# CA-MIRI
# Computer Animation
# 2nd Project

## Exercise 1 - Locomotion

**Alejandro Beacco Porres**
**alejandro.beacco@upc.edu**

# Exercise Statement

- 1-Obtain and import properly 1 animated character and several locomotion animations.
- 2-Create an animation controller with a locomotion *blendtree*.
- 3-Create a motion tracker and feed parameters from the tracker to the *Animator* for proper locomotion.
- 4-Drag your character around and/or move it with the keyboard: the character must be properly animated depending on the velocity vector. Handle abrupt changes in the velocity vector.
- 5-Create an orientation manager and introduce automatic changes in orientation.
- 6-Add an option/key to fix orientation.

# Outline

- 1-Obtaining and importing assets
- 2-Animator controller
- 3-Tracker
- 4-Locomotion script
- 5-Orientation manager
- 6-Fixed orientation
- 7-Debug visuals

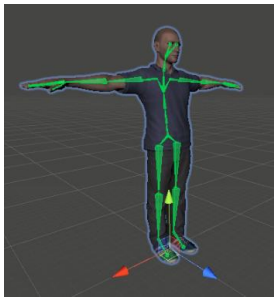
# 1-Obtaining and importing assets

- Character(s) model(s)
  - Rigged model: mesh, skeleton and skin
  - Humanoid (for retargetting)
- Animations
  - Walking and running in different directions
    - Forward, backwards
    - Strafe lef, right
  - Loop cycles
- Recommended FBX format
  - .max, .maya, .blend, need the correspondent program installed for proper import

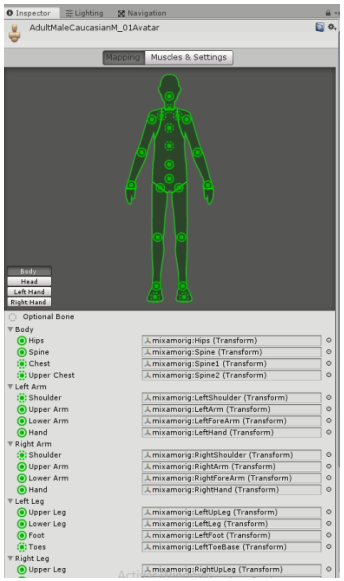
# 1-Obtaining and importing assets

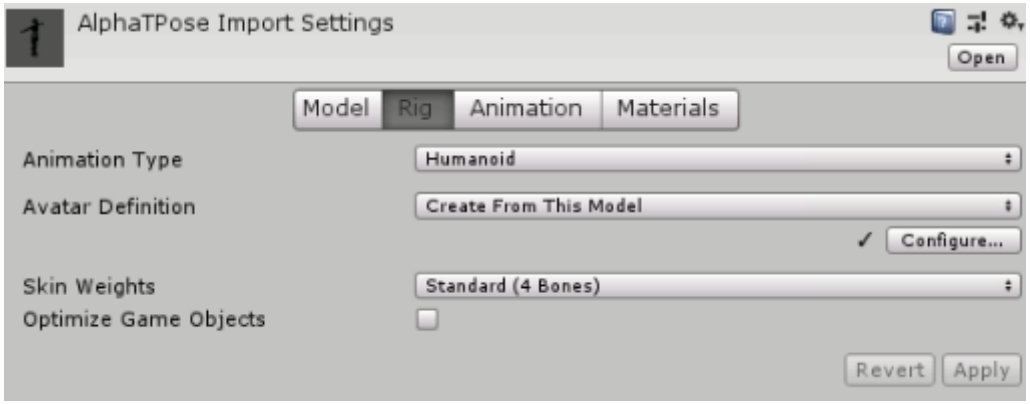
- Ressources:
  - Gonzalez-Franco *et. al.* ***"The Rocketbox library and the utility of freely available rigged avatars."*** *Frontiers in Virtual Reality. 2020.* *DOI: 10.3389/frvir.2020.561558*
    *https://github.com/microsoft/Microsoft-Rocketbox*
    - Free characters for research and education
  - Mixamo: https://www.mixamo.com
    - Free characters and animations
    - Free autorig
  - Autodesk character generator: https://charactergenerator.autodesk.com/
    - Free character editor with student account
  - Unity asset store: https://assetstore.unity.com/
    - Free characters and animations
  - Render People: https://renderpeople.com/free-3d-people/
    - Some free characters
  - Axyz Design: https://secure.axyz-design.com/en/shop/category/free-3d-people
    - Some free characters
  - CMU: http://mocap.cs.cmu.edu/
    - Mocap animations
    - Version for Unity: https://drive.google.com/file/d/1TmVkHuyEdIBWB9McM_xZfK42fJoh-LtO/view?usp=sharing

# 1-Obtaining and importing assets



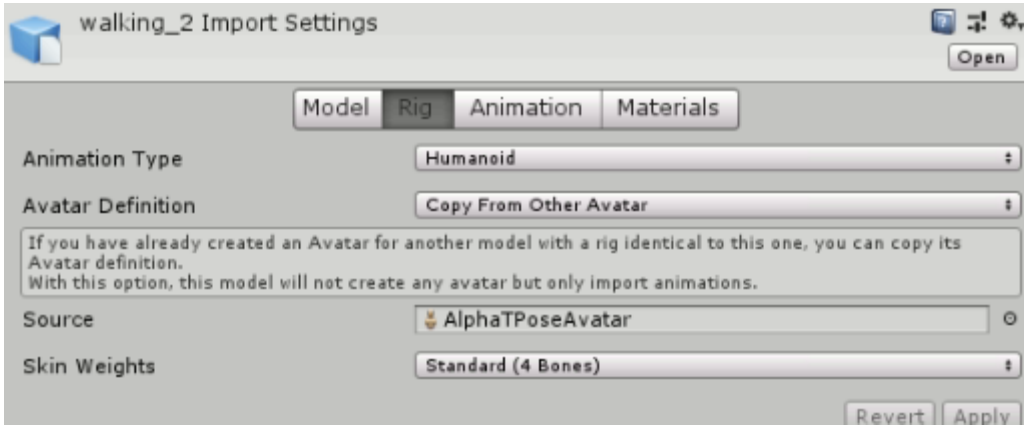
- Importing character
  - Rig
    - Animation Type = Humanoid
    - Avatar Definition = Create From This Model
      - Character must be in T-Pose!
      - Otherwise → Configure → Configuration scene where you can set the proper T-Pose and joints mapping of the avatar
  - Will generate one Avatar



# 1-Obtaining and importing assets

- Importing animations
  - Rig
    - Animation Type = Humanoid
    - Avatar Definition = Create From This Model
      - Only if asset contains also character in T-Pose!!
    - Avatar Definition = Copy From Other Avatar
    - Source = <Avatar for which the animation was created>

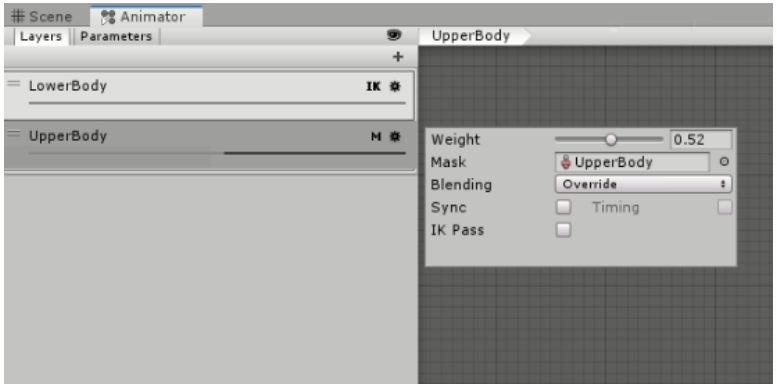


# 1-Obtaining and importing assets

- Importing animations
  - Animation
    - Import animation checked
    - Clips → List of clips in file
    - Start and end frames
    - Loop time (checked)
      - Loop pose → matches start and end frame
      - Cycle offset → to better align clips if necessary
    - Root transform rotation
      - Bake Into Pose checked
    - Root transform position (Y)
      - Bake Into Pose checked
    - Root transform position (XZ)
      - Bake Into Pose unchecked!
  - Experiment with other import options / settings and see what happens

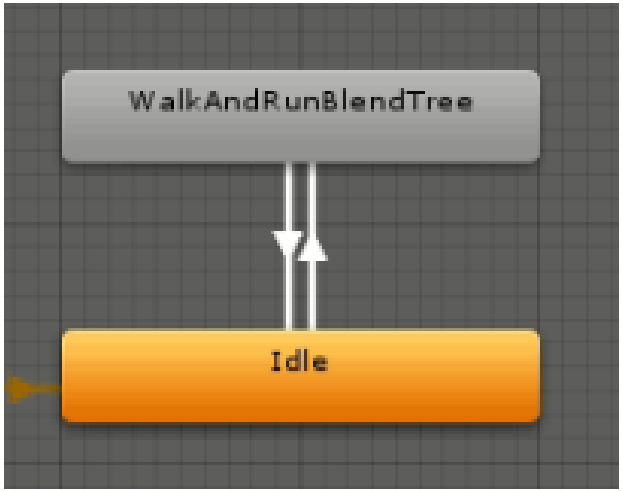
# 2-Animator controller

- Animator controller
  - State machine
    - States
    - Transitions
  - Different layers
    - Masks
    - Override / additive
    - IK pass
    - Weight
    - Default state
  - Sub state machines
    - Entry / Exit states
  - Parameters
    - Int / Float / Bool
    - Trigger → only once!

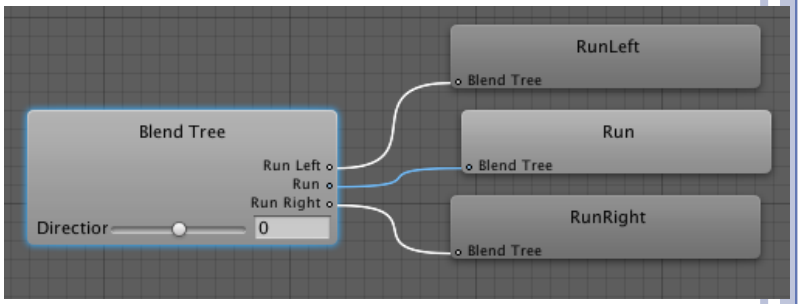


# 2-Animator controller

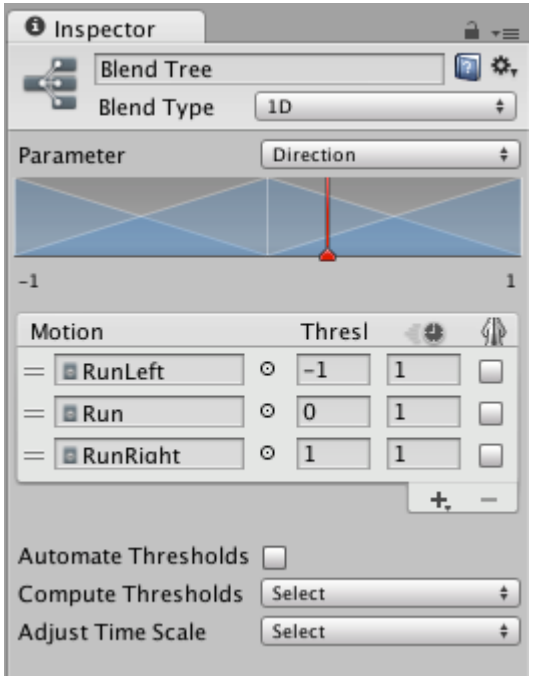
- New locomotion animator controller
- Idle state
- Locomotion (walk/run) blendtree
  - Add locomotion clips
  - Equilibrated in all directions (speeds)
- Transition from Idle to Locomotion
  - When?
- Transition from Locomotion to Idle
  - When?
- We need to use and send parameters from scripts

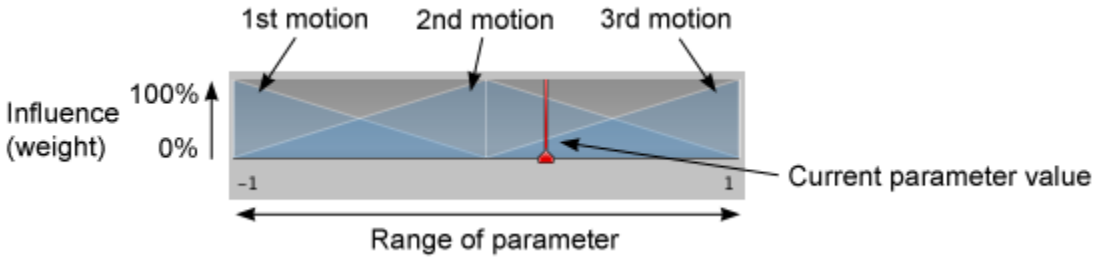


# 2-Animator controller Blendtrees



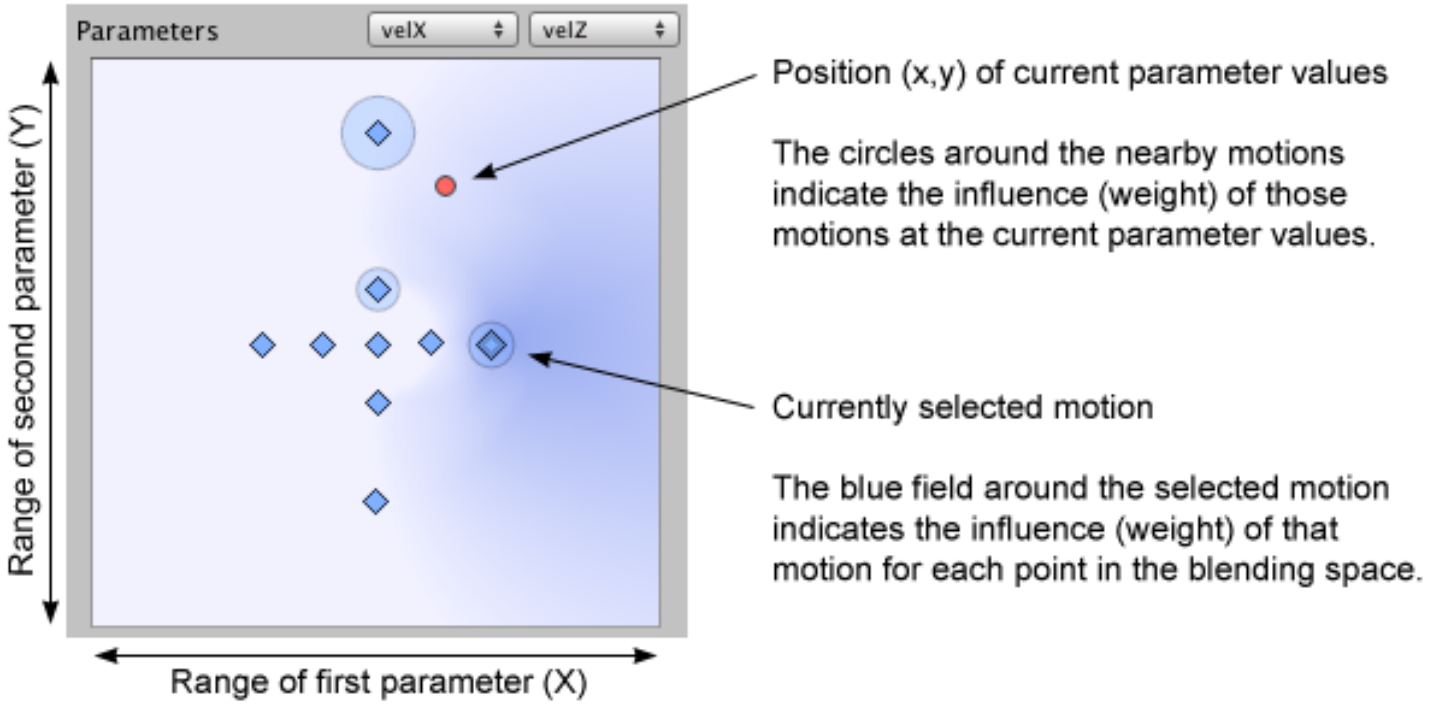

- https://docs.unity3d.com/Manual/class-BlendTree.html
- 1D
  - Linear blend with a single parameter between different motions



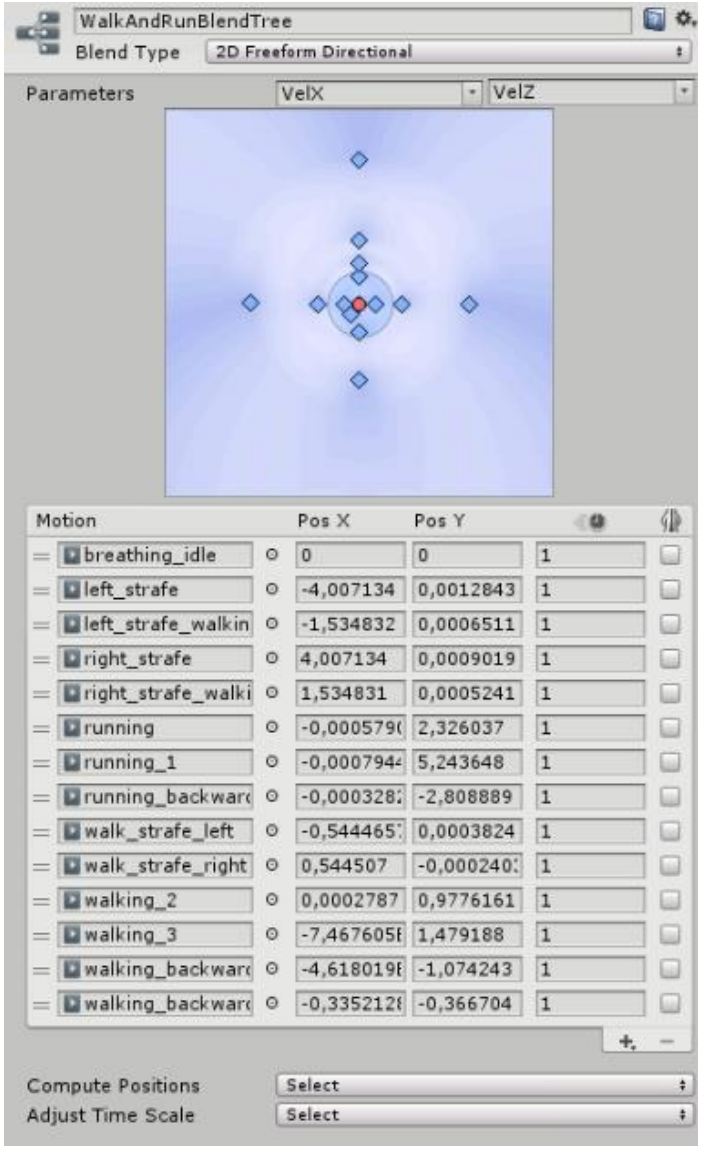
# 2-Animator controller Blendtrees

- 2D Blending:
  - Blends the child motions according to two parameters (X and Y).



# 2-Animator controller Blendtrees

- 2D Simple Directional
  - Different directions (walk forward, backward, left, right…)
  - Not multiple motions in the same direction (no different speeds)
  - Optional idle motion (0,0)
- **2D Freeform Directional**
  - Different directions
  - Can have multiple motions in the same direction (walk forward, run forward)
  - Should always include a single idle motion (0,0)
- 2D Freeform Cartesian
  - Motions do not represent different directions
  - X and Y can represent different concepts (angular speed, linear speed, …).



# 2-Animator controller Blendtrees

- Direct
  - Directly control the weight of each node
  - Allows you to map animator parameters directly to the weight of a BlendTree child
  - Useful if you want to have exact control over the various animations that are being blended rather than blend them indirectly using one or two parameters

# 3-Tracker

- We need to know how the characters is moving in the world
  - Independently from what is moving it
    - AI, simulation, user input, etc.
  - Track displacement with a new Tracker component

- Tracker component
  - Update every frame
    - Update? LateUpdate? FixedUpdate (physics)? → Check the differences
  - Keep track of:
    - Displacement vectors (world and local)
    - Speed
    - Forward vector
    - Anything you could need

# 4-Locomotion script

- Component that relates tracked information with the animator to synthesize proper locomotion animation
  - Reads data from Tracker
  - Sets animator parameters
  - Every frame: after Tracker update

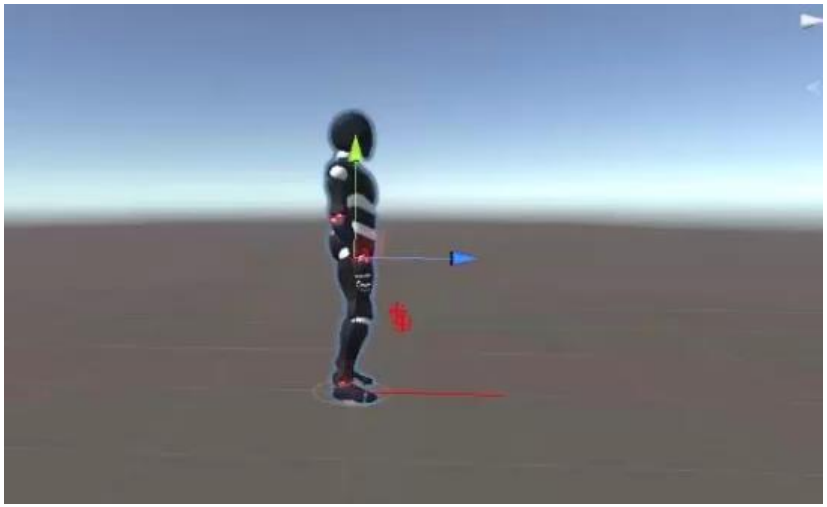
- Examples:
  - _animator.SetBool("Move", _tracker.getSpeed() > 0);
  - _animator.SetFloat("VelX",_tracker.getVelocity().x);

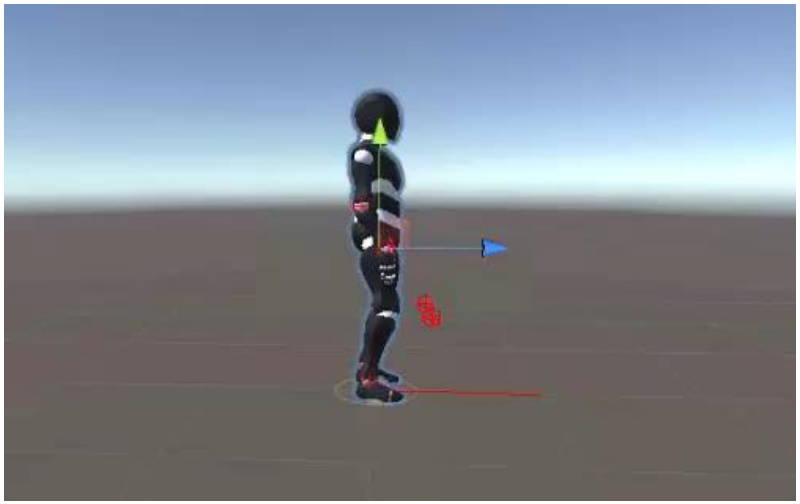
- Character should be animated with a smooth and natural animations that correspond with that displacement

# 4-Locomotion script

- Abrupt changes in the direction or speed of the displacement vector
- Intermittent oscillations between different values
  - Intermittent changes in animation

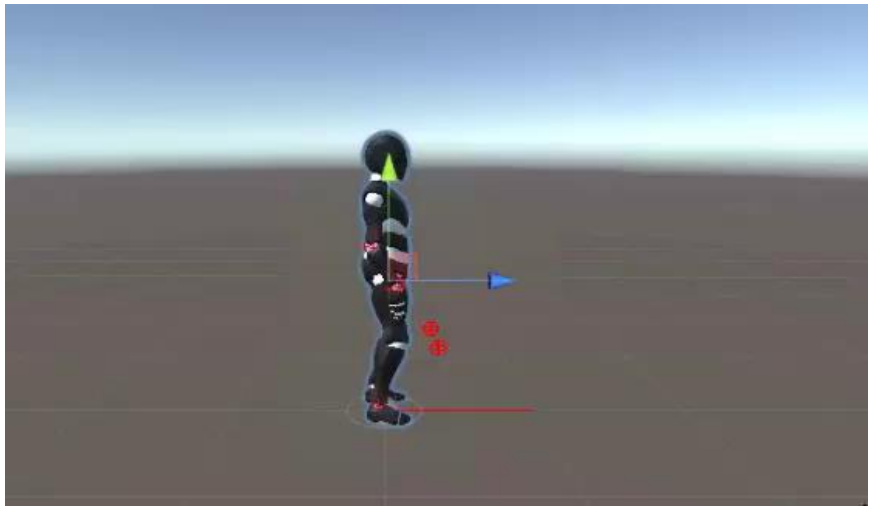
- Handle this cases:
  - Smooth the input displacement vector

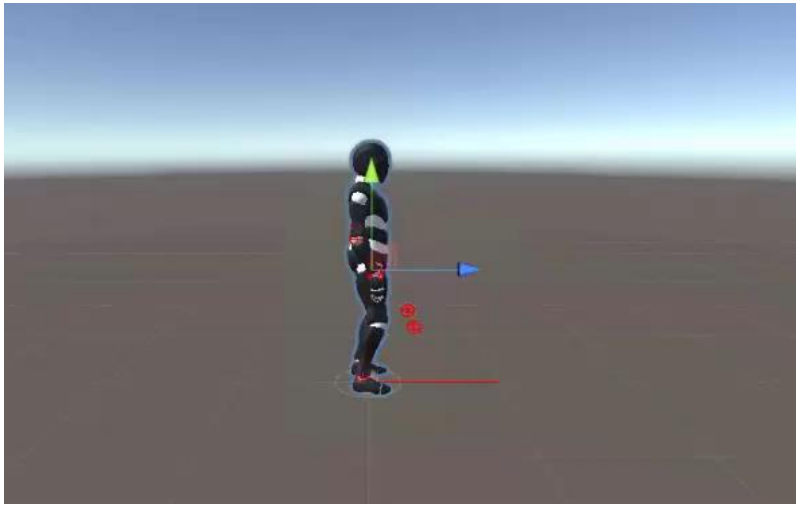




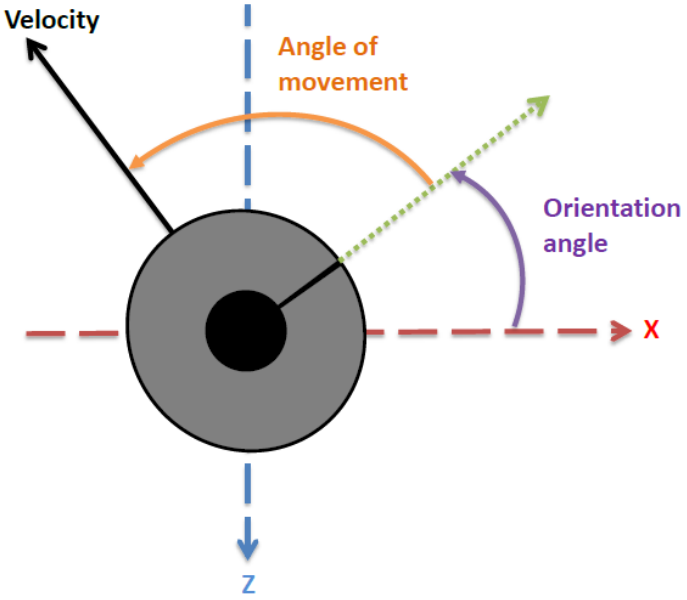
# 5-Orientation manager

- Input displacement vector can change abruptly
- Orientation should not
- We should smoothly turn or lerp the orientation (or forward vector) of our character
  - Use an input orientation smooth factor

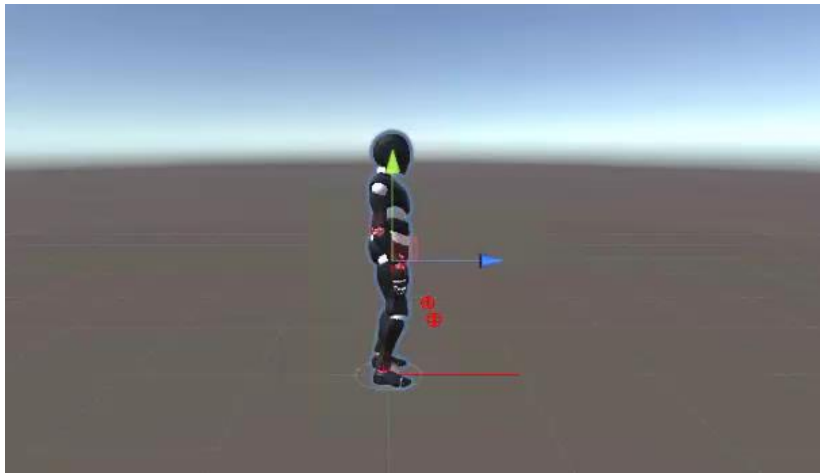


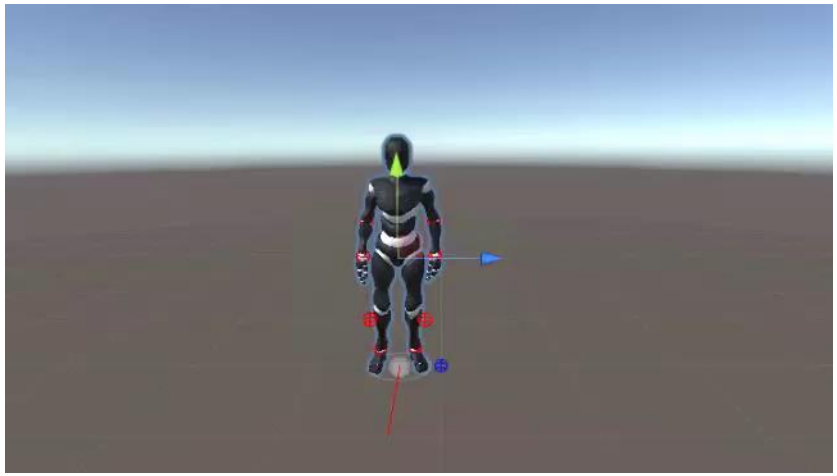


# 6-Fixed orientation




- Character can move in directions he is not facing
- Add option to fix the character orientation
- Your clips and your controller should handle such movements



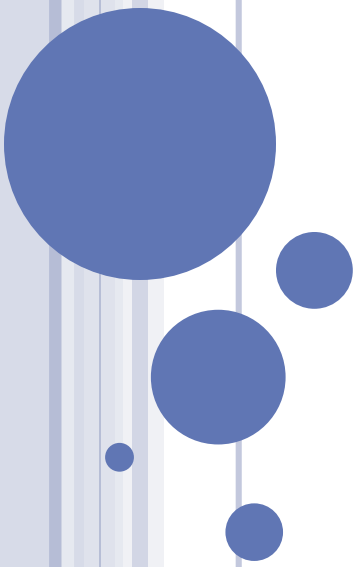


# 7-Debug Visuals

- bool debug atribute
  - Enable / disable debug visuals

- void OnDrawGizmos( )
  {
  // Draw any debug visuals here
  }

- Gizmos.Color = Color.red;
- Gizmos.DrawSphere(position, radius);
- DrawLine, DrawRay, DrawCube, …
- https://docs.unity3d.com/ScriptReference/Gizmos.html



# CA-MIRI
# Computer Animation
# 2nd Project

# Exercise 1 - Locomotion

## get to work!