





Documentation for Submission 1:

21JumpyStreet

1. Goal and Controls

The game can be started by a left click in the pink start screen and then the goal is to jump forward as far as possible. Currently the only way to lose is by falling into the water. Jump onto logs in the water to cross the rivers. The player can jump in all four directions. Pressing ESC in the start screen exits the application.

 /A	Jump left
 /D	Jump right
 /W/SPACE	Jump forward
 /S	Jump backward
ESC	Pause/Resume Game
C	Debug Mode

DEBUG MODE

K	End Game
F1	Toggle Wireframe
F2	Toggle Culling
N	Toggle Normals
T	Toggle Tex Coords
Hold Left Mousedown	Tilting the camera
Hold Right Mousedown	Moving the camera
Scroll Wheel	Zooming In/Out

2. Compulsory Gameplay

- 2.1. 3D Geometry (6 Points): We started to load an external file, but it has not been completely implemented yet.
- 2.2. Playable (3 Points): Game is playable with controls and player can move around.
- 2.3. Min. 60 FPS and Framerate Independence (3 Points): The game is set to 60 fps and is programmed frame rate independent.
- 2.4. Win/Lose Condition (3 Points): If the player is falling into the water the game is lost.
- 2.5. Intuitive Controls (2 Points): The player can move around in all directions using the arrow keys or using WASD. Jumping forward can also

be done by pressing the SPACE bar. The game can be paused by pressing ESC.

- 2.6. Intuitive Camera (2 Points): The camera is fixed and following the player by default and can be moved freely by activating the debug mode (debug camera). Key: C
- 2.7. Illumination Model (2 Points)
- 2.8. Textures (2 Points)
- 2.9. Moving Objects (2 Points): Logs are moving around in the water and can be jumped on and off.
- 2.10. Documentation (1 Point)
- 2.11. Adjustable Parameters (1 Point): Game window is resizable during runtime.

3. Optional Gameplay

- 3.1. Advanced Gameplay (5 Points): Every move forward the score of the current game is increasing, until you die. The map of the game is generated randomly and therefore different every time.
- 3.2. Collision Detection (Basic Physics) (6 Points): If the player is falling into the water, the player is drowning, and the game is over.
- 3.3. Advanced Physics (4 Points)
- 3.4. Scripting Language Integration (6 Points)
- 3.5. Heads-Up Display (4 Points): The score is shown in the top left, and a hit is shown when the debug mode is activated.
- 3.6. Camera Object Tracking (2 Points): The camera is following the players movement.
- 3.7. Camera Pose Estimation (4 Points)

4. Effects

- 4.1. Lighting
 - 4.1.1. Lightmap using Separate Textures (8 Points): not implemented
 - 4.1.2. Lightmap using In-Game Calculation (12 Points)
 - 4.1.3. Shadow Map with PCF (16 Points)
 - 4.1.4. Omnidirectional Shadow Map (20 Points)
 - 4.1.5. Shadow Volumes (16 Points)
 - 4.1.6. Progressive Lightmap (20 Points)
- 4.2. Advanced Modelling
 - 4.2.1. CPU Particle System (8 Points)

- 4.2.2. GPU Particle System using Transform Feedback (12 Points)
- 4.2.3. GPU Particle System using Compute Shader (12 Points)
- 4.2.4. L-System (12 Points)
- 4.2.5. Blobby Object using Raycasting (12 Points)
- 4.2.6. Blobby Object using Marching Cubes (16 Points)
- 4.2.7. Subdivision Surface (20 Points)
- 4.3. Terrain
 - 4.3.1. Tessellation from Height Map (12 Points)
 - 4.3.2. Voxel Terrain using an Octree (20 Points)
 - 4.3.3. Hierarchical Animation (4 Points)
 - 4.3.4. Vertex Shader Animation (8 Points): the waves of the river are animated in the vertex shader.
 - 4.3.5. GPU Vertex Skinning (20 Points)
- 4.4. Texturing
 - 4.4.1. Procedural Texture (8 Points)
 - 4.4.2. Video Texture (8 Points)
 - 4.4.3. Specular Map (4 Points)
 - 4.4.4. Triplanar Texturing (6 Points)
 - 4.4.5. Environment Map (8 Points)
 - 4.4.6. Dynamic Environment Map (12 Points)
 - 4.4.7. Image Based Lighting (10 Points)
- 4.5. Shading
 - 4.5.1. Simple Normal Mapping (4 Points)
 - 4.5.2. Cel Shading (4 Points)
 - 4.5.3. Style Transfer (8 Points)
 - 4.5.4. Brush strokes (4 Points)
 - 4.5.5. Physically Based Shading (6 Points)
- 4.6. Advanced Data Structures
 - 4.6.1. BSP Tree (16 Points)
 - 4.6.2. kd-Tree (16 Points)
 - 4.6.3. LOD using an octree (12 Points)
- 4.7. Post Processing
 - 4.7.1. Bloom/Glow (8 Points)
 - 4.7.2. Lens Flares (8 Points)
 - 4.7.3. Contours via Backfaces (4 Points)
 - 4.7.4. Contours via Edge Detection (12 Points)
 - 4.7.5. Ambient Occlusion (12 Points)
 - 4.7.6. Motion Blur (16 Points)