CS7GV5-A-SEM202-201920 REAL-TIME ANIMATION

Assignment 4 – Interactive Movie with Animated Character

- **Must have 3-dimensional objects and views:**
  The scene contains a 3D terrain which is generated using ***Terrain*** gameobject of Unity. The textures, tress and other details are added explicitly.

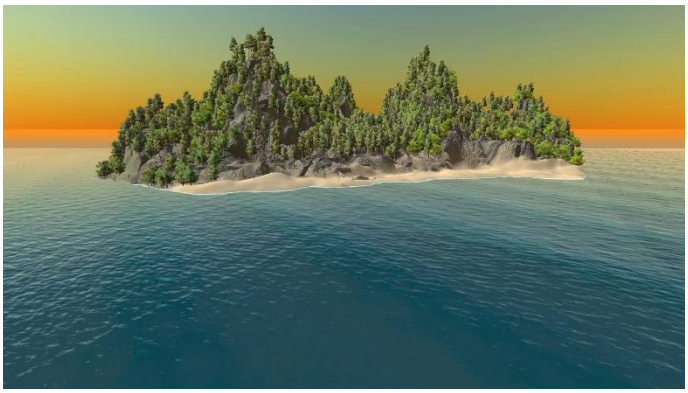


Figure 1. 3D view of generated terrain

- **Must be 30 seconds or longer:**
  The duration of the movie is ***00:01:54***.

- **Must have a non-linear animated camera path.**
  The camera path is animated using the ***Animator + Animation*** component. I have created different frames for different positions which are glued together to make a non-linear camera path. Three cameras are used for better views.

- **Must have one reasonably realistically moving articulated animated character, as part of the storyline and visible during the camera path:**
  The movie contains two animated characters. The Boy is the protagonist whose presence is throughout the movie and the Bee appears for a small time. Both characters have their own animations.
  The Boy's animations are: ***Sitting and Standing Idle, Walking, Turning left and right, Throwing, Heads Up and Shocked Reaction.***
  The Bee's animations are***: Flying Idle and Flying Around the boy***.

Figure 2.The Boy and The Bee

- **Must have an interactive element (e.g., character reacts to user throwing a ball, etc.):**
  There are several things happening on key press:
  1. The boy is turning left when space bar is pressed.
  2. Crowd Simulation is triggered along with boy's reaction to it. (B)
  3. Cameras are switched. (1-Main camera, 2-Second camera, 3-Boy's child camera)
  4. Skybox is changed (K for night, L for day)

- **Must clearly demonstrate the following principles of animation in the context of the movie.**
  - **Squash and Stretch:**
    The ball in the opening scene clearly demonstrate stretch and squash. This animation is made using five frames, where the first and last frame are the original positions of the ball, the second and fourth frames are for stretch, where the ball is in mod air and the third slice is for squash when the ball hits the ground.

Figure 3. The ball demonstrating Stretch and Squash.

  - **Ease in, Ease out:**
    This principle is demonstrated when the Bee enters and exits the scene. The entry is done quickly whereas the remaining motion is comparatively slower, and the exit is again swift.
  - **Arcs:**
    The motion of Bee around the boy shows Arcs. As the intermediate frames are interpolated to give a smooth motion. The motion would have extremely articulated if only the position was changing with any rotation or intermediate frames.
  - **Anticipation:**
    The boy stretches his whole body and arms before throwing the stone.

Figure 4. Anticipation before throwing the stone.

- **Exaggeration:**
  As shown in Figure 2, The boy gets shocked when the bee suddenly enters the vicinity. Also, the boy also turns around happily when the bee revolves around him.
- **Additional implemented principles:**
  I've also tried implementing ***Staging***, where only one thing happens on the screen at a time, so that the audience have a clear vision.

➢ Additional Features:

➢ Motion Capture:

I have downloaded the animations for Boy from Mixamo.com and used them with ***Animation + Animator.*** As we can see on the lower portion of the screen, there are different animations associated with a single avatar.

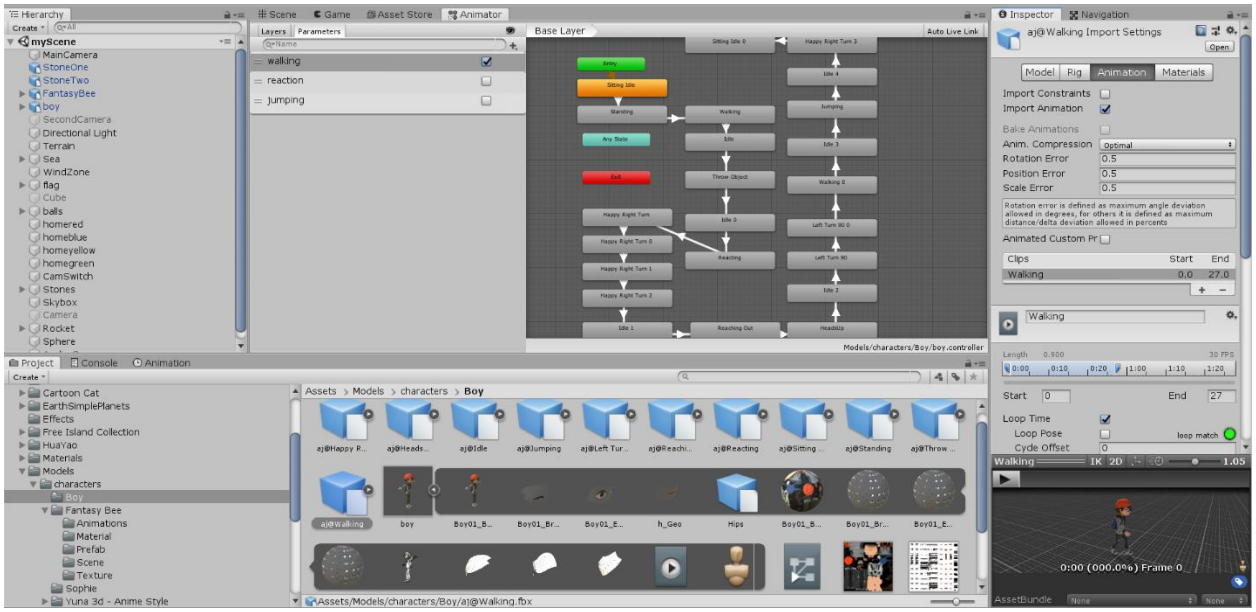


Figure 5. Motion Capture and State Machines

➢ Motion State Machines:

```
void Update()
{
    if(Input.GetKeyDown(KeyCode.Space) && !reaction)
    {
        animator.SetBool("reaction", true);
        reaction = true;
    }
    else if (Input.GetKeyDown(KeyCode.Space) && reaction)
    {
        animator.SetBool("reaction", false);
    }
    if(Input.GetKeyDown(KeyCode.B))
    {
        animator.SetBool("jumping", true);
    }
}
```

Figure 6. Update function for Boy.

From Figure 5, the center window shows state machine for the Boy. Here we can switch from one animation to another. Blending between two animations over time and looping a single animation is also done using the same. Here, we can add conditions for switching from one position to another.

The value can be changed using scripts. To exemplify, I've used different Boolean variables for triggering different animations (Figure 6).

- **Motion Editing:**

The only editing I did on the downloaded animations was Adding an event in the throwing motion of the boy which triggers the stone to fall from the hand.

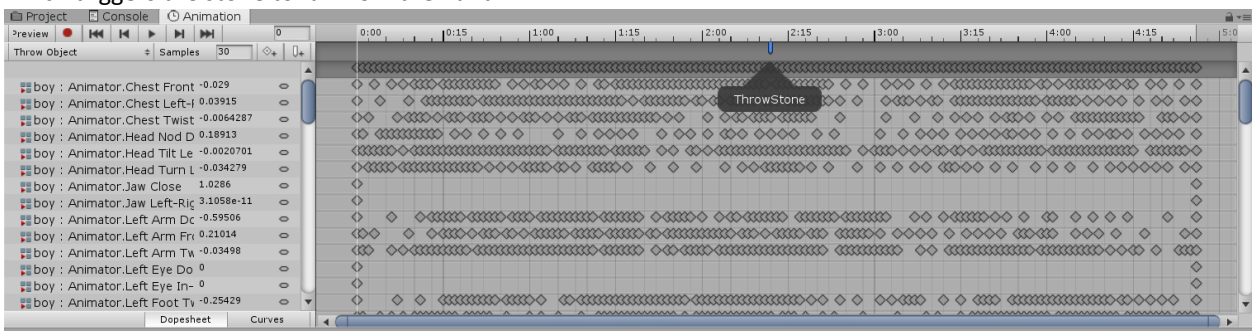


Figure 7. Throwstone Event for trigger

Apart from that, I created the stretch and squash animation of the ball and the whole bee's entry, exit and revolving around the boy from scratch using ***Animation*** tab.

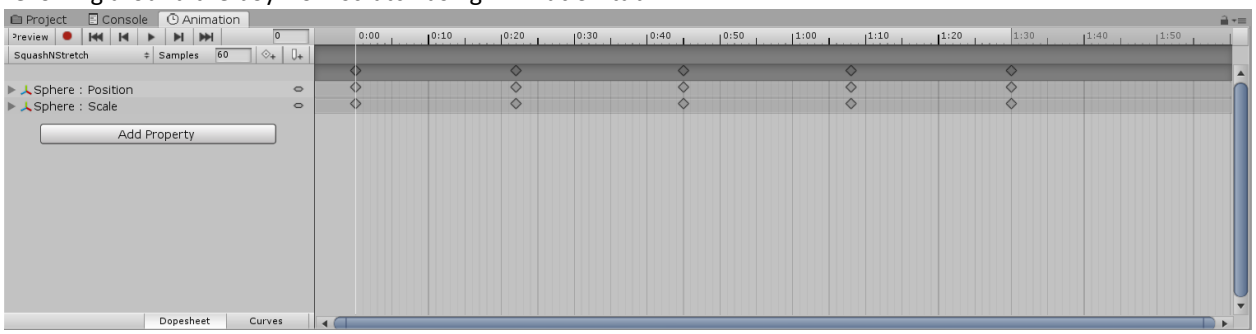


Figure 8. Stretch and Squash for Sphere.

- **Stylized Motion:**

I have added stylized motion by adding glowing particle trails to the spheres that were used for crowd simulation. This is done by adding ***Trail Renderer*** component in the Gameobject and giving it required properties.

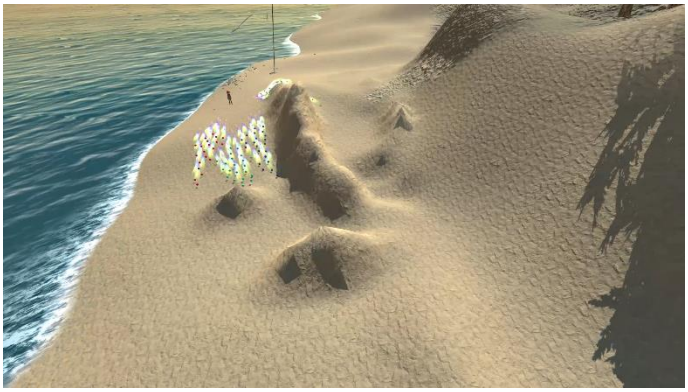


Figure 9. Trails for stylized motion.

- **Crowd Simulation:**

  There are two factors that were considered for doing crowd simulation:

  1. Bi-directional motion: when the path becomes narrower, the crowd tends to follow a line.
  2. Congestion: When more than 6 people are present at a place, turbulence is likely to happen, which will lead to some objects getting stuck.

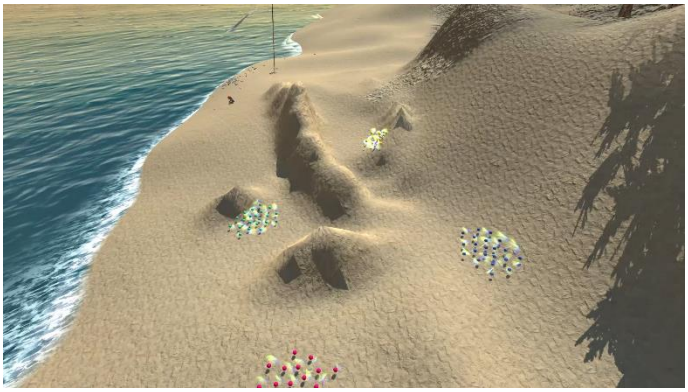
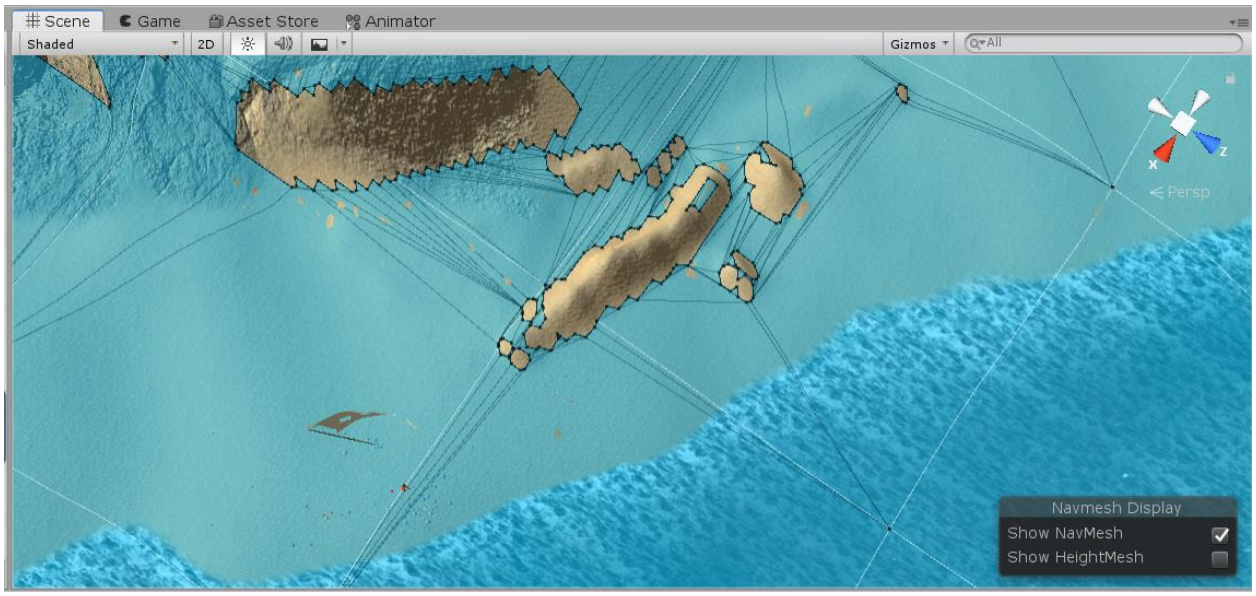


Figure 10. Simulated Crowd in their respective destinations.

- **AI:**

  I have used ***Navigation Mesh*** for the spheres to navigate their ways to their destinations. The blue part is where the balls can move, and the rest is restricted.



Now, the spheres try to find their own way home in this mesh. A script is used which used **UnityEngine.AI** for finding the path.

```
void Update()
{
    if (Input.GetKeyDown(KeyCode.B))
    {
        agent = this.GetComponent<NavMeshAgent>();
        agent.SetDestination(home.position);
    }
}
```

Figure 11. Update for crowd agents

- **Cloth Simulation:**

  This is done using ***Cloth component*** on an empty gameobject and using the required mesh. Here, I used an object file if flag and divided it into two components using ***Blender***, so that the pole and flag cloth are separated. Now I used the cloth mesh for the flag. I added constraints such that the cloth doesn't fall due to gravity and ***external acceleration*** is added for the wind.

Figure 12. Flag for Cloth Simulation

- **Advanced Particle System:**

  I have used three particle systems for creating the fireworks. Here, the first particle system acts as the parent of the other two and is triggered at the beginning. The other two acts as ***sub-emitters*** to the first one. The second particle system starts at the birth of a particle and creates the trail of the particle, which looks like a rocket going in the sky. The last system is activated at the death of a particle and prompts the explosions at the end. I have used maximum ***1000 particles*** because having three particles system was making Unity crash repeatedly.

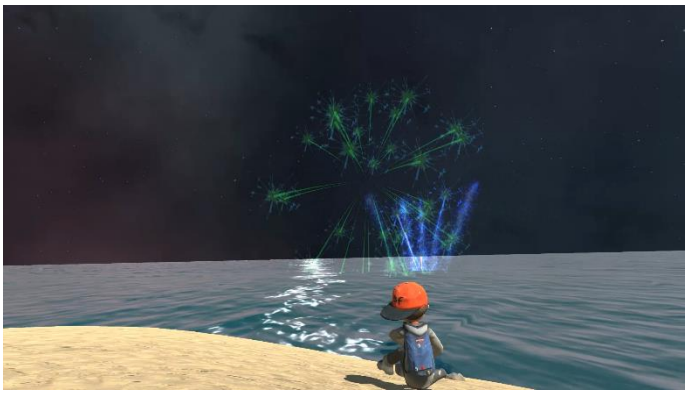


Figure 13. Particle system as Fireworks

- **Background Audio:**

  I have used ***Audio Source*** gameobject to add background music in the movie. Play on awake is turned on, so that the audio starts playing at the beginning of the movie.

- **Skybox change:**

```
if (Input.GetKeyDown(KeyCode.K))
    RenderSettings.skybox = sky2;
if (Input.GetKeyDown(KeyCode.L))
    RenderSettings.skybox = sky1;
```

Figure 14. Script for changing skybox

The skybox is changed on keypress as shown in Figure 14 and 15.

Figure 15. Showing both skyboxes in a single picture.

- **Physics on Objects:**
  As illustrated in Figure. 7, the ThrowStone event triggers a function. It calls this release function from Stone's script. Here, we are changing the parent bone of stone from boy's hand to null. A force is added in z-direction, so that the stone is moved forward.

```
public void Release()
{
    curVel = new Vector3(0, 1, 5);
    transform.parent = null;
    rigid.useGravity = true;
    transform.rotation = parentBone.transform.rotation;
    rigid.AddForce(curVel * 2, ForceMode.Impulse);
    Debug.Log("Throw");
}
```

Figure 16. Script for releasing the stone from boy's hand

Figure 17. Throwing the stone.

- References:
  - The Boy: https://www.mixamo.com/#/?page=1&type=Character
  - The Bee: https://assetstore.unity.com/packages/3d/characters/animals/fantasy-bee-135487
  - Animations: https://www.mixamo.com/#/?page=1&type=Motion%2CMotionPack
  - Night Skybox: https://assetstore.unity.com/packages/2d/textures-materials/sky/starfield-skybox-92717

- Standard Assets: https://assetstore.unity.com/packages/essentials/asset-packs/standard-assets-for-unity-2017-3-32351
- Animator: https://www.youtube.com/watch?v=gON_hhhvheI&feature=youtu.be
- Camera motion: https://www.linkedin.com/learning/unity-3d-essential-training/animation-basics-and-editors-in-unity?u=2217001
- Terrain Generation: https://docs.unity3d.com/Manual/script-Terrain.html
- Particle System: https://www.youtube.com/watch?v=nlLow7DbKkY
- Principles of Animation: https://www.youtube.com/watch?v=uDqjIdI4bF4
- Navigation mesh/ Crowd Simulation: https://www.youtube.com/watch?v=4Kj6YUPLWCw