Milestone 1 – Character Animation

This is an *INDIVIDUAL* assignment.

***** SPECIAL NOTE *****
Please remove all unnecessary assets from the projects you are submitting.

Due Date

January 27 2017 @ 11:55 PM

Late Policy

Starting with Milestone 1, we will use a 2^(*n*+1) late policy point penalty, where *n* is the number of days late. For example, 1 day late is a reduction of 2^(1+1) or minus 4 points. Assignments are considered late if submitted any amount of time beyond the due date/time. Each 24 hours late beyond the due date determines the number of days late. More examples: 2 days late is minus 8, 3 days late is minus 16, 4 days late is minus 32, 5 days late is minus 64 (no assignments accepted any later). Note that this late policy does not apply to assignments that involve in-class presentation, and is subject to change for any specific assignment.

Description

For this assignment you will be creating a 3D humanoid character that can walk and run around an environment.

You will be obtaining (or creating) a rigged 3D human character model from Mixamo, applying a variety of locomotion animations with root motion, and combining these assets with a Mecanim animation controller (Unity's state-based animation blending system). This character will transition smoothly between walking and running according to user input. You will also configure an appropriate chase camera (or write your own). Lastly, you will find an appropriate demonstration level for your character to move around in (or create your own).

Mechanim is Unity3d's animation system. It combines Blend Trees and Finite State Machines (FSMs) to support creation of fluid animations for characters. Mechanim can also be leveraged to work with non-humanoid objects in the world and even the GUI system.

Requirements

(Please note that this assignment will serve as the foundation for future assignments, including environment interaction and physics simulation as well as artificial intelligence.)

For this assignment you will need to complete the following requirements:

Obtain (or create with Mixamo Fuse) a rigged 3D character model from Mixamo (https://www.mixamo.com) that you feel might be useful in the future (e.g. upcoming assignments). Even a character that will not be player-controlled later could be useful for a non-player character (NPC). **Note that you MUST use a Mixamo model for this assignment. However, you can optionally use animations from other sources.** When in doubt about what is allowed, always contact your TA!
10 pts

Utilize a chase camera. Your chase camera can come from the Unity standard assets, tutorials, etc. Or write your own. In either case, make sure you configure it so that it works appropriately with your character. Please document your source in your submission readme file.
5 pts

Utilize an appropriate level. Your level you use should at minimum provide a variety of terrain to traverse, including obstacles that block the character's path, or require climbing to navigate. Specifically, your level should have hills or ramps of various angles. Lastly, your level should provide opportunities for your character to fall from high places. If you want to build your own, consider: ProBuilder Basic
https://www.assetstore.unity3d.com/en/#!/content/11919
5 pts

Create a Mechanim Animation Controller for the model and animations that:

- Contains at least 2 floating point input parameters defined for the controller (e.g. forward/backward [-1.0,1.0], left/right [-1.0,1.0]). These parameters represent continuous analog-style input values. (**5 pts)**
- You must also have animations with root motion for the following (at minimum):
    - Walk forward
    - Run forward
    - Walk backward
    - Run backward
    - Run left forward
    - Run right forward
    - Walk left forward
    - Walk right forward
    - Run left backward
    - Run right backward
    - Walk left backward
    - Walk right backward
    - Idle (standing in place)
    - Turn left in place
    - Turn right in place
    - Falling

  (10 pts)
- From the default/idle animation state, the character animation transitions to (and possibly through) various animations using the values of the mecanim parameters discussed above. You must implement a combination of animation states and blend trees to support the locomotion speeds from walking to full run. Also, you must support various turn angles from no turn (straight ahead) up to full turn. Additionally, your character should be able to fall from high places. **(30 pts)**
- Position translation of the avatar must occur via root motion (animation of relative position and orientation). **(5 pts)**

Implement a keyboard input testing script that sets the parameters you use in the animation controller based on keyboard inputs from the player. For demonstration purposes, use the keyboard numerals to set different global speeds and use keys W,A,S,D,Q,E to move. This will create a standardized pseudo-analog way for the graders to assess your character control. As is, this approach is a pretty cumbersome way to control a character. It is only in place to make grading easier.

You may want to additionally support video game controllers such as the PS4 controller, but that isn't required for this assignment.

Test control details:
"1" sets slowest speed (walking speed). "2" through "9" set progressively higher speeds (blend between walking and running at appropriate ratio). "0" is 100% full speed (full run).

Direction controls are defined as:
W-forward
S-backward (not required except for extra credit)
A-hard left turn
D-hard right turn
Q-light left turn
E-light right turn

If the player presses W (forward) then the character should walk/run forward based on the current speed setting (set by numerals). If the player presses W+A then the character walk/run forward and turn left at the maximum turn rate. W+Q will result in the character walking/running forward while turning slightly left. If the player presses Q, the character will turn in place slowly to the left. If the player presses A, the character will turn in place quickly to the left.
10 pts

The overall aesthetics of your character control should meet the following expectations:

- Demonstrate smooth transitions between all animations and animations states
- Character's feet should appear to land with every step across all blend ratios
- No visible moon walking or feet sliding along the ground
- No teleporting, glitching, etc., that breaks the illusion of a human moving naturally in space

20 pts

Extra Credit Opportunities:

Do one of the options below for **UP TO** 5 points. Do both of the options below for **UP TO** 10 points. Be sure your readme clearly documents that you have completed extra credit and want the grader to assess. Actual credit awarded is determined by the grader, specifically determining if the spirit of the extra credit task is met and the interaction and aesthetic requirements are met.

1.) Implement jump that works from arbitrary heights as well as falling from arbitrary heights (e.g. walk off ledge without pressing jump) both with simulated gravity acceleration. You will need to abandon the use of root motion while in these animation states in order to work with differing heights. The horizontal distance of your jump should be directly related to the horizontal velocity at the time of the jump. For instance, if standing in place the jump goes straight up. Also, a walking jump is shorter than a running jump. Please use spacebar for jump. Also, make sure your map provides the appropriate test environment for jumping.
2.) Add an environment-specific animation such as pulling up and over an overhead ledge or pressing a button on a wall, etc. Please email instructor(s)/TA(s) to confirm your choice is appropriate.

Resources:

- Setting up the model in Unity - https://www.youtube.com/watch?v=4oloh4x7qH0&feature=youtu.be

- Scripting and Mecanim with animations - https://www.youtube.com/watch?v=HjkFfkHCgCc&feature=youtu.be&list=PLVjHjbbAP9Bsgib7NpfYWAL6fxIa2IF79

- Adobe Mixamo: https://www.mixamo.com
- Mixamo's Animation Overview - https://www.youtube.com/watch?v=mskE0ywnGMA&feature=youtu.be&list=PLVjHjbbAP9Bsgib7NpfYWAL6fxIa2IF79
- Mixamo's Advanced Anim Tutorial - https://www.youtube.com/watch?v=mBbXPB_6SWs&feature=youtu.be
- Character Animation – (old Unity 4 tutorial, but still useful) https://www.youtube.com/watch?v=Xx21y9eJq1U

The following aren't for this assignment, but might be useful if you want to do your own modeling/rigging/animation in the future.

- Unity Animations From Blender Rigify - http://docs.unity3d.com/Manual/BlenderAndRigify.html
- More Blender: http://zakjr.com/blog/blender-to-unity-workflow-part-1
- Blender - https://www.blender.org/
- Autodesk Education Free Software - http://www.autodesk.com/education/free-software/all

Tips:

General:

- Pay careful attention to any tutorial instructions you follow. Often if you miss one little step, you'll have a lot of trouble.

Animations in General:

- All blended animations must be similar (e.g. start with the same foot and take the same number of steps). Otherwise, blending will give wonky results.
- You only need a turn left (or a turn right) animation for the directional requirements. You can duplicate the turning animation (Edit->Duplicate), then set it to be mirrored. Lastly, adjust "Cycle Offset" under the Animations tab such that the correct foot takes a step first.

Mixamo:

- Export your character model as FBX for Unity. It should come with a TPose "animation" that doesn't actually animate.
- When you add the model to your Unity assets, immediately change the rig to Humanoid and the avatar definition to "Create from this Model"
- Export each of your animations as FBX for Unity.
- When you add the animations to Unity assets, immediately change the rig to Humanoid and the avatar definition to "Create from Other Avatar". Drag the avatar from the original model to the "Source" avatar field.
- For all animations that are intended to be blended for on-foot locomotion (e.g. walking, running), set the "Loop Time"==true. You will also probably want to set "Root Transform Position (Y)" to "Bake into Pose"==true. This will allow natural falling behavior.

- For straight ahead locomotion you can set “Root Transform Rotation” “Bake into Pose”==true which can sometimes smooth out the chase camera from rocking back and forth.

Blendtree:

- A 2D Freeform Directional Blendtree might work well for you for locomotion. I make one for the forward variants of locomotion (and turn in place) and a separate one for backward variants (also with turn in place).
- The inspector is great for debugging blendtrees.
- Previewing a blendtree might work just fine, then look bad when running the game. If so, you probably didn’t set “Loop Time”==true as described in Mixamo section above for ALL animations being blended.

Mecanim:

- Be wary of transition settings between states. These can create long delays between animations with the default crossfade settings
- Consider enabling “Foot IK” on your animation states to get better foot placement

InputManager:

- Read up on the InputManager, in particular the filtering effects that cause delays in input values changing (e.g. gravity, sensitivity, snap, etc.)

Animator Component:

- Make sure “apply root motion” is turned on

Putting your name of the HUD:

How to make a screen space overlay canvas with text:
File Menu: GameObject>UI>Canvas
(defaults to screen space overlay, which you want)
select the canvas in scene hierarchy
File Menu: GameObject>UI>Text
select the text in scene hierarchy
optionally set your view to 2D mode in your scene view
set min/max anchors of text's rect transform to 0,1 (top left corner)
play with font type, size, color until appropriate
drag text box to top left; resize box to fit your name if it's getting cropped
(turn 2d mode back off when you are done)

Submission:

You should submit a 7ZIP/ZIP file of your Unity project directory via t-square. **Please clean the project directory to remove unused assets, intermediate build files, etc., to minimize the file size and make it easier for the TA to understand.**

The submissions should follow these guidelines:

a) Your name should appear on the HUD of your game when it is running.
b) ZIP file name: <lastName_firstInitial>_mX.zip (X is milestone #)
c) A /build/ directory should contain either a Windows or OSX build of your game.

d) Readme file should be in the top level directory: < lastName_firstInitial >_m1_readme.txt and should contain the following
    i. Full name, email, and TSquare account name
    ii. Detail which requirements you have completed, which are incomplete, and which are buggy (be specific)
    iii. Detail any and all resources that were acquired outside of class and what it is being used for (e.g. Asset Bundles downloaded from the Asset Store for double sided cutout shaders, or this file was found on the internet has link http://example.com/test and does the orbit camera tracking).
    iv. Detail any special install instructions the grader will need to be aware of for building and running your code, including specifying whether your developed and tested on Windows or OSX
    v. Detail exact steps grader should take to demonstrate that your game meets assignment requirements.
    vi. Which scene file is the main file that should be opened first in Unity
e) Complete Unity project (any file you acquired externally should be attributed with the appropriate source information)

Submission total: (**up to 20 points deducted** by grader if submission doesn't meet submission format requirements)

Be sure to save a copy of the Unity project in the state that you submitted, in case we have any problems with grading (such as forgetting to submit a file we need).