UBISOFT

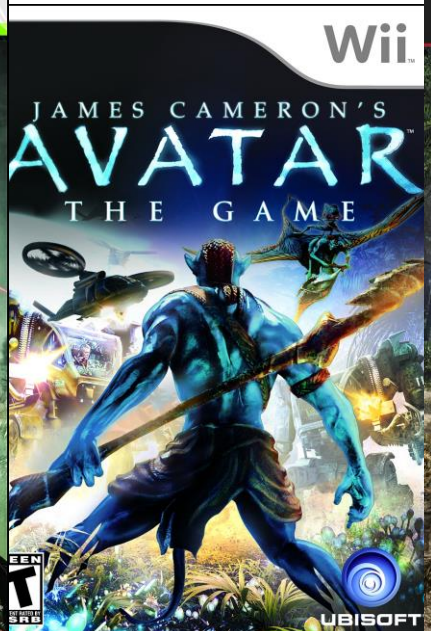
Motion Matching

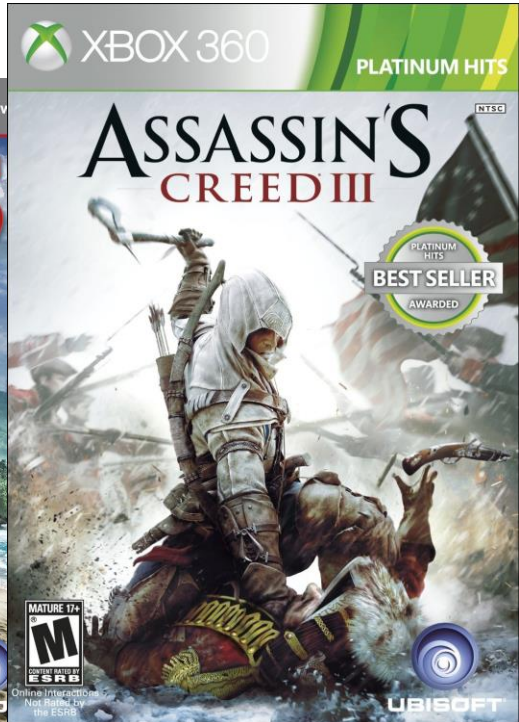
road to next-gen animation

Simon Clavet
Ubisoft Montreal

Who is this guy

animation programmer at Ubisoft Montreal since 2005






FOR
HONOR


Animation Goals
Precise Gameplay
Believable Animation

This Talk:

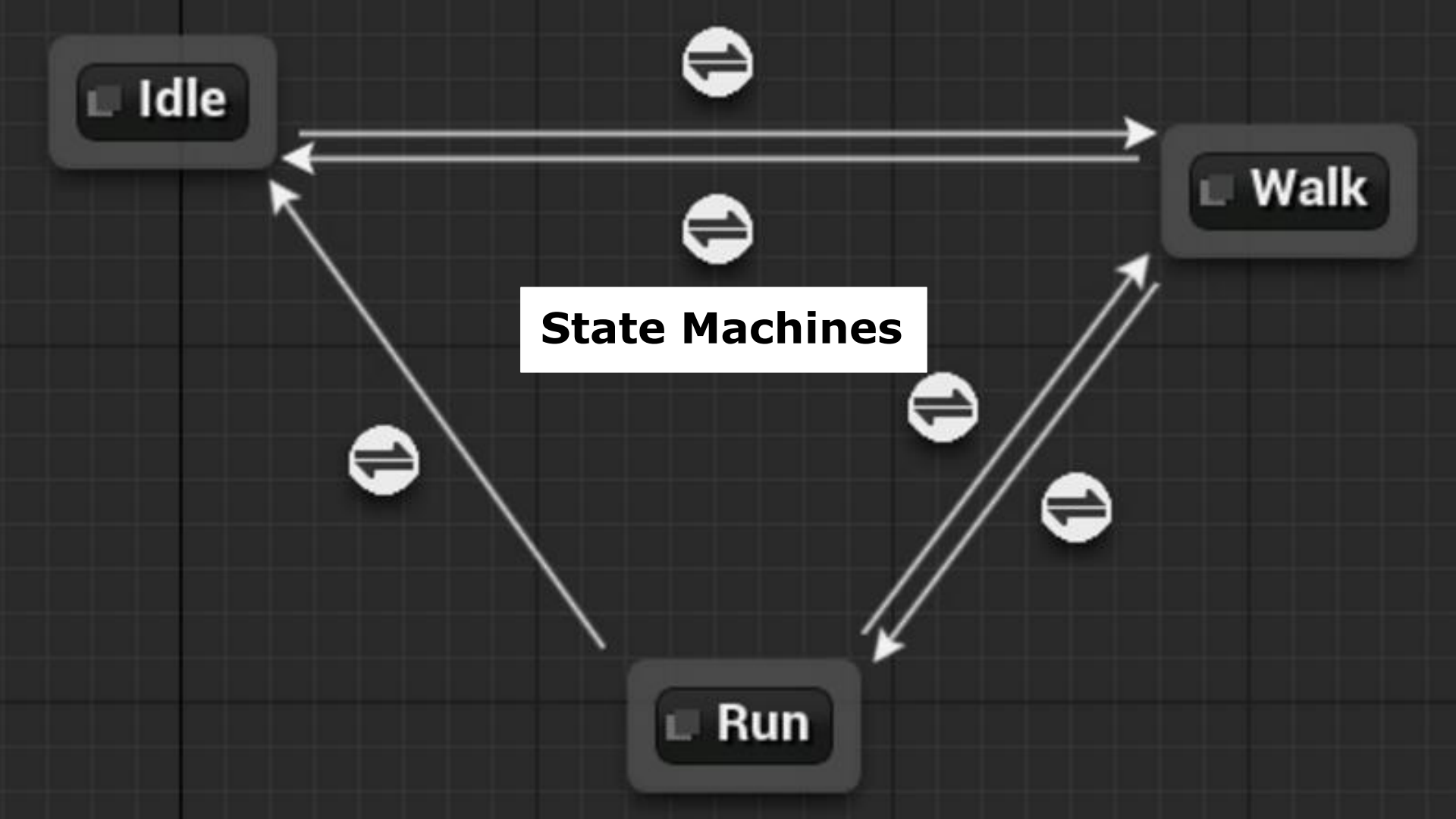
- Little history of animation systems
- Motion Matching
- Workflow
- Procedural Touchups

On the first day,

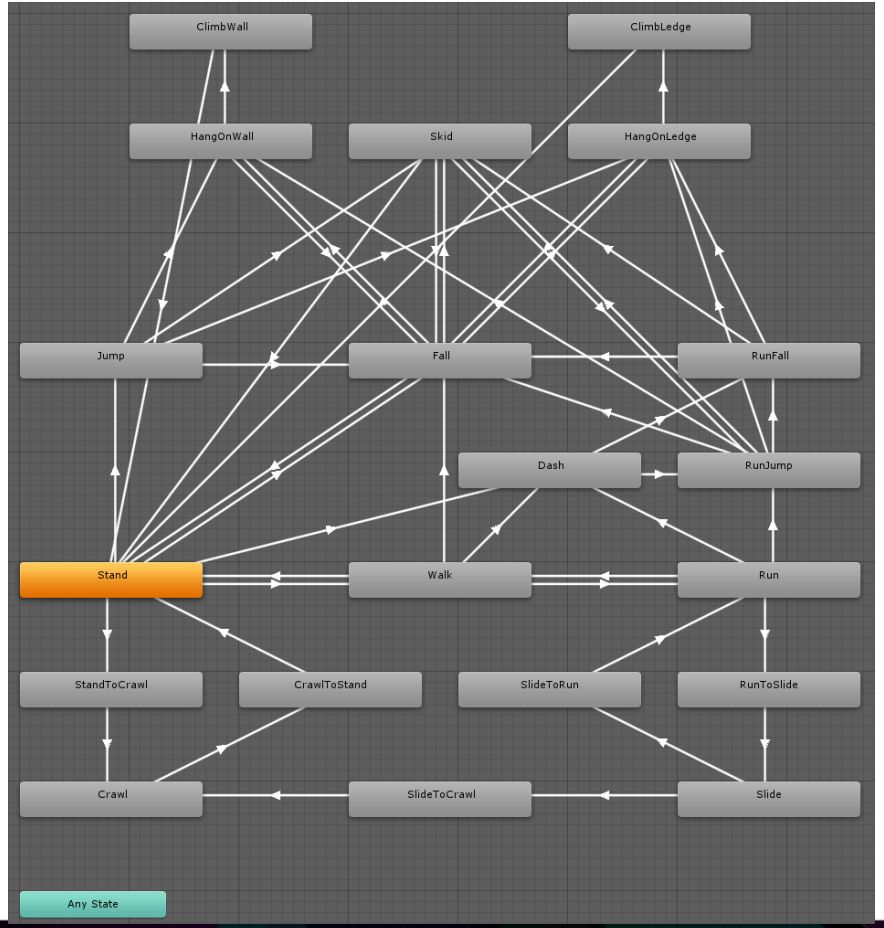
God created the function PlayAnim()

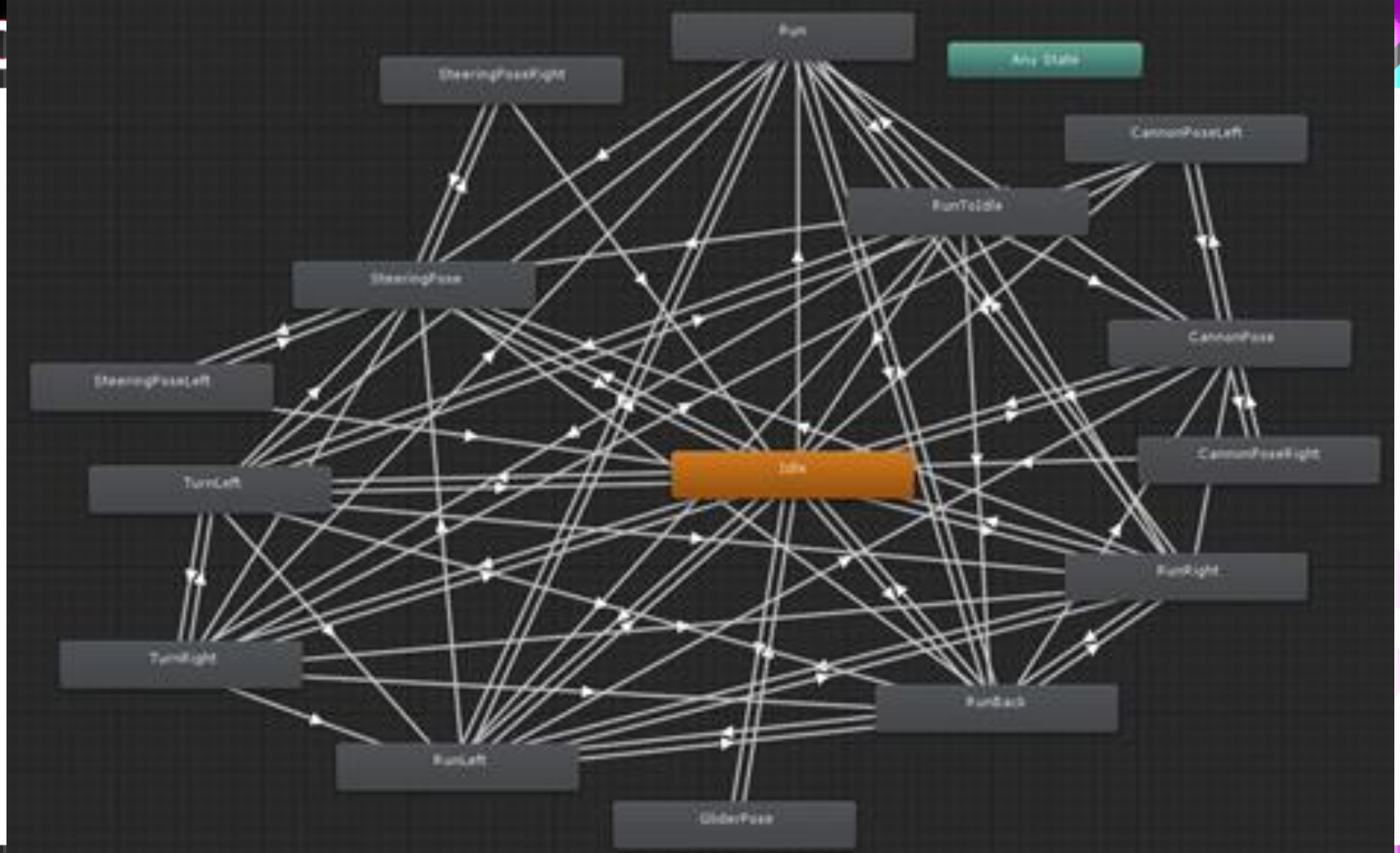
```
PlayAnim(RunAnimation);
```

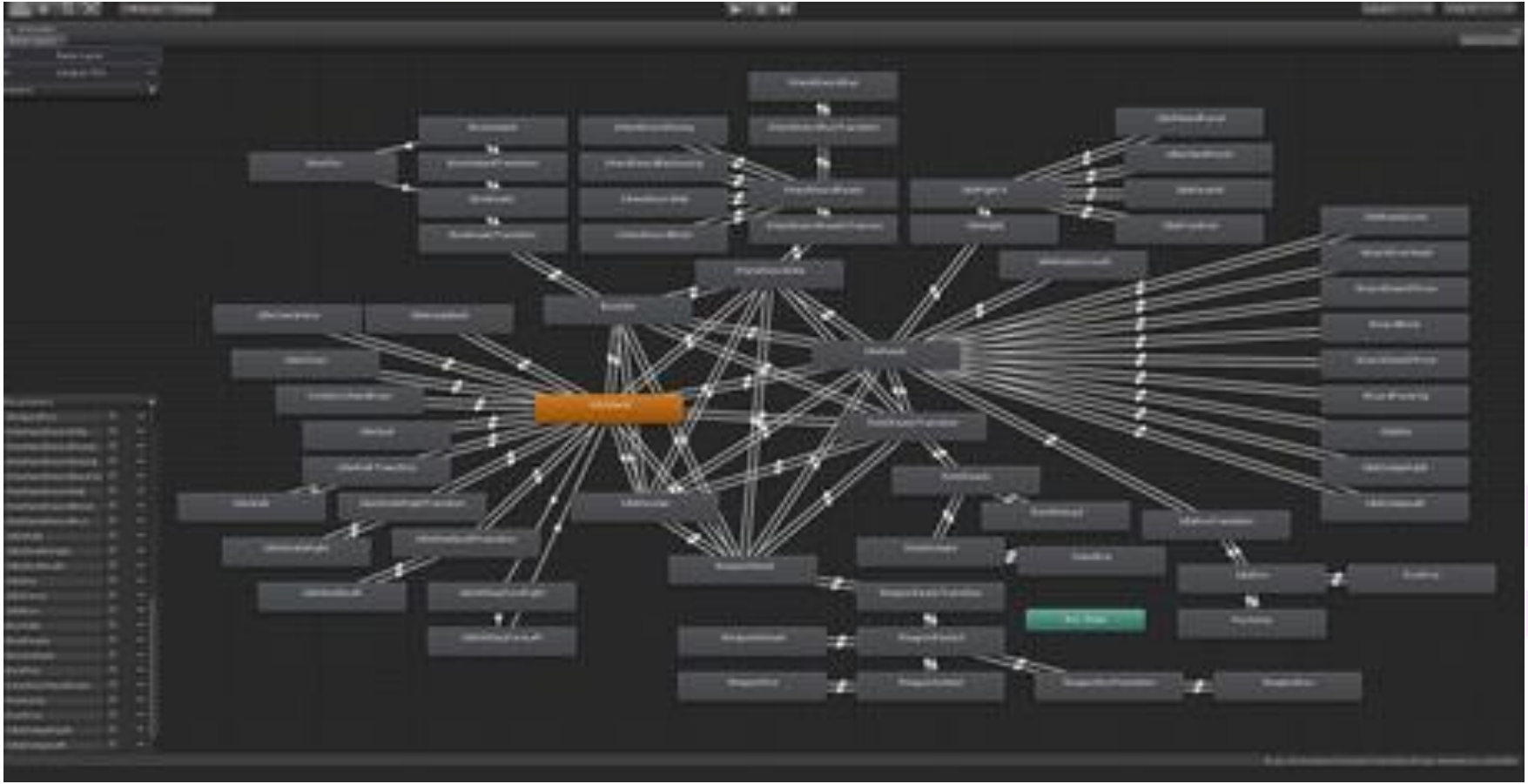
```
// start
if (!walking && wantToWalk)
{
    PlayAnim(StartAnim);
    walking = true;
}

// walk loop
if (IsPlaying(StartAnim) && IsAtEndOfAnim())
{
    PlayAnim(WalkLoopAnim);
}

// stop
if (walking && !wantToWalk)
{
    PlayAnim(StopAnim);
    walking = false;
}
```



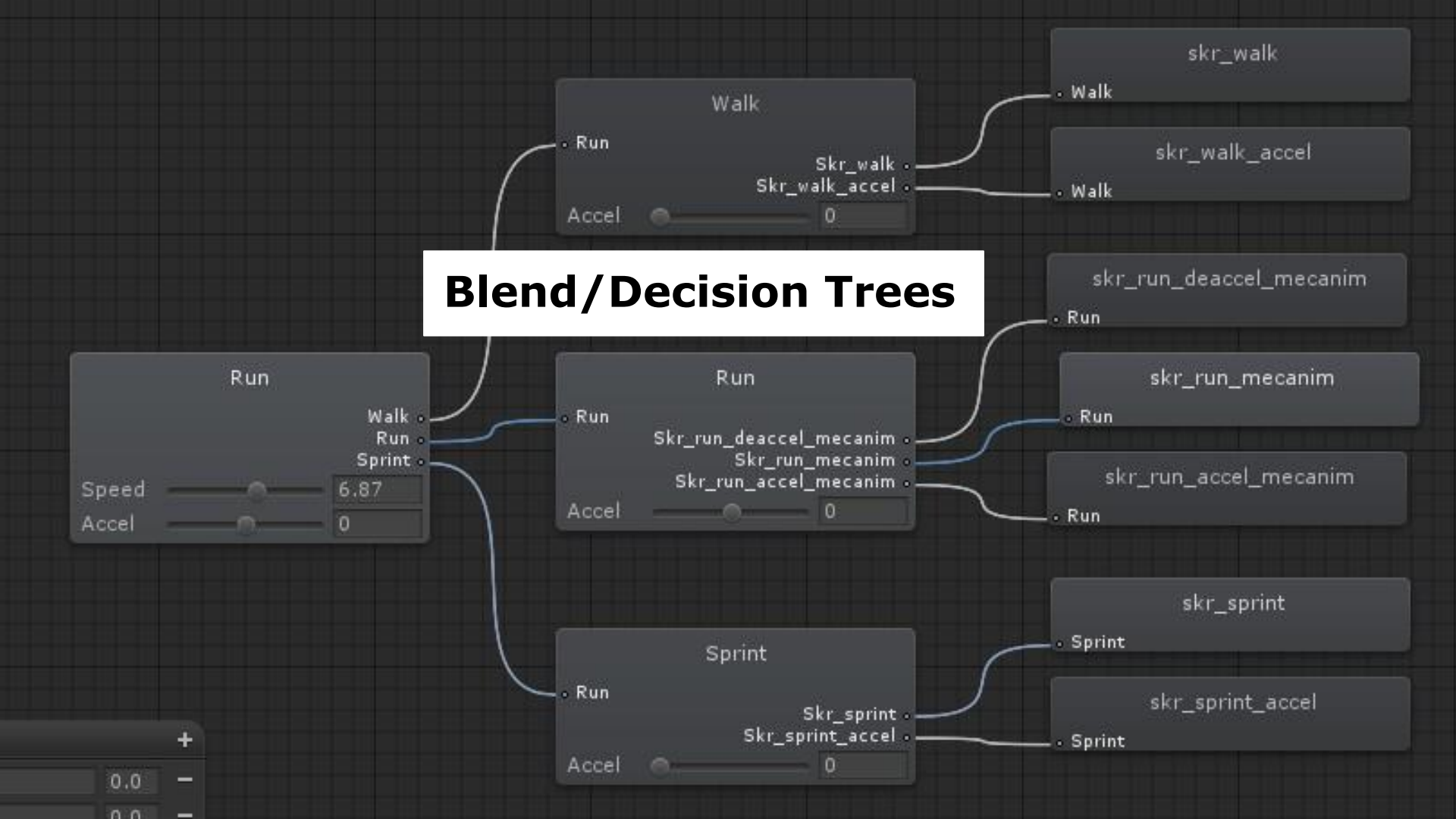
State Machines

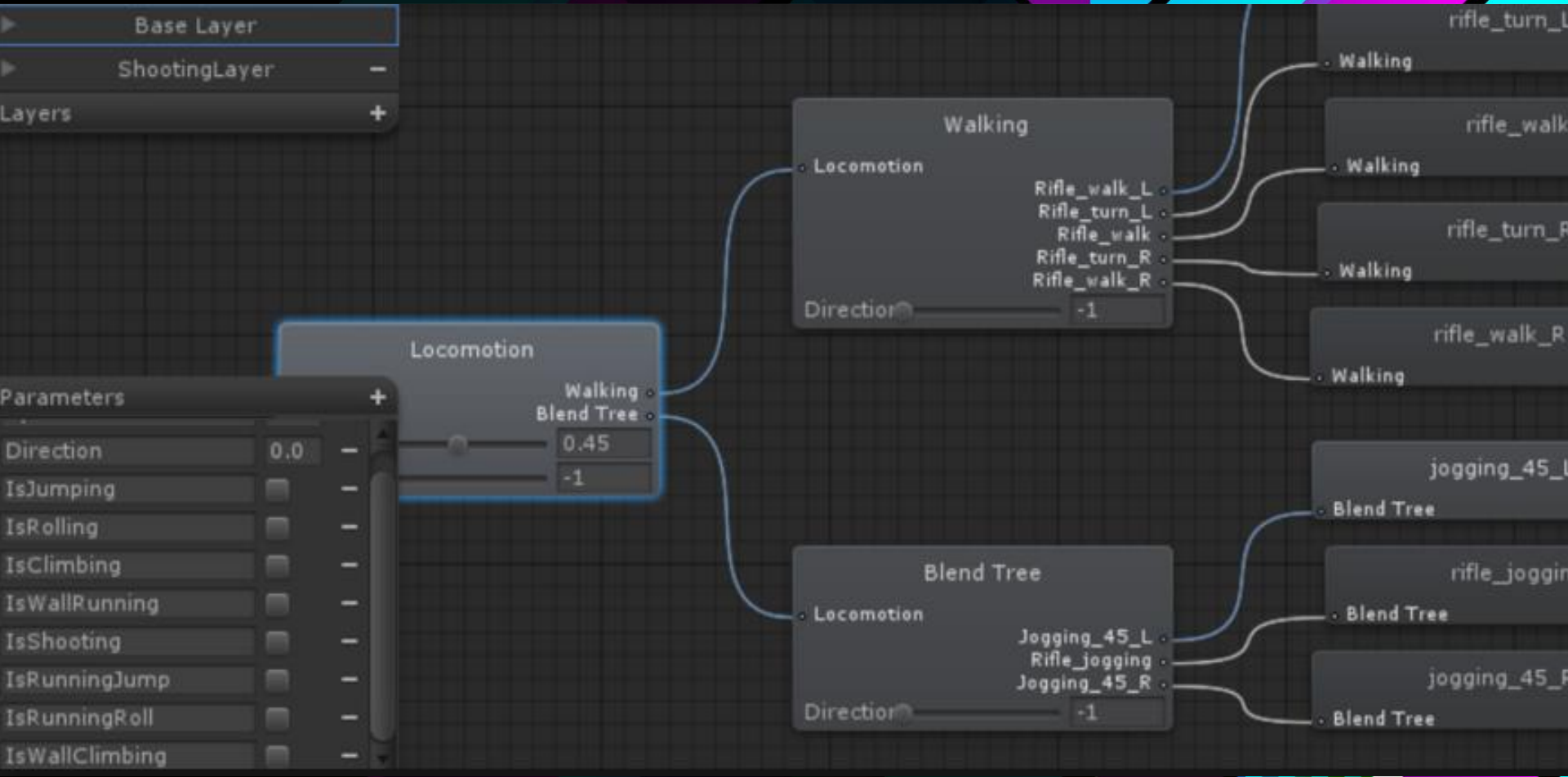

ClimbWall
ClimbLedge
HangOnWall
Skid
HangOnLedge
Jump
Fall
RunFall
Dash
RunJump
Stand
Walk
Run
StandToCrawl
CrawlToStand
SlideToRun
RunToSlide
Crawl
SlideToCrawl
Slide
Any State

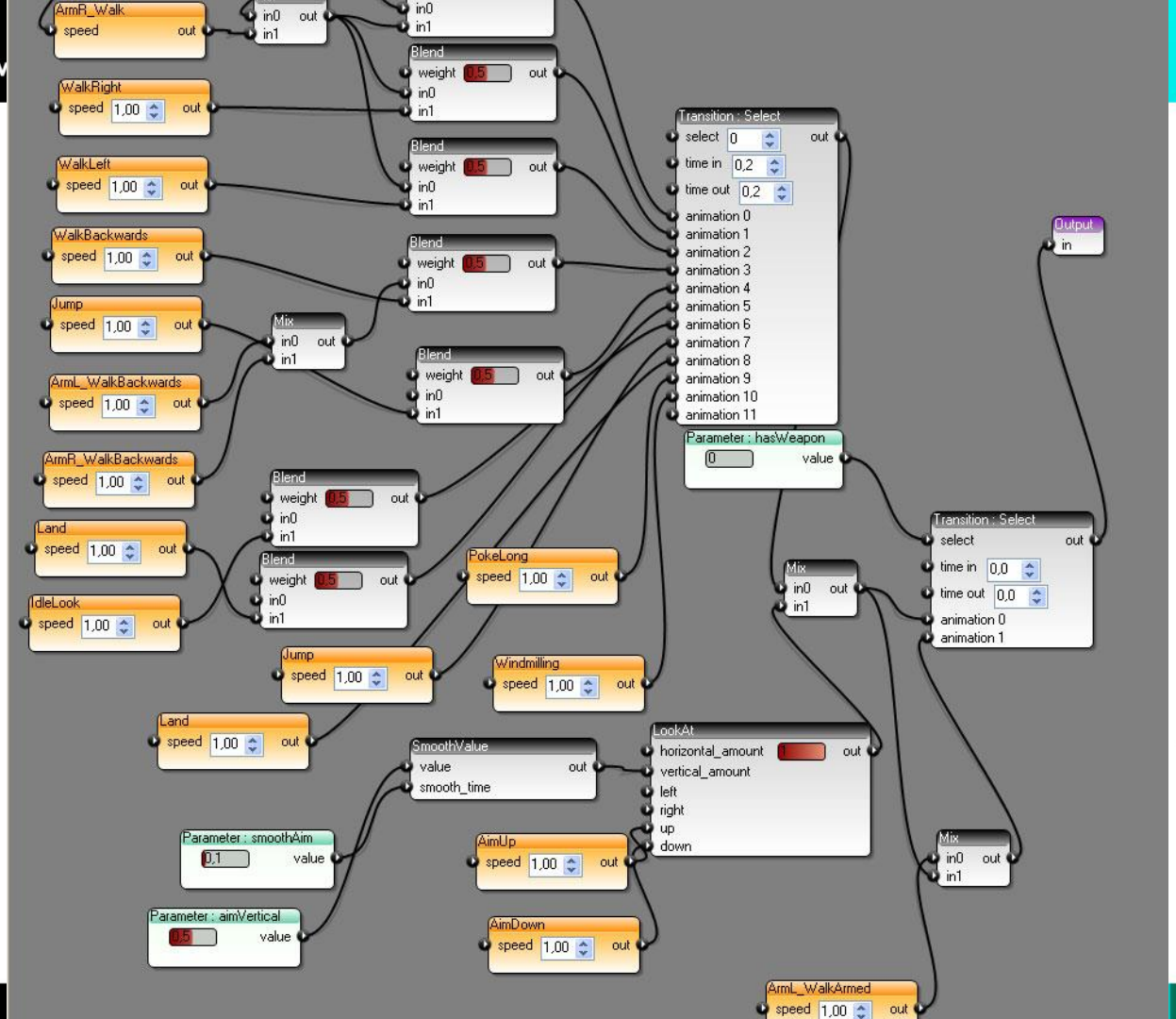

Run
Any State
SteeringPoseRight
CannonPoseLeft
RunToIdle
SteeringPose
CannonPose
SteeringPoseLeft
CannonPoseRight
Idle
TurnLeft
RunRight
TurnRight
RunBack
RunLeft
GliderPose



```
if (speed > 3.0f)
{
    PlayAnim(RunAnim);
}
else if (speed > 0.0f)
{
    PlayAnim(WalkAnim);
}
else
{
    PlayAnim(IdleAnim);
}
```


Blend/Decision Trees
skr_walk
Walk
Walk
Run
Skr_walk
Skr_walk_accel
Accel
0
skr_walk_accel
Walk
skr_run_deaccel_mecanim
Run
Run
Walk
Run
Sprint
Speed
6.87
Accel
0
Run
Run
Skr_run_deaccel_mecanim
Skr_run_mecanim
Skr_run_accel_mecanim
Accel
0
skr_run_mecanim
Run
skr_run_accel_mecanim
Run
skr_sprint
Sprint
Sprint
Run
Skr_sprint
Skr_sprint_accel
Accel
0
skr_sprint_accel
Sprint
0.0

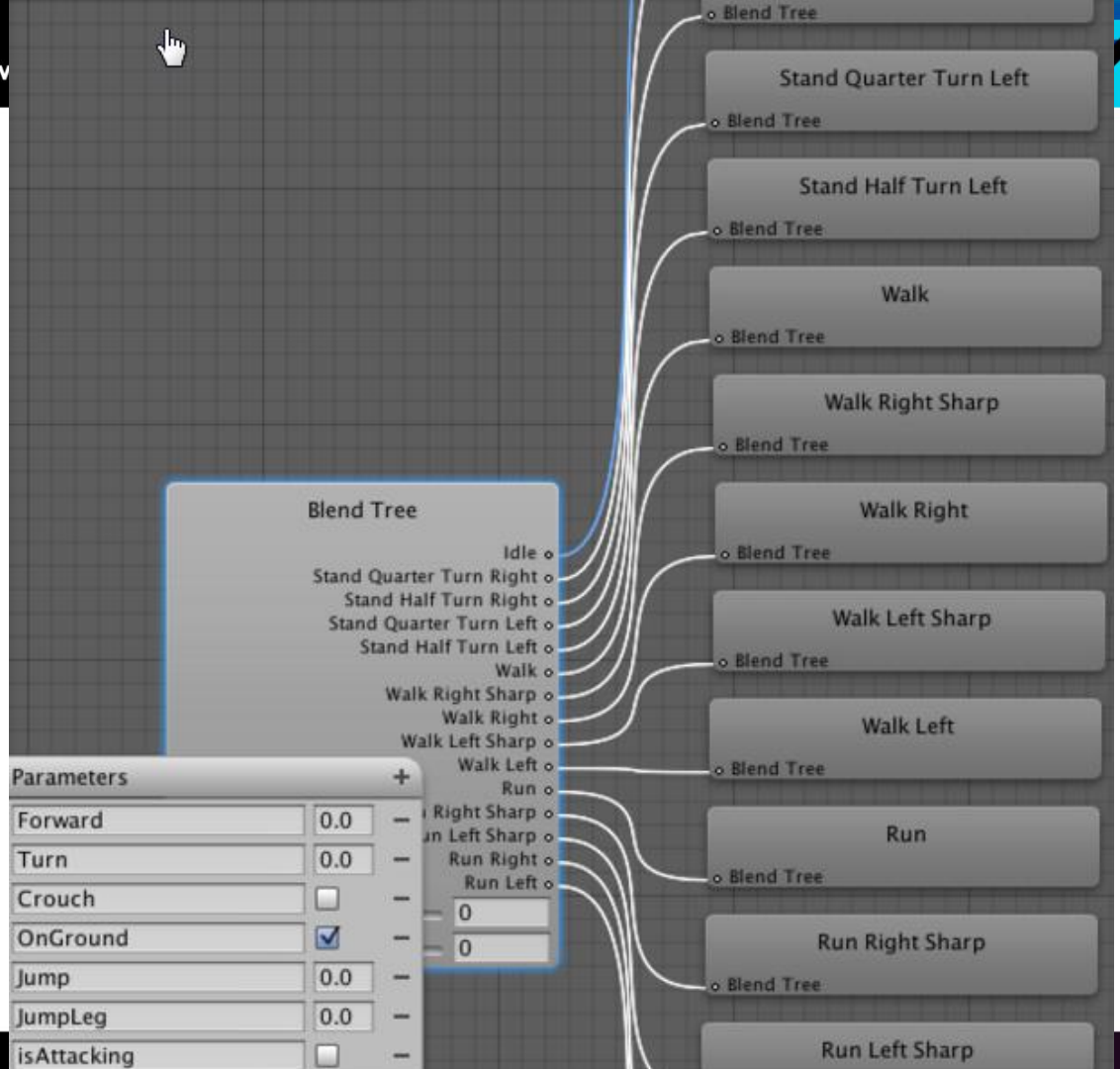

Base Layer
ShootingLayer
Layers
Parameters
Direction 0.0
IsJumping
IsRolling
IsClimbing
IsWallRunning
IsShooting
IsRunningJump
IsRunningRoll
IsWallClimbing
Locomotion
Walking
Blend Tree
0.45
-1
Walking
Locomotion
Rifle_walk_L
Rifle_turn_L
Rifle_walk
Rifle_turn_R
Rifle_walk_R
Direction
-1
rifle_turn_L
Walking
rifle_walk
Walking
rifle_turn_R
Walking
rifle_walk_R
Walking
Blend Tree
Locomotion
Jogging_45_L
Rifle_jogging
Jogging_45_R
Direction
-1
jogging_45_L
Blend Tree
rifle_joggin
Blend Tree
jogging_45_R
Blend Tree


ArmR_Walk
speed
out
in0
in1
WalkRight
speed 1,00
WalkLeft
speed 1,00
WalkBackwards
speed 1,00
Jump
speed 1,00
ArmL_WalkBackwards
speed 1,00
ArmR_WalkBackwards
speed 1,00
Land
speed 1,00
IdleLook
speed 1,00
Mix
Blend
weight 0,5
Transition : Select
select 0
time in 0,2
time out 0,2
animation 0
animation 1
animation 2
animation 3
animation 4
animation 5
animation 6
animation 7
animation 8
animation 9
animation 10
animation 11
Output
in
Parameter : hasWeapon
0
value
PokeLong
speed 1,00
Windmilling
speed 1,00
Transition : Select
time in 0,0
time out 0,0
animation 0
animation 1
LookAt
horizontal_amount 1
vertical_amount
left
right
up
down
SmoothValue
smooth_time
Parameter : smoothAim
0,1
Parameter : aimVertical
0,5
AimUp
AimDown
ArmL_WalkArmed

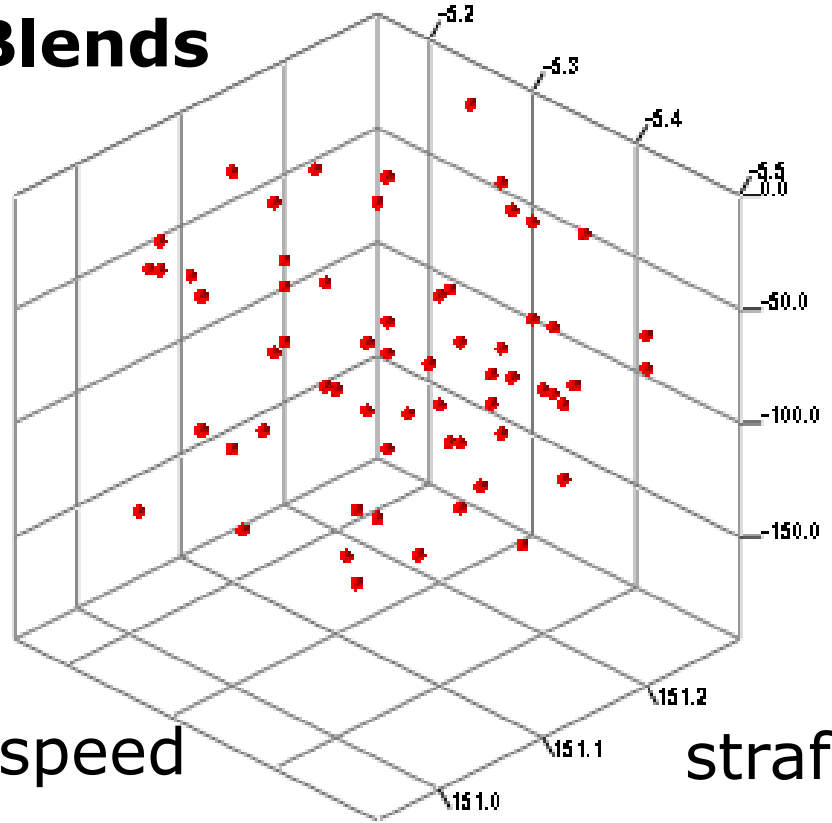
Question 1

Where would you put this animation in your structure?

Start-Strafe90-TurnOnSpot45-Stop


Blend Tree
Stand Quarter Turn Left
Blend Tree
Stand Half Turn Left
Blend Tree
Walk
Blend Tree
Walk Right Sharp
Blend Tree
Walk Right
Blend Tree
Walk Left Sharp
Blend Tree
Walk Left
Blend Tree
Run
Blend Tree
Run Right Sharp
Blend Tree
Run Left Sharp
Blend Tree
Blend Tree
Idle
Stand Quarter Turn Right
Stand Half Turn Right
Stand Quarter Turn Left
Stand Half Turn Left
Walk
Walk Right Sharp
Walk Right
Walk Left Sharp
Walk Left
Run
Run Right
Run Left
0
0
Parameters
Forward 0.0
Turn 0.0
Crouch
OnGround
Jump 0.0
JumpLeg 0.0
isAttacking

Parametric Blends



slope

speed

strafe angle

Name	Speed	SightVelocityAngle	SlopeVelocityAngle	Banking	AwarenessContext	Stances	Bump
Hero_Nav_RunInField_5Msec	5	0	0	0	Tall Grass	Run	0
Hero_Nav_WalkFwd1Msec_BankRt	1	0	0	1	Default	Walk	0
Hero_Nav_WalkFwd1Msec_BankLt	1	0	0	-1	Default	Walk	0
Hero_Nav_MoCapNarrativeWalkSlow_1Msec	1	0	0	0	Default	Walk	0
Hero_Nav_JogSlow_3Msec	3	0	0	0	Default	Run	0
Hero_Nav_JogSlow_4Msec							
Hero_Jst_WalkFast_3Msec							
Hero_Fight_WalkFwd_BankR							
Hero_Fight_WalkFwd_BankL							
Hero_Nav_WalkSlope_30De							
Hero_Nav_TransitionLoop_R							
Hero_Nav_RunBump_LtSide							
Hero_Nav_RunBump_RtSide							
Hero_Nav_RunFwdDirection							
Hero_Nav_RunFwdDirection							
Hero_Agg_SprintFwd_BankR							
Hero_Jst_walkAggresive_2M							
Hero_Nav_Run_5m_NewSty							
Hero_Agg_SprintFwd_BankL							
Hero_Agg_SprintBump8m_L							
Hero_Agg_SprintBump8m_R							
Hero_Agg_SprintBump6m_L							
Hero_Agg_SprintBump6m_R							
NPC_Walk_Slow							
Hero_Nav_Run_5m_NewSty							
Hero_Nav_Sprint_stop_6m							
Hero_Nav_Sprint_7m_NewS							
Hero_Jst_walkAggresive_2M							
Hero_Nav_Run_Slope_30DegUp_5m	5	0	-30	0	Default	Sprint	0
Hero_Nav_Run_Slope_30DegDown_5m	5	0	30	0	Default	Sprint	0
Hero_Nav_Sprint_8m	8	0	0	0	Default	Run	0
Hero_sprint_10m	10	0	0	0	Default	Run	0
Hero_Nav_Run_5m_NewStyle_LongStride_20Fps	5	0	0	0	Default	Sprint	0

```
// a parametric-blend query
AnimQuery query;

// let's strafe up a slope
query.AddDesiredFeature("speed", 3.0f);
query.AddDesiredFeature("slope", 35.0f);
query.AddDesiredFeature("strafeAngle", 90.0f);

Array<float> blendFactors = ComputeBlendFactors(query);

SetBlendFactors(blendFactors);
```

Question 2

Is there a way to deal with loops and transitions in a uniform way?

To appear in Proceedings of SIGGRAPH '02

Motion Graphs

Lucas Kovar
University of Wisconsin-Madison

Michael Gleicher*
University of Wisconsin-Madison

Frédéric Pighin†
University of Southern California
Institute for Creative Technologies

***Unstructured* list of animations**

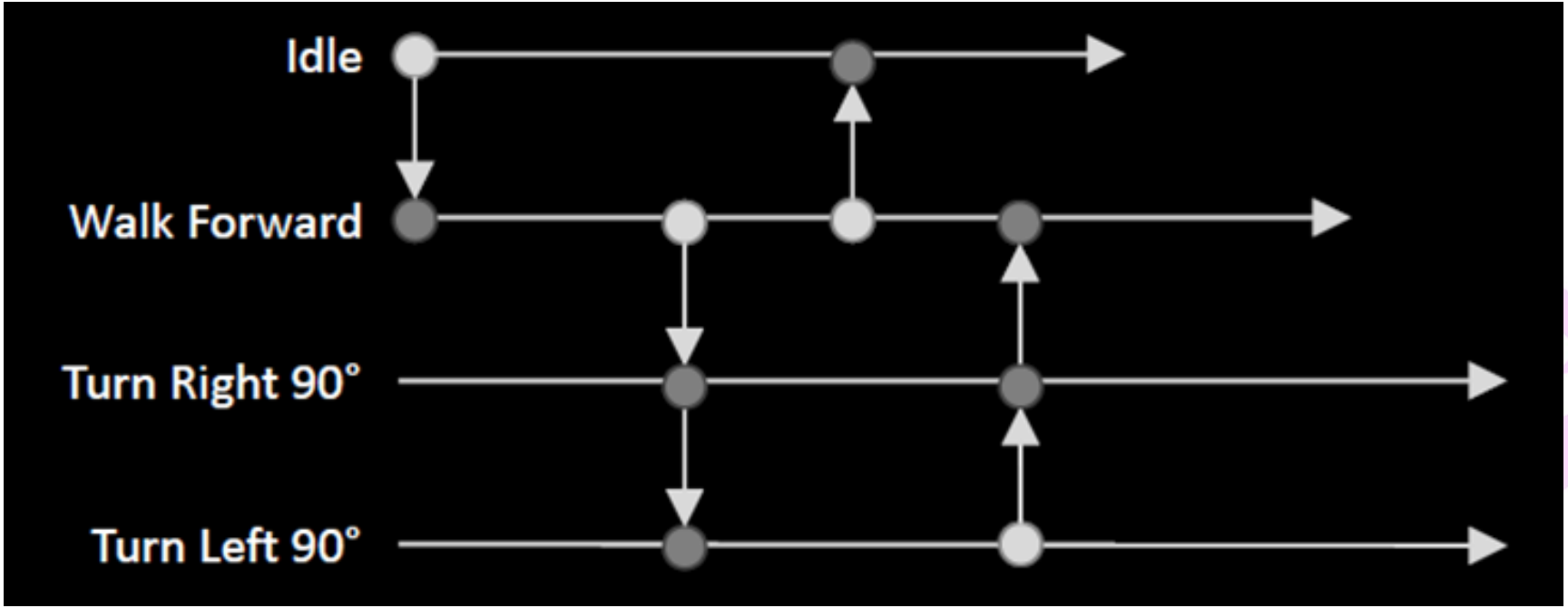
is desired, then often there is little that can be done
ire more data, a time-consuming and expensive pro-
articular is a problem for applications that require

Preprocess:
Figure out *when* you can *jump* to somewhere else

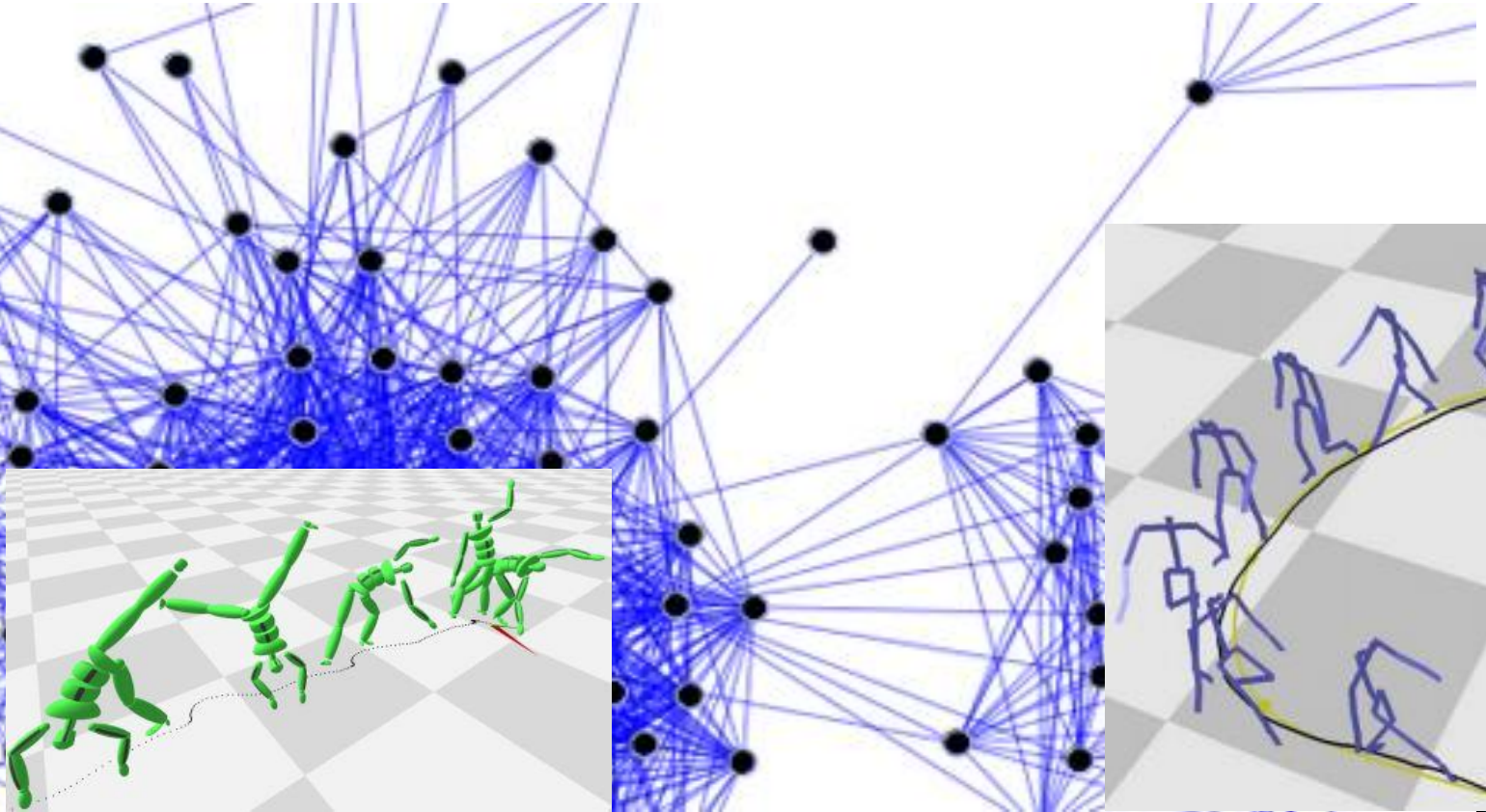
on the graph. We present a general framework for extracting particular graph walks that meet a user's specifications. We then show how this framework can be applied to the specific problem of gen-

without worrying about having a piece of motion data that contains the correct number of steps and travels in the right directions. We also need to be able to direct characters who can perform multiple

When reaching a transition point, decide where to go from a list of possibilities



A path on the graph gives us our animation



Question 3

How do we choose the next animation?

GDC 13

Reinforcement Learning based Character Locomotion in Hitman: Absolution

Michael Büttner
Technical Director - IO Interactive / Square Enix

GAME DEVELOPERS CONFERENCE 2013
SAN FRANCISCO, CA
MARCH 25-29, 2013
EXPO DATES: MARCH 27-29

For comfortable control, we need a dense graph

Motion Fields for Interac

Yongjoon Lee[1,2*] Kevin Wampler[1†] Gilbert
[1]University of Washington

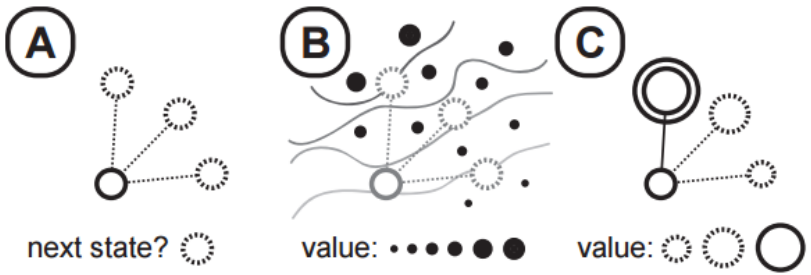


Figure 2: Action search using a value function.

Abstract

We propose a novel representation of motion data and control which gives characters highly agile responses to user input and allows a natural handling of arbitrary external disturbances. Our representation organizes samples of motion data into a high-dimensional generalization of a vector field which we call a *motion field*. Our run-time motion synthesis mechanism freely flows through the motion field in response to user commands.

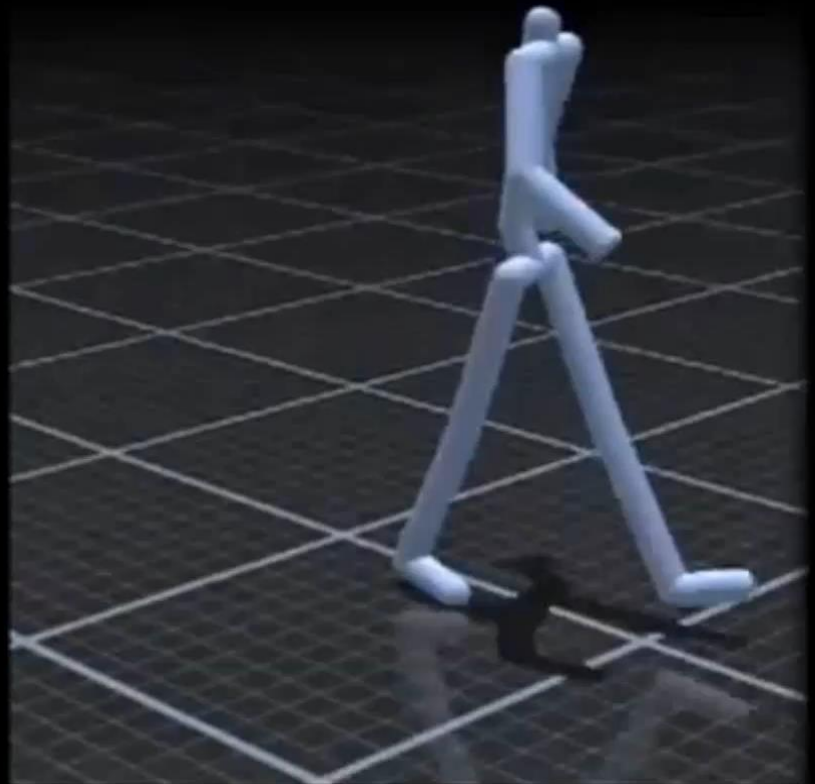
1 Introduction

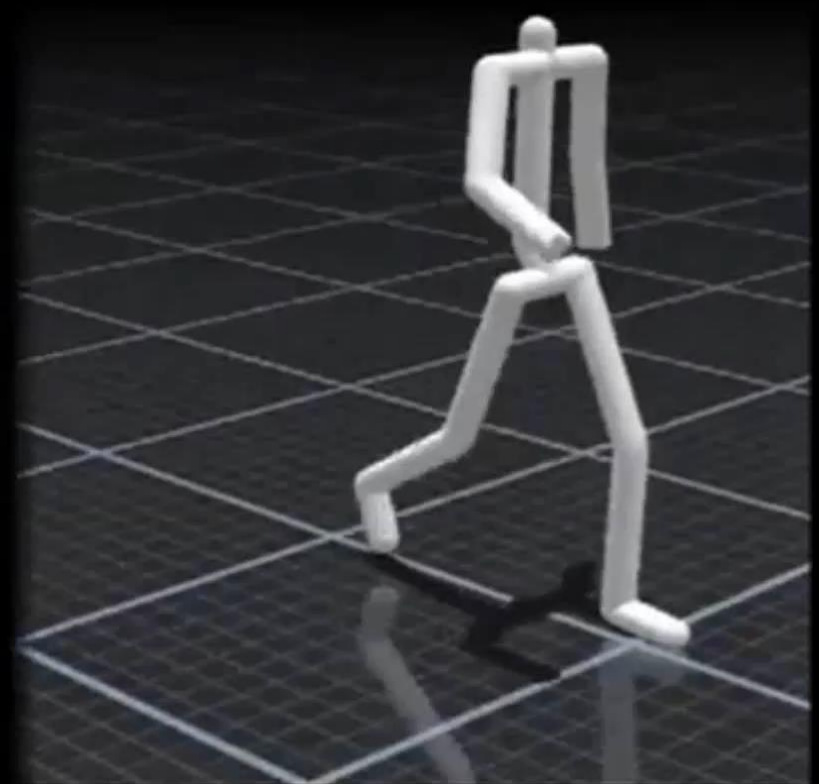
Human motion is a highly-varied and continuous phenomenon: it

continuous state representation with an optimal control framework. We find that this approach provides many advantages for character animation.

graph

field





The problem with Motion Fields:

the equations are scary…

Let’s just use the main idea of the paper:

We can jump to any other frame whenever we want

Question 4


How to choose the '*correct*' start time in a stop animation?
pose matching?
velocity matching?
precise end position matching?
a mix of all of the above?

Introducing *Motion Matching*

A ridiculously brute-force approach to animation selection

Algorithm:

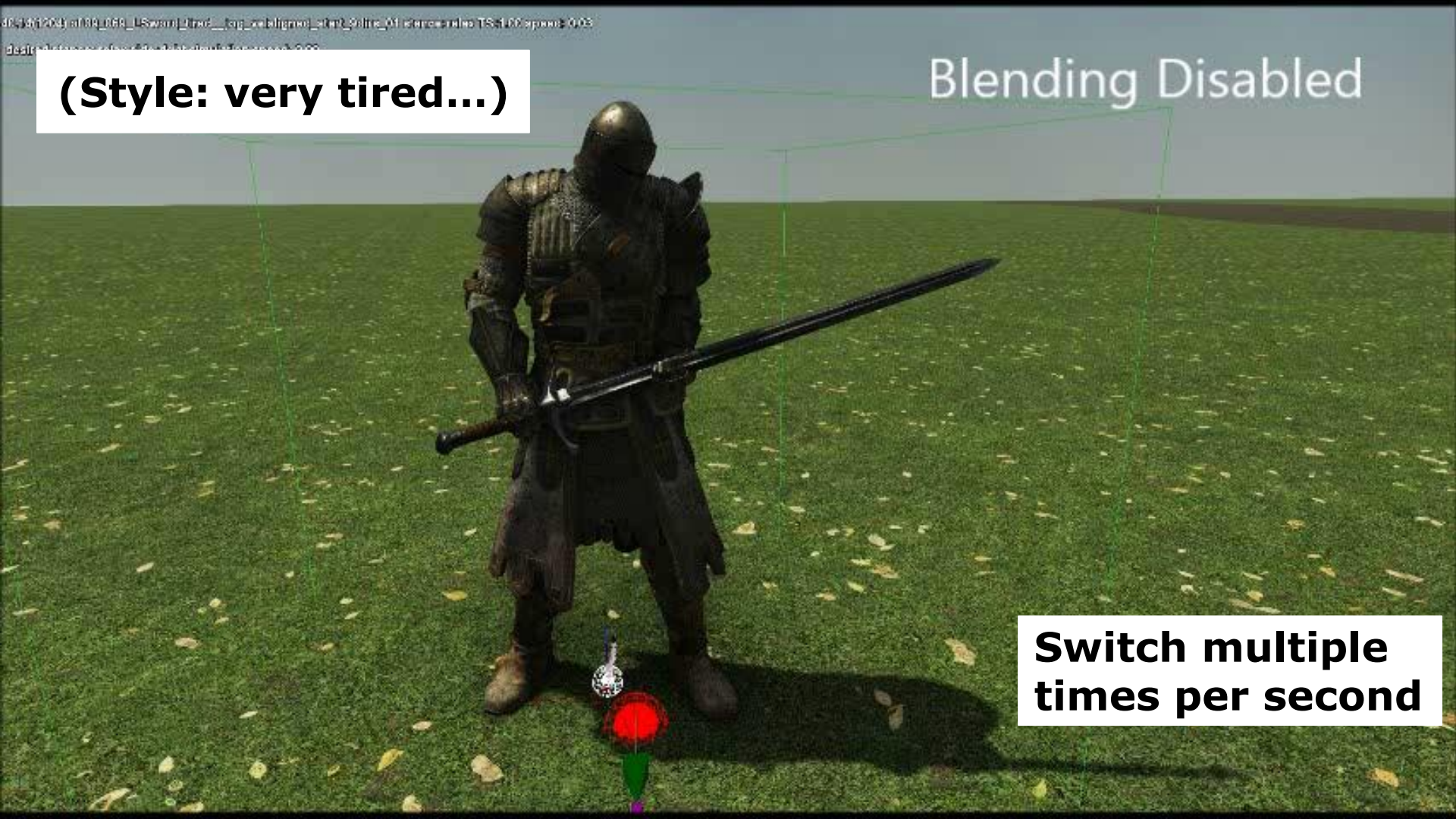
Every frame, look at all mocap and jump at the best place.

Every candidate jumping point has a *cost.*

If the candidate matches the current situation
and
the piece of motion that follows brings us where we want,

then the *cost* is zero.


52_059_LSword_Nav_Walk_Reposition45Deg_8Dir_01
52_060_LSword_Nav_Walk_Stance_Dir_02
52_059_LSword_Nav_Walk_Reposition45Deg_8Dir_01
20_014_FPP_LSword_jog_reposition_1meter_8dirs_01
52_061_LSword_Nav_Walk_Circle_01
52_062_LSword_Nav_Walk_Plant45_135_90_180Deg_01
52_062_LSword_Nav_Walk_Plant45_135_90_180Deg_01_B.mov - VLC media player
Media
Playback
Audio
Video
Tools
View
Help
TCG 07:16:35:00
UBISOFT
EXT-LK
00:08
00:44
75%


(Style: very tired...)
Blending Disabled
Switch multiple times per second

Step 1: Mocap


02:53:19:07
TC 02:53:19:09
EXT-4K
TC 02:53:19:08
EXT-LX
NO SIGNAL
Turn on spot

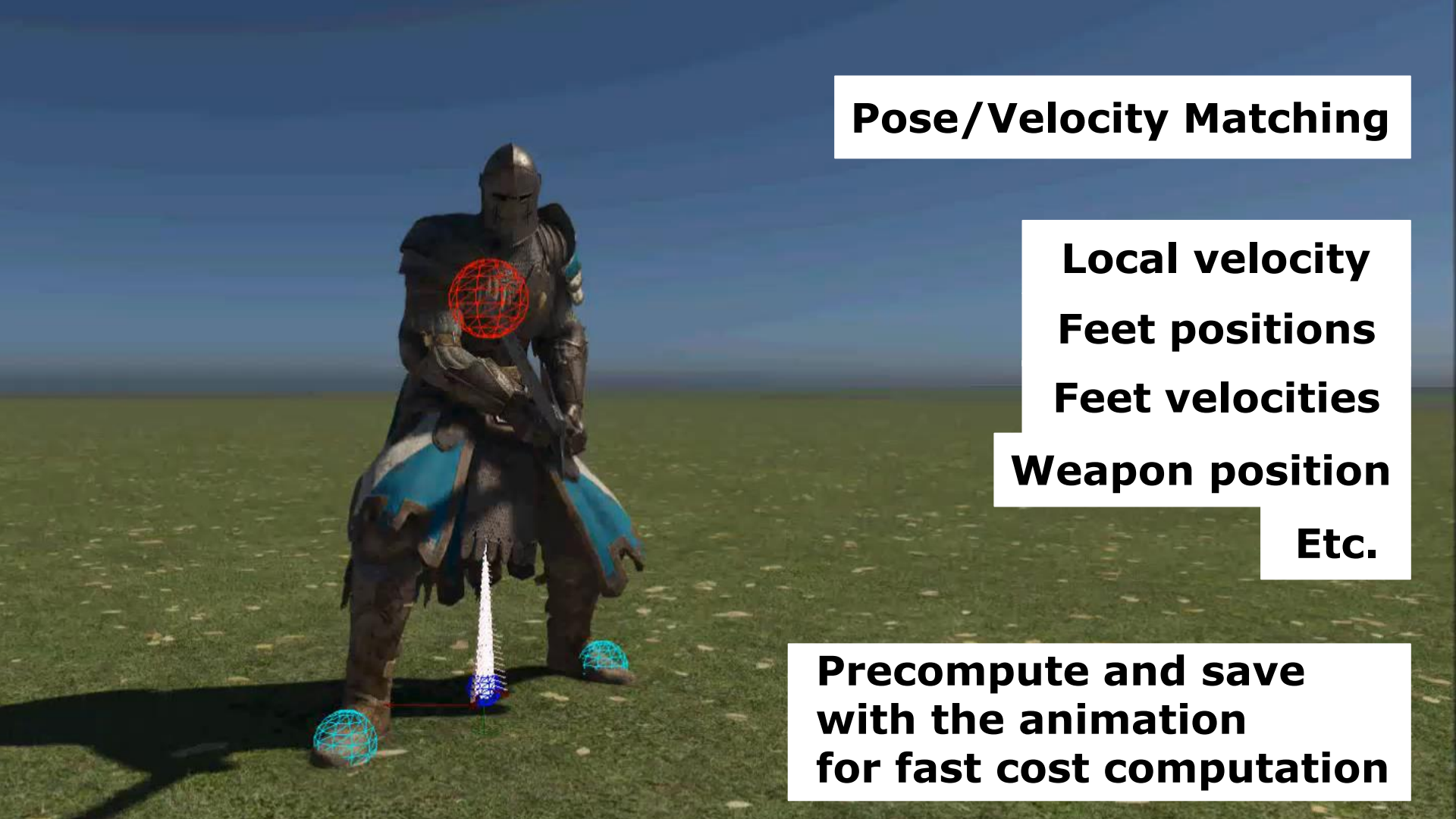
Step 2: Code

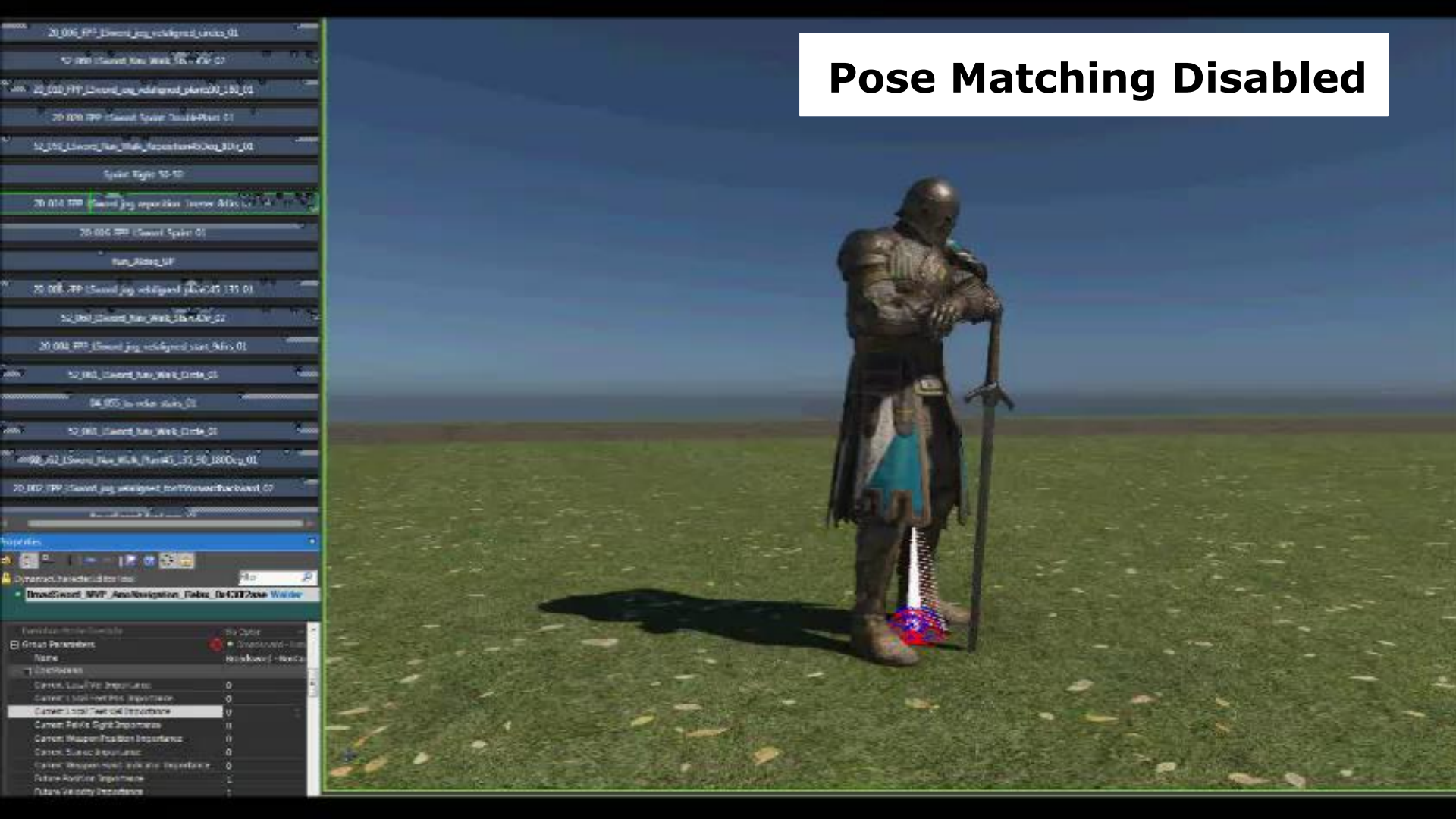
```
int m_CurrentAnimIndex;
float m_CurrentAnimTime;

void AmoUpdate(Goal goal, float dt)
{
    m_CurrentAnimTime += dt;
    Pose currentPose = EvaluateLerpedPoseFromData(m_CurrentAnimIndex, m_CurrentAnimTime);

    float bestCost = 1000000;
    Pose bestPose;

    // loop on all mocap
    for (int i = 0; i < m_Poses.Size(); i++)
    {
        Pose candidatePose = m_Poses[i];

        // every candidate jumping point has a cost
        float thisCost = ComputeCost(currentPose, candidatePose, goal);
        if (thisCost < bestCost)
        {
            // remember the best candidate
            bestCost = thisCost;
            bestPose = candidatePose;
        }
    }
```

```
    bool theWinnerIsAtTheSameLocation =
        m_CurrentAnimIndex == bestPose.m_AnimIndex &&
        fabs(m_CurrentAnimTime - bestPose.m_AnimTime) < 0.2f;

    if (!theWinnerIsAtTheSameLocation)
    {
        // blend to the winning location
        m_CurrentAnimIndex = bestPose.m_AnimIndex;
        m_CurrentAnimTime = bestPose.m_AnimTime;
        PlayAnimStartingAtTime(m_CurrentAnimIndex, m_CurrentAnimTime, 0.25f);
    }
}
```

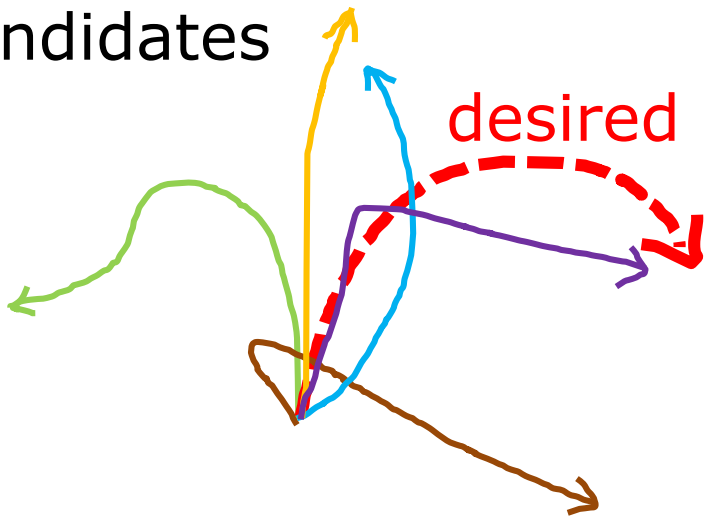
```
float ComputeCost(Pose currentPose, Pose candidatePose, Goal goal)
{
    float cost = 0.0f;

    // how much the candidate jumping position matches the current situation
    cost += ComputeCurrentCost(currentPose, candidatePose);









    return cost;
}
```

Trick 1: Posematch only a few bones


Pose/Velocity Matching
Local velocity
Feet positions
Feet velocities
Weapon position
Etc.
Precompute and save
with the animation
for fast cost computation

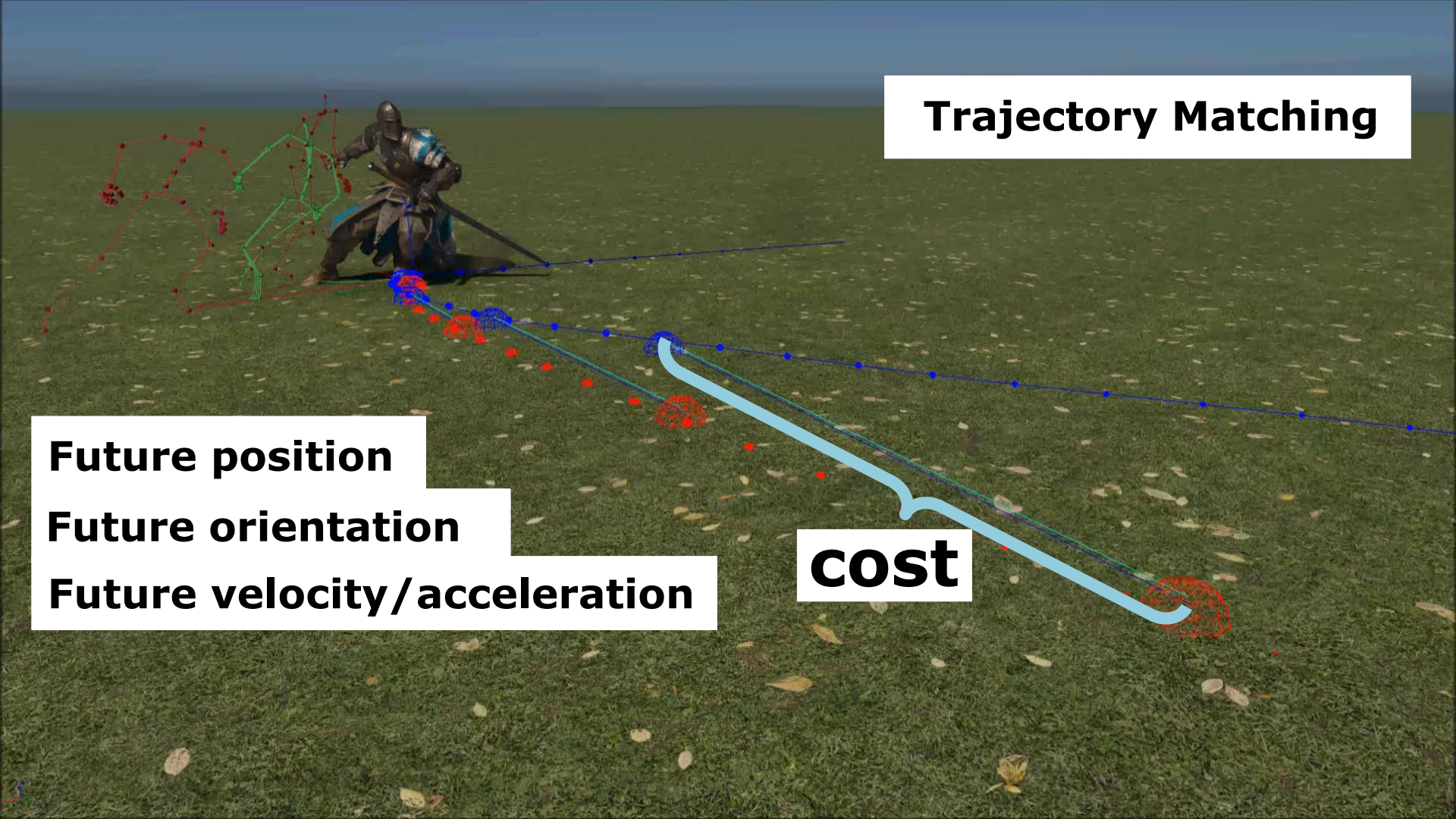

Pose Matching Disabled

How to follow a desired trajectory?

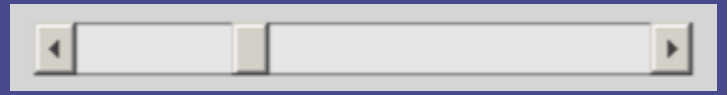


Trick 2: Just check where a piece of animation brings you if you play it

```
class TrajectoryPoint
{
    Vector3 m_Position
    float m_Sight;
    float m_TimeDelay;
};

// desired goal, sent by gameplay each frame
class Goal
{
    Array<TrajectoryPoint> m_DesiredTrajectory;

    Stance m_DesiredStance;

    // ...
};
```


Trajectory Matching
Future position
Future orientation
Future velocity/acceleration
cost

```
float ComputeCost(Pose currentPose, Pose candidatePose, Goal goal)
{
    float cost = 0.0f;

    // how much the candidate jumping position matches the current situation
    cost += ComputeCurrentCost(currentPose, candidatePose);

    // this is our responsivity slider
    static float responsivity = 1.0f;

    // how much the candidate piece of motion matches the desired trajectory
    cost += responsivity * ComputeFutureCost(candidatePose, goal);

    return cost;
}
```

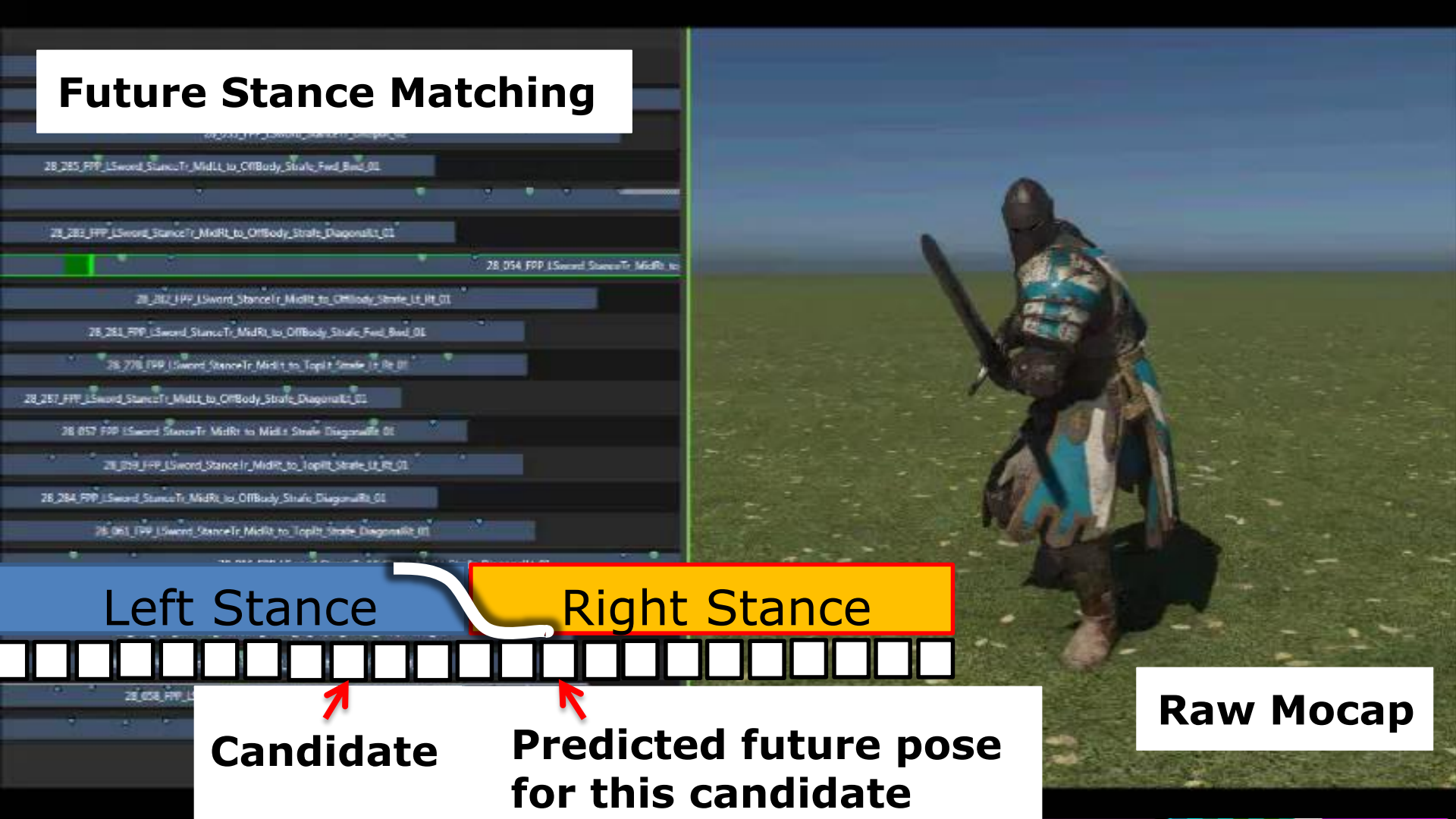


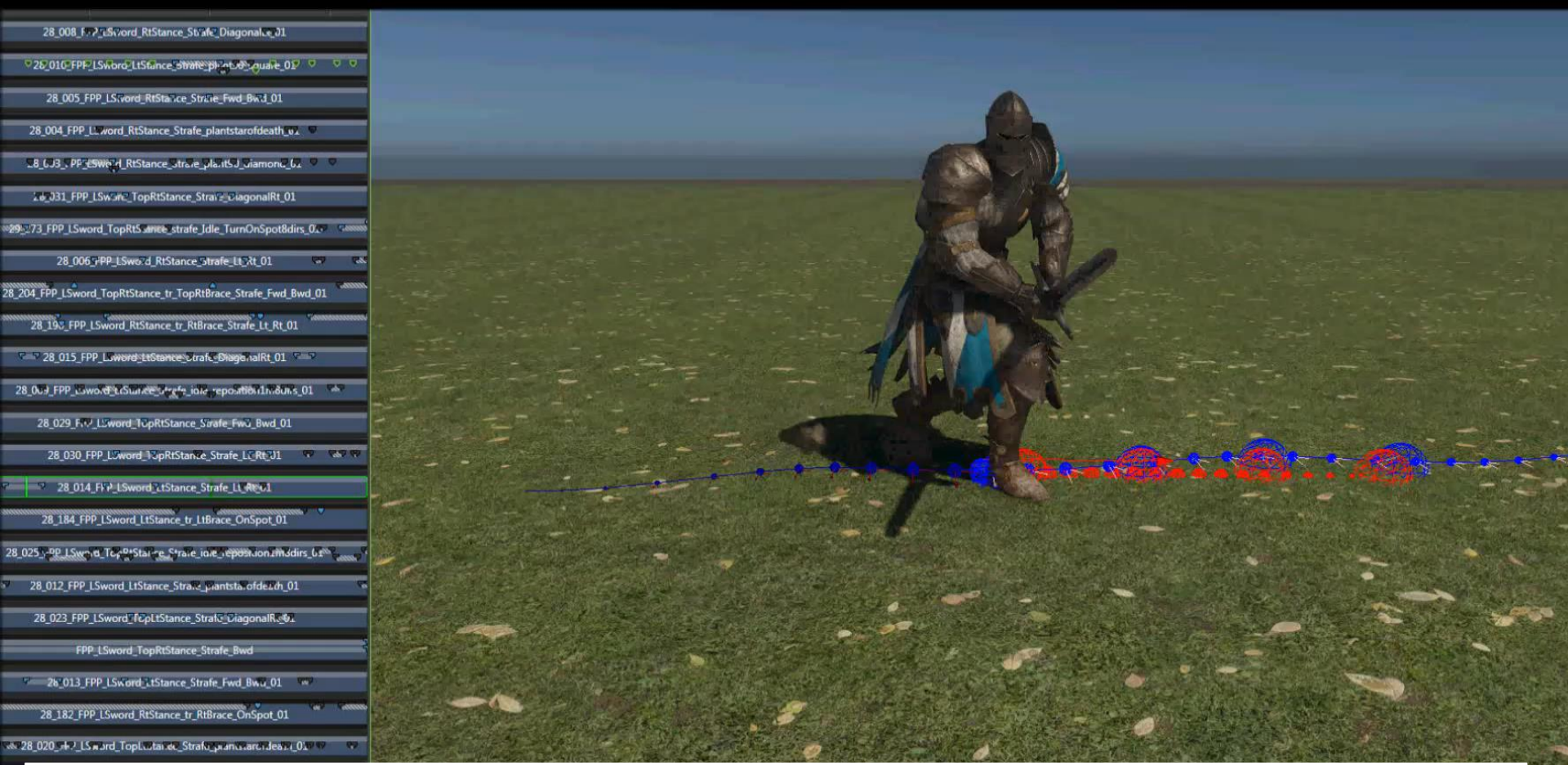
Realism

Comfort

Now we can match more things

Future Stance Matching





Elegantly find transitions when they exist

Optimizations (subject of a future talk)

- ***LOD***
- ***KD-Tree***
- ***Motion Shaders***
- ***Etc.***

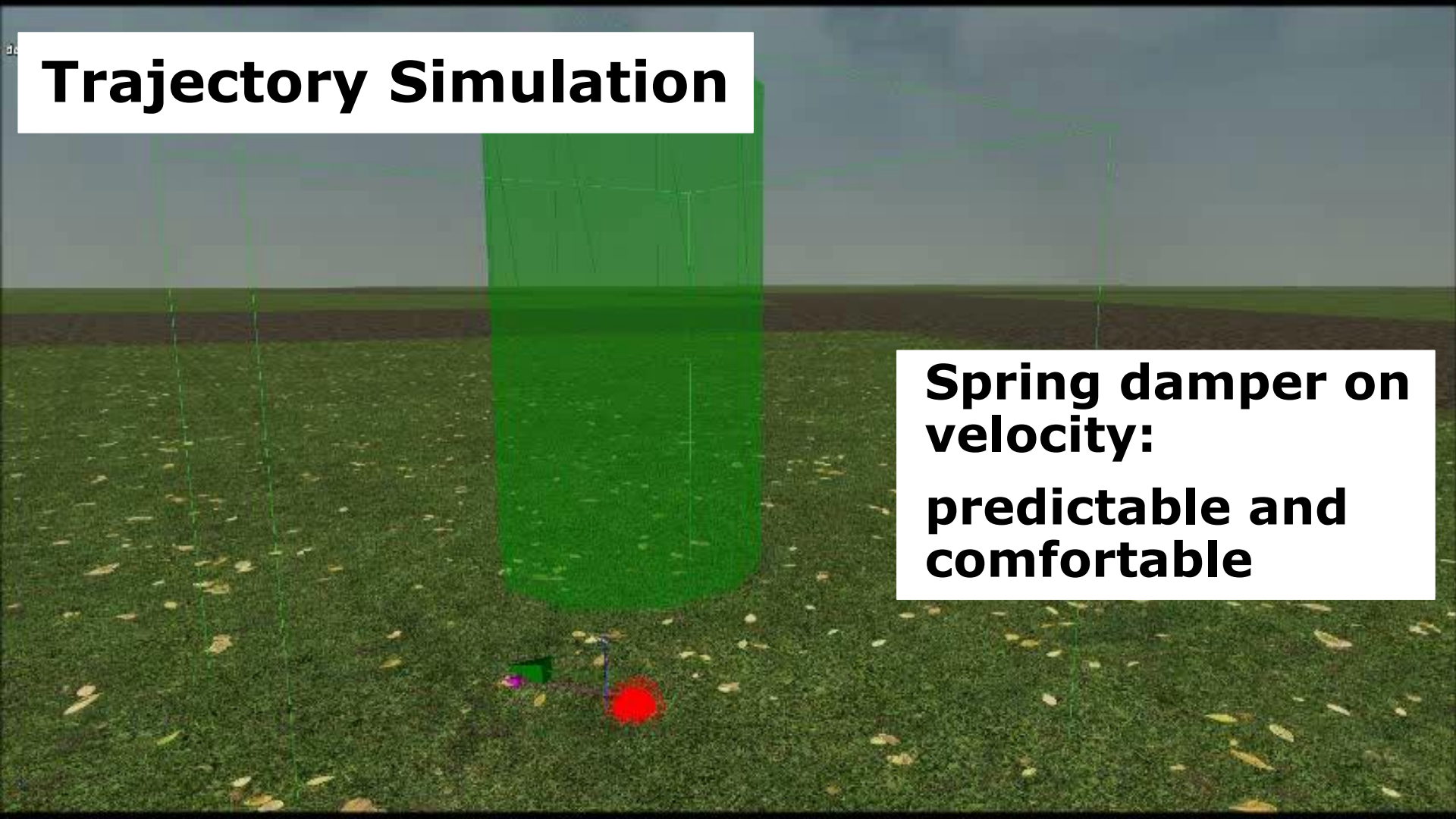
Trajectory Simulation Choices

- ***Displacement from Animation*?**
- ***Displacement from Simulation*?**

Our choice:

The code decides the real simulation point

Trajectory Simulation

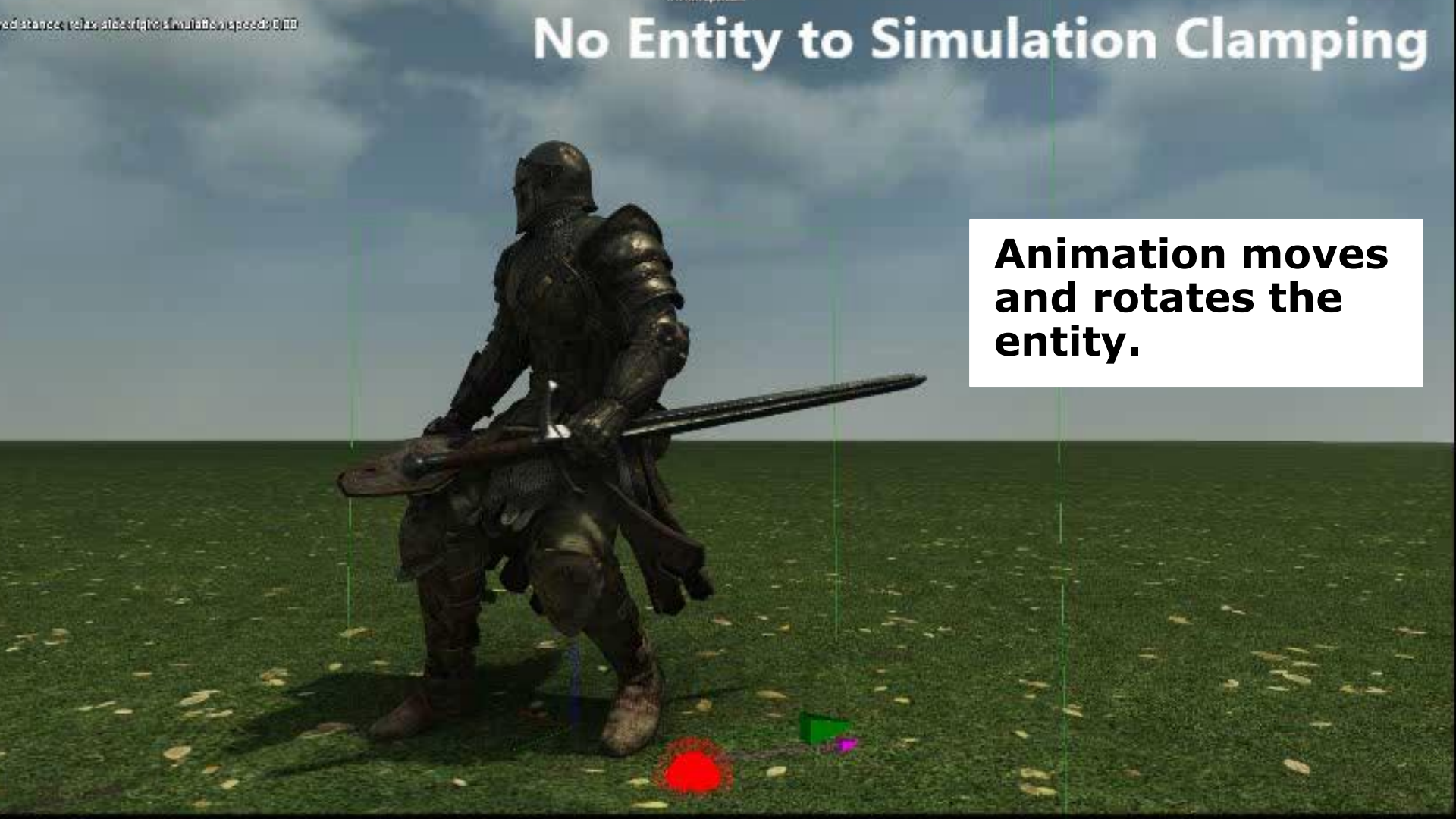


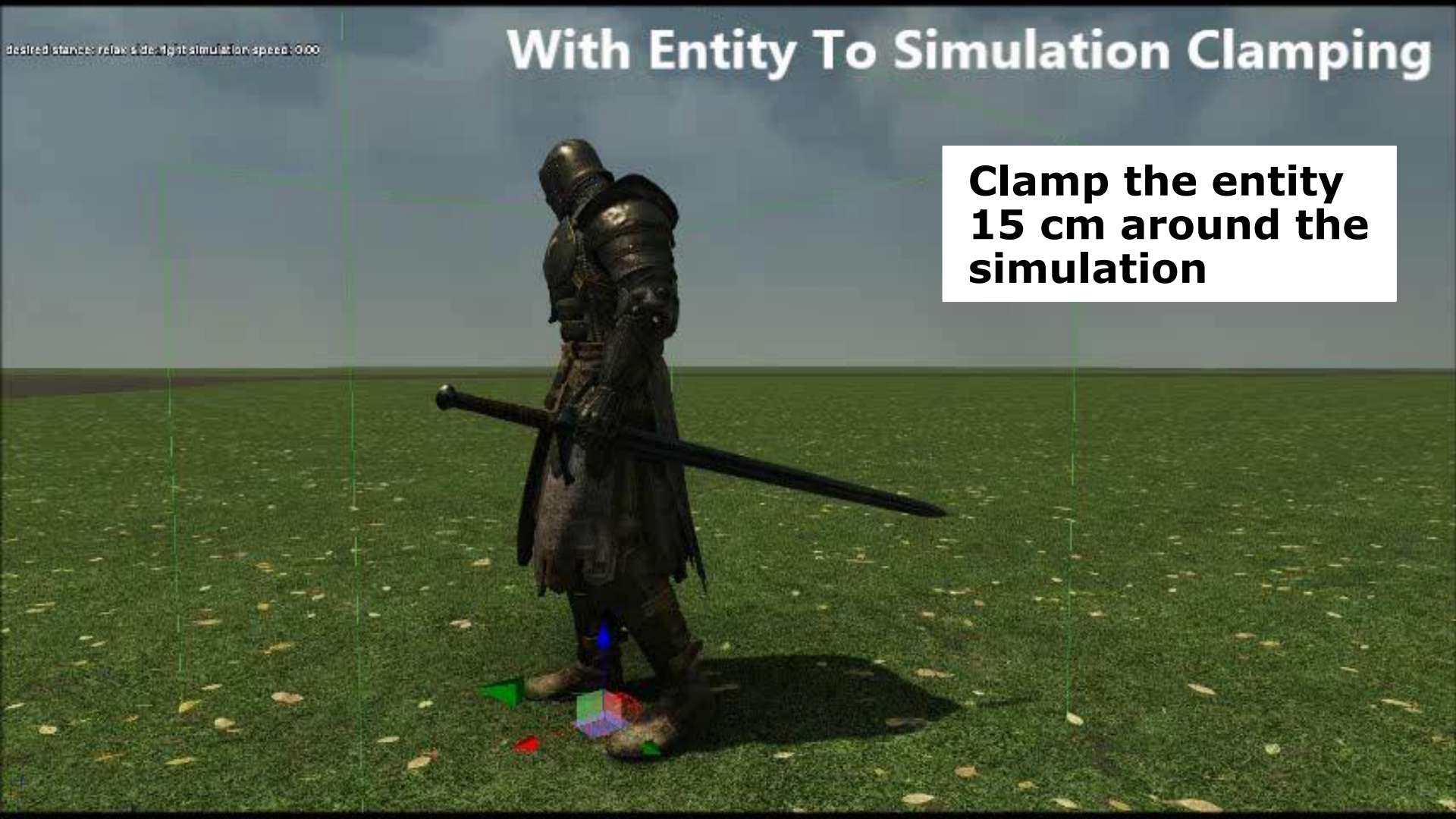
Spring damper on velocity:

predictable and comfortable

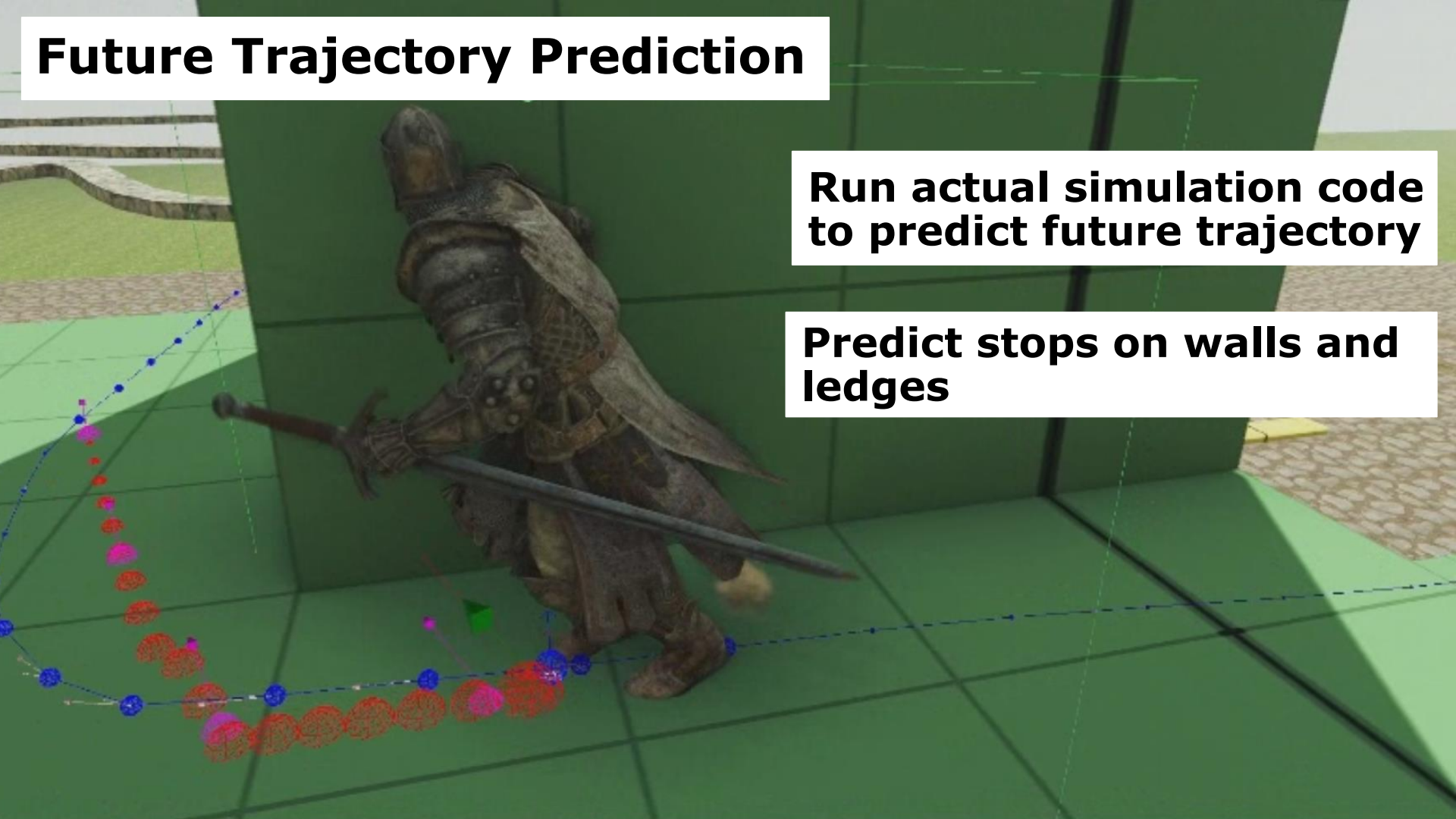
Choose animations that follow the simulation

And do *Displacement from Animation*


No Entity to Simulation Clamping
Animation moves and rotates the entity.


desired stance: relax side: right simulation speed: 0.00
With Entity To Simulation Clamping
Clamp the entity 15 cm around the simulation

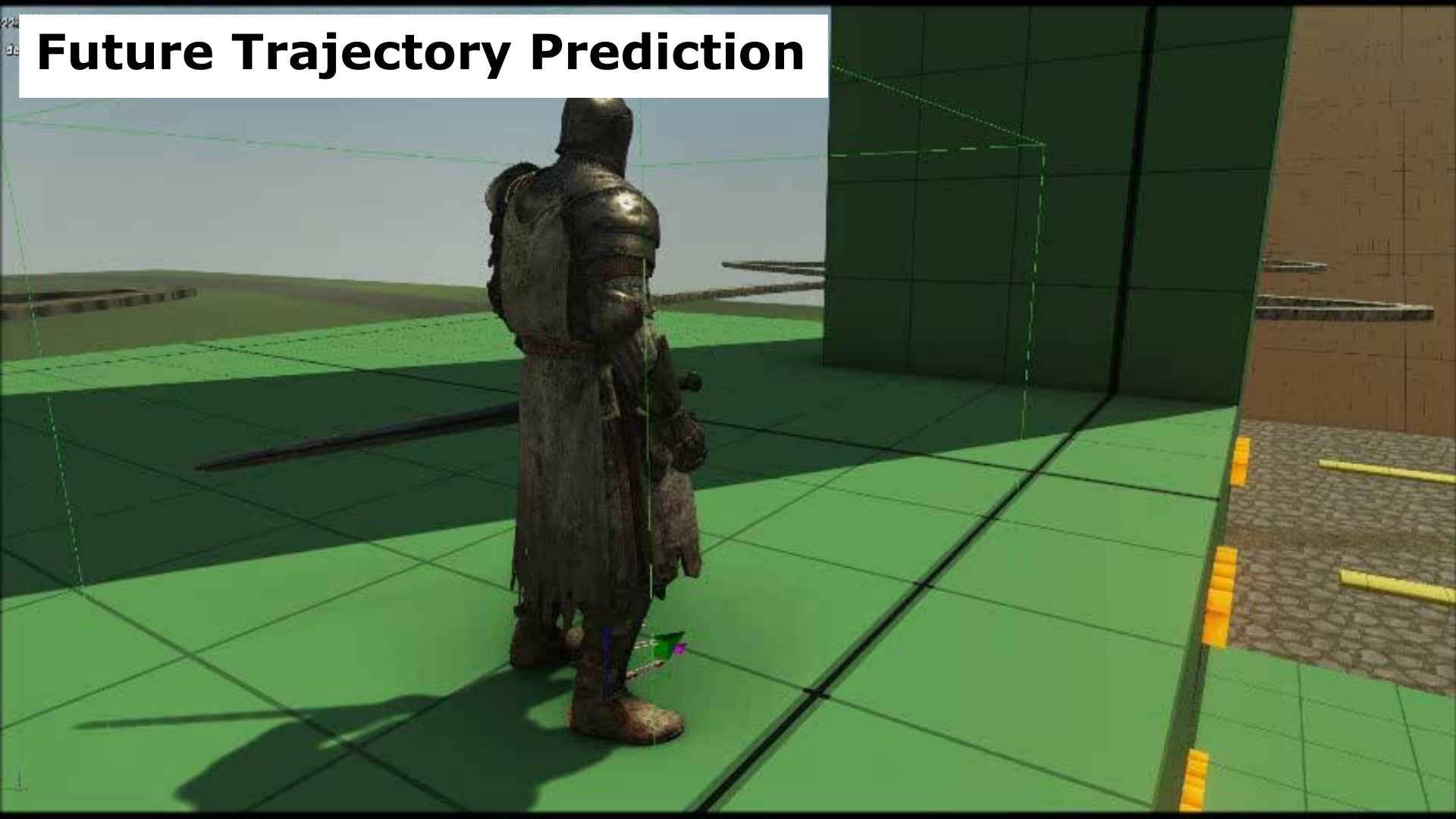
Future Trajectory Prediction



Run actual simulation code to predict future trajectory

Predict stops on walls and ledges

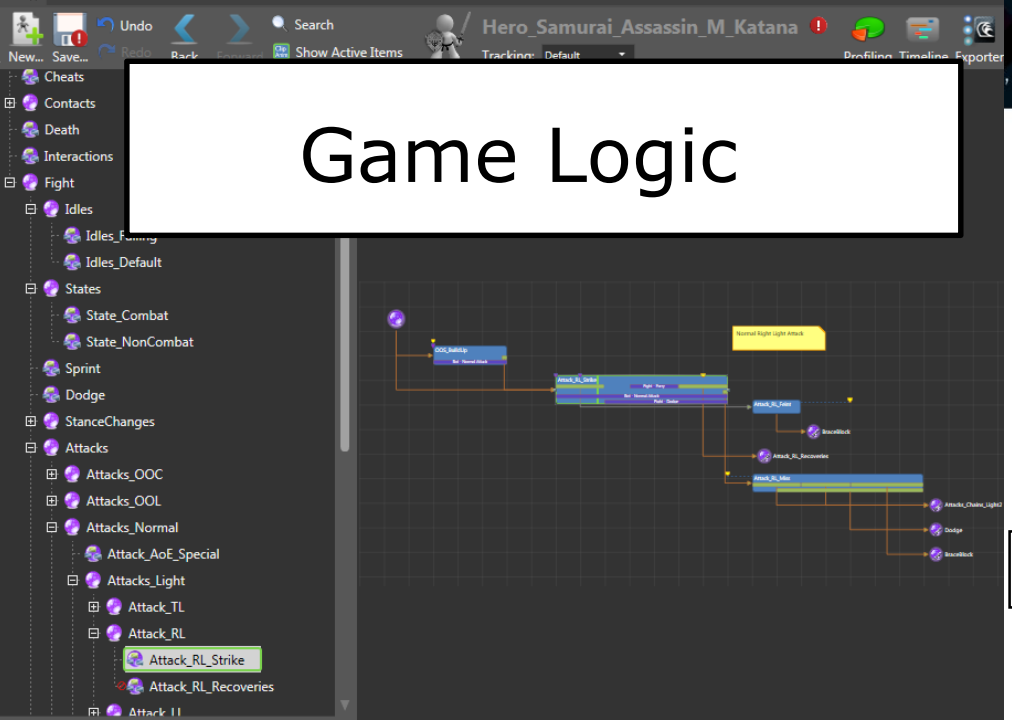
Future Trajectory Prediction



- Little history of animation systems
- Motion Matching
- **Workflow**
- **Procedural Touchups**

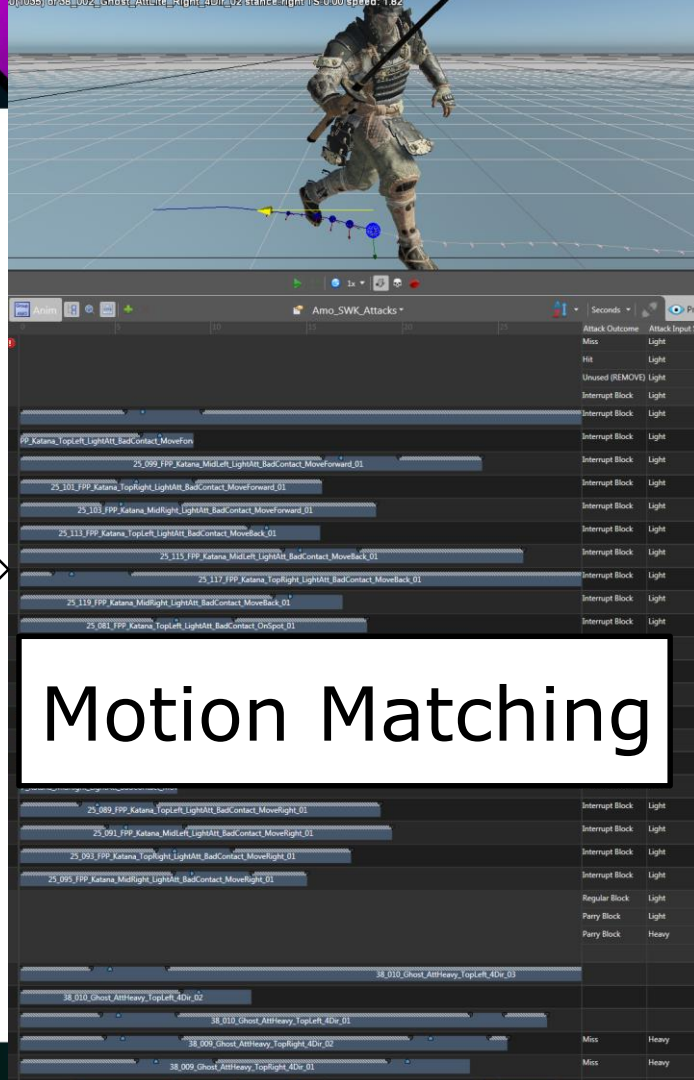
Pipeline

- Mocap is tweaked, imported, and marked up
- At runtime Gameplay makes a *request* (desired trajectory and event constraints)
- The animation system continuously finds the best piece of data to play
- We modify the result to precisely match gameplay and environment



Game Logic

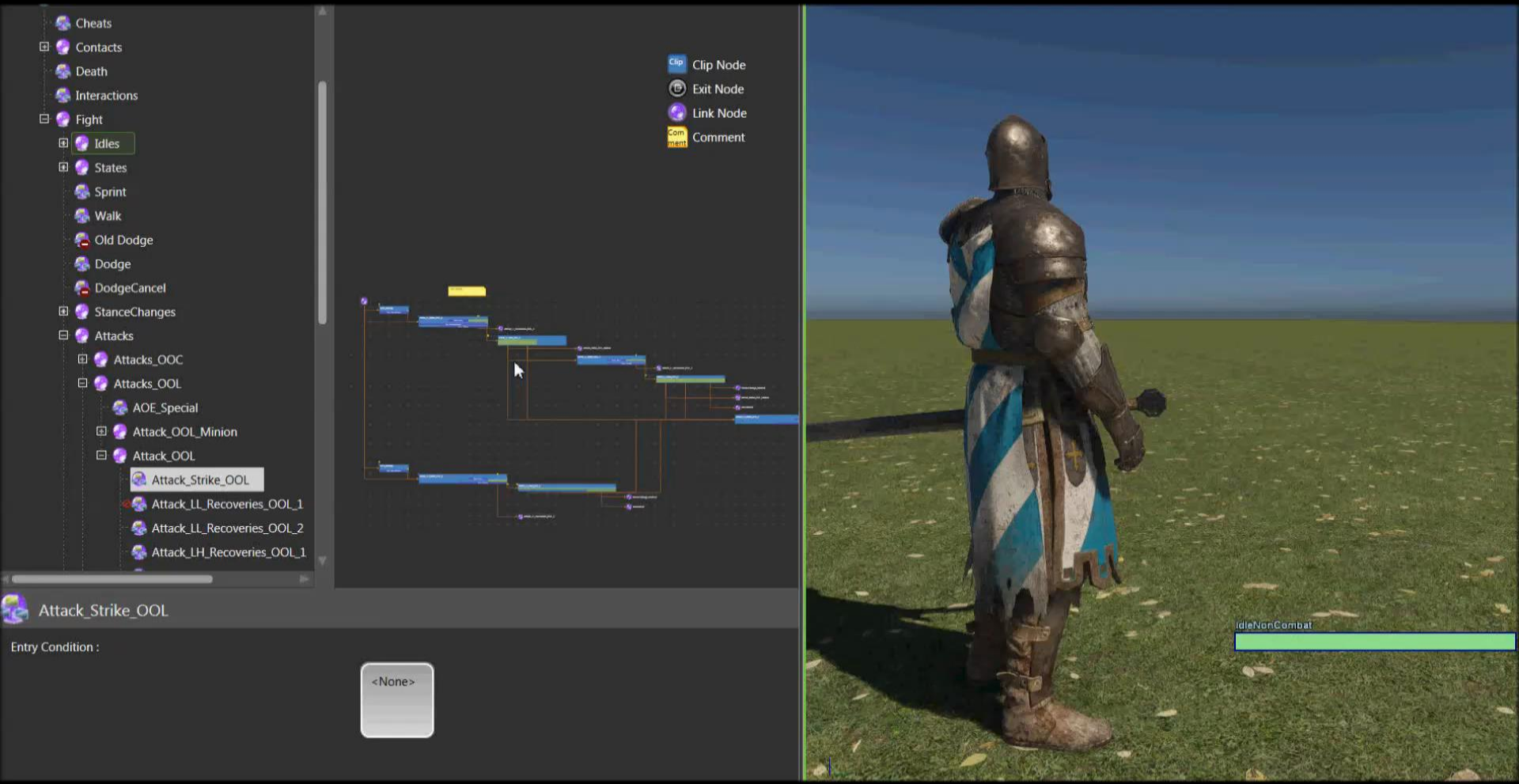
request



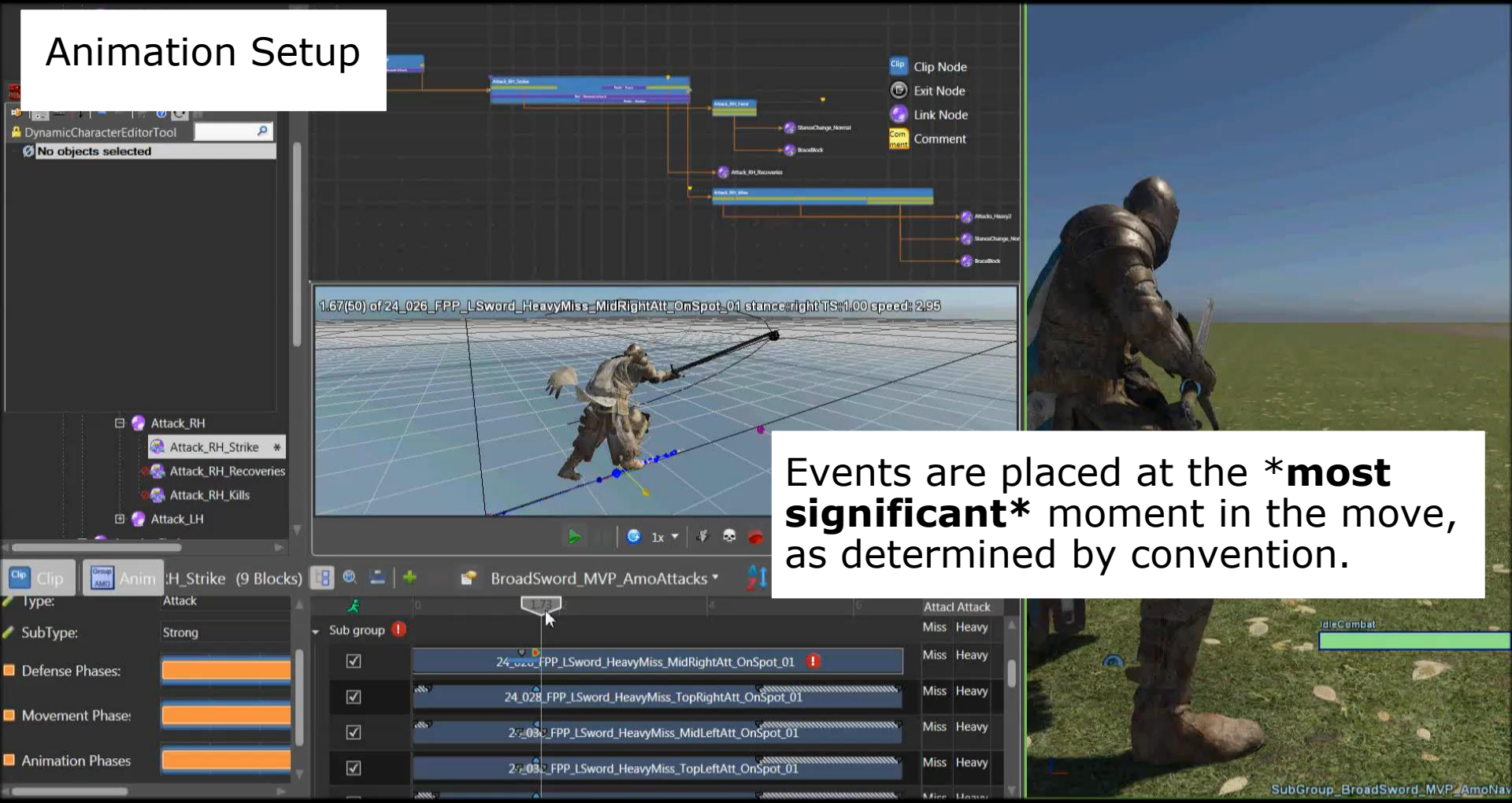
Motion Matching

Simple timed state-machine

Possible to play the game without animation! (just debug display)


Cheats
Contacts
Death
Interactions
Fight
Idles
States
Sprint
Walk
Old Dodge
Dodge
DodgeCancel
StanceChanges
Attacks
Attacks_OOC
Attacks_OOL
AOE_Special
Attack_OOL_Minion
Attack_OOL
Attack_Strike_OOL
Attack_LL_Recoveries_OOL_1
Attack_LL_Recoveries_OOL_2
Attack_LH_Recoveries_OOL_1
Clip Node
Exit Node
Link Node
Comment
Attack_Strike_OOL
Entry Condition :
<None>
IdleNonCombat

Animation Setup



Events are placed at the ***most significant*** moment in the move, as determined by convention.

Clip Event Request

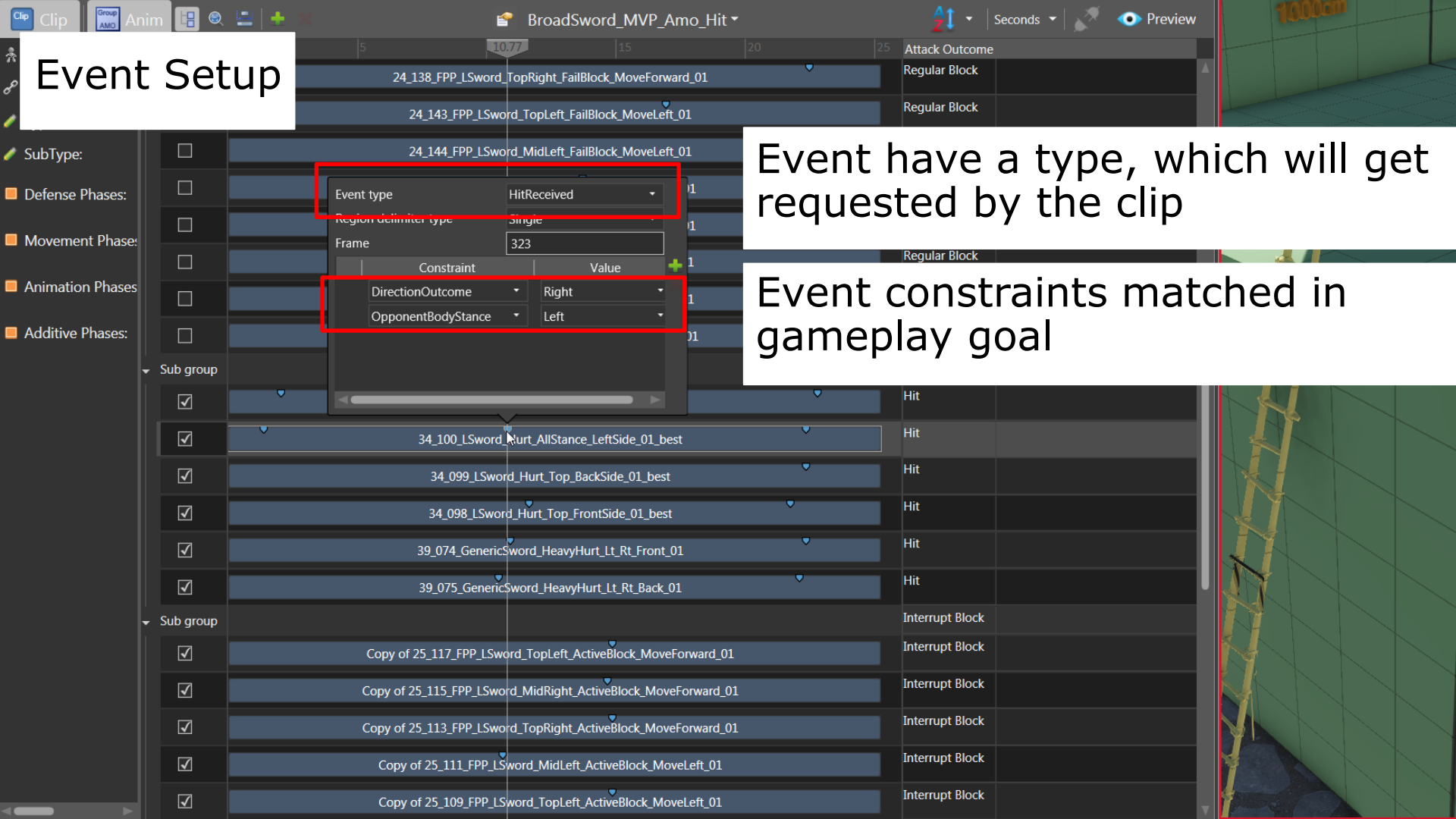
0.8

The clip says: **I want this event to happen exactly 0.8 seconds after I start**

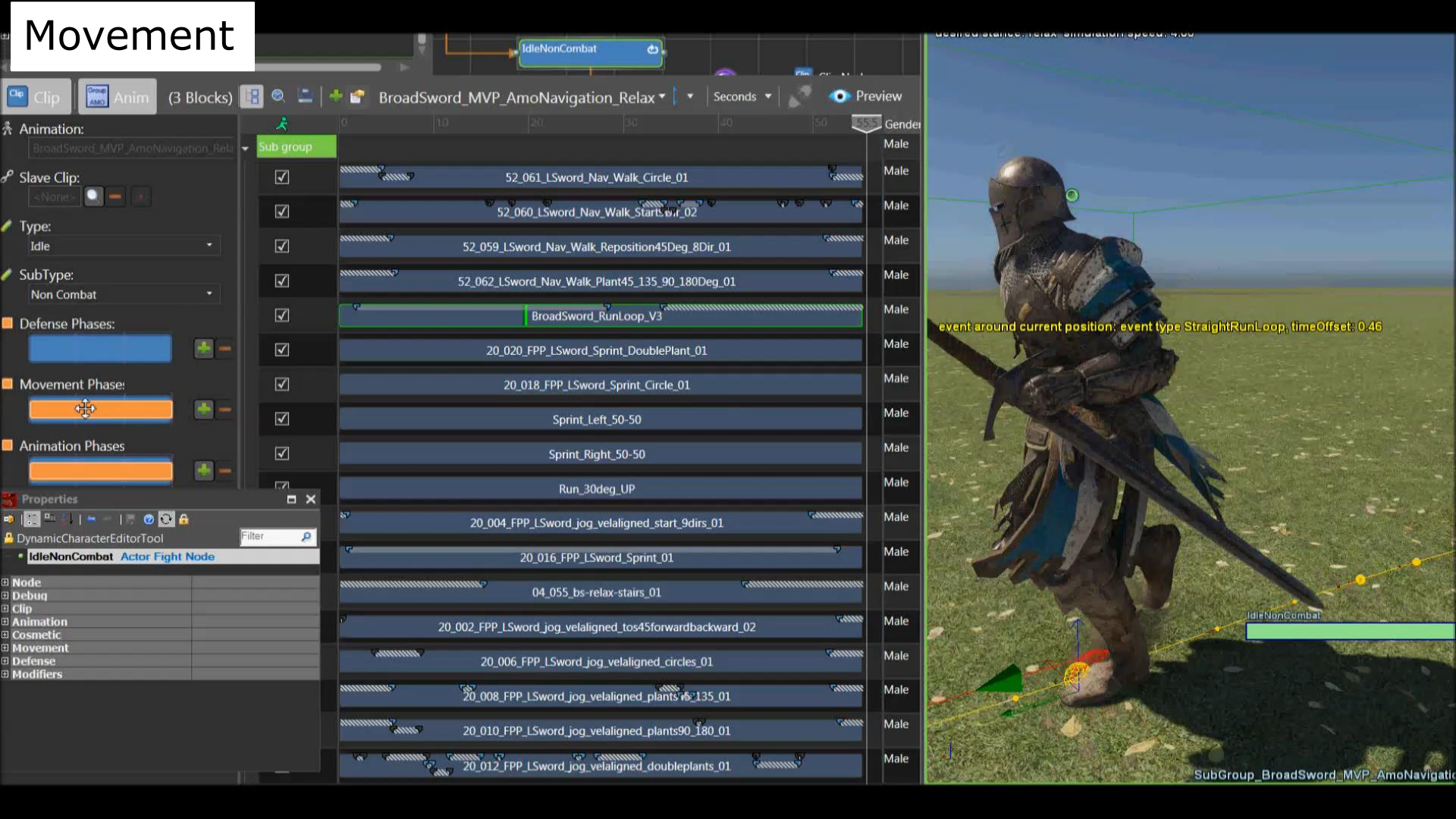
Event Setup

Event have a type, which will get requested by the clip

Event constraints matched in gameplay goal



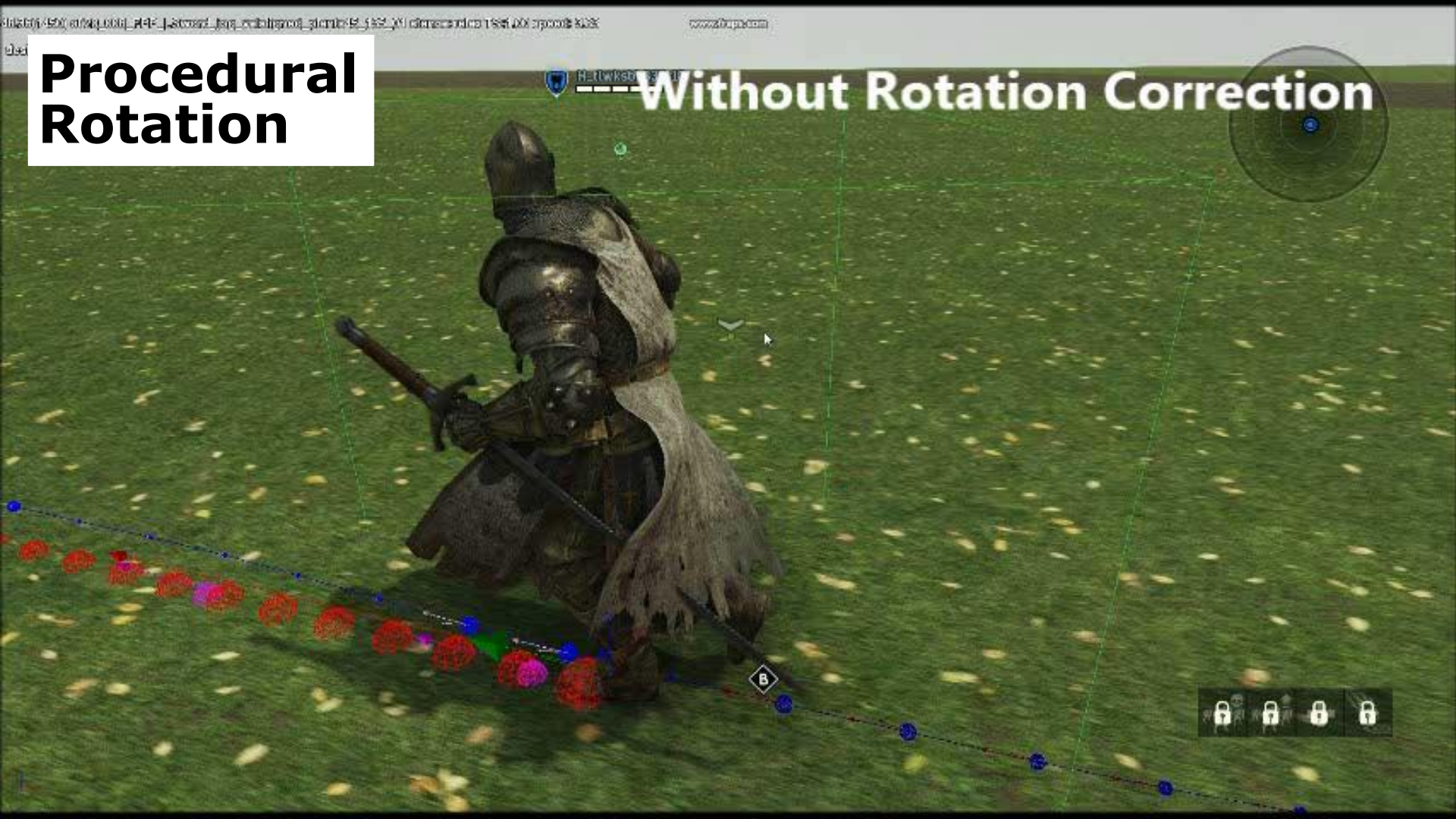
Movement

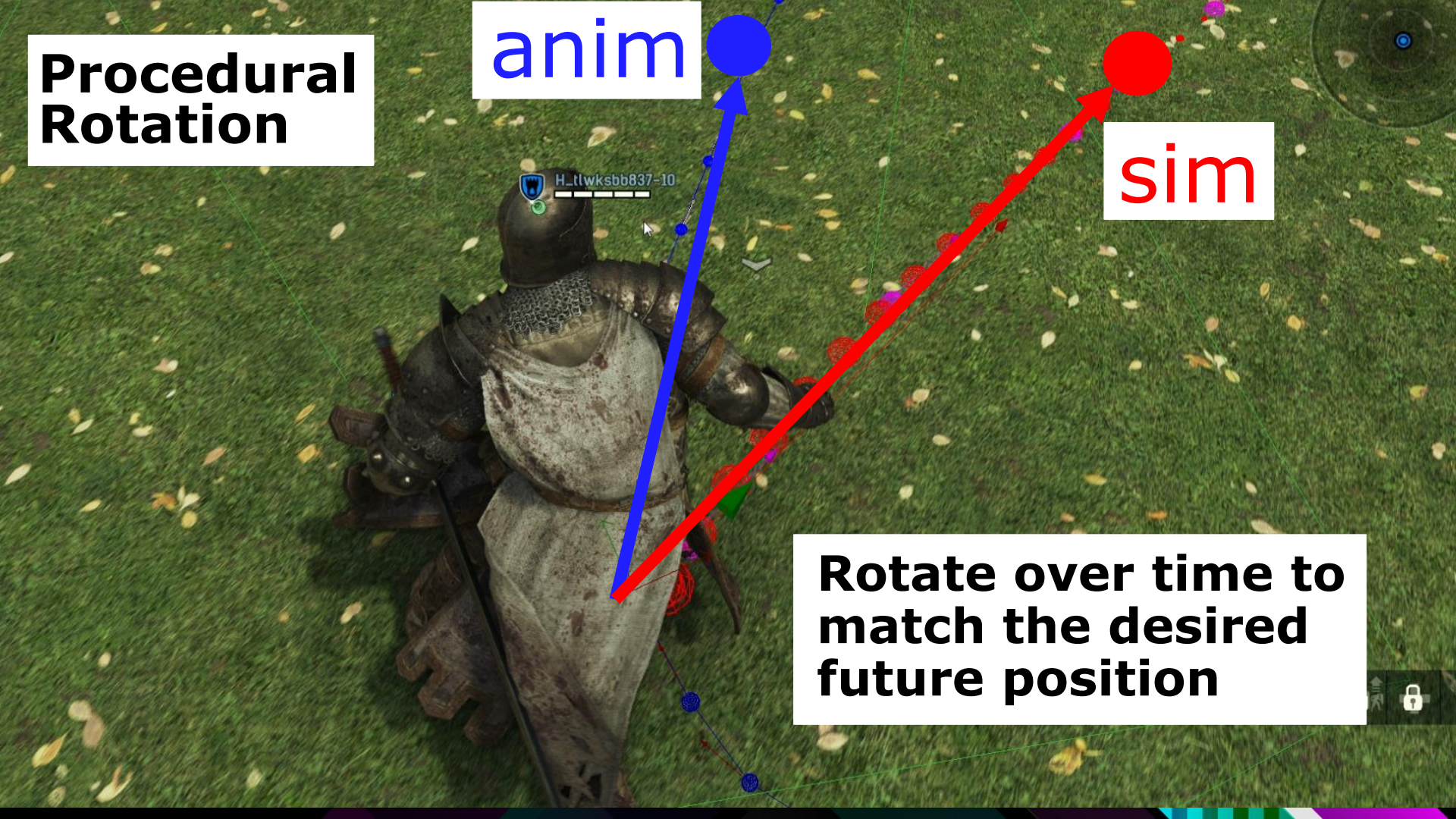

IdleNonCombat
Clip
Anim
(3 Blocks)
BroadSword_MVP_AmoNavigation_Relax
Seconds
Preview
Animation:
Slave Clip:
Type:
Idle
SubType:
Non Combat
Defense Phases:
Movement Phase:
Animation Phases
Sub group
Gender
52_061_LSword_Nav_Walk_Circle_01
52_060_LSword_Nav_Walk_Starts_Dir_02
52_059_LSword_Nav_Walk_Reposition45Deg_8Dir_01
52_062_LSword_Nav_Walk_Plant45_135_90_180Deg_01
BroadSword_RunLoop_V3
20_020_FPP_LSword_Sprint_DoublePlant_01
20_018_FPP_LSword_Sprint_Circle_01
Sprint_Left_50-50
Sprint_Right_50-50
Run_30deg_UP
20_004_FPP_LSword_jog_velaligned_start_9dirs_01
20_016_FPP_LSword_Sprint_01
04_055_bs-relax-stairs_01
20_002_FPP_LSword_jog_velaligned_tos45forwardbackward_02
20_006_FPP_LSword_jog_velaligned_circles_01
20_008_FPP_LSword_jog_velaligned_plants45_135_01
20_010_FPP_LSword_jog_velaligned_plants90_180_01
20_012_FPP_LSword_jog_velaligned_doubleplants_01
Male
Properties
DynamicCharacterEditorTool
Filter
IdleNonCombat Actor Fight Node
Node
Debug
Clip
Animation
Cosmetic
Movement
Defense
Modifiers
event around current position: event type StraightRunLoop, timeOffset: 0.46
IdleNonCombat
SubGroup_BroadSword_MVP_AmoNavigati

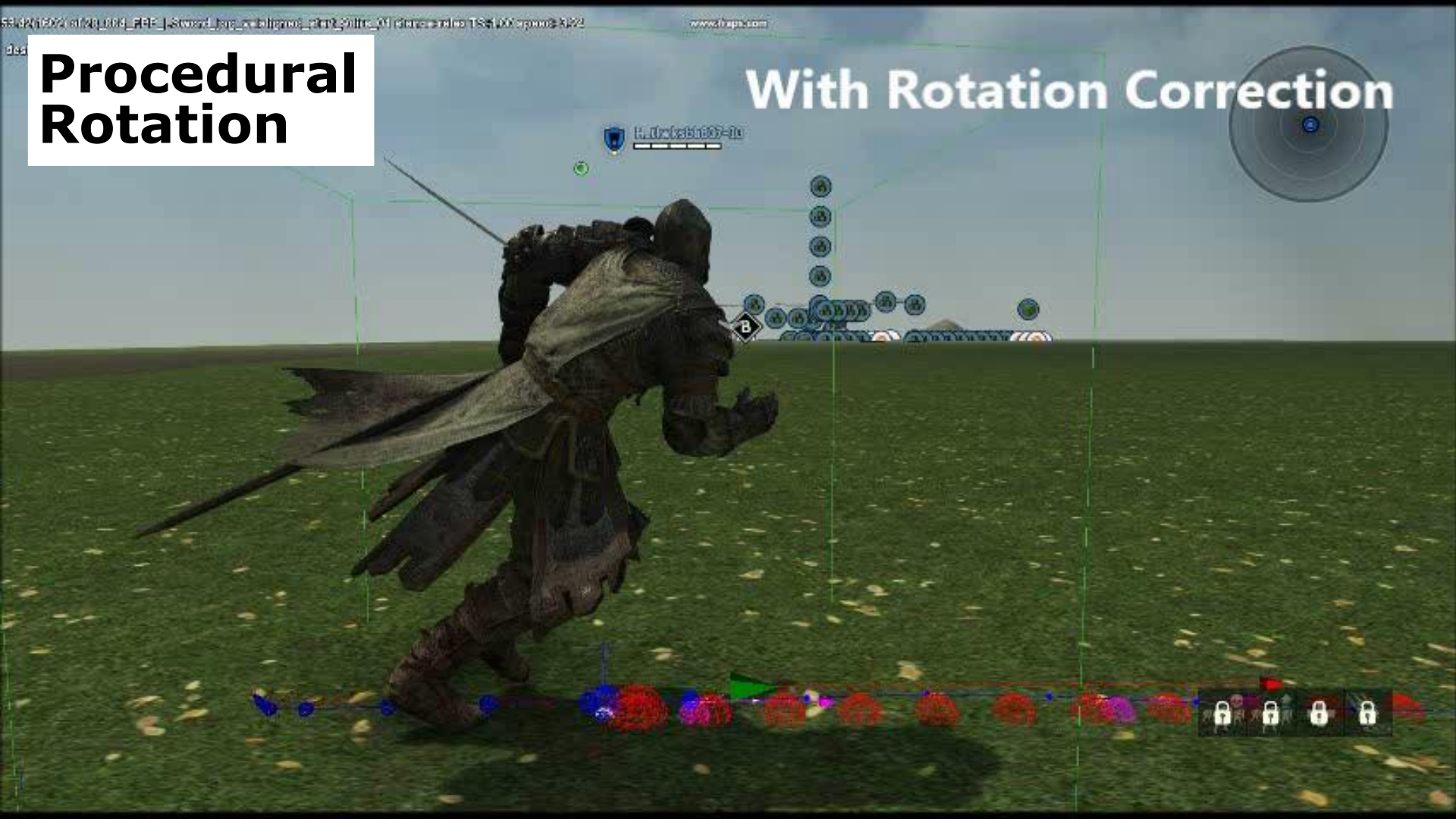
- Little history of animation systems
- Motion Matching
- Workflow
- **Procedural Touchups**

Orientation corrections

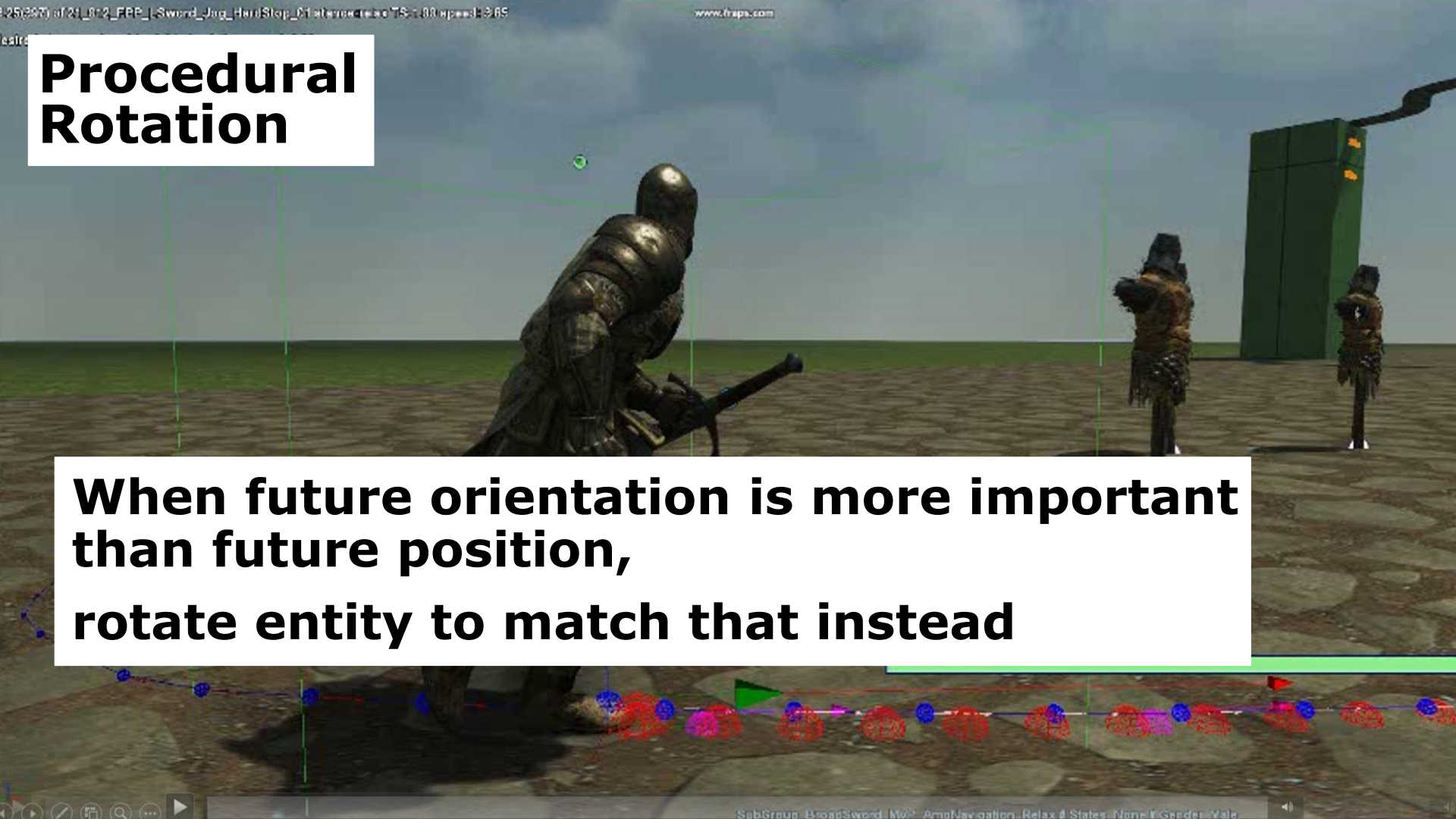
- Let the anim decide rotation but
- Correct to match future desired position
- Or to match future desired orientation


Procedural Rotation
Without Rotation Correction


Procedural Rotation
anim
sim
H_tlwksbb837-10
Rotate over time to match the desired future position


Procedural Rotation
With Rotation Correction

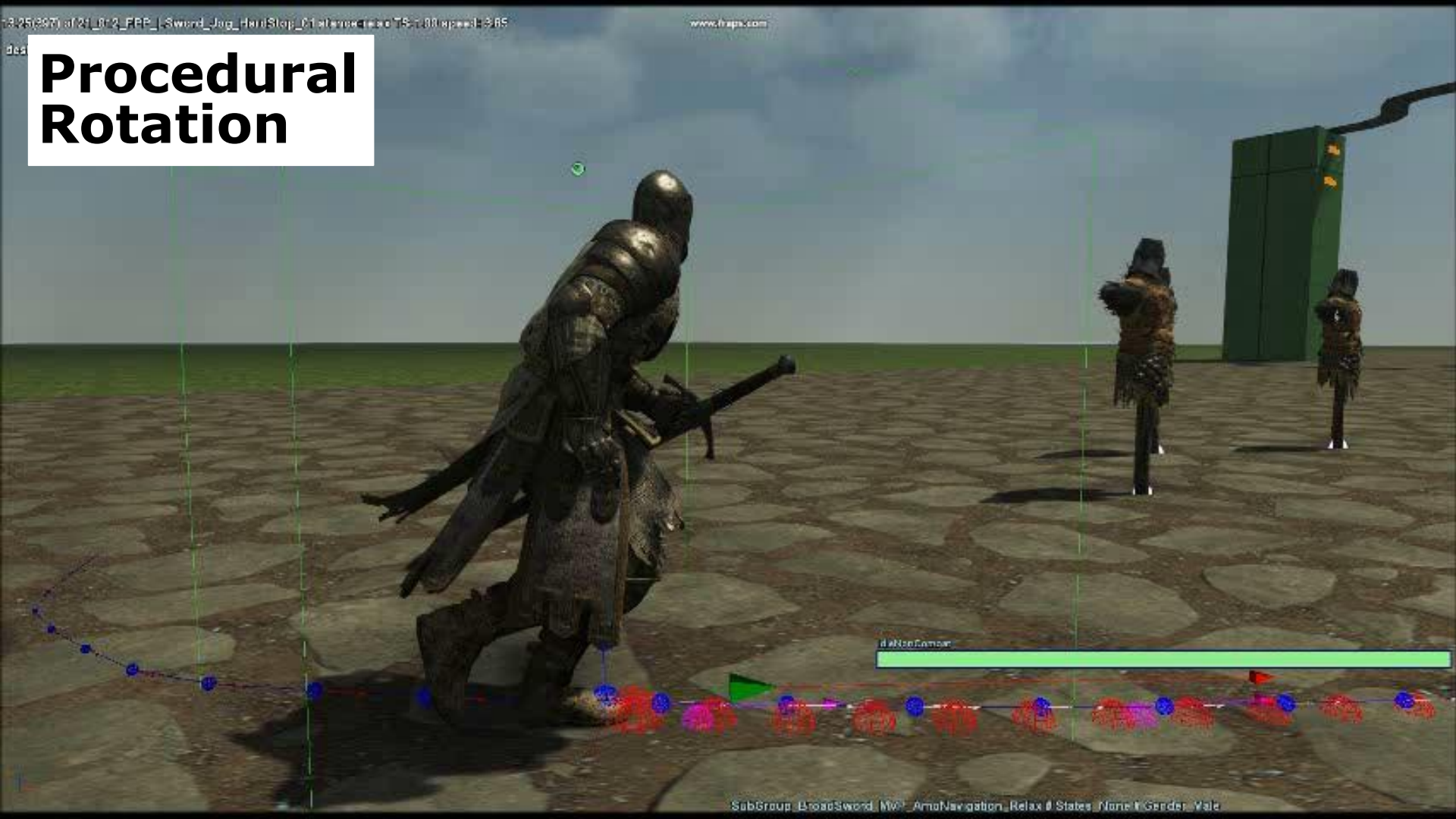
Procedural Rotation

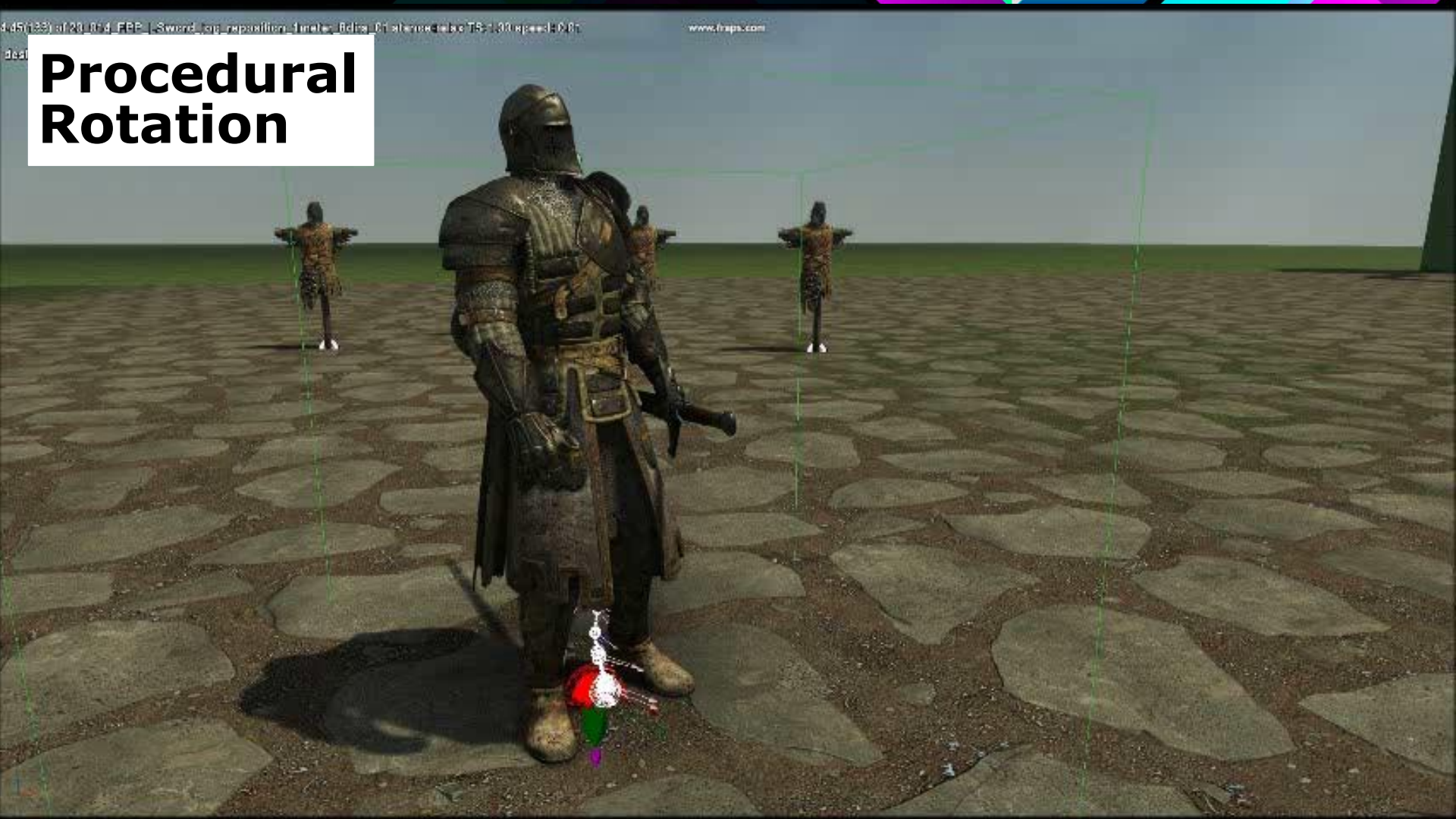


When future orientation is more important than future position,

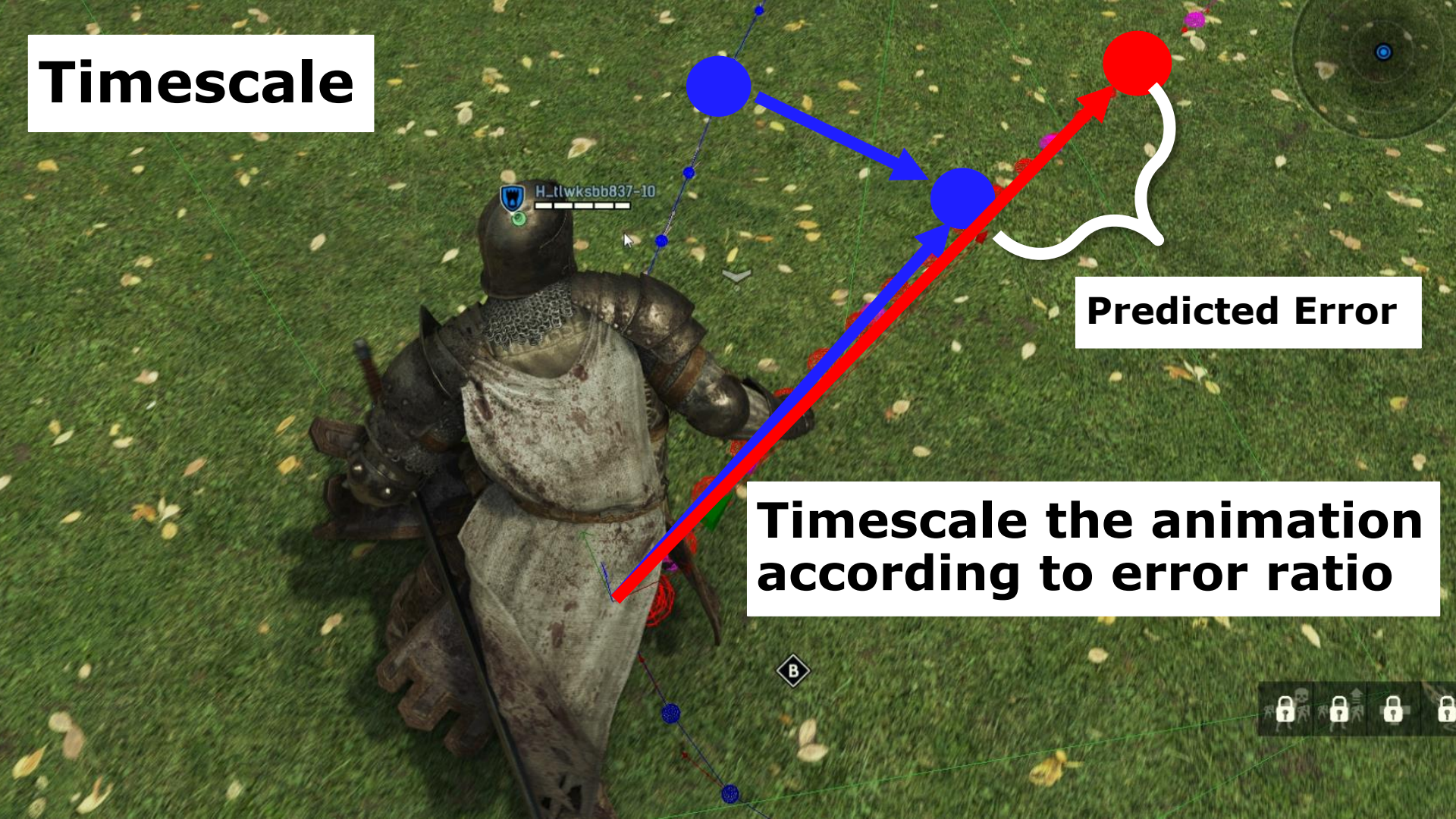
rotate entity to match that instead

Procedural Rotation




Procedural
Rotation

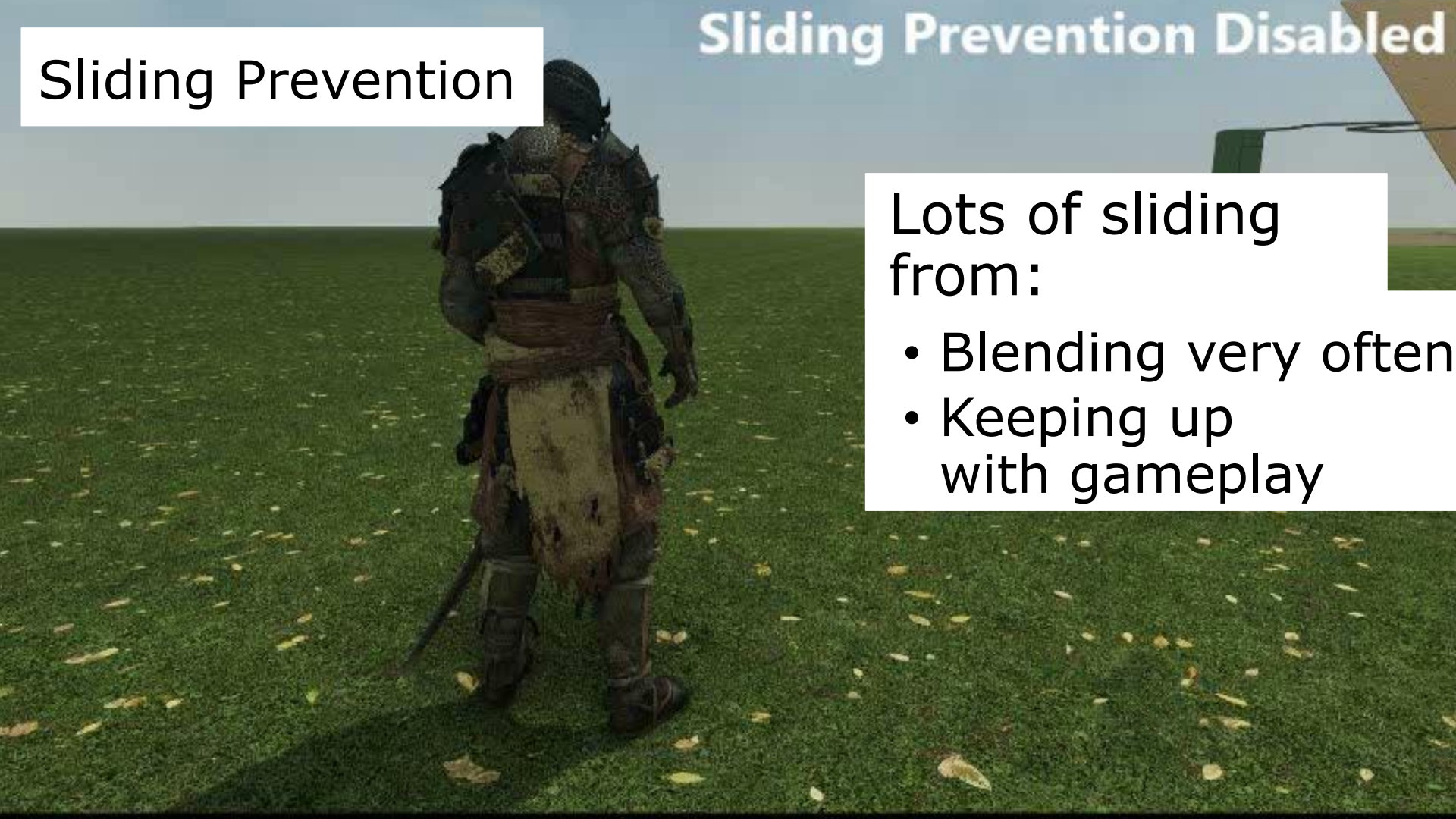
Timescale


Timescale
H_tlwksbb837-10
Predicted Error
Timescale the animation according to error ratio

Timescale

Just sliding often gives better results

Sliding Prevention



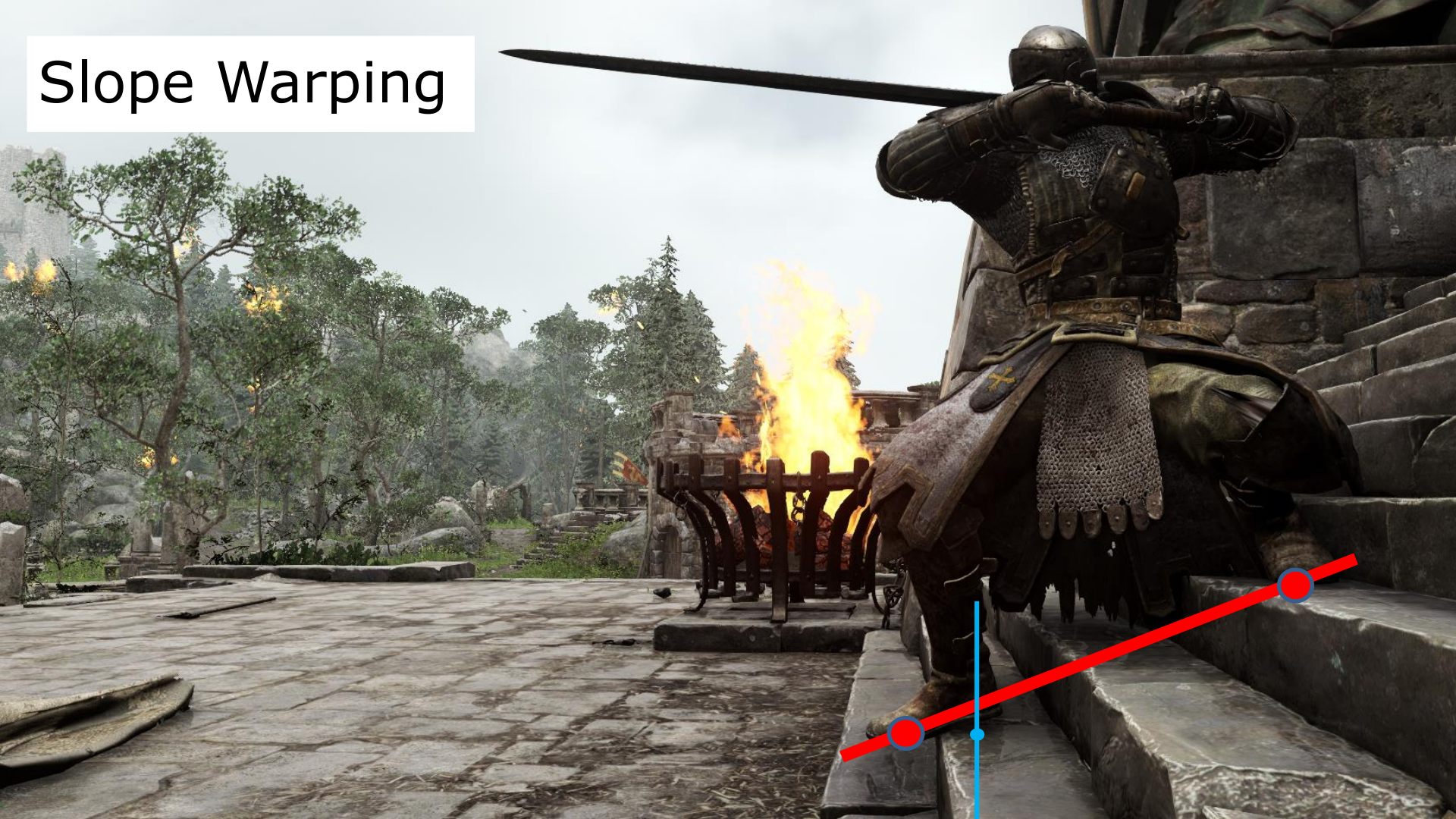
Lots of sliding from:

- Blending very often
- Keeping up with gameplay

Sliding Prevention

Lock the toe when it ***doesn’t move too much*** in the main animation

Slope Warping



Slope Warping

Carefully tweak ground smoothing

Smoothly pull the hips down

Slope Warping

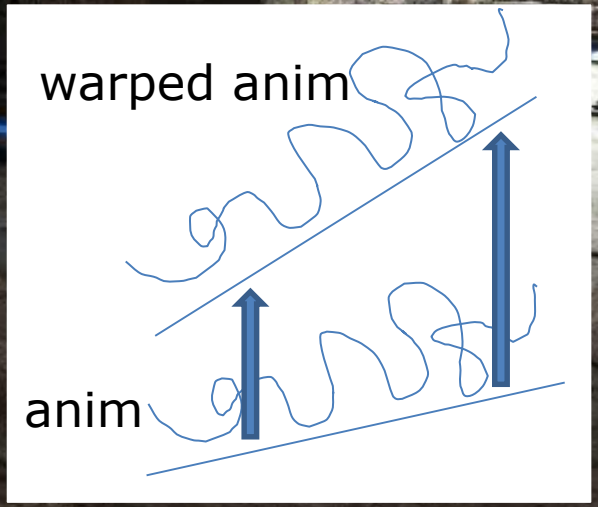
Prefer smoothness over absolute penetration prevention

Don’t break the pose:

Never hyper-extend the knee

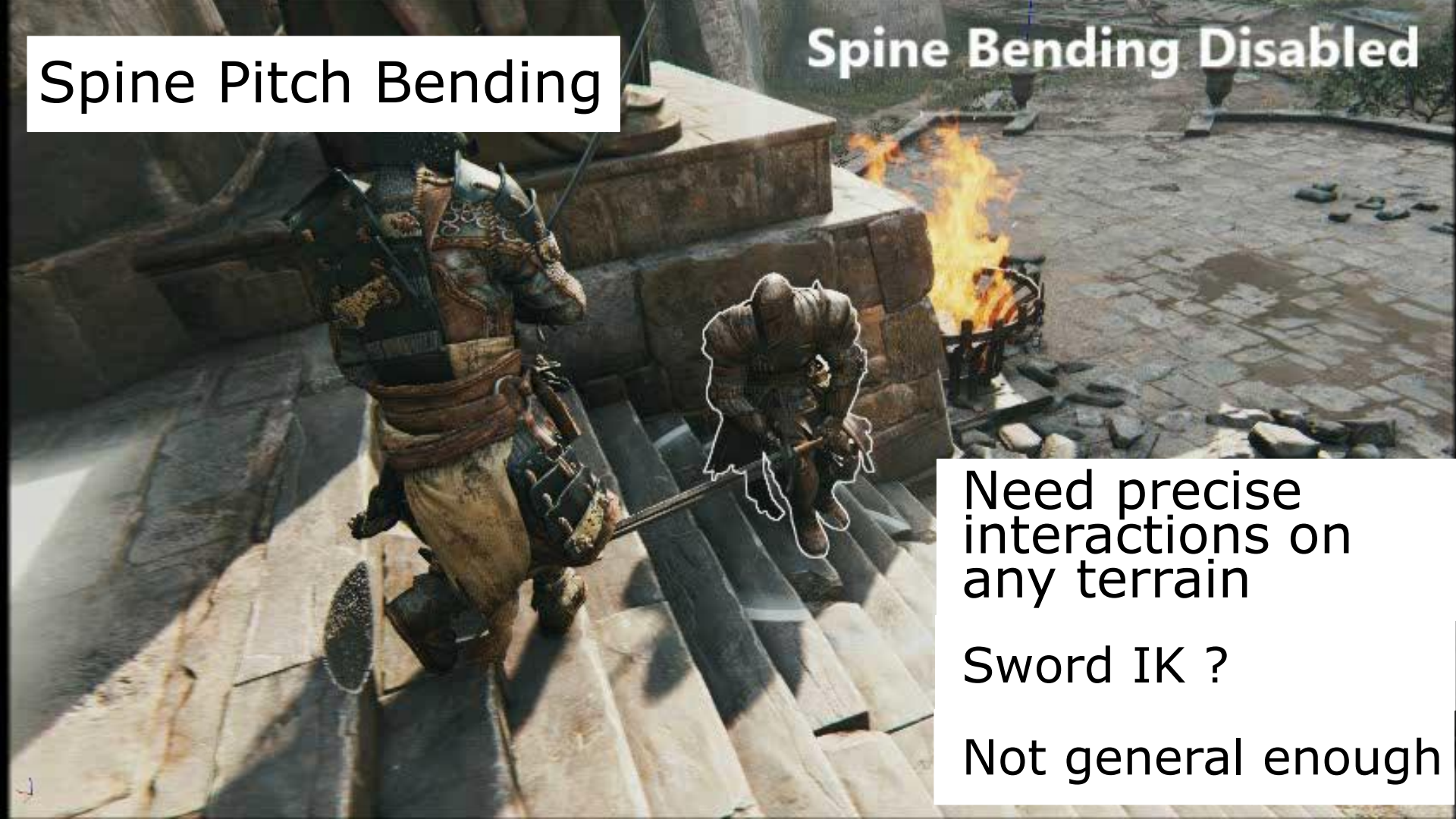
Slope Animations

Slope anims are automatically chosen by 3D trajectory matching



IK only compensate for what is missing from the anim

Spine Pitch Bending



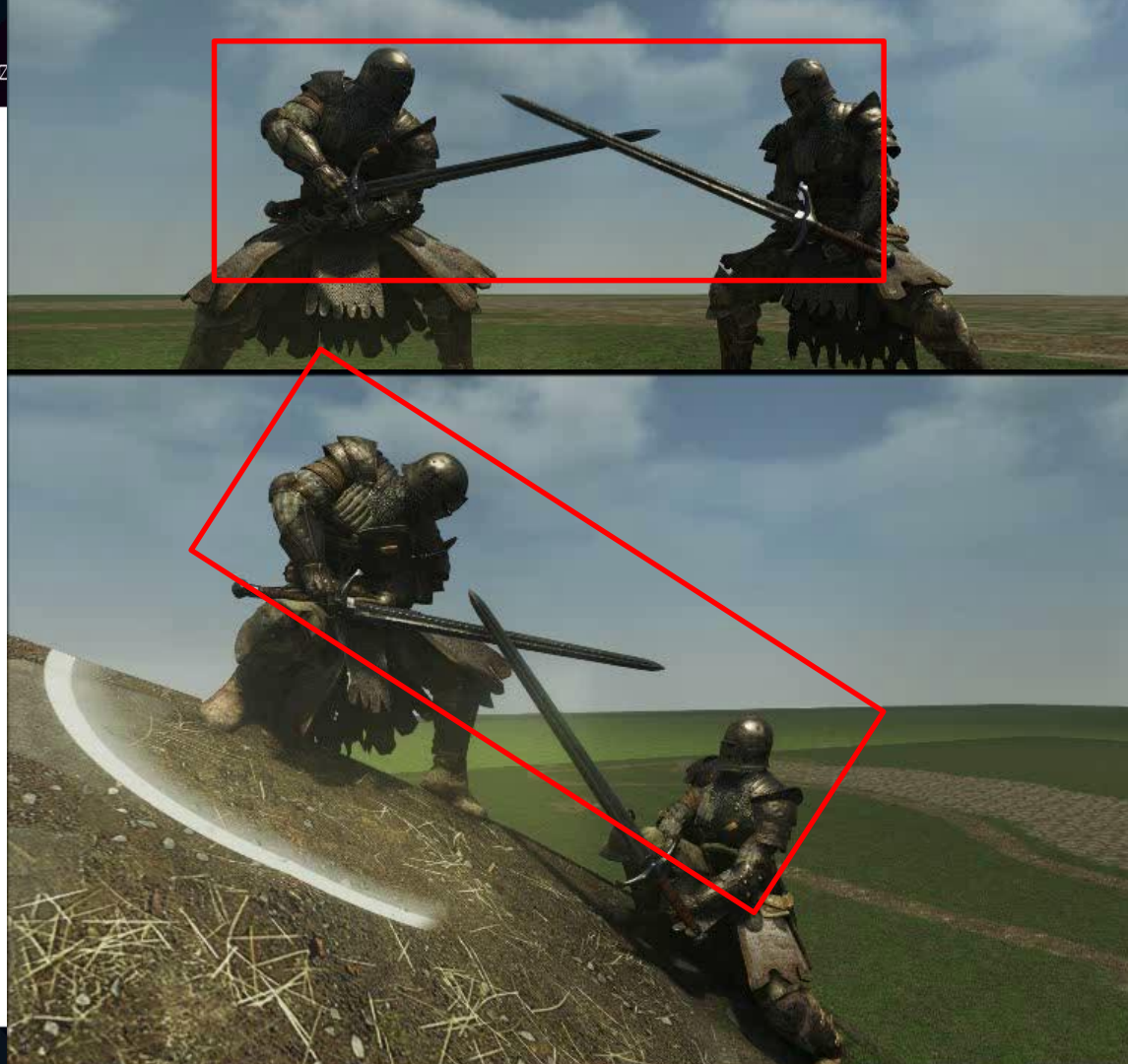
Need precise interactions on any terrain

Sword IK ?

Not general enough

Spine Pitch Bending

Keep whole upper body in sync



All together now

Future Work

- Match more stuff
  - Surfaces
  - Interaction Partners

Conclusion

- **Motion Matching is a simple idea, that helps us reason about movement *description* and *control***

Conclusion

- It’s also a new type of animation system, with three advantages:
  - High quality
  - Controllable responsiveness
  - Minimal manual work

Final Shower Thought

Let’s just markup the mocap with the *inputs* that should trigger the moves.

And generate everything automatically…

UBISOFT

Questions?

Thanks to
Jonathan Bapst
Michael Buttner
Sebastien Comte
Serge Doré
Sean Gomez
Xavier Guilbeault
Yves Jacquier
Dany Joannette
Karim Kochen
Marc-Antoine Lamarche
Xavier Lemaire
James Michel
Pascal Mimeault
Khai Nguyen
Guillaume Picard
Alexandre Pichette
Jason VandenBerghe
all *For Honor* team
and all mocap actors...

More Questions?

Meet me at the Ubisoft Lounge West Hall 2nd floor from 2PM to 3PM