

Universidad Nacional Autónoma de México

Facultad de Ingeniería

Computación Grafica Avanzada

## Practica 4. Blend Map.



Alumna: Anahi Hillary Gil González.

Profesor: Reynaldo Martell Ávila.

Edo. De México a 6 de octubre de 2020

## Desarrollo.

1. Utilizando el modelo animados con las dos poses (descanso y corriendo), colocar la animación en la práctica de terreno, también debe agregar los movimientos para mover al personaje (girar y avanzar) **(10 Puntos)**.

Para este ejercicio fue necesario cargar cada animación por separado, así que lo primero fue declarar ambas variables del modelo cuando está descansando y corriendo:

```
86 //----->Cargando ambas animaciones de Pacman
87 Model modelPacManDescanso;
88 Model modelPacManCorriendo;
```

También se declararon dos nuevas matrices para hacer las transformaciones necesarias de ambas animaciones:

```
125 //----->Matrices para Pacman
126 glm::mat4 modelMatrixPacManDescanso = glm::mat4(1.0f);
127 glm::mat4 modelMatrixPacManCorriendo = glm::mat4(1.0f);
```

Se cargan y referencian las variables de los modelos a su correspondiente archivo fbx:

```
294 //----->Cargando modelos de Pacman
295 modelPacManDescanso.loadModel("../models/PacMan/Pac-Man_Descanso.fbx");
296 modelPacManDescanso.setShader(&shaderMullighting);
297 modelPacManCorriendo.loadModel("../models/PacMan/Pac-Man_Corriendo.fbx");
298 modelPacManCorriendo.setShader(&shaderMullighting);
```

Se declaró una nueva variable llamada banderaPacman, la cual será igual a 0 cuando Pacman esté en modo descanso y en 1 cuando Pacman este corriendo.

```
133 int banderaPacman = 0; //----->Para indicar el estado de PacMan
```

Para controlar a Pacman y poder moverlo libremente en el escenario se ocuparon las teclas flechas del teclado:

Tecla	Movimiento
↑	Adelante
↓	Atrás
→	Gira hacia la derecha
←	Gira hacia la izquierda

```

865 //----->Controles del modelo de PacMan
866 if (glfwGetKey(window, GLFW_KEY_LEFT) == GLFW_RELEASE && glfwGetKey(window, GLFW_KEY_RIGHT) == GLFW_RELEASE
867     && glfwGetKey(window, GLFW_KEY_UP) == GLFW_RELEASE && glfwGetKey(window, GLFW_KEY_DOWN) == GLFW_RELEASE)
868     banderaPacman = 0;
869 else if (glfwGetKey(window, GLFW_KEY_LEFT) == GLFW_PRESS) {
870     modelMatrixPacmanCorriendo = glm::rotate(modelMatrixPacmanCorriendo, 0.02f, glm::vec3(0, 1, 0));
871     modelMatrixPacmanDescanso = glm::rotate(modelMatrixPacmanDescanso, 0.02f, glm::vec3(0, 1, 0));
872     banderaPacman = 1;
873 }
874 else if (glfwGetKey(window, GLFW_KEY_RIGHT) == GLFW_PRESS) {
875     modelMatrixPacmanCorriendo = glm::rotate(modelMatrixPacmanCorriendo, -0.02f, glm::vec3(0, 1, 0));
876     modelMatrixPacmanDescanso = glm::rotate(modelMatrixPacmanDescanso, -0.02f, glm::vec3(0, 1, 0));
877     banderaPacman = 1;
878 }
879 else if (glfwGetKey(window, GLFW_KEY_UP) == GLFW_PRESS) {
880     modelMatrixPacmanCorriendo = glm::translate(modelMatrixPacmanCorriendo, glm::vec3(0.0, 0.0, -0.04));
881     modelMatrixPacmanDescanso = glm::translate(modelMatrixPacmanDescanso, glm::vec3(0.0, 0.0, -0.04));
882     banderaPacman = 1;
883 }
884 else if (glfwGetKey(window, GLFW_KEY_DOWN) == GLFW_PRESS) {
885     modelMatrixPacmanCorriendo = glm::translate(modelMatrixPacmanCorriendo, glm::vec3(0.0, 0.0, 0.04));
886     modelMatrixPacmanDescanso = glm::translate(modelMatrixPacmanDescanso, glm::vec3(0.0, 0.0, 0.04));
887     banderaPacman = 1;
888 }

```

Como se muestra en la imagen, mientras ninguna de las teclas antes mencionadas se presione banderaPacman será igual a 0, cuando alguna de ellas se presione banderaPacman será igual a 1.

Para poder hacer el cambio entre ambas animaciones se tomó en cuenta banderaPacman durante la renderización de los modelos de Pacman descansando y corriendo:

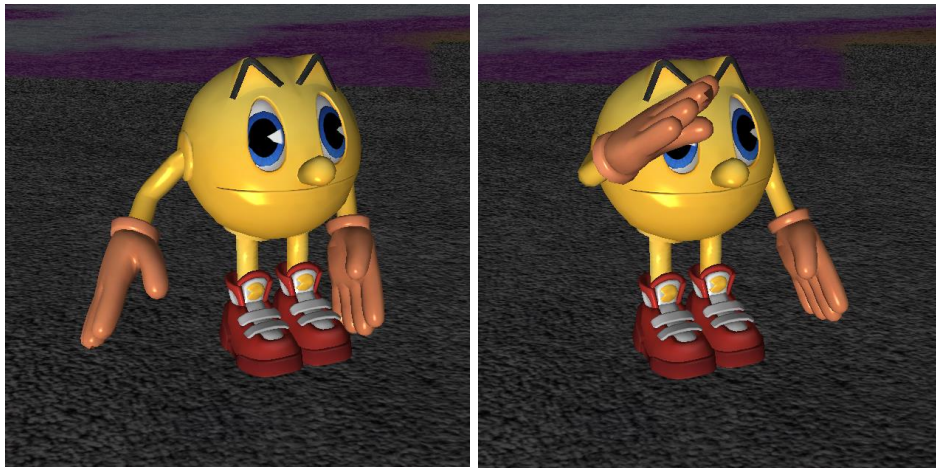
```

1130 //----->renderizando de Pacman según el estado de banderaPacman
1131
1132 if (banderaPacman == 1) {
1133     modelMatrixPacmanCorriendo[3][1] = terrain.getHeightTerrain(modelMatrixPacmanCorriendo[3][0], modelMatrixPacmanCorriendo[3][2]);
1134     glm::vec3 aux = glm::vec3(terrain.getNormalTerrain(modelMatrixPacmanCorriendo[3][0], modelMatrixPacmanCorriendo[3][2]));
1135     modelMatrixPacmanCorriendo[1][0] = aux[0];
1136     modelMatrixPacmanCorriendo[1][1] = aux[1];
1137     modelMatrixPacmanCorriendo[1][2] = aux[2];
1138     glm::mat4 modelMatrixPacmanCorriendoBody = glm::mat4(modelMatrixPacmanCorriendo);
1139     modelMatrixPacmanCorriendoBody = glm::scale(modelMatrixPacmanCorriendoBody, glm::vec3(0.005, 0.005, 0.005));
1140     modelPacmanCorriendo.render(modelMatrixPacmanCorriendoBody);
1141 }
1142 else {
1143     modelMatrixPacmanDescanso[3][1] = terrain.getHeightTerrain(modelMatrixPacmanDescanso[3][0], modelMatrixPacmanDescanso[3][2]);
1144     glm::vec3 aux = glm::vec3(terrain.getNormalTerrain(modelMatrixPacmanDescanso[3][0], modelMatrixPacmanDescanso[3][2]));
1145     modelMatrixPacmanDescanso[1][0] = aux[0];
1146     modelMatrixPacmanDescanso[1][1] = aux[1];
1147     modelMatrixPacmanDescanso[1][2] = aux[2];
1148     glm::mat4 modelMatrixPacmanDescansoBody = glm::mat4(modelMatrixPacmanDescanso);
1149     modelMatrixPacmanDescansoBody = glm::scale(modelMatrixPacmanDescansoBody, glm::vec3(0.005, 0.005, 0.005));
1150     modelPacmanDescanso.render(modelMatrixPacmanDescansoBody);
1151 }

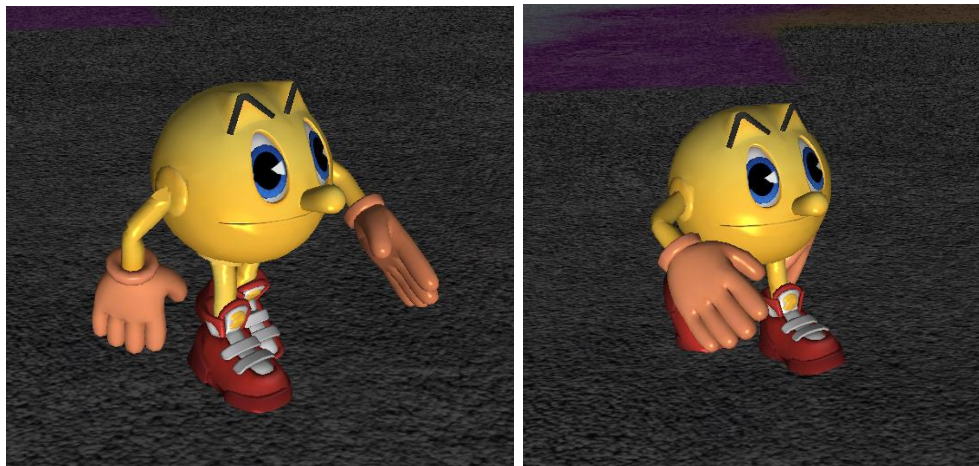
```

El resultado final es el siguiente:

Pacman en modo descanso.



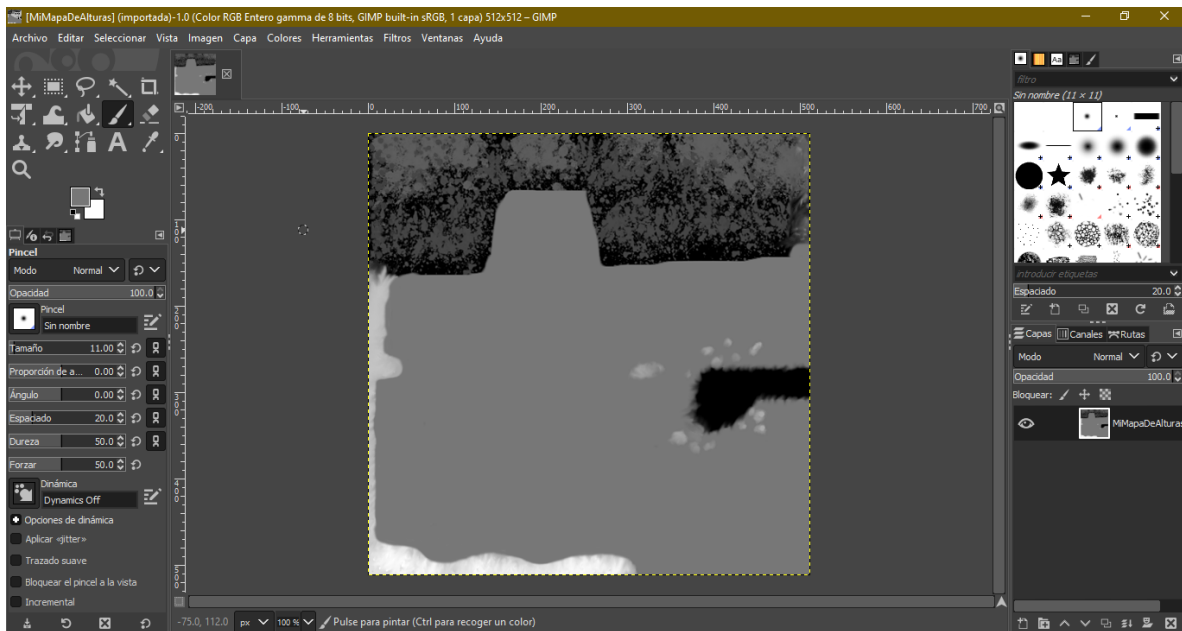
Pacman Corriendo.



2. Realizar con gimp de un mapa de alturas diferente al que se realizó y el cual servirá como base para las siguientes prácticas **(10 Puntos)**.

Se creo el mapa de altura de 512x512px y se guardó en la carpeta de texturas del proyecto.

Para este ejercicio se intento representar el mapa Polus del juego AmongUs.



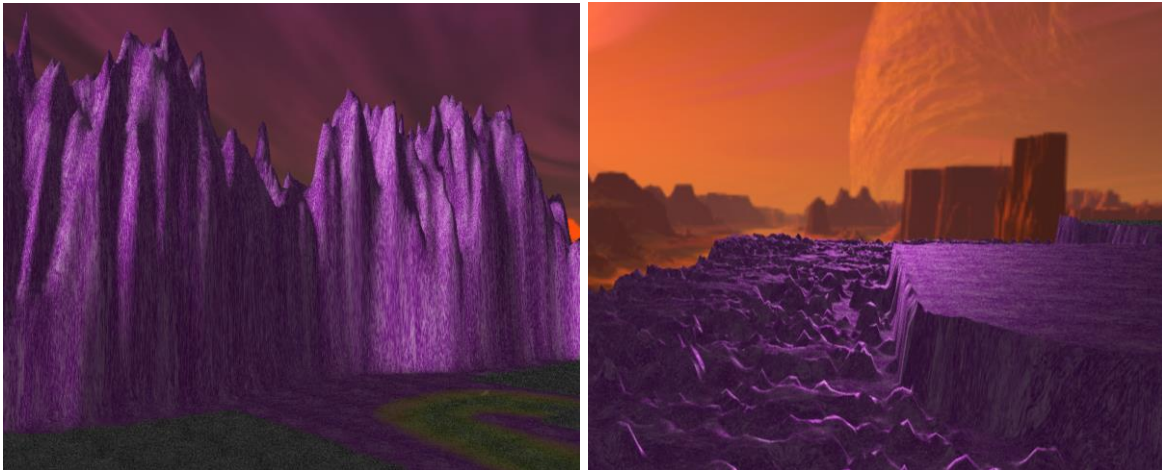
3. Cargar el mapa de alturas personalizado en OpenGL y ajustarlo (Tamaño, y altura máxima) acorde a su diseño **(10 Puntos)**.

Se cargo el mapa de alturas definiendo una rejilla de 200 y una altura máxima de 40. Tambien se situó en la parte central del escenario:

```
91 // Terrain model instance
92 Terrain terrain(-1, -1, 200, 40, "../Textures/Texturas de Terreno/MiMapaDeAlturas.png");
```

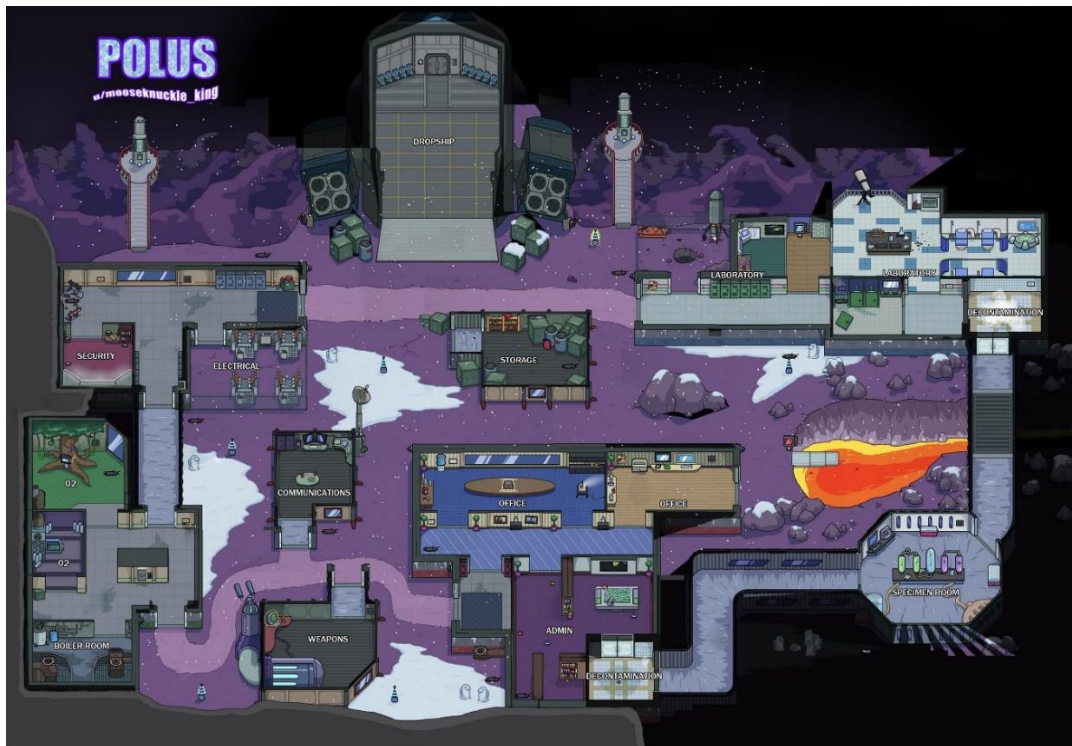


El resultado es el siguiente:

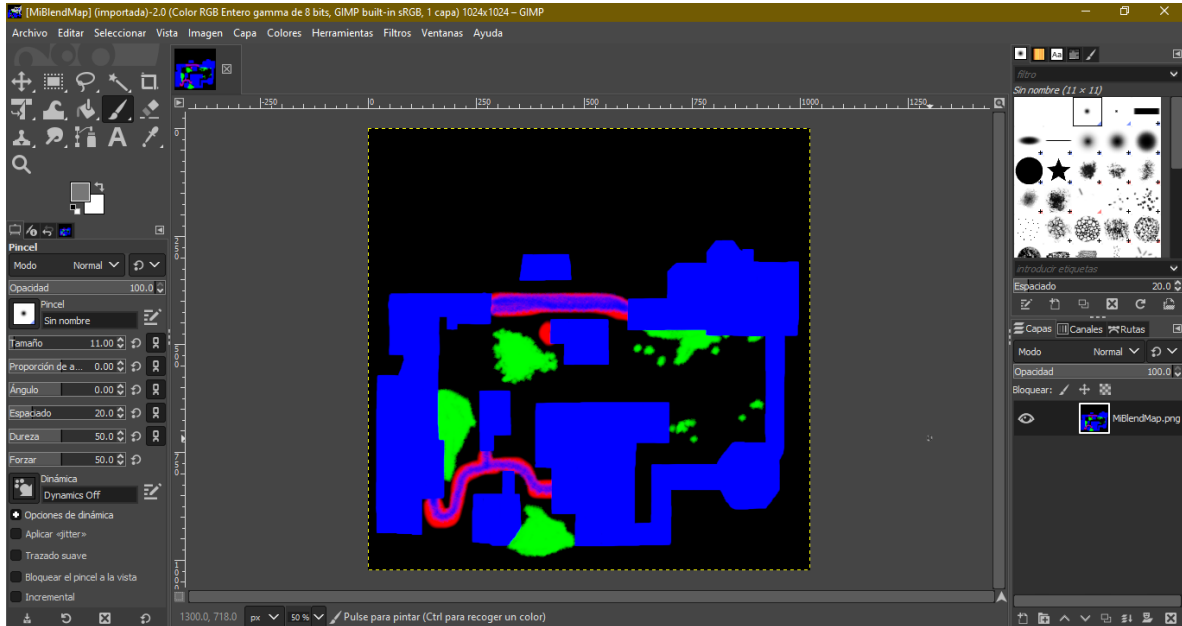


4. Con gimp como se explicó en la práctica crear un mapa de mezcla personalizado tomando en cuenta el terreno de los puntos anteriores **(20 Puntos)**.

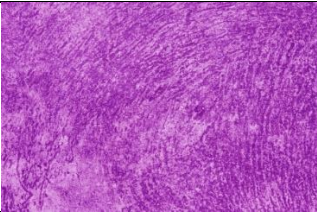


En este ejercicio se elaboro el mapa de mezcla para representar el terreno del planeta Polus de AmongUs.




Los caminos y las partes construidas se pintarán de azul, la nieve se pinto de color verde y la parte erosionada alrededor de los caminos se le asigno el color rojo. Todo lo que esta en negro será la textura general del fondo.



5. Descargar 4 texturas, terreno, camino, tierra y flores, colocarlo sustituyendo estas texturas a las que se tienen por defecto (también se puede representar otro tipo de terreno por ejemplo en nieve) **(20 Puntos)**.

Textura	Lo que representa
	Terreno general (color negro)
	Nieve (color verde)
	Camino/Parte construida (Azul)

		<p>Erosión del suelo alrededor del camino (Rojo)</p>
--	---	--

Cambiando las texturas por defecto:

```

494 // Definiendo la textura a utilizar
495 Texture textureTerrainBackground("../Textures/Texturas de Terreno/terreno.jpg");

526 //-----> Definiendo la textura a utilizar para R
527 Texture textureTerrainR("../Textures/Texturas de Terreno/tierra.jpg");

558 //-----> Definiendo la textura a utilizar para G
559 Texture textureTerrainG("../Textures/Texturas de Terreno/nieve.jpg");

590 //----->Definiendo la textura a utilizar para B
591 Texture textureTerrainB("../Textures/Texturas de Terreno/suelo.jpg");

622 //-----> Definiendo la textura a utilizar para mapa de mezcla
623 Texture textureTerrainBlendMap("../Textures/Texturas de Terreno/MiBlendMap.png");
624 // Carga el mapa de bits (FIBITMAP es el tipo de dato de la libreria)
625 bitmap = textureTerrainBlendMap.loadImage(true); //-----> se agrego el true para bliquear la textura

```

Activando las texturas correspondientes:

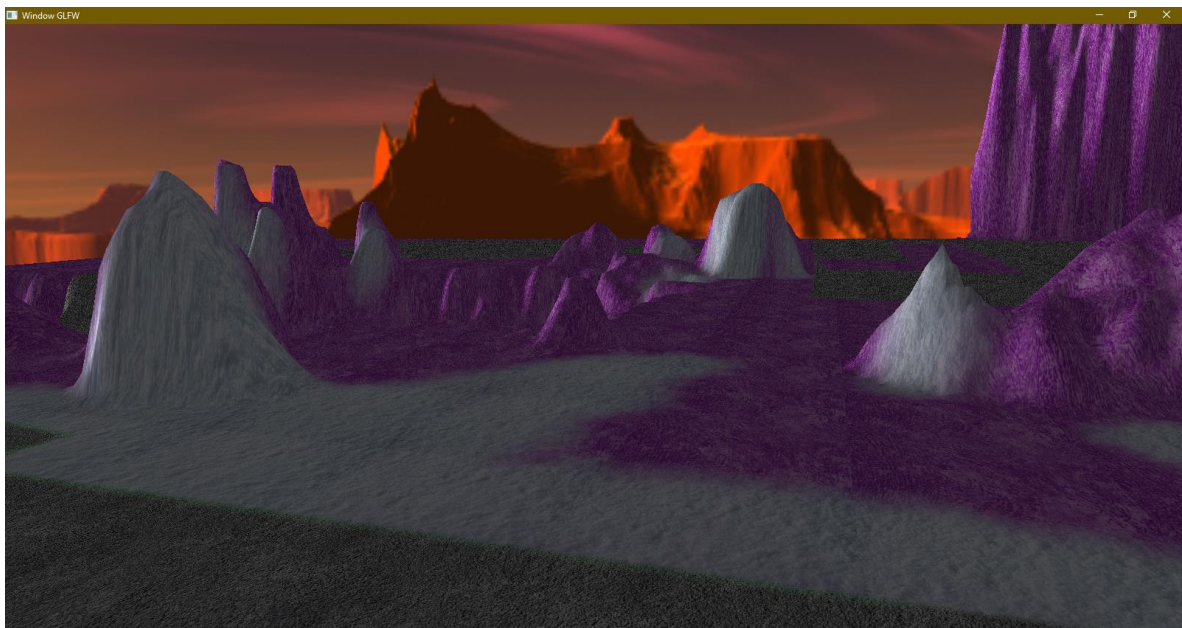
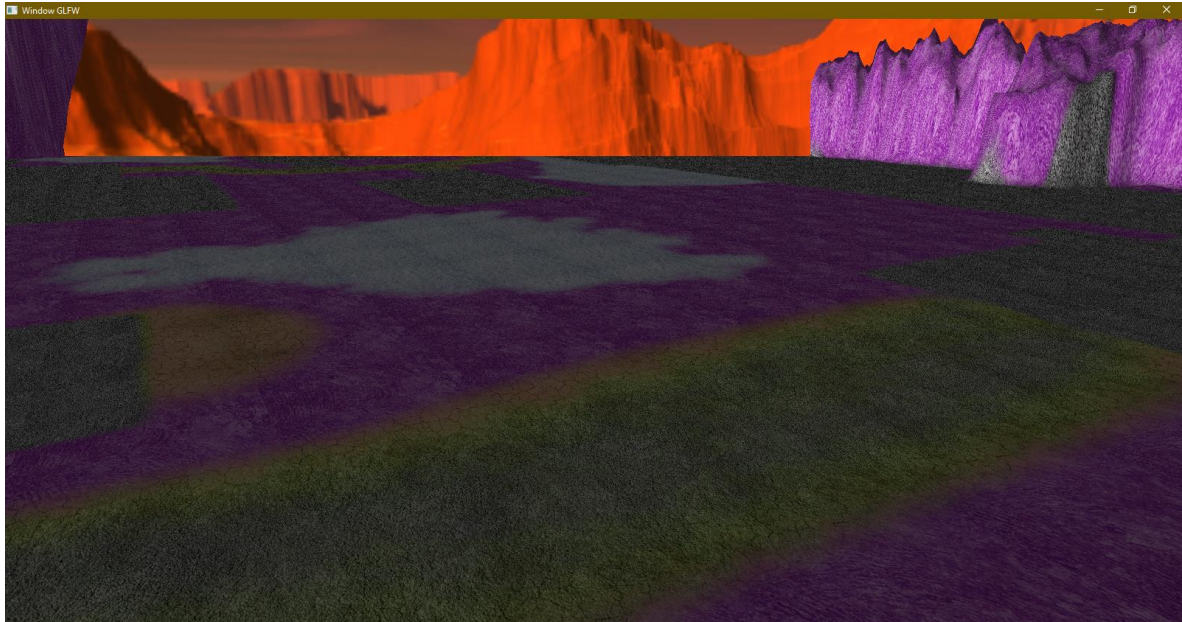
```

994 // Se activa la textura del background
995 glActiveTexture(GL_TEXTURE0);
996 glBindTexture(GL_TEXTURE_2D, textureTerrainBackgroundID);
997 shaderTerrain.setInt("backgroundTexture", 0);
998 //----->se activa la textura de la tierra
999 glActiveTexture(GL_TEXTURE1);
1000 glBindTexture(GL_TEXTURE_2D, textureTerrainRID);
1001 shaderTerrain.setInt("rTexture", 1);
1002 //----->se activa la textura de flor
1003 glActiveTexture(GL_TEXTURE2);
1004 glBindTexture(GL_TEXTURE_2D, textureTerrainGID);
1005 shaderTerrain.setInt("gTexture", 2);
1006 //----->se activa la textura de la comino
1007 glActiveTexture(GL_TEXTURE4);
1008 glBindTexture(GL_TEXTURE_2D, textureTerrainBID);
1009 shaderTerrain.setInt("bTexture", 4);
1010 //----->se activa la textura de mapa de mezcla
1011 glActiveTexture(GL_TEXTURE5);
1012 glBindTexture(GL_TEXTURE_2D, textureTerrainBlendMapID);
1013 shaderTerrain.setInt("blendMapTexture", 5);
1014
1015 shaderTerrain.setVectorFloat2("scaleUV", glm::value_ptr(glm::vec2(40, 40))); // se escalan las coordenadas de texturas n-veces
1016 terrain.render();
1017 shaderTerrain.setVectorFloat2("scaleUV", glm::value_ptr(glm::vec2(0, 0)));
1018 glBindTexture(GL_TEXTURE_2D, 0);

```

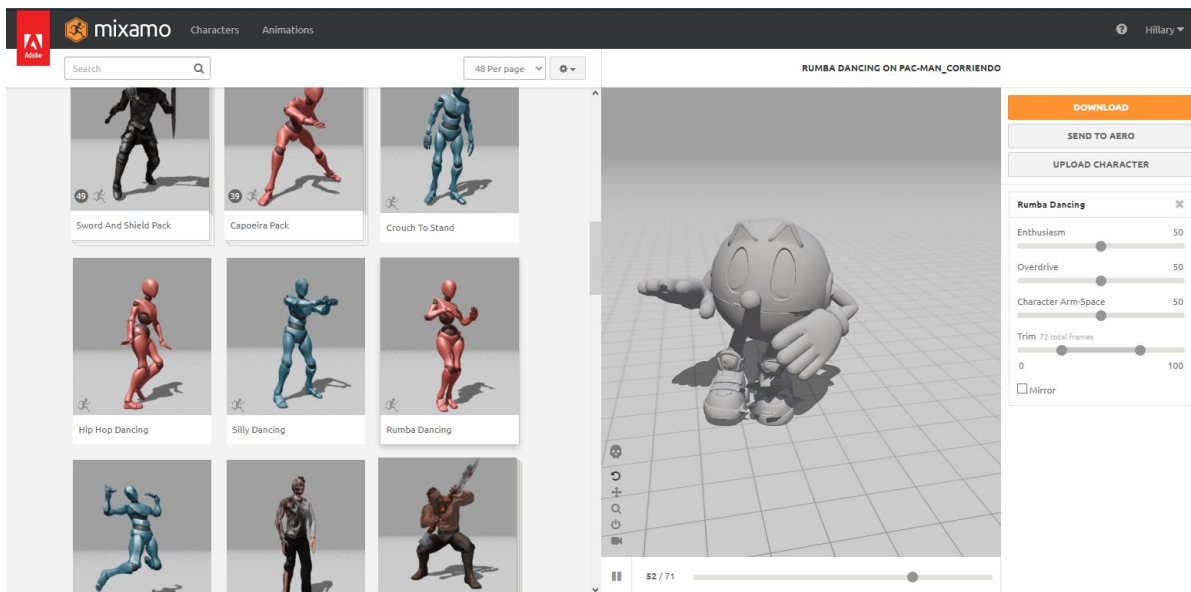


Este es el resultado:

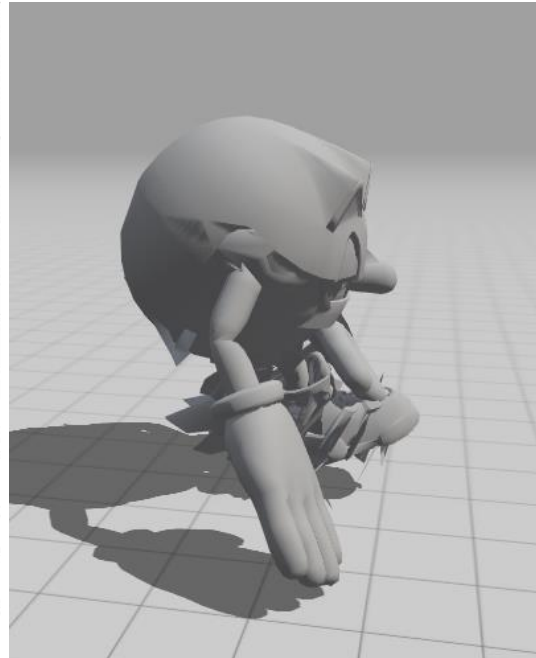
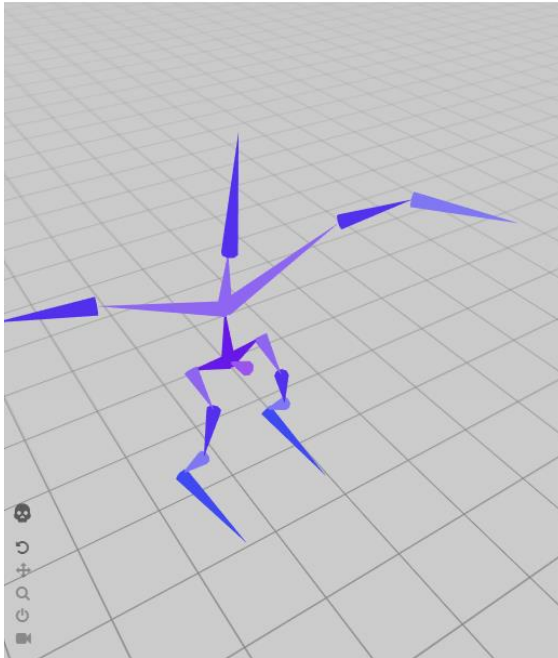


6. Investigar y crear una animación con la herramienta Mixamo (<https://www.mixamo.com/>) (Hint. se puede exportar a blender y visualizar y ajustar las animaciones) (30 Puntos)

Para poder utilizar la herramienta se tiene que crear una cuenta de acceso gratuito, luego de esto la herramienta te brinda varios esqueletos con animaciones de todo tipo. Se puede cargar un modelo que tenga un esqueleto para poder pasarle la animación que escojas:



El esqueleto del modelo se ajusta de forma automática por lo que pueden existir problemas referentes a los pesos y deformación de la malla como se muestra en las siguientes imágenes:



Para este ejercicio se escogio una animación de baile:



Para cargarlo al proyecto se añadieron las siguientes líneas.

Primero se declaró el modelo nuevo:

```
89      Model modelBaile;
```

Se declaró una nueva matriz para las transformaciones del modelo:

```
128      glm::mat4 modelMatrixBaile = glm::mat4(1.0f);
```

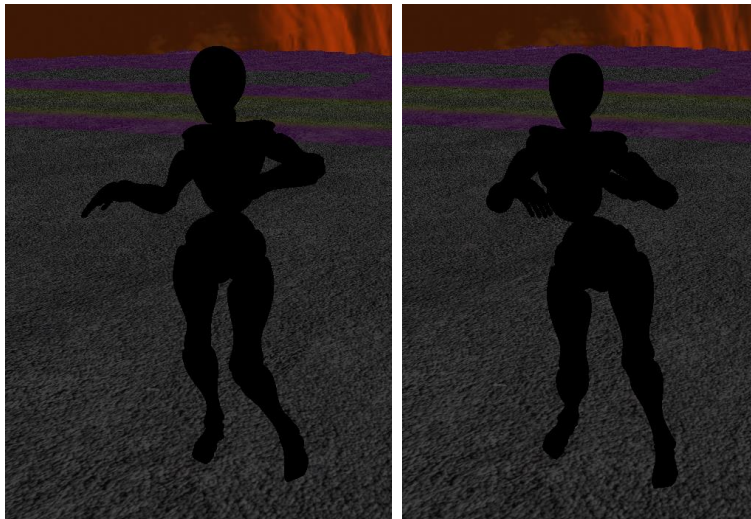
Se referencio y cargo el modelo de baile hacia su correspondiente archivo fbx:

```
300      modelBaile.loadModel("../models/Mis modelos/mixamo/baile.fbx");  
301      modelBaile.setShader(&shaderMulLighting);
```

Para finalizar se escaló, traslado, se le hicieron los ajustes necesarios para que tome en cuenta la altura del terreno en la que se encuentra y se renderizo:

```
1124      modelMatrixBaile[3][1] = terrain.getHeightTerrain(modelMatrixBaile[3][0], modelMatrixBaile[3][2]); //-----  
1125      glm::mat4 modelMatrixBaileBody = glm::mat4(modelMatrixBaile);  
1126      modelMatrixBaileBody = glm::scale(modelMatrixBaileBody, glm::vec3(0.015, 0.015, 0.015));  
1127      modelMatrixBaileBody = glm::translate(modelMatrixBaileBody, glm::vec3(0, 0, 1));  
1128      modelBaile.render(modelMatrixBaileBody);
```

El resultado es el siguiente:



## Conclusión.

Mixamo es una herramienta muy útil para poder descargar animaciones ya diseñadas que demuestran un mejor flujo en el movimiento, además te ayuda aplicando de forma automática un modelo con esqueleto que ya tengas hecho previamente. En esta practica pude hacer la descarga de la animación de Mixamo y tambien la pude cargar al proyecto, sin embargo, no logre hacer que un modelo previo guarde la animación que escogí.

Enlace de demostración en video.

<https://youtu.be/g3Sya1NvRgQ>

Enlace a repositorio

<https://github.com/HillaryGil97/ComputacionGraficaAvanzada>