Specification Compliance Report - Gaming Technologies

Zakirhussain Kutianawala - 210137839

+30%: **Physics**

- +5%: Appropriate use of Newtonian physics
  - ☑ Usage of Rigid Bodies
    - Rigid bodies can be found on my player character, NPCs and on moveable obstacles (cubes) used to demonstrate physics and mass on level 4 and 5
  - ☑ Correct application of impulses to bodies
    - An example of this can be seen when my player performs a jump/double jump.
  - ☑ Objects have appropriate mass quantities
    - The player object and cube used to activate the mass trigger have appropriate mass attached to them.
  - ☑ Game mechanics is physics-driven (e.g., immediate response to collisions).
    - The player or NPCs cannot phase through solid objects, The NPC projectile attack and Player use addForce to shoot and jump. Changing the mass of the projectile affects its speed and distance covered.

- +5%: Advanced Physics (multiple gravity areas / changing mass / *et ceteram*)
  - ☑ Physics properties are changed via scripts
    - The jump of the player is affected by momentum, The initial speed before the jump affect how far you can go and also affects the level of control the player has.
  - ☑ Mass/Physics is a gameplay mechanic
    - The jump being affected by momentum is used in parkour segments. There are triggers on level 4 and 5 that require an appropriate amount of mass to be triggered.
  - ☑ Additional forces beyond simple motion-driven accelerations are provided (projectile trajectories, gravity interfering with the velocity).
    - The projectiles fired by the NPCs are force driven and affected by gravity.
  - ☑ AI uses calculations to determine projectile forces.
    - The AI calculates the players last location and adjusts the speed and force applied on the projectile to make sure it hits the player.

- +5%: Basic Collision Volumes
  - ☑ There is at least one collision volume
    - Player has a collision volume.
  - ☑ There is more than one collision volume
    - The NPCs and movable objects also have colliders, so do the pillars, platforms, walls and doors.
  - ☑ The collision volume is appropriate and matches the Game Object’s mesh.
    - Almost all game objects have the collider attached perfectly, NPCs have a capsule collider that covers their body appropriately.

- +5%: Advanced collision volumes.
  - ☑ A single Game Object has multiple colliders.
    - The trigger on level 4 and 5 have multiple colliders.
  - ☑ Colliders are enabled/disabled via scripts.
    - Instatiate hologram when teleport function is active. Door game objects and water are disables on trigger. For collider only, there is a cube behind player spawn on level1 that a collider enabled and disabled every 5 seconds.
  - ☑ Colliders can change their position programmatically.
    - Player collider moves with player, similar with NPCs. If this is for collider only, the black cube on level 1 has a collider moving without the gameObject moving itself.
  - ☑ Trigger volumes are used as part of player mechanics.
    - The player has to pickup the teleport ability and final key. The trigger for water drain and door open work on pass but also have a trigger collider. Trigger collider also used to change the player movement when in water or in the sand.

- +5%: Appropriate collision response and feedback.
  - ☑ Rigidbody responds to collisions realistically
    - Rigidbody cannot phase through objects and can stand on all platforms. The cubes on level 4 and 5 move around when pushed by the player.

  - ☑ An object makes use of OnCollisionEnter / Exit
    - The projectile gameObjects are destroyed on collision with anything. This also reduces the load minorly or else it could infinitely spawn projectiles that miss the player.

☑ Collision Layers are used to separate out collision types

- Cube object cannot fall off platform due to collider set to the cube layer, Also passes through the trigger collider due to layer settings.

☑ Physics materials are used

- Water physics and sand physics are seen in level 4 and level 5. These had to be coded manually as the player movement was made manually and would overwrite the physics material.

+5%: Advanced collision response and feedback

☑ Multiple physics materials appearing in the game.

- Physics material for water and sand on level 4 and 5.

☑ Physics materials are changed at run-time via script

- Phsyics material for sand is default at start but enabled when the teleport ability is picked up changing to sand physics material.

☑ Trigger volumes are used to trigger gameplay events

- Triggers use a is trigger collider to activate events like drain water and open door.

+20%: **Graphics**

- +10%: Appropriate Use of Graphical Elements
  - ☑ Multiple textures appearing in game
    - Water, sane, hologram, player, walls, floor, pillars etc all have different textures
  - ☑ Appropriate use of lighting
    - There is lightsournce around the map making sure all parts are visible. Intensity and direction are affected by day and night cycle.
  - ☑ GameObjects moving & rotating via script
    - Npc has movement and rotation
  - ☑ A navigable camera moving in 3D
    - Camera can be controlled by mouse and is attached to the camera.

- +10%: Advanced Graphics
  - ☑ Environment appears to extend infinitely
    - At the end of the game, where the player escapes the dungeon the environment appears to extend infinitely.
  - ☑ A body of realistic looking water
    - Body of realistic water on level 4 and end of the game.
  - ☑ Scripted lighting/effects (e.g. weather, day/night cycle)
    - Day and night cycle that lasts appx 5 mins(changeable). Direction, color and intensity of light changes
  - ☑ Change object appearance via script
    - Textures change for the floor on level 5 when the ability is picked up. Material for NPC also changes when takedown in available.
  - ☑ Geometry that changes over time (e.g. plants growing)
    - Ice cube on level 4 melts over time and becomes smaller. Can be pushed around and go under water as well. Melts over 5 mins (changeable)

+10%: Pathfinding

☑ NavMeshAgents are used

- All NPCs have NavMesh agent added onto them.

☑ NavMeshObstacles are used

- Pillars have a NavMeshObstacle to prevent NPCs from walking through or into them.

☑ Custom pathfinding code, or external library with some modifications.

- The NPC chases the player and finds the fastest path to the players last known location.

☑ AI makes decisions based on pathfinding

- NPC follows shortest path to the Player. When lost the NPC randomly travels to the last known location and does a search.

+30%: Artificial Intelligence

- ## +10%: State Machines
  - ☑ Usage of simple state-machines
    - The NPC has 3 states, Chasing Patrolling and Searching.

  - ☑ Usage of boolean or state-driven state machines
    - Chasing is activated when the player is detected. Player detected is true puts the NPC in the chase state

  - ☑ Usage of object-encapsulation for modelling states
    - The states are in their own individual classes and scripts.

  - □ Usage of hierarchical state machines or usage of external tools for generating state machines.
    - Triend implementing it but could not due to time constraints, State-smith was not a option as it kept getting blocked due to virus detection.

  - ☑ State machines are triggered by external events or timeouts.
    - Chase state is triggered on enemy detection. Searching state is limited to 5s(changeable)

  - ☑ Usage of probabilistic/stochastic state transitions.
    - While in the pareolling state the NPC has a 30% chance to switch to Searching.

- ## +10%: Advanced AI
  - □ Usage of Planning techniques (Real Planners, GOAPs)
  - □ Usage of Non-Cooperative Game Strategies (Min-Max trees, α-βpruning)
  - □ Usage of basic Reinforcement Learning techniques.

- +10%: Structuring NPCs
    - ☑ NPCs are not orchestrated, and mainly react directly to the player or the environment.
        - NPCs chase the player on detection and move to the last known location when searching.

    - □ NPCs are coordinated as a group or in tandem, to fulfil the same goal or task

    - □ An orchestrator or game manager is used to handle multiple and contrasting behaviours, thus including the generation of random events.

Advanced Features

☑ **[+2%]** Appropriate usage of Prefabs.

  - Walls, pillars, doors etc are all examples of prefabs.

☑ **[+2%]** Levels/Menus are separated into distinct scenes.

  - Menu is in MenuScene and the game is in SampleScene.

☑ **[+3%]** Evidence of code for limiting expensive computations (e.g., *raycasting*).

  - Raycasting used to detect when on ground is limited. NPC detection is only active when player is in a close range so it isnt active at all times and does not need to check if the player is in the detection range.

□ **[+2%]** Flocking techniques.

☑ **[+2%]** Usage of Vector Fields.

  - A vector field affects the fire particles on the door between level 5 and level 6

☑ **[+2%]** Usage of Particle Systems.

  - Dust particle effect visible when the player starts sprinting. Objects like the final Key and TeleportItem also have a glow utilising the particle system for added magical effect

□ **[+2%]** Implementation of custom AI tools (AverageMinMax, *et ceteram*)