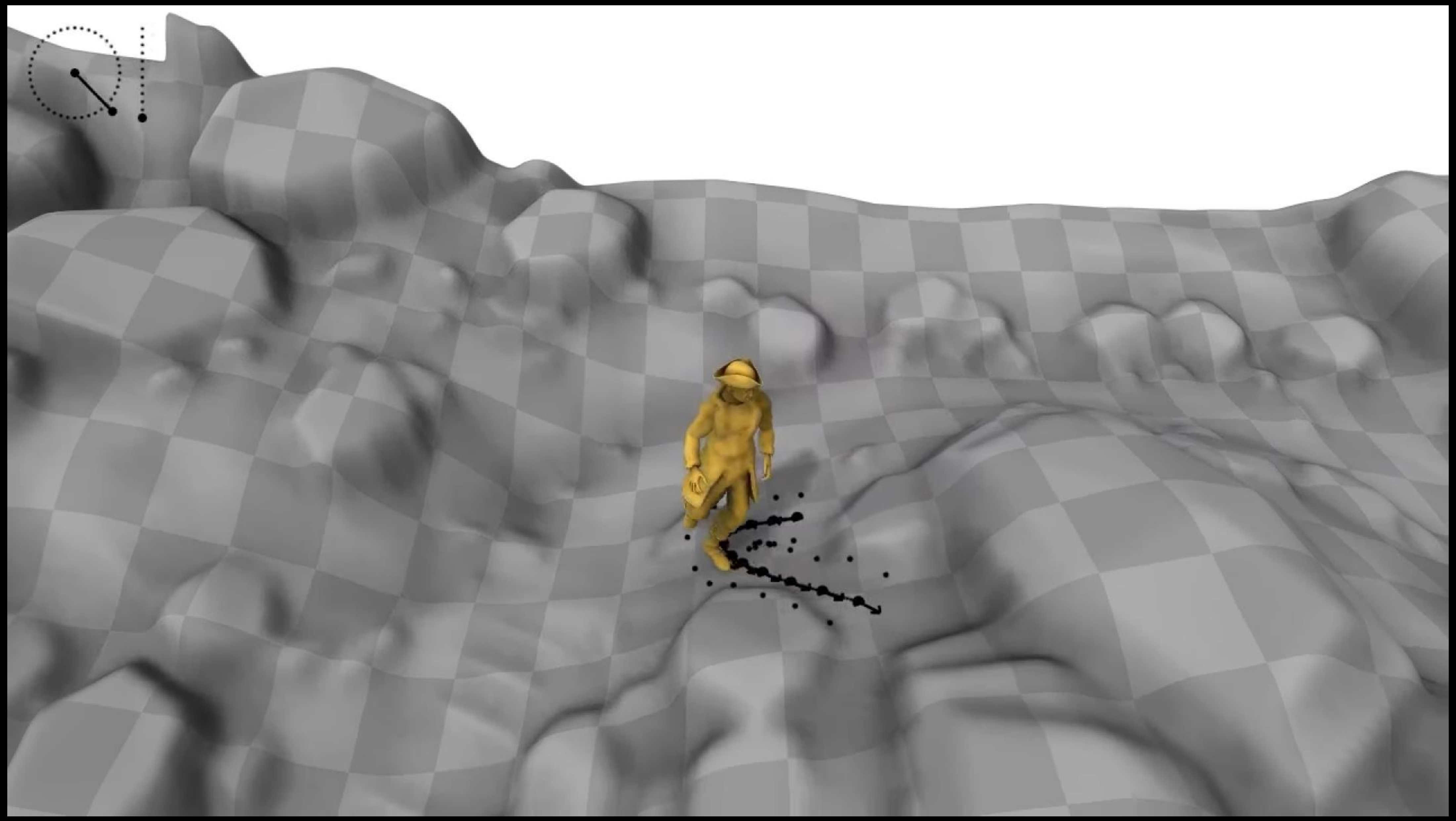

Perspective
Take 001
Open
34
721
Settings

Autoregressive methods



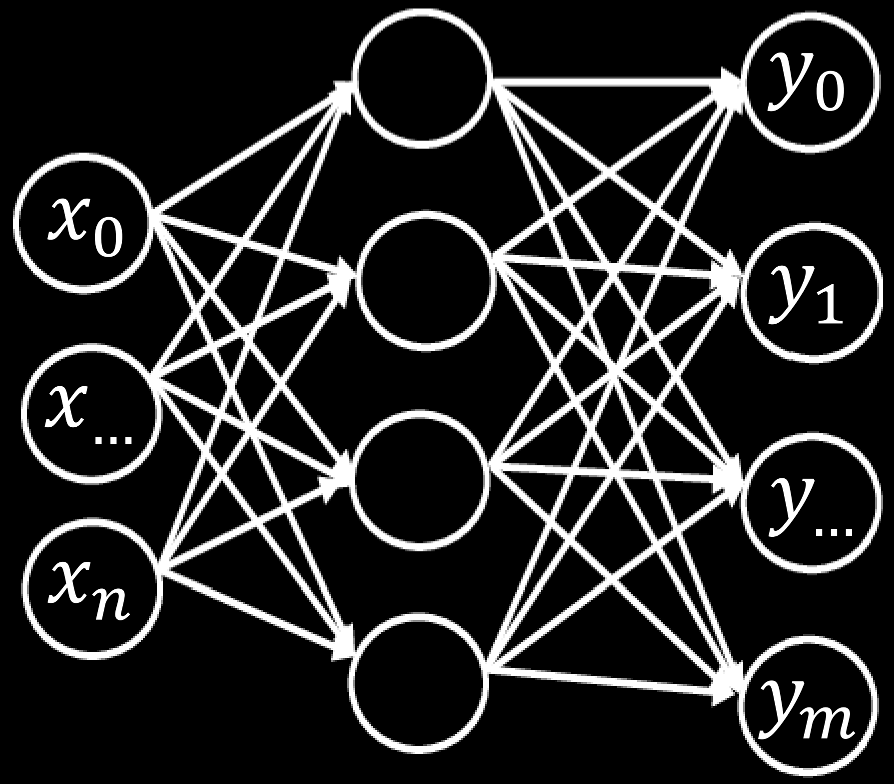

Center
Local
Collab
Account
Layers
Layout
Hierarchy
Create
DeepMotion
Main Camera
Directional Light
character
Plane
Scene
Game
Asset Store
Display 1
Free Aspect
Scale
1x
Maximize On Play
Mute Audio
Stats
Gizmos
Inspector
Navigation
Services
character
Static
Tag
Untagged
Layer
Default
Model
Select
Revert
Open
Transform
Position X 0 Y 0 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1
Deep Motion Controller (Scrip
Script
DeepMotionContr
Height 2
Distance 3
Turn Speed 40
Walk Speed 1.8
Run Speed 3.2
Draw Debug
Debug Draw 2 (Script)
Script
DebugDraw2
Add Component
Project
Create
Favorites
All Materia
All Models
All Prefabs
All Conflict
Assets
Materials
Scenes
NPC
Scripts
Assets ▸ Scenes
NPC
DeepMotion
NPC
PlayerCont...
Console
Clear
Collapse
Clear on Play
Error Pause
Connected Player
Stop for Assert
Stop for Error
0
2
1
Error loading launcher://unity/C:/Users/Mickey/AppData/Roaming/Unity/Packages/node_modules/unity-editor-home/dist/index.html?code=jvlZnuNpPAnsV6tAOJZwVw019f&locale=en&session_state=e72a27f5f69cbbb1bb63d1d76d9fb41f77b41a44c0b08b4b7c563dc57c433df8.U1bbNWrH2JWM7cFB--C1DA003f#/login
Allocated: 53.0 MB Objects: 3170

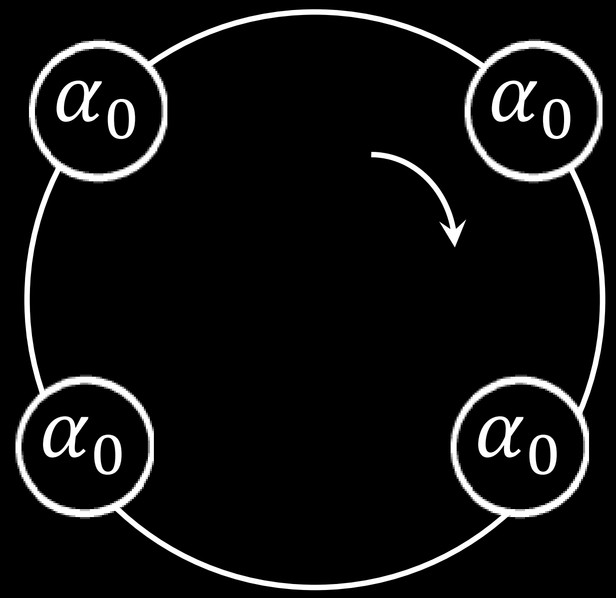
Phase Function Neural Networks



A neural network where the weights are generated as a function of the phase

The "phase" is the scalar variable in the range 0 to 2π representing the point in time of the current pose in the locomotion cycle





$$W_i = \Theta_i\ (p)$$

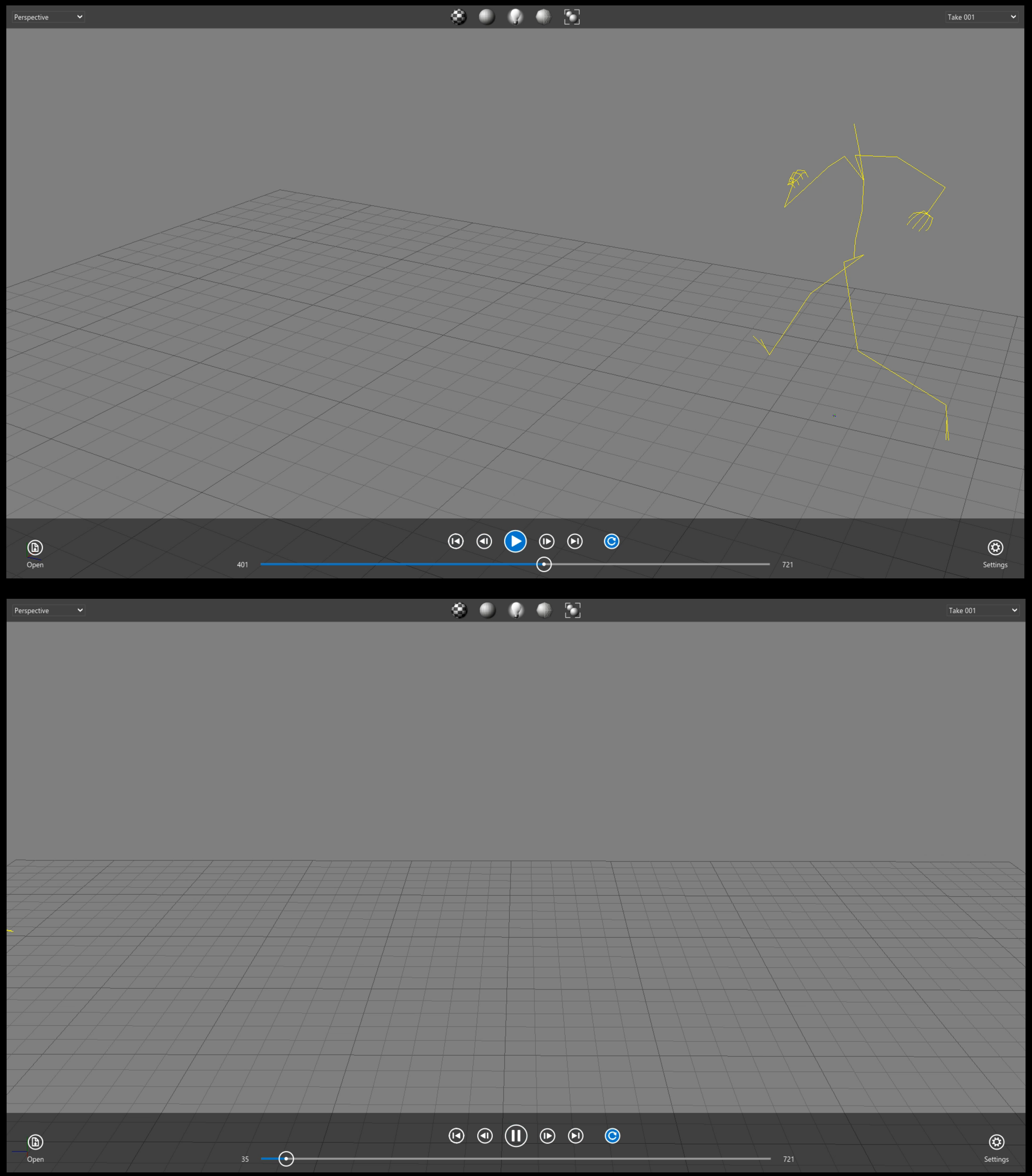
unity

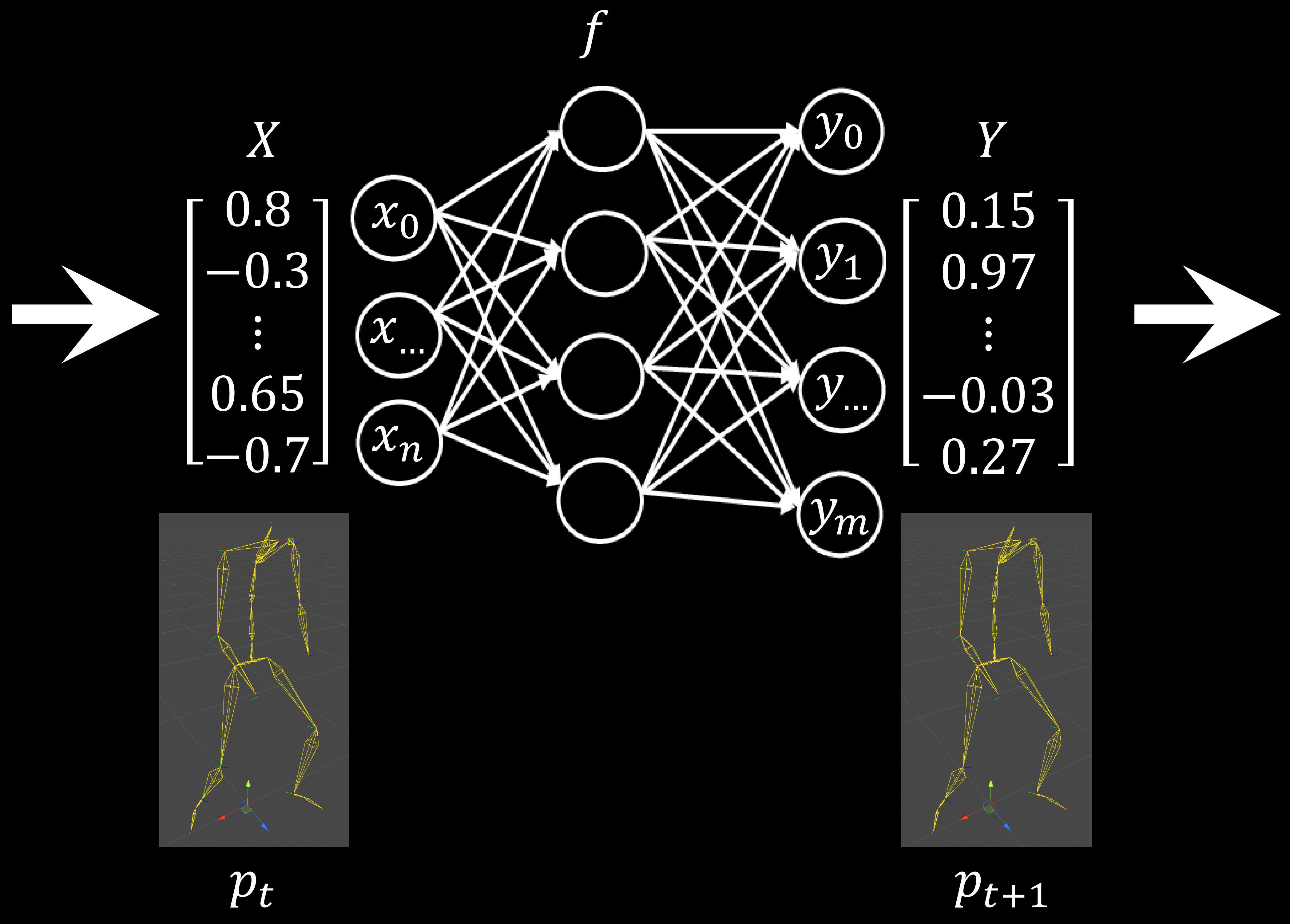

Ik OFF
Prediction ON
Target speed 5.50

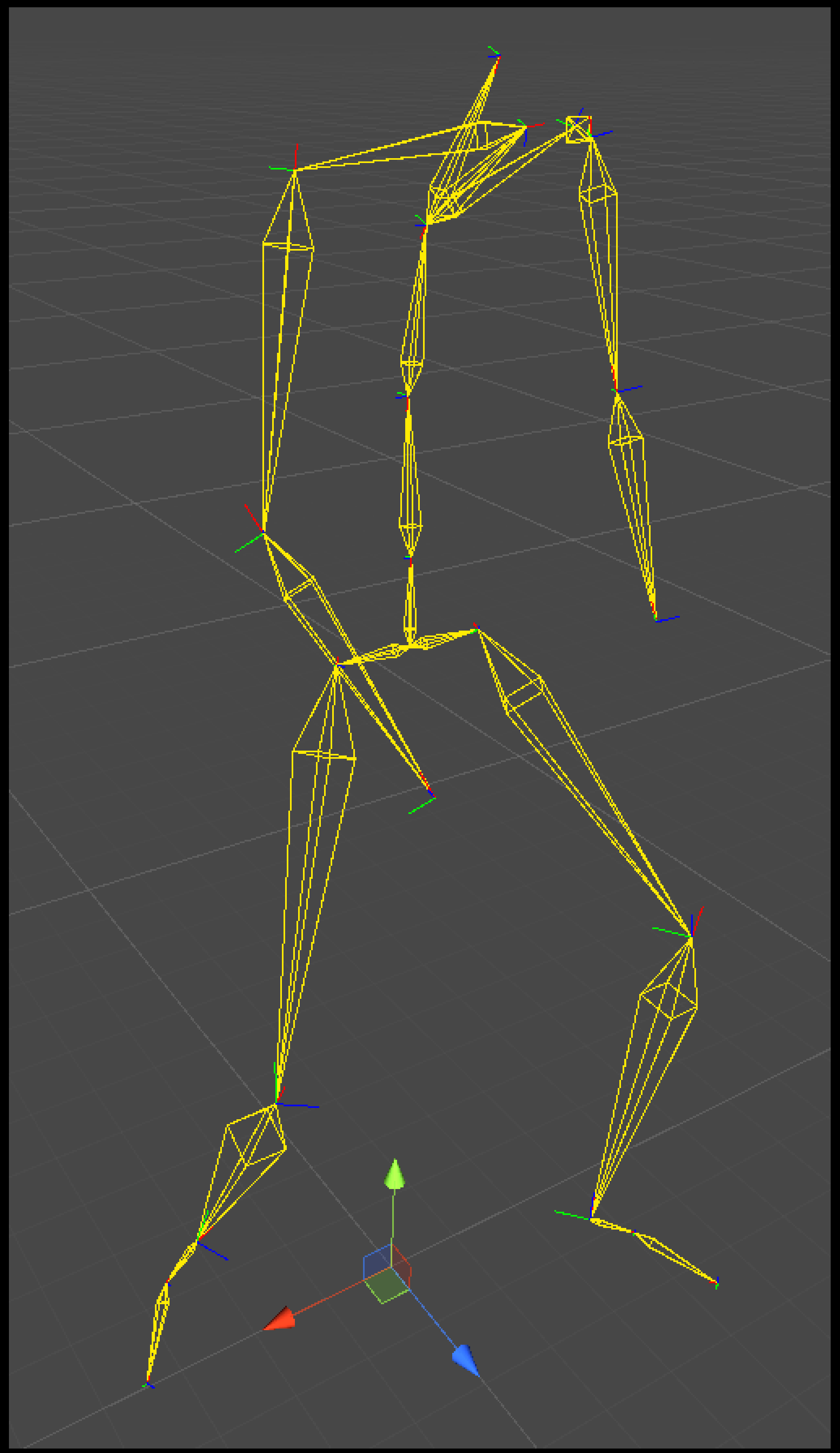

Ik OFF
Prediction ON
Target speed 5.50


Center
Local
Collab
Account
Layers
Default
Hierarchy
Create
SampleScene
Main Camera
Directional Light
GameObject
Scene
Game
Asset Store
Shaded
2D
Gizmos
Persp
Inspector
GameObject
Static
Tag Untagged
Layer Default
Transform
Position X 0 Y 0 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1
Motion Synthesis (Script)
Script MotionSynthesis
Pose Index 5
Add Component

Autoregressive methods & Pose merging







Autoregressive methods

Autoregressive network training averages the possible continuation candidates -> Loss of quality

“…NN is compact, requiring only a few megabytes of memory, even when trained on gigabytes of motion capture data…”

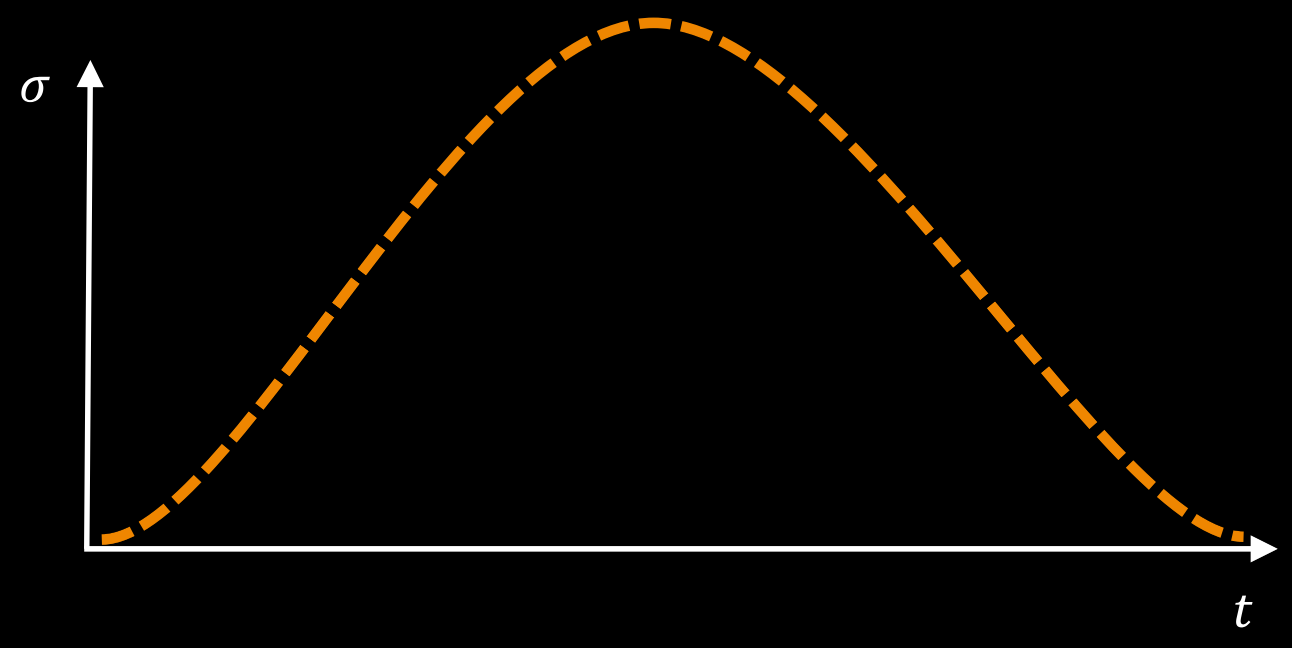
“…requires keeping all the motion data…”

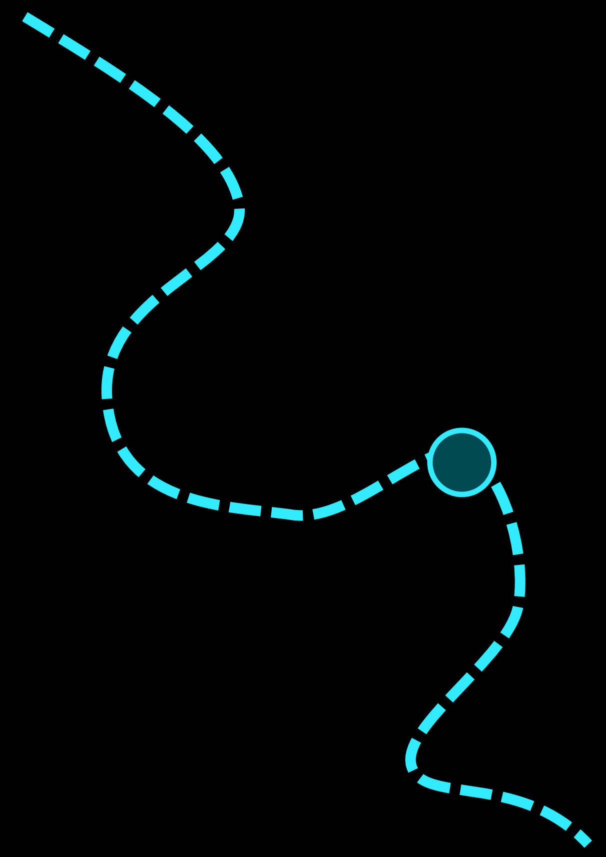
Trajectory Control

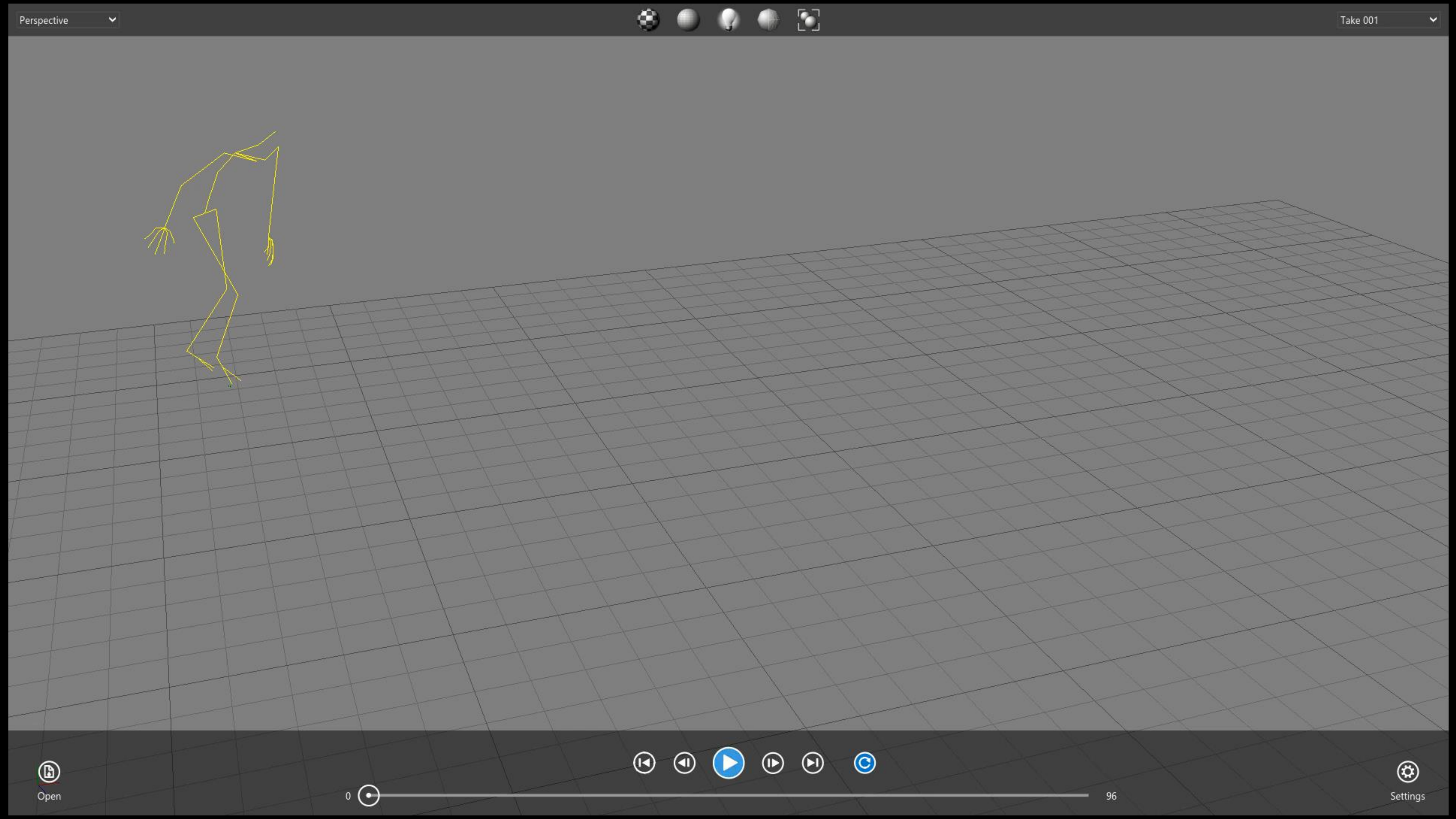
Trajectory annotations are used to guide the pose generation process

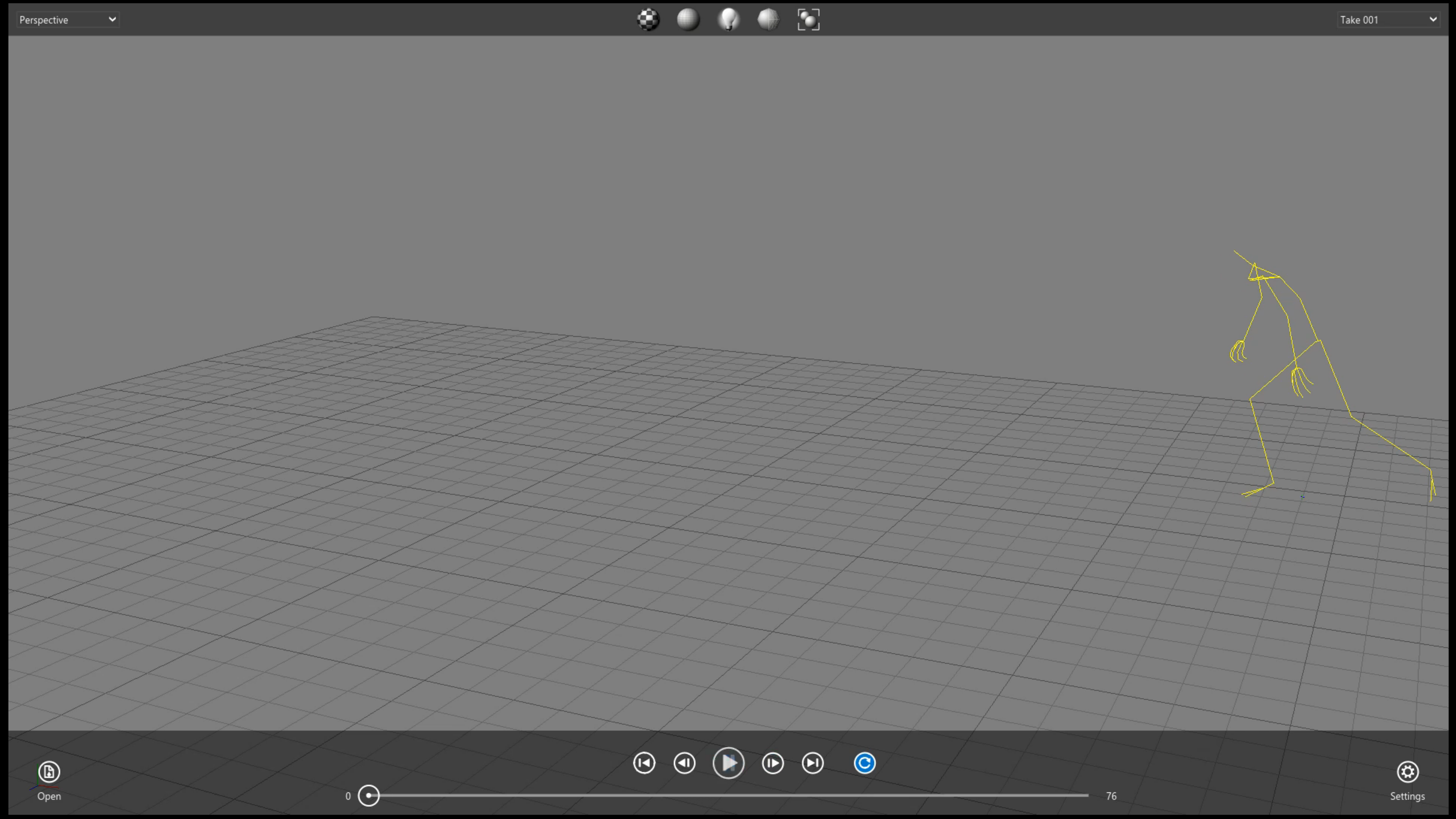
Expert gates can merging different movements (locomotion & jumping) -> Loss of quality

Trajectory dictates timing, but instead animation needs to dictate overall timing









Current time: 0.00
Blend weight: 0.00

Phase Function NN | Mode adaptive NN

Both approaches only work for locomotion

Climbing for example has no obvious phase

Basic assumption – any pose that corresponds to the same "phase" is similar

Poor quality

"Floating"

Don't use exponential maps in NN's

Variations won't be preserved but get averaged

Memory footprint is determined by number of weights – not the amount of animations used

Neural Networks do not memorize anything!

Slow runtime performance

my SSE implementation was ~0.9ms

4 samples of Θ yields average result, more samples require a higher memory footprint

Phase mispredictions result in a tendency towards the mean pose

Extraordinary long training times (6+ hours)

Edit data, retrain, hope it appears

We can't predict the output

…or ask why it was produced

Mode adaptive NN's don't work for bipeds

24
20
10 Antidote
6080
70


Lightning Blade
15
Estus Flask+10
0


W


MKIceAndFire

The holy grail

- Fast turn-around times
- Ground-truth motion synthesis
- Minimal memory footprint
- Fast runtime
- Style
- Scaleable
- Controllable
- Versatile
- Precise

Kinematica

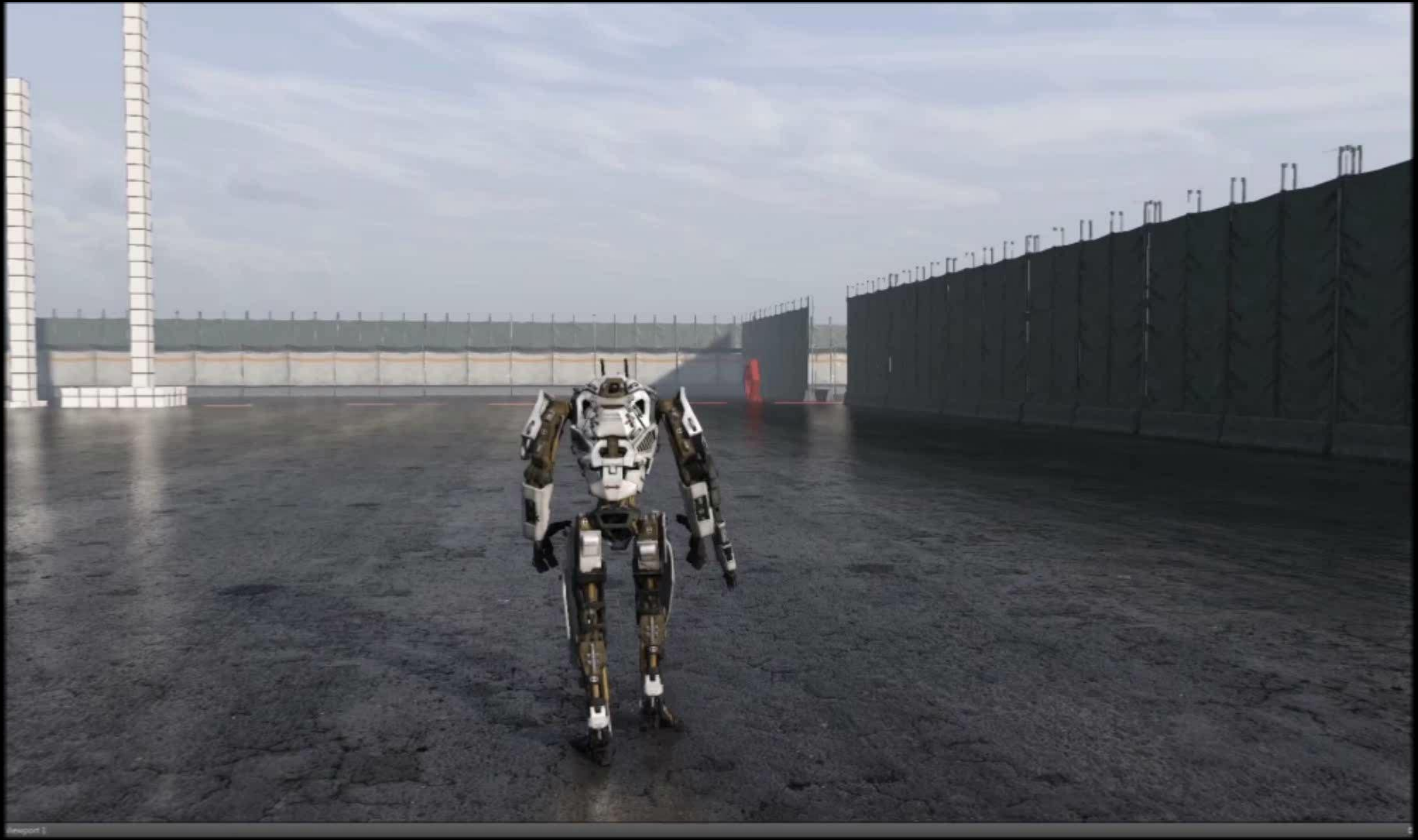
Motion Matching

Pros

- Preserves high quality result
- Does not rely on phases
- Relatively easy to implement

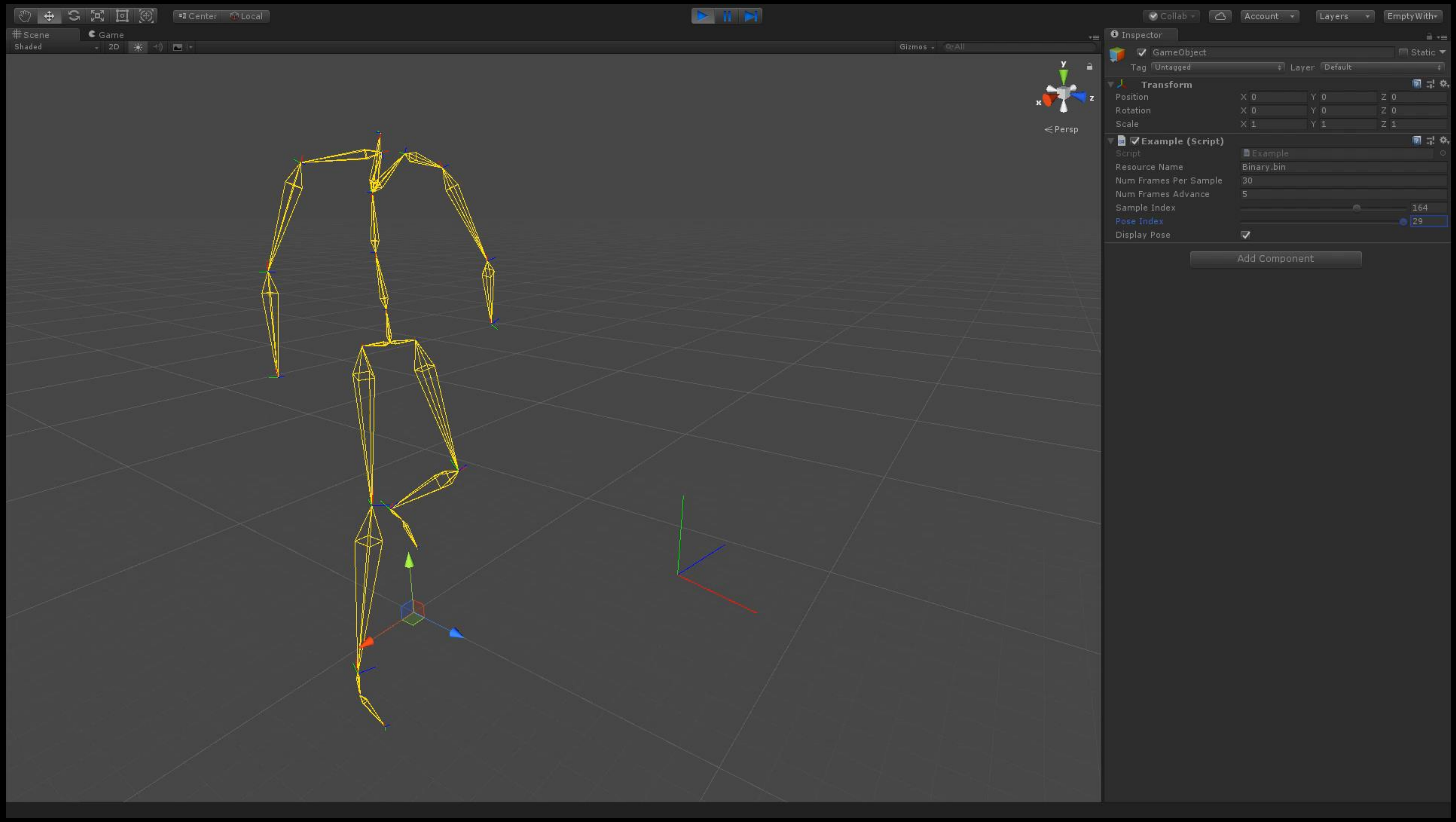
Cons

- Prediction must match data
- Construction of cost functions
- Requires a lot of tweaking
- Doesn’t scale well
- Duplicate data problem
- “Back-in-time” problem




Pivot
Local
Collab
Account
Layers
Layout
Scene
Game
Shaded
2D
Gizmos
All
Persp
Inspector
Navigation
GameObject
Static
Tag Untagged
Layer Default
Transform
Position X 0 Y 0 Z 0
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1
Fragment (Script)
Script Fragment
Resource Name Binary.bin
Display Root Trajectory
Display Joint Trajectories
Display Relative
Display Fragment
Display Matching Fragment
Policy Index 0
Timeline
.00
85.00
90.00
95.00
Offset 0
Add Component
Auto Generate Lighting On

Motion Fragments



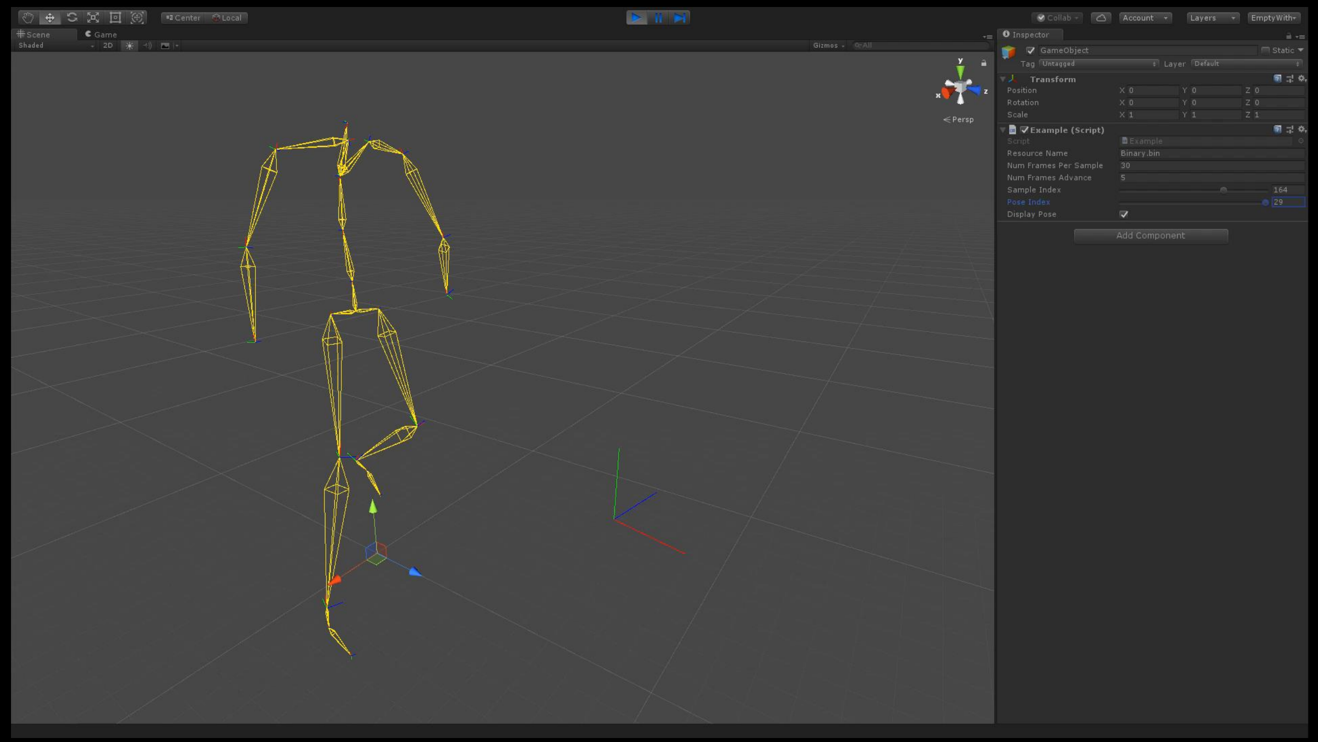
A motion fragment is represented as a matrix $F_i \in M^{\tau\times(m+1)}(\mathbb{R})$ where each entry v_j^t contains the velocity of joint j at some time $t \in \{i-\tau, \dots, i+\tau\}$, and $j = 0$ represents the root transform

$$F_i = \begin{bmatrix} v_0^i & v_0^{i+1} & \cdots & v_0^{i+\tau} \\ v_1^{i-\tau} & v_1^{i-\tau+1} & \cdots & v_1^i \\ v_2^{i-\tau} & v_2^{i-\tau+1} & \cdots & v_2^i \\ \vdots & \vdots & \vdots & \vdots \\ v_m^{i-\tau} & v_m^{i-\tau+1} & \cdots & v_m^i \end{bmatrix}$$

$$v_j^t = J_j^t - J_j^{t+1}\left({T_r^{t+1}}^{-1} T_r^t\right)$$

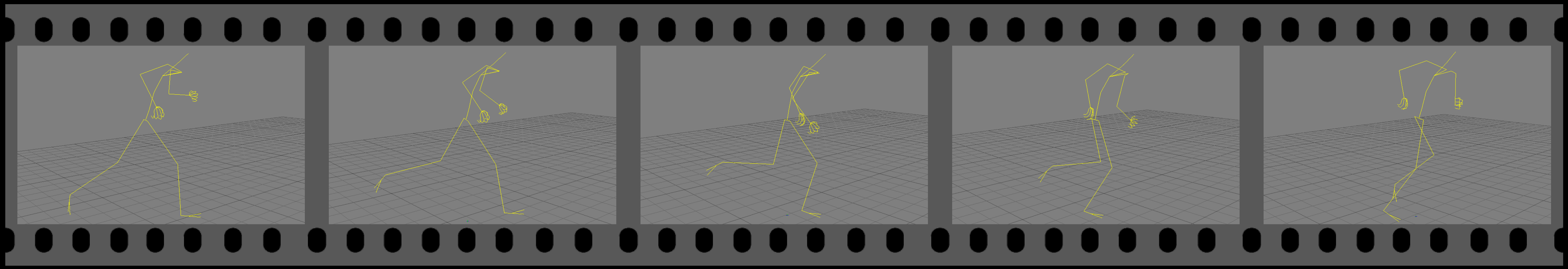
Kinematica Algorithm

$$s(F, F') = \sum_{k=1}^{n} F_k - F'_k{}^2$$



Query -> $F(y)$

$$\arg\min x \; s(F(x), F(y))$$



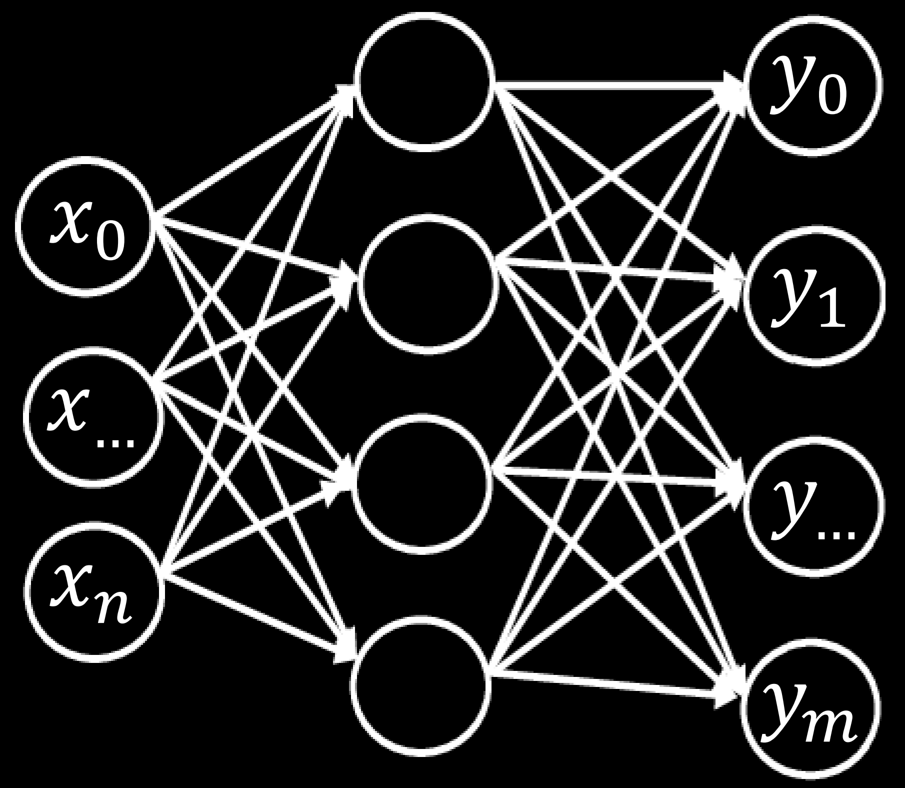
Motion Database

Nearest Neighbor Search

$$F_i = \begin{bmatrix} v_0^{i} & v_0^{i+1} & \cdots & v_0^{i+\tau} \\ v_1^{i-\tau} & v_1^{i-\tau+1} & \cdots & v_1^{i} \\ v_2^{i-\tau} & v_2^{i-\tau+1} & \cdots & v_2^{i} \\ \vdots & \vdots & \vdots & \vdots \\ v_m^{i-\tau} & v_m^{i-\tau+1} & \cdots & v_m^{i} \end{bmatrix} \longrightarrow$$

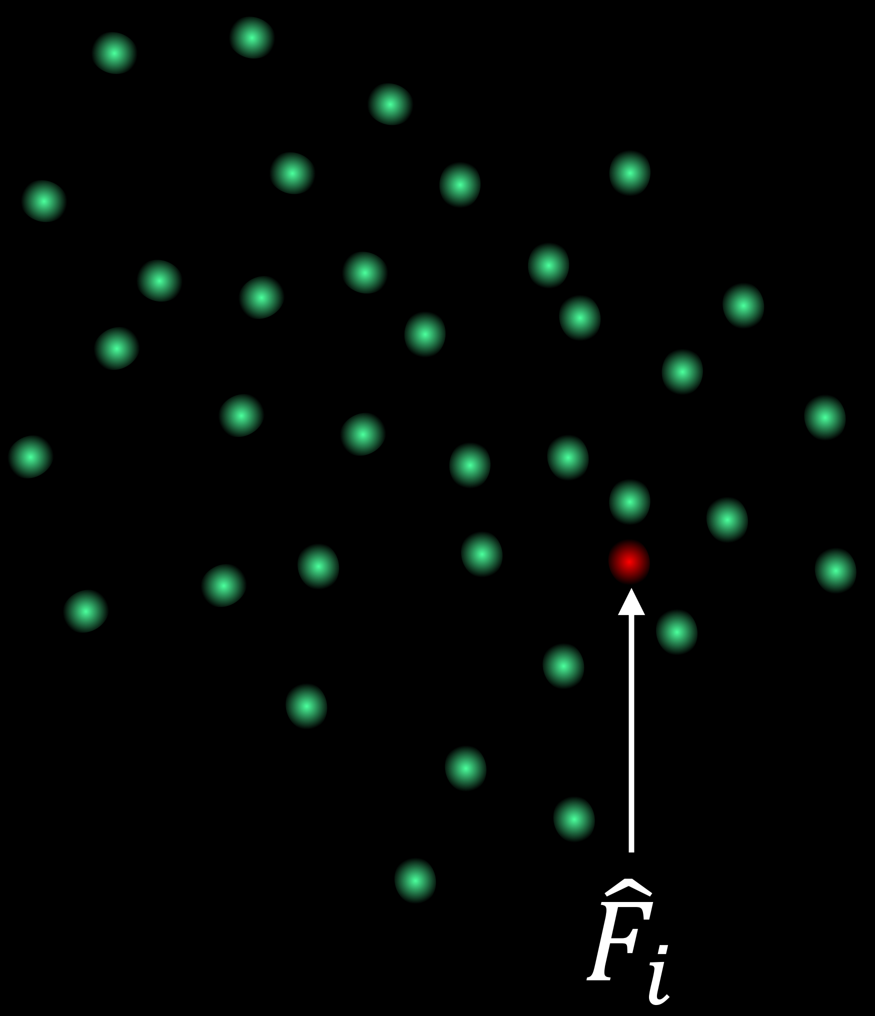
300+ Scalar Values
1200 bytes per fragment
For 70.000 poses -> 82 Mb

Product Quantization for Nearest Neighbor Search
[Jegou, Douze, Schmid, 2011]



$$\longrightarrow \hat{F}_i = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}$$

< 64 bytes per fragment
For 70.000 poses -> ~3 Mb



Nearest Neighbor Search time < 0.05 ms (HP C#)

Sub-Linear Nearest Neighbor Search - $k = 1$

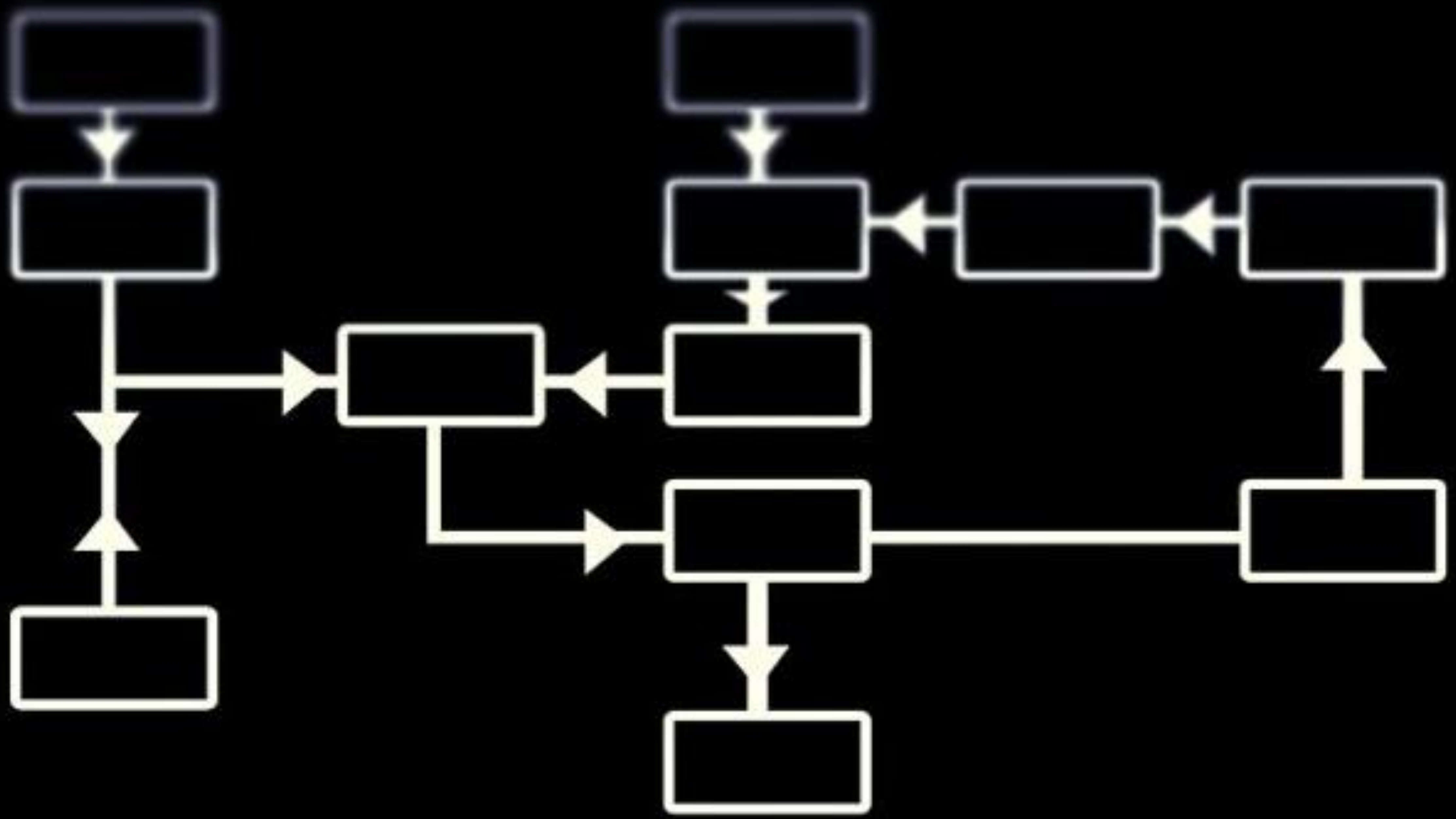
Short training time (< 5 minutes)


Pivot
Local
Collab
Account
Layers
Default
Scene
Game
Asset Store
Display 1
Free Aspect
Scale
1x
Maximize On Play
Mute Audio
VSync
Stats
Gizmos
Inspector
Contact Threshold 0.15
Maximum Linear Error 0.3
Maximum Angular Error 25
Debug settings
Enable Debugging
Debug Index 0
Debug Pose Index 0
Fractional Time 0
Locomotion Ability (Script)
Script LocomotionAbility
Prediction settings
Desired Speed Slow 3.9
Desired Speed Fast 5.5
Velocity Percentage 1
Forward Percentage 1
Debug setting
Display Desired Traject
Display Anchor Trajecto
Display Candidate Traje
Player Character (Script)
Script PlayerCharacter
Resource Name Binary.bin
Root Root (Transform)
Blend Duration 0.3
Record & Rewind
Enable Recording
Recorder State Record
Debugger Timeline
2.50
5.00
7.50
10
Movement Controller (Script)
Is Enabled
Use Transform
Actor Properties
Width (as radius) 0.35
Mass 2
Gravity
Is Gravity Enabled
Dampen Factor 1
Skin Width 0.01
Grounding
Use Grounding Layers
Grounding Layers Default
Grounding Radius 0.1
Grounding Ray 0.5 0.75
Force Grounding 0.3
Collisions
Is Collision Enabled
Stop Rotations
Allow push back
Collision Layers Mixed...
Auto Generate Lighting Off

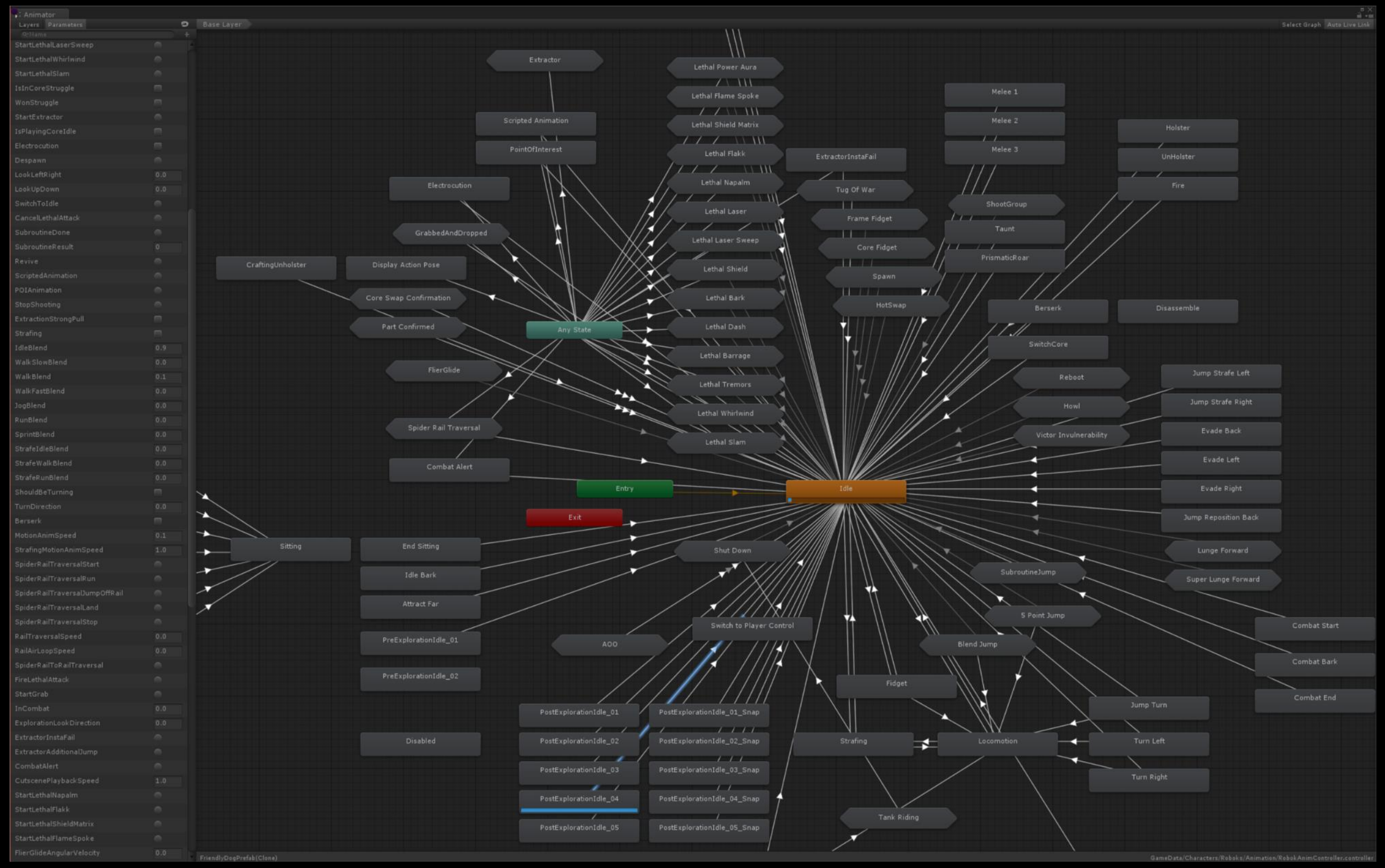
Pivot
Local
Collab
Account
Layers
Default
Scene
Game
Asset Store
Shaded
2D
Gizmos
All
Persp
Inspector
Contact Threshold 0.15
Maximum Linear Error 0.3
Maximum Angular Error 25
Debug settings
Enable Debugging
Debug Index 0
Debug Pose Index 0
Fractional Time 0
Locomotion Ability (Script)
Script LocomotionAbility
Prediction settings
Desired Speed Slow 3.9
Desired Speed Fast 5.5
Velocity Percentage 1
Forward Percentage 1
Debug setting
Display Desired Traject
Display Anchor Trajecto
Display Candidate Traje
Player Character (Script)
Script PlayerCharacter
Resource Name Binary.bin
Root Root (Transform)
Blend Duration 0.3
Record & Rewind
Enable Recording
Recorder State Rewind
Debugger Timeline
2.50
5.00
7.5
Movement Controller (Script)
Is Enabled
Use Transform
Actor Properties
Width (as radius) 0.35
Mass 2
Gravity
Is Gravity Enabled
Dampen Factor 1
Skin Width 0.01
Grounding
Use Grounding Layers
Grounding Layers Default
Grounding Radius 0.1
Grounding Ray 0.5 0.75
Force Grounding 0.3
Collisions
Is Collision Enabled
Stop Rotations
Allow push back
Collision Layers Mixed...
Auto Generate Lighting Off

Game State | Animation Graphs

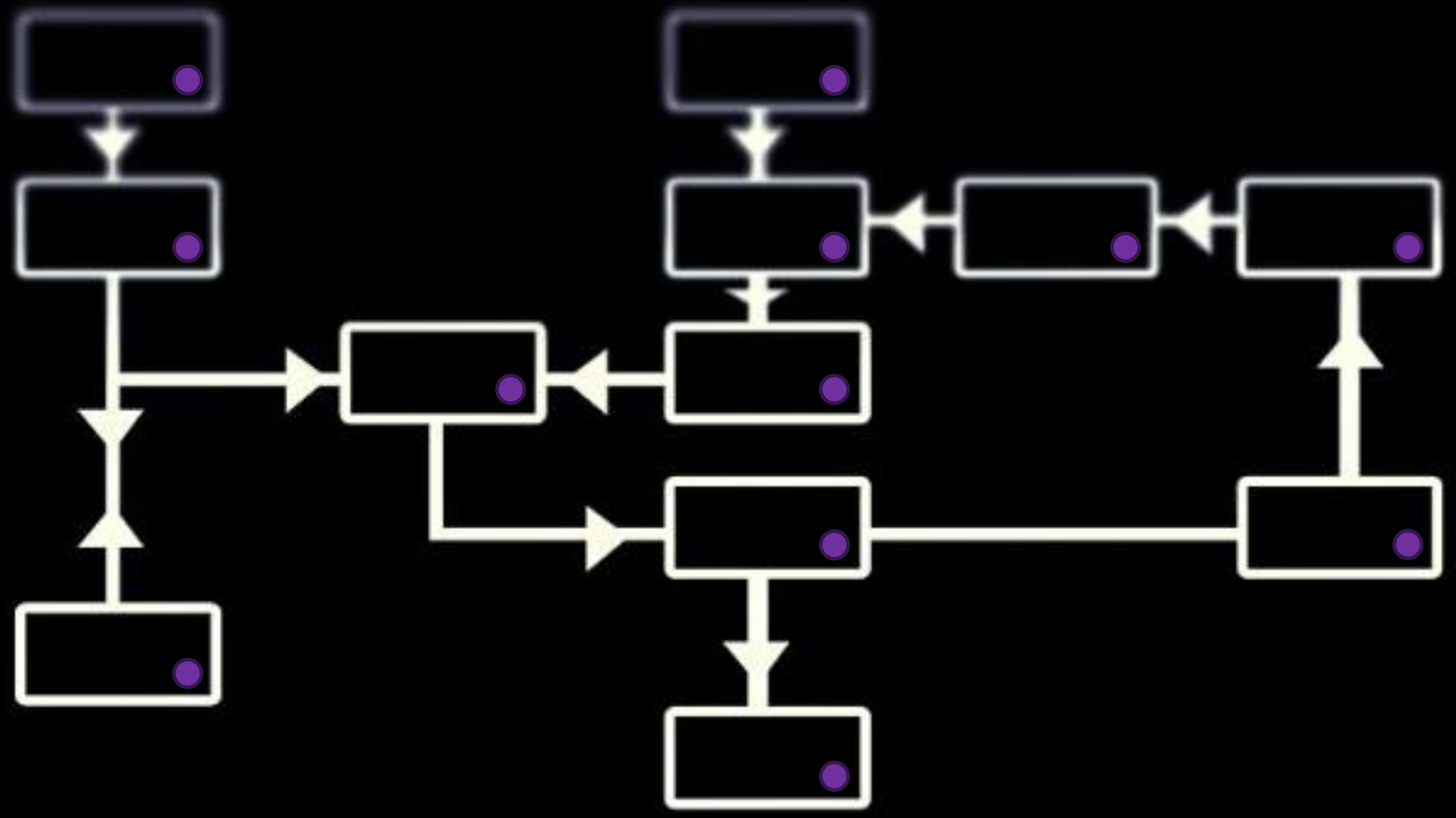
Game Logic



Animation Logic



Game State



Abilities

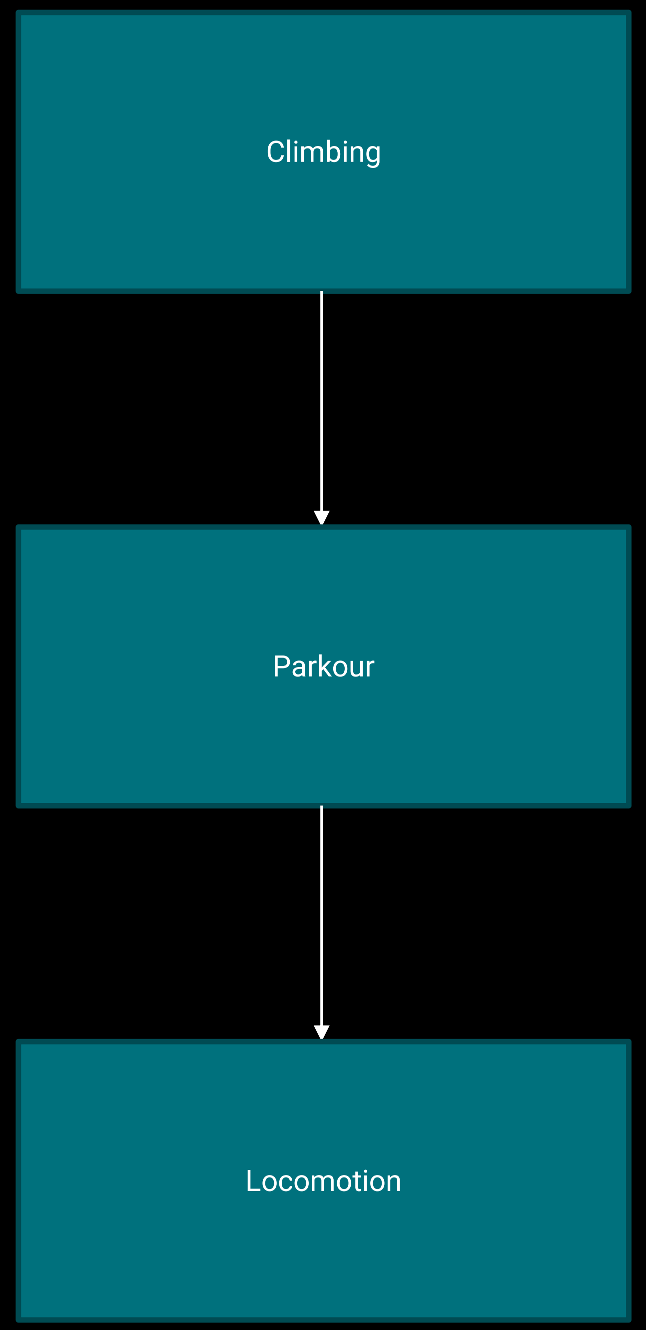
Kinematica’s goal is to provide a **complete** alternative to animation graphs

Parkour, Climbing, Melee Combat, Synchronized movements, One-off actions, etc…

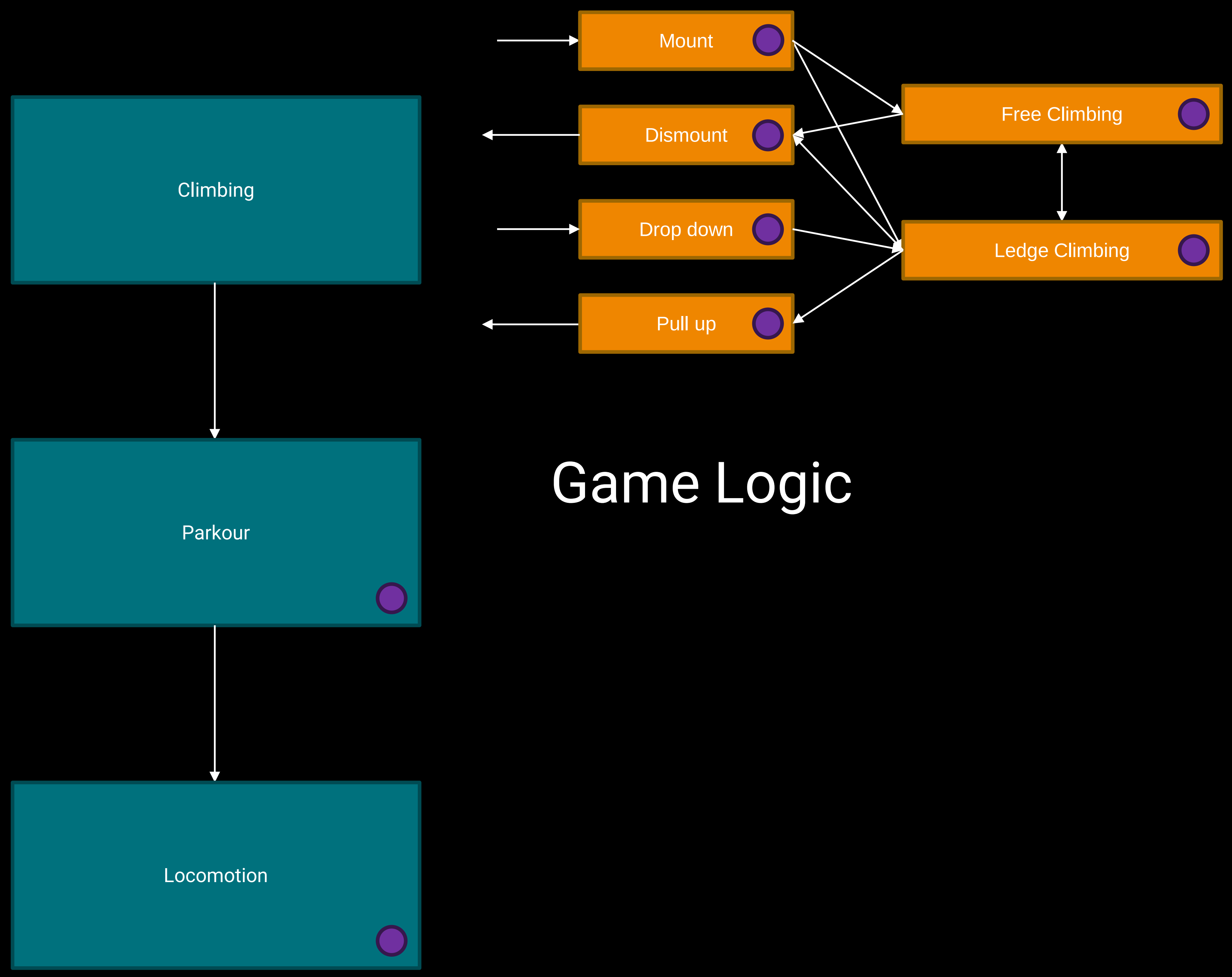
Prioritized list of abilities

Executed in order

This is not a super-imposed concept, i.e. Kinematica does not call into abilities



Kinematica Components



Game Logic

Policies execute as part of the game code (ideally as C# jobs)

Policies execute variations of similarity searches depending on game logic

Similarity searches can be based on 1:1 or n:m fragments

Motion Library

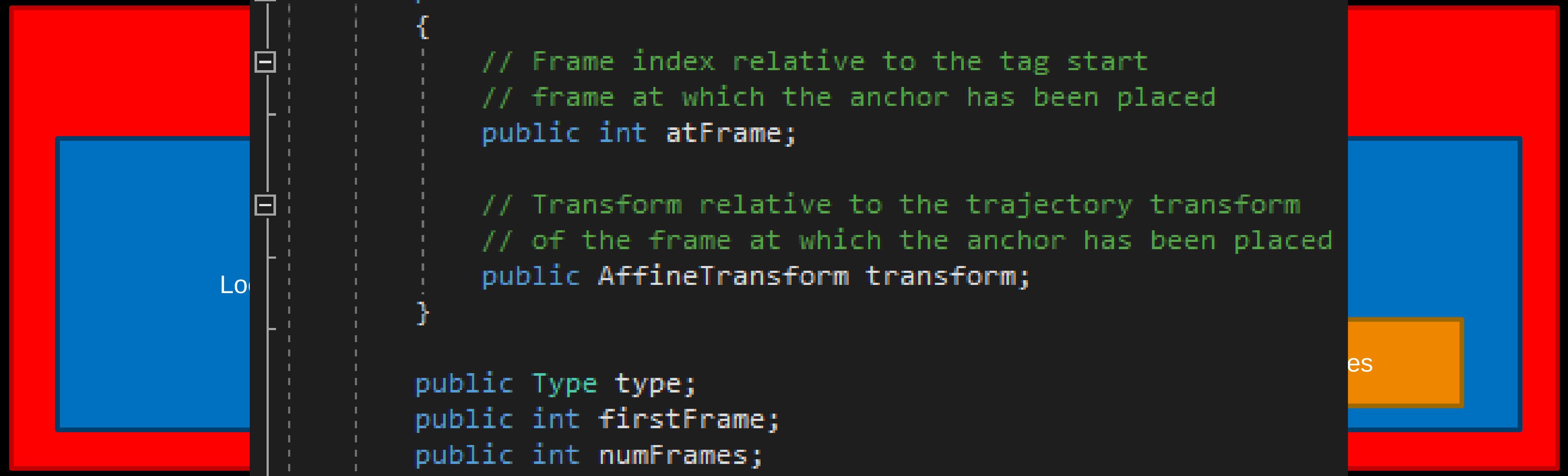
All poses are arranged into a large matrix $\boldsymbol{D} \in M^{(m+1)\times D}(\mathbb{R})$ where each column corresponds to a pose $\mathcal{J} = \{J_i; i =$

$$\boldsymbol{D} = \begin{array}{c} \text{Time} \\ \begin{bmatrix} T_r^1 & T_r^2 & \cdots & T_r^D \\ J_1^1 & J_1^2 & \cdots & J_1^D \\ \vdots & \vdots & \ddots & \vdots \\ J_m^1 & J_m^2 & \cdots & J_m^D \end{bmatrix} \end{array} \text{Poses}$$

Tagging segregates the motion library into addressable islands

Markers carry an arbitrary user-defined payload and are associated with discrete frames

Policies utilize tags and markers in user-defined similarity searches



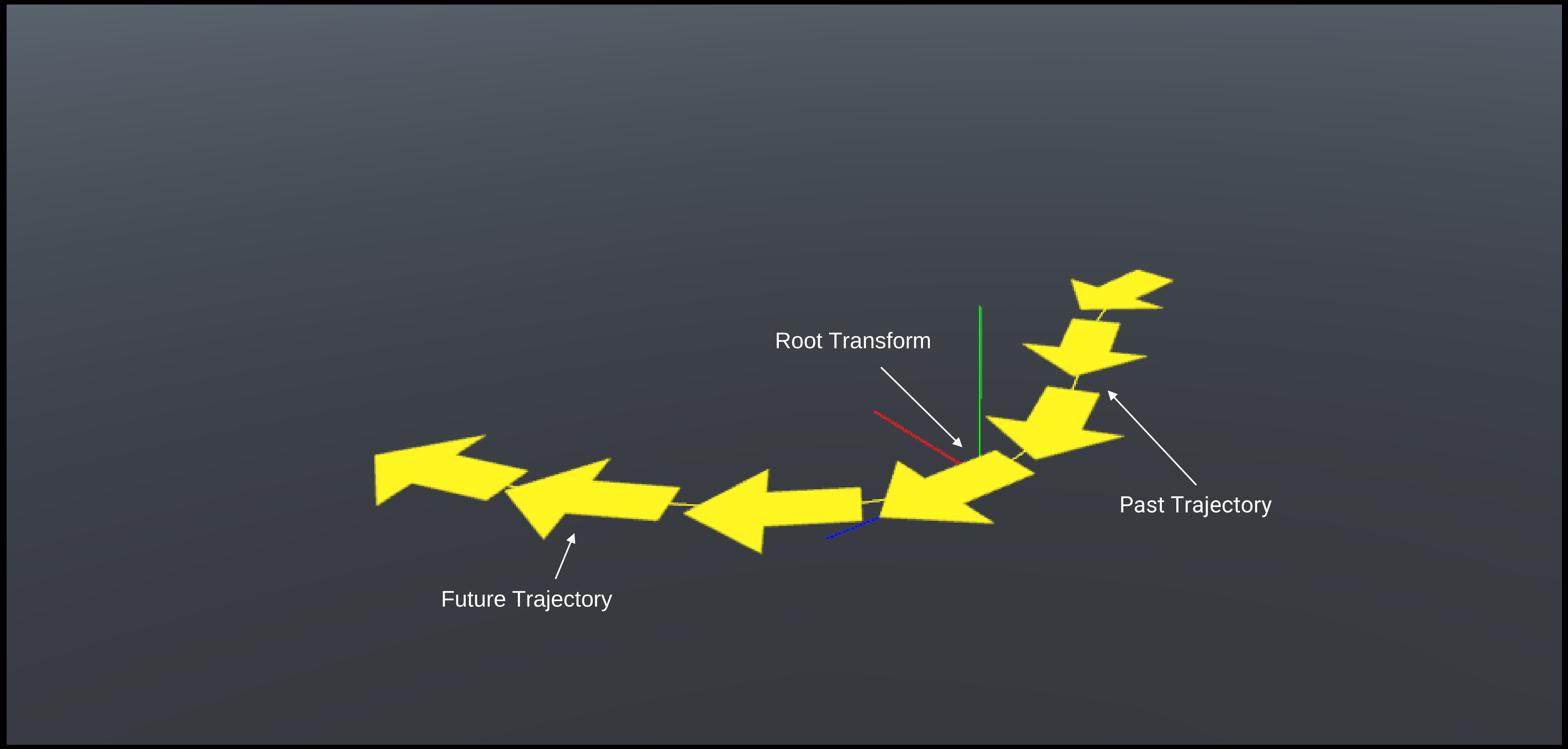
```
namespace Parkour
{
    public struct Contact
    {
        public struct Anchor
        {
            // Frame index relative to the tag start
            // frame at which the anchor has been placed
            public int atFrame;

            // Transform relative to the trajectory transform
            // of the frame at which the anchor has been placed
            public AffineTransform transform;
        }

        public Type type;
        public int firstFrame;
        public int numFrames;
        public int firstContact;
        public int numContacts;
        public int layerMask;
        public Anchor anchor;
    }
}
```

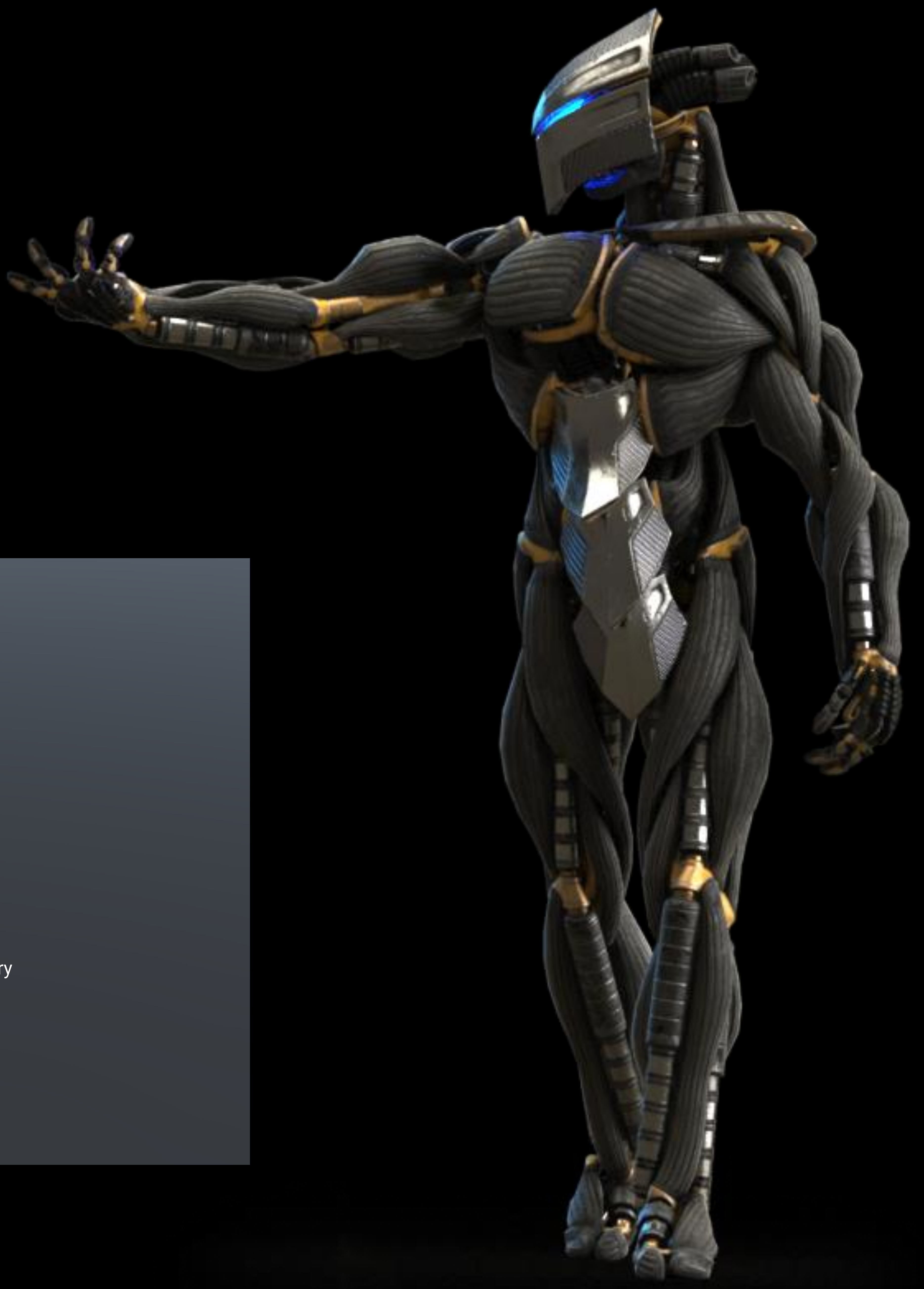
unity

Locomotion

Trajectory Prediction

Using our knowledge about the intended target location (NPC) or desired velocity (PC) we can generate a predicted path over the time horizon



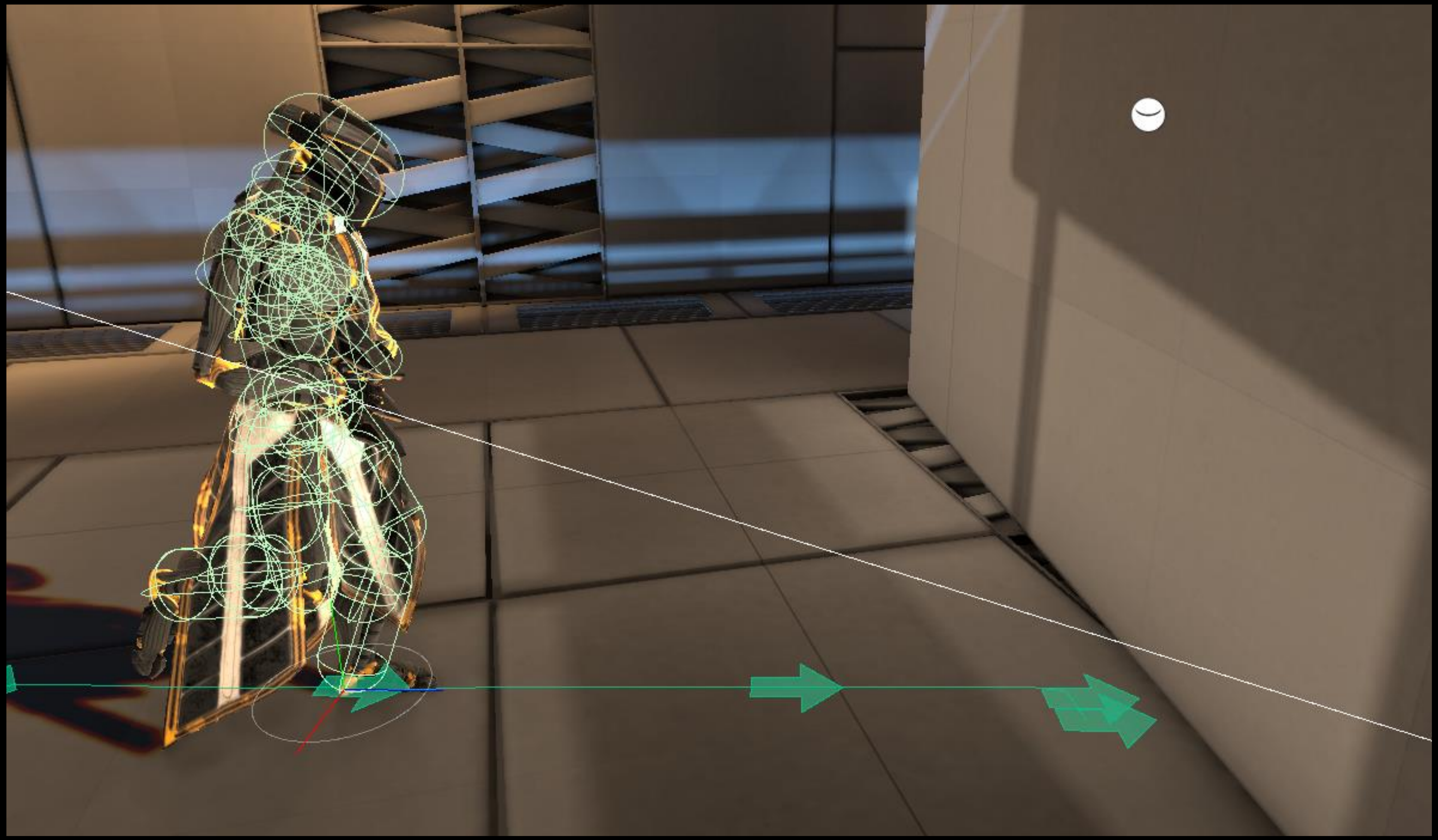


unity


Pivot
Local
Collab
Account
Layers
EmptyWith
Scene
Game
Display 1
Free Aspect
Scale
1x
Maximize On Play
Mute Audio
Stats
Gizmos
Inspector
Navigation
GameObject
Static
Tag
Untagged
Layer
Default
Transform
Position
X -0.1391961
Y 0
Z 0.07112044
Rotation
X 0
Y 119.4
Z 0
Scale
X 1
Y 1
Z 1
Root Motion 3 (Script)
Script
RootMotion3
Maximum Speed
1
Position Bias
0.02
Direction Bias
0.02
Add Component

Collision detection

During the generation of the predicted future trajectory we can perform collision detection with the environment and other characters



unity

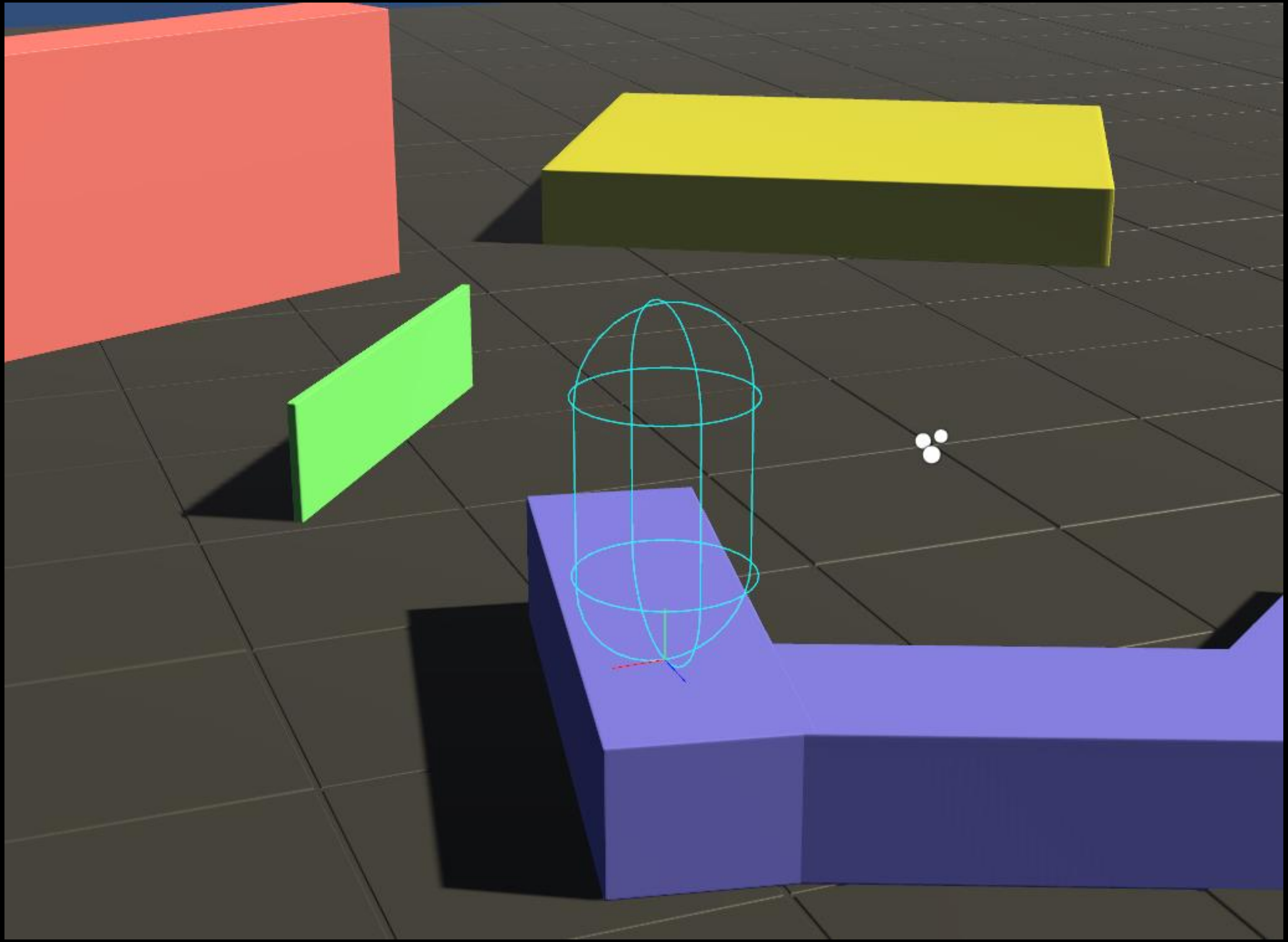

Pivot
Local
Collab
Account
Layers
EmptyWith
Scene
Game
Display 1
Free Aspect
Scale
1x
Maximize On Play
Mute Audio
Stats
Gizmos
Inspector
Navigation
Debug settings
Debug Index 0
Pose Index 0
Locomotion Ability (Script)
Script LocomotionAbility
Prediction settings
Desired Speed Slow 3.9
Desired Speed Fast 5.5
Velocity Factor 1.92
Direction Bias 0.16
Policy settings
Linear Error Multiplier 1
Linear Velocity Error Multipl 1.5
Angular Error Multiplier 1.4
Time Between Jumps 0.15
Animator
Controller None (Runtime Animator Controller)
Avatar Avatar
Apply Root Motion Handled by Script
Update Mode Normal
Culling Mode Always Animate
Clip Count: 0
Curves Pos: 0 Quat: 0 Euler: 0 Scale: 0 Muscles: 0 Generic: 0 PPtr: 0
Curves Count: 0 Constant: 0 (0.0%) Dense: 0 (0.0%) Stream: 0 (0.0%)
Motion Synthesizer (Script)
Script MotionSynthesizer
Resource Name Animations/Binary.bin
Root Root (Transform)
Blend Duration 0.3
Record & Rewind
Enable Recording
Recorder State Record
Debug setting
Display Trajectory
Display Current
Current Time Offset 0
Display Candidate
Candidate Time Offset 0
Display Pose Error
Display Policy Error
Debug Policy
Candidate Index 0
Actor Controller (Script)
Is Enabled
Use Transform
Actor Properties
Width (as radius) 0.35
Mass 2
Gravity
Bake paused in play mode

Trajectory Prediction

During trajectory prediction we use a character controller with the ability to forward simulate the collision world

Allows us to detect collisions in advance and plan accordingly

Controller has full knowledge of which objects it collides with during normal frame-by-frame processing as well as during the prediction phase and can be safely rolled back in time to return to a previous simulation step

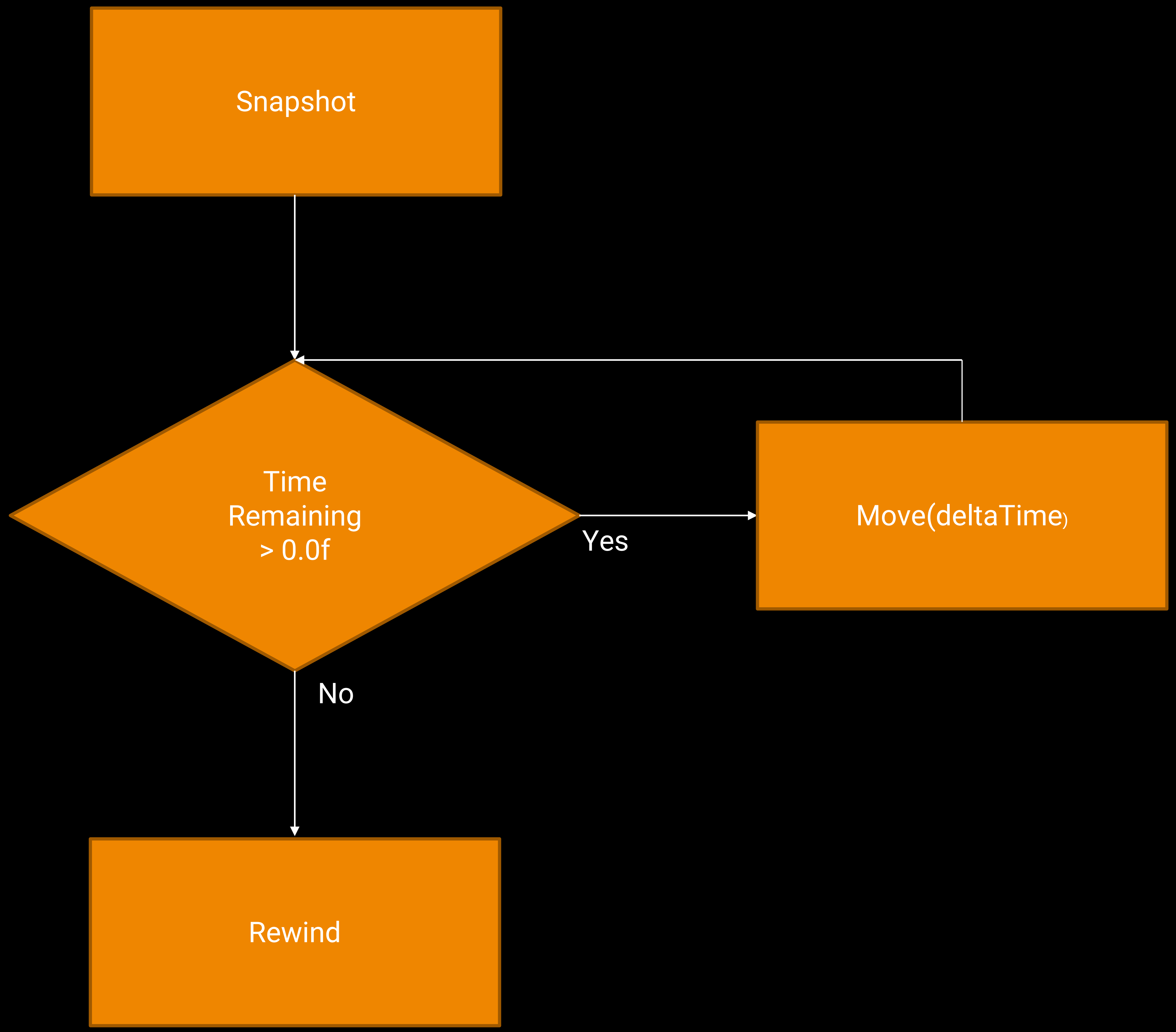


unity


Pivot
Local
Collab
Account
Layers
Empty
Game
Display 1
Free Aspect
Scale
1x
Maximize On Play
Mute Audio
Stats
Gizmos
Bake paused in play mode


Pivot
Local
Collab
Account
Layers
Empty
Game
Display 1
Free Aspect
Scale
1x
Maximize On Play
Mute Audio
Stats
Gizmos
Bake paused in play mode

Forward Prediction



Parkour

Anchors & Contacts

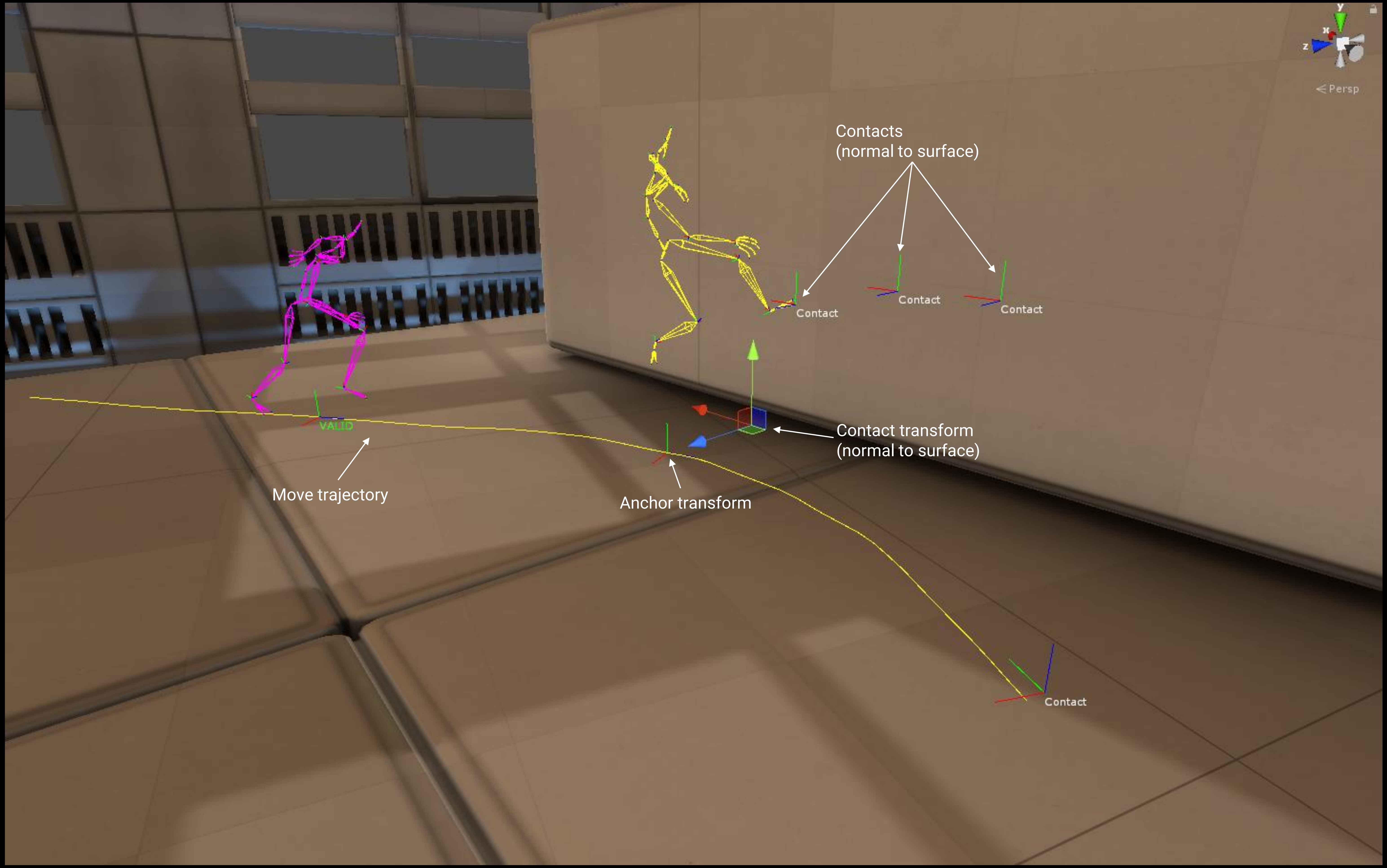
Parkour moves are designed to make precise contacts with the environment

The goal is to generate a predicted future trajectory for a specific parkour move

We use pose annotations to indicate which joint makes contact including the corresponding surface normal

We denote the transform between the contact transform and the root transform of the first contact point as “anchor transform”

Given contact transform -> Move trajectory



unity


Pivot
Local
Collab
Account
Layers
Mickey
Scene
Game
Shaded
2D
Gizmos
All
Debugger
Root (TraversalAbility)
AnchorTransform (Transform)
GameObject (Transform)
Action
5
Display Source Pose
Display Target Pose
Display Contact Pose
Verify Move
Action Pose
39
Synthesizer Pose
1
Transition Time
0
Candidate Pose
1704
<Position x="0" y="0" z="0"/><Rotation x="0" y="0" z="0" w="0.9999999"/>
Persp
Contact
Contact
Contact
Contact
Bake paused in play mode

Anchors & Contacts

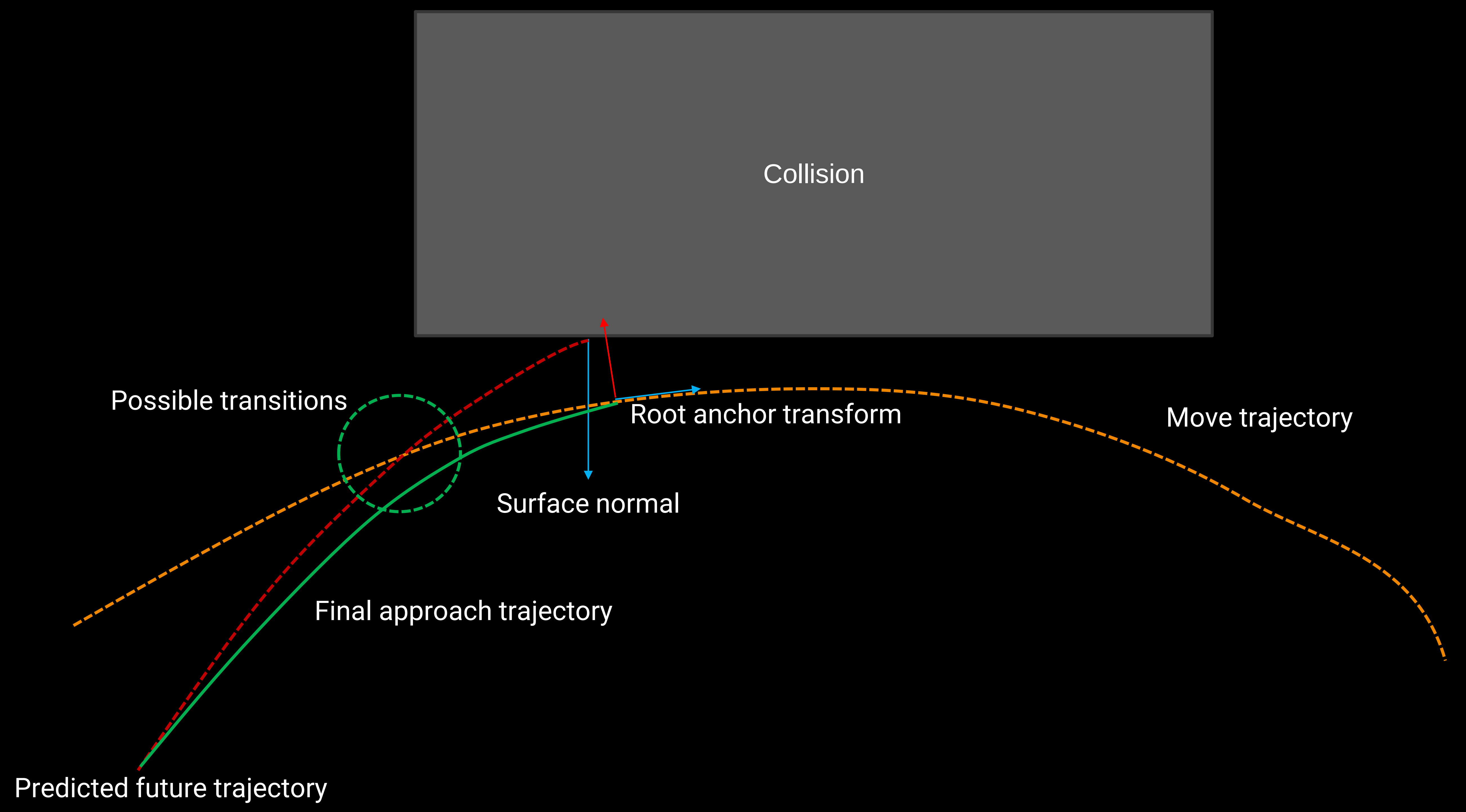
The goal is to generate a predicted future trajectory that “leads into” a specific parkour move

In case the predicted future trajectory detects a collision during the prediction phase…

We generate a “contact transform” which in turn allows us to anchor the entire move in world space

Now we can find possible transitions between the predicted future trajectory and the move trajectory

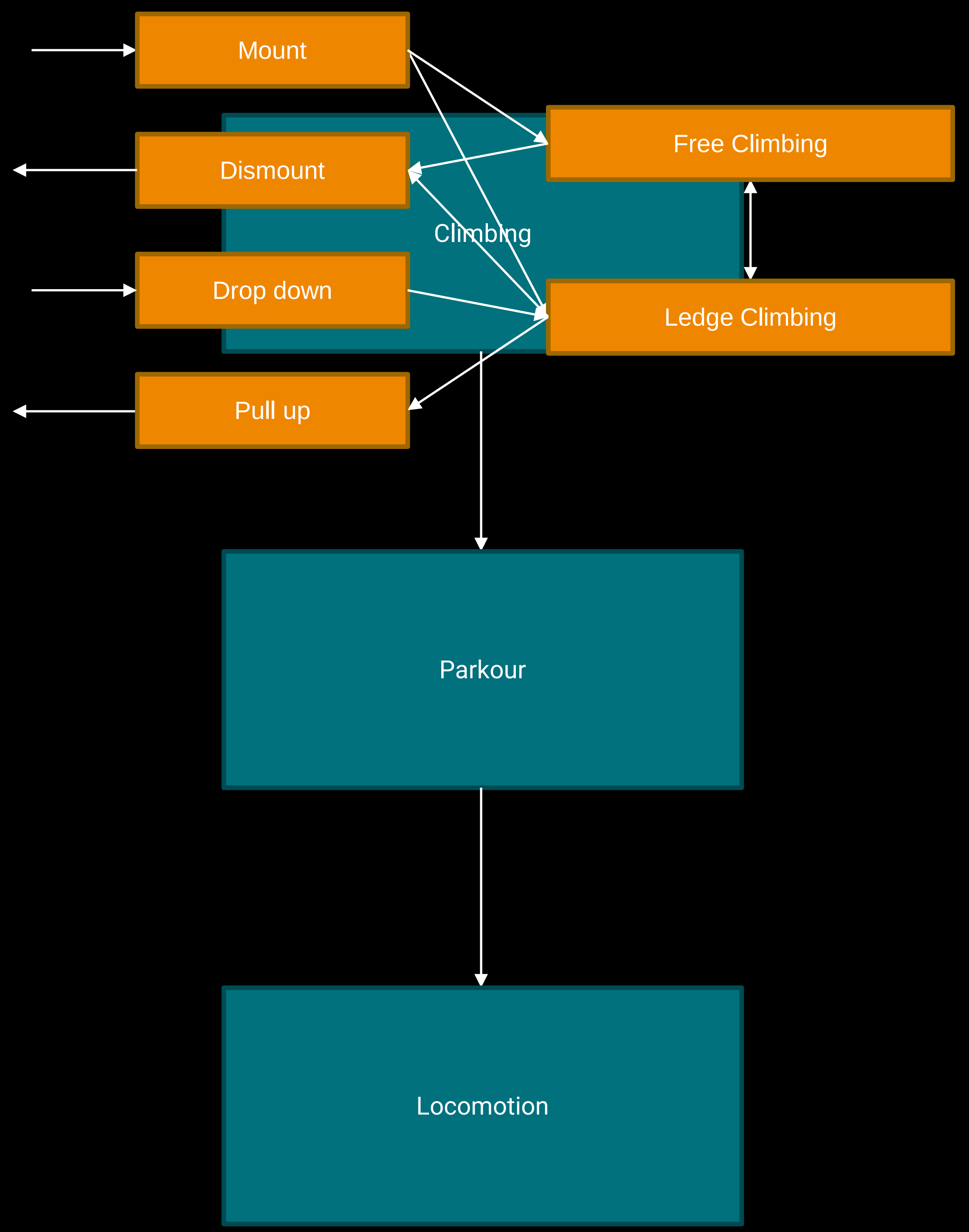
We generate a new predicted future trajectory based on the result



Pivot
Local
Collab
Account
Layers
Default
Hierarchy
Create
All
Kinematica_Siggraph
Root
Unit_Sword
Unit_Staff
Unit_Shield
Prefab_Unit_Red
_Scene
Geometry
Reflections
Room_Labyrynth
Collisions
Scene
Game
Asset Store
Display 1
Free Aspect
Scale
1x
Maximize On Play
Mute Audio
VSync
Stats
Gizmos
Inspector
Root
Static
Tag Joint
Layer
Transform
Position X 5.987 Y 0.0049 Z 19.704
Rotation X 0 Y 0 Z 0
Scale X 1 Y 1 Z 1
Rigidbody
Mass 1
Drag 0
Angular Drag 0
Use Gravity
Is Kinematic
Interpolate Interpolate
Collision Detection Discrete
Constraints
Info
Capsule Collider
Animator
Builder (Script)
Script Builder
Time Horizon 1
Sample Rate 30
Resource Name Binary.bin
Animation Library
Policies
Traversal Ability (Script)
Script TraversalAbility
Transition settings
Contact Threshold 0.15
Maximum Linear Errc 0.3
Maximum Angular Er. 25
Debug settings
Enable Debugging
Debug Index 1
Debug Pose Index 0
Fractional Time 0
Locomotion Ability (Script)
Script LocomotionAbility
Prediction settings
Desired Speed Slow 3.9
Desired Speed Fast 5.5
Velocity Percentage 1
Forward Percentage 1
Debug setting
Display Desired Traj
Display Anchor Traje
Display Candidate Tr
Player Character (Script)
Script PlayerCharacter
Resource Name Binary.bin
Root Root (Transform)
Blend Duration 0.3
Auto Generate Lighting Off

Climbing

Climbing States



Climbing is the most complex state

Several internal states

Transitions

Multiple movement types

unity


Pivot
Local
Collab
Account
Layers
Empty
Game
Display 1
Free Aspect
Scale
1x
Maximize On Play
Mute Audio
Stats
Gizmos
Mount
Dismount
Drop down
Pull up
Free Climbing
Ledge Climbing
Bake paused in play mode


Pivot
Local
Collab
Account
Layers
Empty
Game
Display 1
Free Aspect
Scale
1x
Maximize On Play
Mute Audio
Stats
Gizmos
Mount
Dismount
Drop down
Pull up
Free Climbing
Ledge Climbing
Bake paused in play mode


Pivot
Local
Collab
Account
Layers
Empty
Game
Display 1
Free Aspect
Scale
1x
Maximize On Play
Mute Audio
Stats
Gizmos
Mount
Dismount
Drop down
Pull up
Free Climbing
Ledge Climbing
Bake paused in play mode

Game
Display 1
Free Aspect
Scale
1x
Maximize On Play
Mute Audio
Stats
Gizmos
Pivot
Local
Collab
Account
Layers
Empty
Mount
Dismount
Drop down
Pull up
Free Climbing
Ledge Climbing
Bake paused in play mode

Pivot
Local
Collab
Account
Layers
EmptyWith
Scene
Game
Display 1
Free Aspect
Scale
1x
Maximize On Play
Mute Audio
Stats
Gizmos
Mount
Dismount
Drop down
Pull up
Free Climbing
Ledge Climbing
Inspector
Navigation
Policy settings
Linear Error Multiplier 1
Linear Velocity Error Multipl 1
Angular Error Multiplier 2
Time Between Jumps 1
Debug settings
Debug Index 0
Pose Index 0
Traversal Ability (Script)
Script TraversalAbility
Transition settings
Contact Threshold 0.15
Maximum Linear Error 0.5
Maximum Angular Error 45
Debug settings
Debug Index 0
Pose Index 0
Locomotion Ability (Script)
Script LocomotionAbility
Prediction settings
Desired Speed Slow 3.9
Desired Speed Fast 5.5
Velocity Factor 1.92
Direction Bias 0.16
Policy settings
Linear Error Multiplier 1
Linear Velocity Error Multipl 1.5
Angular Error Multiplier 1.4
Time Between Jumps 0.15
Animator
Motion Synthesizer (Script)
Script MotionSynthesizer
Resource Name Animations/Binary.bin
Root Root (Transform)
Blend Duration 0.3
Record & Rewind
Enable Recording
Recorder State Record
Debug setting
Display Trajectory
Display Current
Current Time Offset 0
Display Candidate
Candidate Time Offset 0
Display Pose Error
Display Policy Error
Debug Policy
Candidate Index 0
Bake paused in play mode


Scene
Game
Display 1
Free Aspect
Scale
1x
Maximize On Play
Mute Audio
Stats
Gizmos
Pivot
Local
Collab
Account
Layers
EmptyWith
Mount
Dismount
Drop down
Pull up
Free Climbing
Ledge Climbing
Inspector
Navigation
Policy settings
Linear Error Multiplier 1
Linear Velocity Error Multipl 1
Angular Error Multiplier 2
Time Between Jumps 1
Debug settings
Debug Index 0
Pose Index 0
Traversal Ability (Script)
Script TraversalAbility
Transition settings
Contact Threshold 0.15
Maximum Linear Error 0.5
Maximum Angular Error 45
Debug settings
Debug Index 0
Pose Index 0
Locomotion Ability (Script)
Script LocomotionAbility
Prediction settings
Desired Speed Slow 3.9
Desired Speed Fast 5.5
Velocity Factor 1.92
Direction Bias 0.16
Policy settings
Linear Error Multiplier 1
Linear Velocity Error Multipl 1.5
Angular Error Multiplier 1.4
Time Between Jumps 0.15
Animator
Motion Synthesizer (Script)
Script MotionSynthesizer
Resource Name Animations/Binary.bin
Root Root (Transform)
Blend Duration 0.3
Record & Rewind
Enable Recording
Recorder State Record
Debug setting
Display Trajectory
Display Current
Current Time Offset 0
Display Candidate
Candidate Time Offset 0
Display Pose Error
Display Policy Error
Debug Policy
Candidate Index 0
Bake paused in play mode

Ledge Climbing Model

Ledge geometry gets constructed on-the-fly

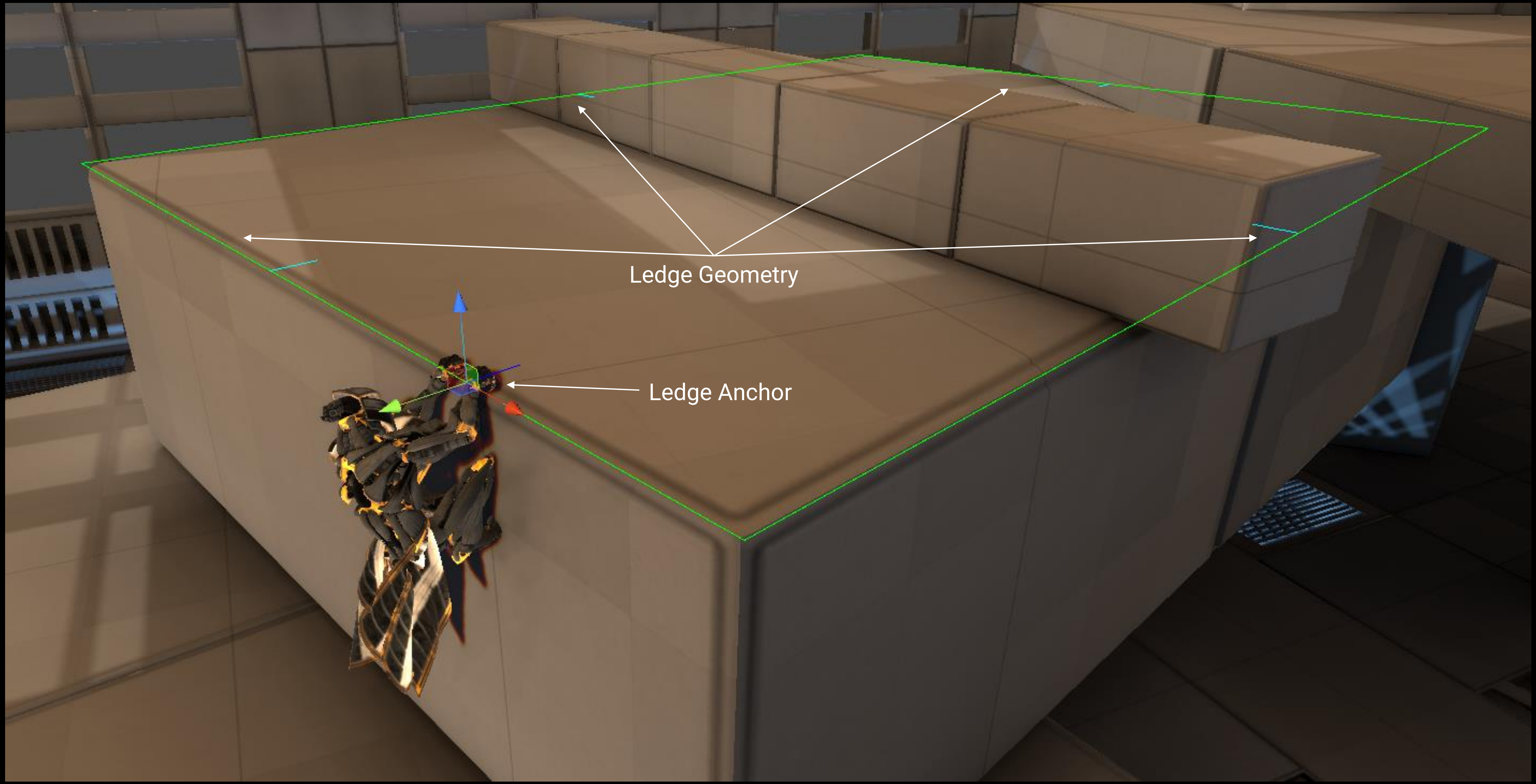
Ledge anchor is (line index, fraction)

Ledge anchor can be advanced based on distance

Ledge anchor can be constructed from world space position

Full predictive model

Snapshot() / Move() / Rewind()



Free Climbing Model

Wall geometry gets constructed on-the-fly

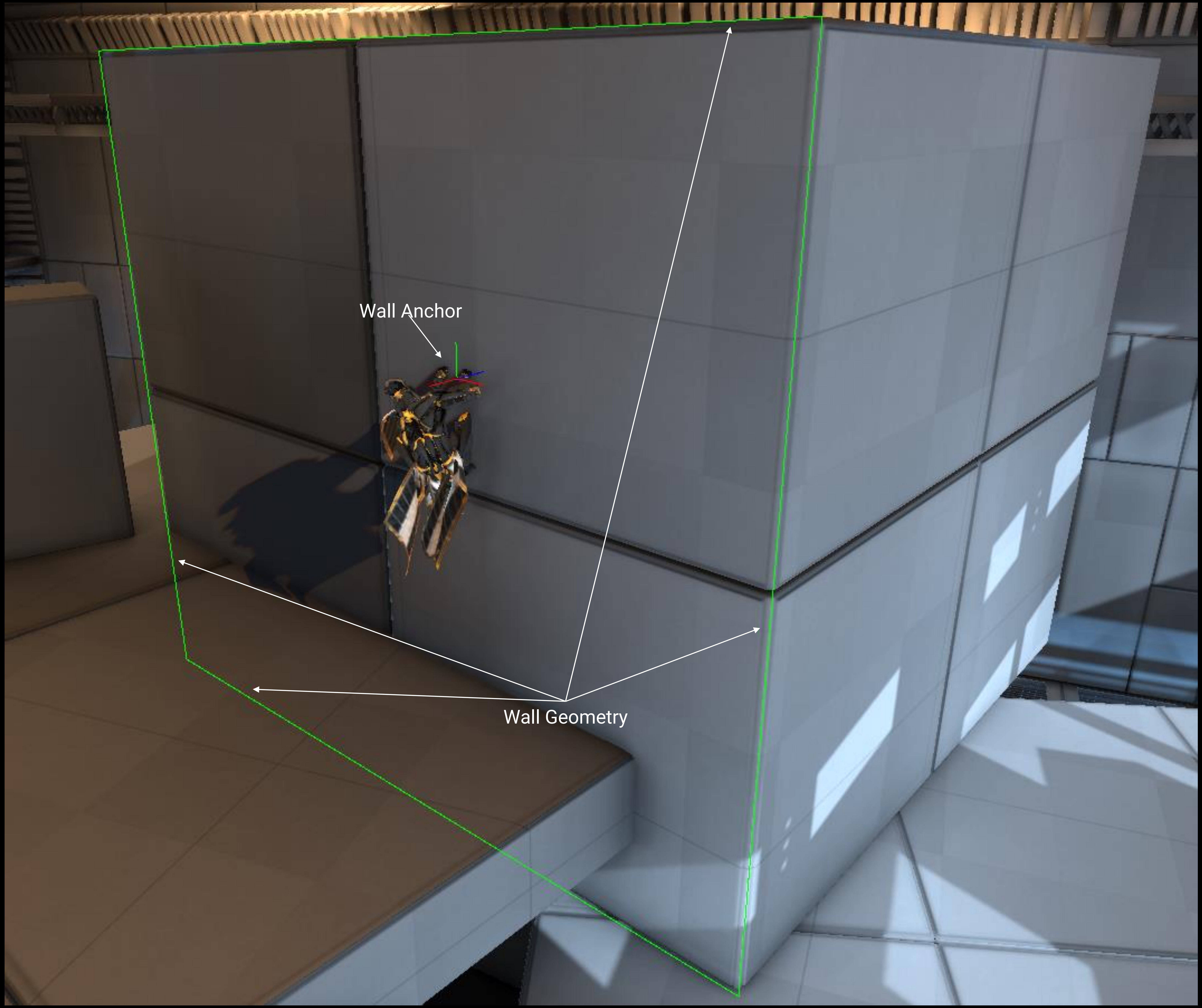
Wall anchor is (Normalized UV coordinate)

Wall anchor can be moved inside geometry bounds (2d displacement vector)

Wall anchor can be constructed from world space position

Full predictive model

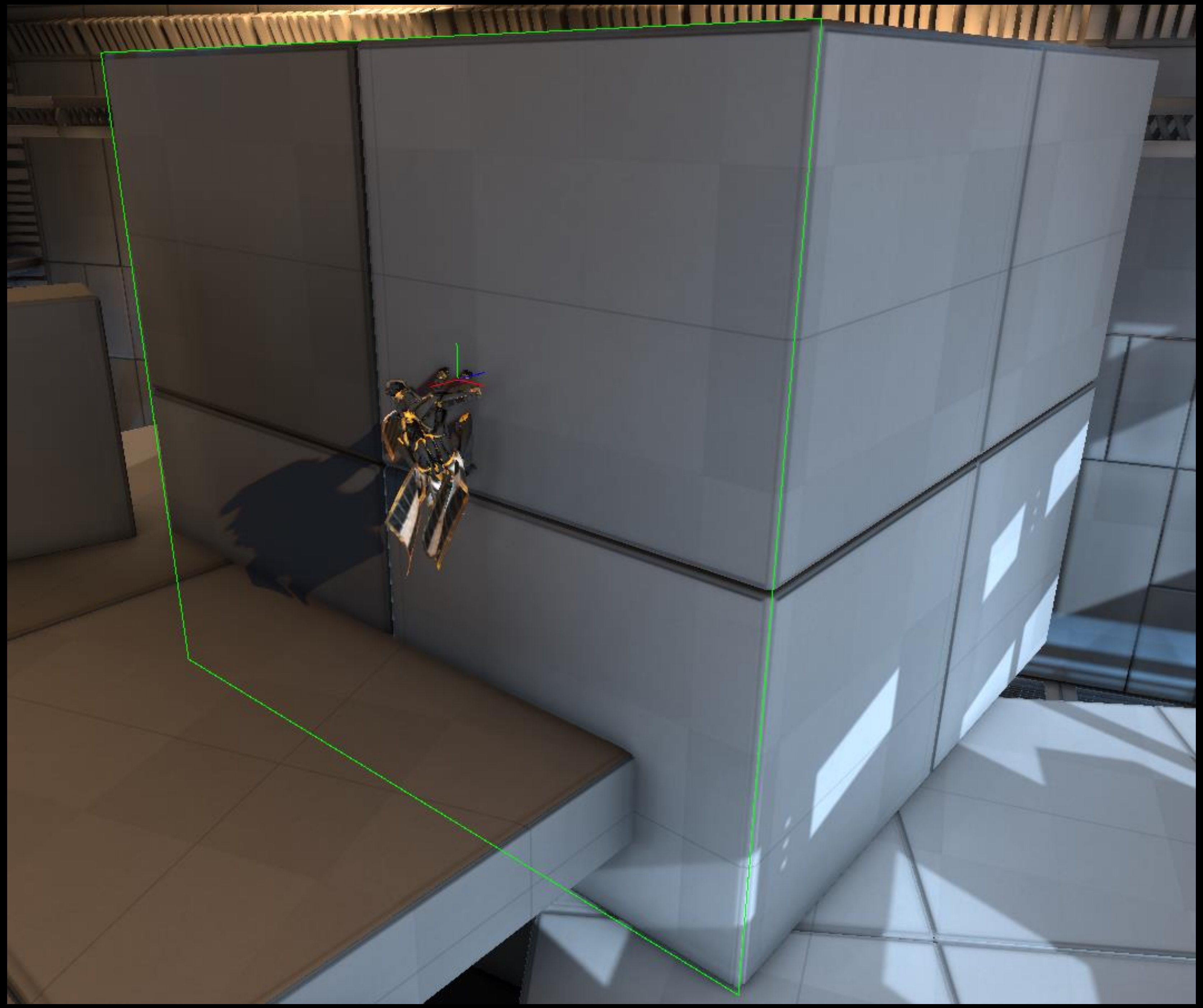
Snapshot() / Move() / Rewind()

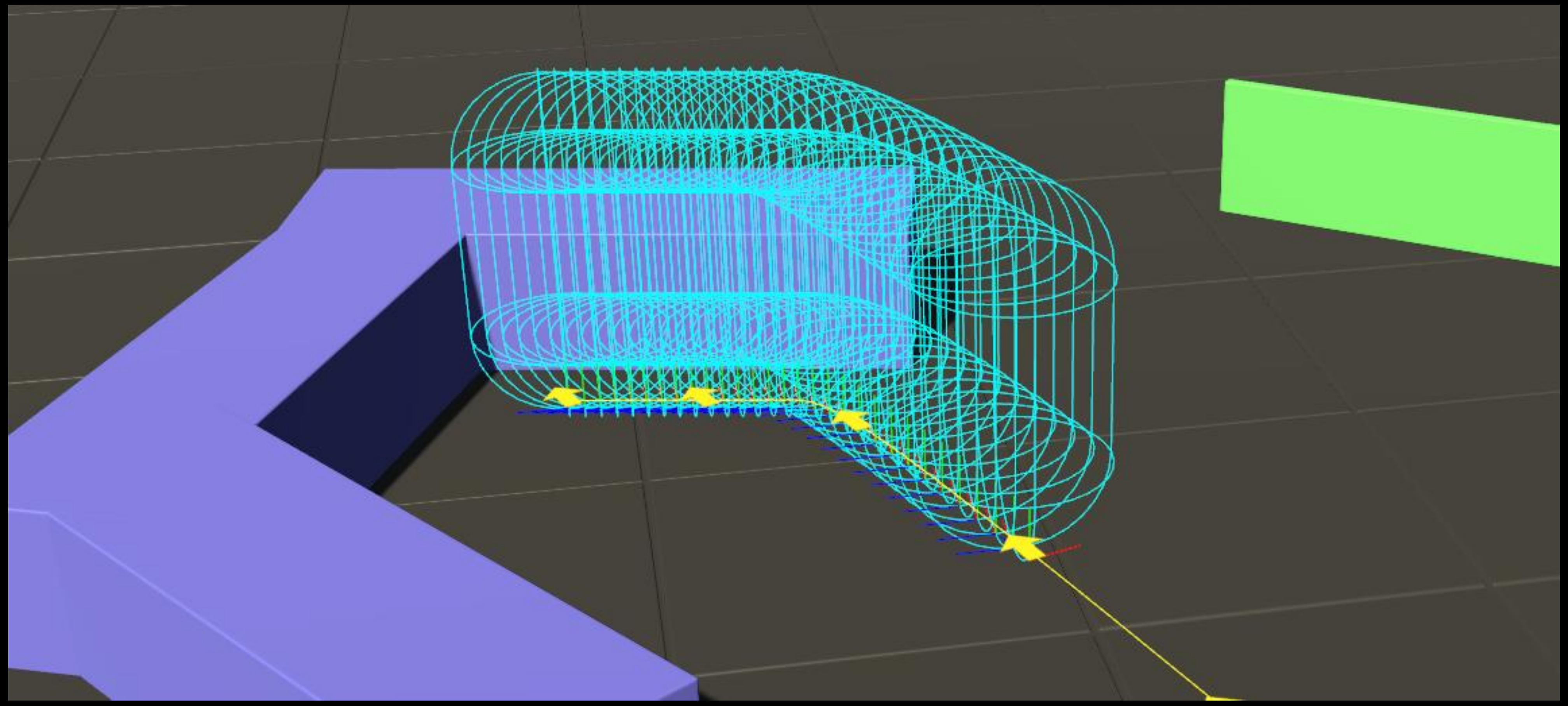


World Model

It is important to note that this is an unavoidable complexity for any non-trivial character navigation

Any game will require this kind of structure in one form or another





unity

Q & A

Machine Learning for Motion Synthesis and Character Control in Games

Michael Buttner, Unity Labs
michaelbu@unity3d.com