

Universidad Nacional Autónoma de México

Facultad de Ingeniería

Computación Grafica Avanzada

Practica 1. Animación por keyframes.



Alumna: Anahi Hillary Gil González.

Profesor: Reynaldo Martell Ávila.

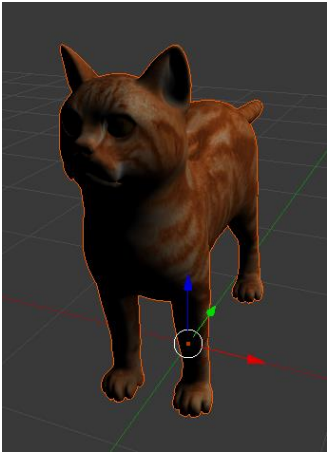

Edo. De México a 16 de octubre de 2020

Desarrollo.

1. Colocar dos modelos obj con el cargador de modelos de la clase, los modelos deben buscarlos en internet o que tengan de su curso previo, aquí les dejo las rutas donde pueden encontrar modelos. Los modelos deben tener texturas y normales, ajusten las texturas para que puedan cargarse usando el cargador de la clase.

- <https://www.turbosquid.com/es/Search/3D-Models/free/obj>
- <https://free3d.com/es/modelos-3d/obj>
- <https://open3dmodel.com/es/3d-models/obj>

Para este ejercicio se busco en la página turboSquid se encontraron dos modelos gratuitos los cuales son los siguientes:

Modelo de Gato	Modelo de una Mole de piedra
	

Con cada uno se hizo un trabajo de escalamiento y rotación en blender para que luciera mejor en el escenario. Ambos se encuentran en la ruta ..\ComputacionGraficaAvanzada\models\Mis modelos

Para cargarlos en OpenGL se utilizaron los siguientes fragmentos de código:

```
108 //----->Nuevos modelos
109 Model modelGato;
110 Model modelMole;

143 glm::mat4 modelMatrixGato = glm::mat4(1.0f);//----->Para los
144 glm::mat4 modelMatrixMole = glm::mat4(1.0f);//----->Modelos nuevos
```

```

369 //----->Cargando los dos modelos nuevos
370 modelGato.loadModel("../models/Mis modelos/Gato/Gato.obj");
371 modelGato.setShader(&shaderMultilighting);
372 modelMole.loadModel("../models/Mis modelos/Mole/Mole.obj");
373 modelMole.setShader(&shaderMultilighting);

1166 //-----> AGREGANDO EL MODELO DEL GATO
1167 glm::mat4 modelMatrixGatoBody = glm::mat4(modelMatrixGato);
1168 modelMatrixGatoBody = glm::translate(modelMatrixGatoBody, glm::vec3(0.0, 0.0, -15.0));
1169 modelGato.render(modelMatrixGatoBody);
1170 //-----> AGREGANDO EL MODELO DE LA MOLE DE PIEDRA
1171 glm::mat4 modelMatrixMoleBody = glm::mat4(modelMatrixMole);
1172 modelMatrixMoleBody = glm::translate(modelMatrixMoleBody, glm::vec3(5.0, 0.0, -15.0));
1173 modelMole.render(modelMatrixMoleBody);

```

Al compilar y correr los resultados fueron los siguientes:



2. Realizar la animación del lamborghini por máquina de estados que haga un recorrido por el circuito que ya se tiene, animar las llantas por separado como se hace con el eclipse, al completar el circuito debe terminar la animación y abrir una de sus puertas.

Para este ejercicio lo primero fue encontrar los pivotes con ayuda de Blender de las 4 llantas del Lamborghini, estos pivotes se agregaron en las siguientes lineas:

```
1035 glm::mat4 modelMatrixLamboLeftDor = glm::mat4(modelMatrixLamboChasis);
1036 modelMatrixLamboLeftDor = glm::translate(modelMatrixLamboLeftDor, glm::vec3(1.08676, 0.707316, 0.982601));
1037 modelMatrixLamboLeftDor = glm::rotate(modelMatrixLamboLeftDor, glm::radians(dorRotCount), glm::vec3(1.0, 0, 0));
1038 modelMatrixLamboLeftDor = glm::translate(modelMatrixLamboLeftDor, glm::vec3(-1.08676, -0.707316, -0.982601));
1039 modelLamboLeftDor.render(modelMatrixLamboChasis);
1040 modelLamboRightDor.render(modelMatrixLamboChasis);
1041
1042 glm::mat4 modelMatrixLamboFrontLeftWheel = glm::mat4(modelMatrixLamboChasis);
1043 modelMatrixLamboFrontLeftWheel = glm::translate(modelMatrixLamboFrontLeftWheel, glm::vec3(0.948337, 0.3771, 1.39999));
1044 modelMatrixLamboFrontLeftWheel = glm::rotate(modelMatrixLamboFrontLeftWheel, rotWheelsX, glm::vec3(1, 0, 0));
1045 modelMatrixLamboFrontLeftWheel = glm::translate(modelMatrixLamboFrontLeftWheel, glm::vec3(-0.948337, -0.3771, -1.39999));
1046 modelLamboFrontLeftWheel.render(modelMatrixLamboChasis);
1047
1048 glm::mat4 modelMatrixLamboFrontRightWheel = glm::mat4(modelMatrixLamboChasis);
1049 modelMatrixLamboFrontRightWheel = glm::translate(modelMatrixLamboFrontRightWheel, glm::vec3(-0.948337, 0.3771, 1.39999));
1050 modelMatrixLamboFrontRightWheel = glm::rotate(modelMatrixLamboFrontRightWheel, rotWheelsX, glm::vec3(1, 0, 0));
1051 modelMatrixLamboFrontRightWheel = glm::translate(modelMatrixLamboFrontRightWheel, glm::vec3(0.947495, -0.378069, -1.40386));
1052 modelLamboFrontRightWheel.render(modelMatrixLamboChasis);
1053
1054 glm::mat4 modelMatrixLamboRearLeftWheel = glm::mat4(modelMatrixLamboChasis);
1055 modelMatrixLamboRearLeftWheel = glm::translate(modelMatrixLamboRearLeftWheel, glm::vec3(0.948827, 0.398055, -1.60098));
1056 modelMatrixLamboRearLeftWheel = glm::rotate(modelMatrixLamboRearLeftWheel, rotWheelsX, glm::vec3(1, 0, 0));
1057 modelMatrixLamboRearLeftWheel = glm::translate(modelMatrixLamboRearLeftWheel, glm::vec3(-0.948827, -0.398055, 1.60098));
1058 modelLamboRearLeftWheel.render(modelMatrixLamboChasis);
1059
1060 glm::mat4 modelMatrixLamboRearRightWheel = glm::mat4(modelMatrixLamboChasis);
1061 modelMatrixLamboRearRightWheel = glm::translate(modelMatrixLamboRearRightWheel, glm::vec3(-0.948827, 0.398055, -1.60098));
1062 modelMatrixLamboRearRightWheel = glm::rotate(modelMatrixLamboRearRightWheel, rotWheelsX, glm::vec3(1, 0, 0));
1063 modelMatrixLamboRearRightWheel = glm::translate(modelMatrixLamboRearRightWheel, glm::vec3(0.948827, -0.398055, 1.60098));
1064 modelLamboRearRightWheel.render(modelMatrixLamboChasis);
```

También a cada llanta se le agregó la variable `rotWheelsX` para girar cada llanta a medida que avanza el automóvil.

La máquina de estados de este ejercicio es la siguiente:

```
1235 switch (state) {
1236     case 0:
1237         if (numberAdvance == 0)
1238             maxAdvance = 5.5; //primer avance
1239         else if (numberAdvance == 1)
1240             maxAdvance = 40.5; //primer vuelta
1241         else if (numberAdvance == 2)
1242             maxAdvance = 35.5; //segundo avance
1243         else if (numberAdvance == 3)
1244             maxAdvance = 40.5; //segunda vuelta
1245         else if (numberAdvance == 4)
1246             maxAdvance = 30.0; //tercer avance
1247         state = 1;
1248         break;
1249     case 1:
1250         modelMatrixLambo = glm::translate(modelMatrixLambo, glm::vec3(0.0, 0.0, 0.1)); //se traslada sobre el eje Z
1251         advanceCount += 0.1;
1252         rotWheelsX += 0.05;
1253         rotWheelsY -= 0.02;
1254         if (rotWheelsY < 0)
1255             rotWheelsY = 0;
1256         if (advanceCount >= maxAdvance) {
1257             advanceCount = 0;
1258             numberAdvance++;
1259             state = 2;
1260         }
1261         break;
```

```

1262     case 2:
1263         if (numberAdvance == 5)
1264             state = 3;
1265         else {
1266             modelMatrixLambo = glm::translate(modelMatrixLambo, glm::vec3(0.0, 0.0, 0.025));
1267             modelMatrixLambo = glm::rotate(modelMatrixLambo, glm::radians(-0.5f), glm::vec3(0, 1, 0));
1268             rotCount += 0.5;
1269             rotWheelsX += 0.05;
1270             rotWheelsY += 0.02;
1271             if (rotWheelsY > 0.25)
1272                 rotWheelsY = 0.25;
1273             if (rotCount >= 90.0) {
1274                 rotCount = 0;
1275                 state = 0;
1276             }
1277         }
1278         break;
1279     case 3:
1280         dorRotCount += 0.5;
1281         if (dorRotCount > 75)//esta en grados
1282             state = 4;
1283         break;

```

Al compilar y ejecutar este código se tiene el siguiente comportamiento:





3. Guardar los keyframes del dart vader entrando al lamborghini cuando haya completado el recorrido y cerrar la puerta del carro.

Lo primero fue guardar los keyframes de Dart Vader mientras camina y se desplaza en el escenario, estos keyframes se grabaron con ayuda del programa que se habia hecho durante clase y se guardaron en los archivos correspondientes:

Para el desplazamiento por el escenario.

```
animation_dart: Bloc de notas
Archivo Edición Formato Ver Ayuda
0.999806,0,0.0199989,0,0,1,0,0,-0.0199989,0,0.999806,0,3,0,20,1|
-0.265966,0,0.963992,0,0,1,0,0,-0.963992,0,-0.265966,0,3,0,20,1|
-0.265966,0,0.963992,0,0,1,0,0,-0.963992,0,-0.265966,0,5.63833,0,10.4374,1|
-0.63593,0,0.77176,0,0,1,0,0,-0.77176,0,-0.63593,0,11.0564,0,3.86197,1|
-0.994857,0,0.101419,0,0,1,0,0,-0.101419,0,-0.994857,0,19.0641,0,5.14991,1|
-0.961511,0,-0.27483,0,0,1,0,0,-0.27483,0,-0.961511,0,24.7747,0,4.56774,1|
-0.416155,0,0.909315,0,0,1,0,0,-0.909315,0,-0.416155,0,24.7747,0,4.56774,1|
0.999222,0,-0.0399895,0,0,1,0,0,0.0399895,0,0.999222,0,25.6595,0,0.40054,1|
0.999222,0,-0.0399895,0,0,1,0,0,0.0399895,0,0.999222,0,23.1614,0,0.500512,1|
-0.108989,0,-0.994067,0,0,1,0,0,0.994067,0,-0.108989,0,23.1614,0,0.500512,1|
```

Para la animación del caminado.

```
animation_dart_joints: Bloc de notas
Archivo Edición Formato Ver Ayuda
0|-1.52|0.32|0.54|-0.02|-1.06|0.66|
0|0.42|0.32|-1.32|-0.02|0.68|-0.46|
```

Para reproducir los keyframes se hizo una continuación en la máquina de estado del ejercicio anterior (el desplazamiento del lamborghini), el código es el siguiente:

```

1284 case 4:
1285     if (keyFramesDartJoints.size() > 0) {
1286         //para reproducir los frames
1287         interpolationDartJoints = numPasosDartJoints / (float)maxNumPasosDartJoints;
1288         numPasosDartJoints++;
1289         if (interpolationDartJoints > 1.0) {
1290             numPasosDartJoints = 0;
1291             interpolationDartJoints = 0.0;
1292             indexFrameDartJoints = indexFrameDartJointsNext;
1293             indexFrameDartJointsNext++;
1294         }
1295         if (indexFrameDartJointsNext > keyFramesDartJoints.size() - 1)
1296             indexFrameDartJointsNext = 0;
1297         rotDartHead = interpolate(keyFramesDartJoints, indexFrameDartJoints, indexFrameDartJointsNext, 0, interpolationDartJoints);
1298         rotDartLeftArm = interpolate(keyFramesDartJoints, indexFrameDartJoints, indexFrameDartJointsNext, 1, interpolationDartJoints);
1299         rotDartLeftHand = interpolate(keyFramesDartJoints, indexFrameDartJoints, indexFrameDartJointsNext, 2, interpolationDartJoints);
1300         rotDartRightArm = interpolate(keyFramesDartJoints, indexFrameDartJoints, indexFrameDartJointsNext, 3, interpolationDartJoints);
1301         rotDartRightHand = interpolate(keyFramesDartJoints, indexFrameDartJoints, indexFrameDartJointsNext, 4, interpolationDartJoints);
1302         rotDartLeftLeg = interpolate(keyFramesDartJoints, indexFrameDartJoints, indexFrameDartJointsNext, 5, interpolationDartJoints);
1303         rotDartLeftLeg = interpolate(keyFramesDartJoints, indexFrameDartJoints, indexFrameDartJointsNext, 6, interpolationDartJoints);
1304     }
1305     if (keyFramesDart.size() > 0) {
1306         //para reproducir el frame de todo el modelo
1307         interpolationDart = numPasosDart / (float)maxNumPasosDart;
1308         numPasosDart++;
1309         if (interpolationDart > 1.0) {
1310             numPasosDart = 0;
1311             interpolationDart = 0.0;
1312             indexFrameDart = indexFrameDartNext;
1313             indexFrameDartNext++;

```

```

1314     }
1315     if (indexFrameDartNext > keyFramesDart.size() - 1) {
1316         indexFrameDartNext = 0;
1317         state = 5;
1318     }
1319     modelMatrixDart = interpolate(keyFramesDart, indexFrameDart, indexFrameDartNext, 0, interpolationDart);
1320 }
1321 break;
1322 case 5:
1323     dorRotCount -= 0.5;
1324     if (dorRotCount < 0) {
1325         dorRotCount = 0.0;
1326         state = 6;
1327     }
1328     break;
1329 default:
1330     break;
1331 }

```

Al compilar y ejecutar este código se tiene el siguiente resultado:







4. Haga una máquina de estados para realizar la animación del helicóptero que se acerque a la zona de aterrizaje y aterrice, cuando finalice el aterrizaje detener poco a poco las hélices.

En este ejercicio fue necesario agregar dos nuevas variables, una para hacer que el helicóptero descendiera y otra para ir frenando de a poco las helices:

```
804 float desHeli = 10;  
805 float freno = 1;
```

También se necesitó añadir la variable de descenso en la traslación del modelo del helicóptero:

```
1014 modelMatrixHeliChasis = glm::translate(modelMatrixHeliChasis, glm::vec3(0.0, desHeli, 0.0));
```

Teniendo todo lo anterior ya solo se codificó la máquina de estado para el helicóptero:

```
1214 //ANIMACIÓN DEL HELICOPTERO.  
1215  
1216 switch (stateHeli) {  
1217  
1218     case 0:  
1219         rotHelHeliY += 0.5;  
1220         desHeli -= 0.02;  
1221         if (desHeli < 0)  
1222             stateHeli = 1;  
1223         break;  
1224     case 1:  
1225         rotHelHeliY = rotHelHeliY + (0.5*(freno));  
1226         freno -= 0.001;  
1227         if (freno < 0)  
1228             stateHeli = 2;  
1229         break;  
1230     default:  
1231         break;  
1232 }
```

Los resultados fueron los siguientes:



5. Completar el modelado para todas las articulaciones y los movimientos del buzz tomar como base lo visto en clase.

Para el este ejercicio se retomó el código hecho en clase, pero se agregaron los siguientes eventos del teclado:

```
762 //----->CONTROL DEL MODELO BUZZ
763 if (modelSelected == 3 && glfwGetKey(window, GLFW_KEY_LEFT) == GLFW_PRESS)
764     modelMatrixBuzz = glm::rotate(modelMatrixBuzz, 0.02f, glm::vec3(0, 1, 0)); //gitar
765 else if (modelSelected == 3 && glfwGetKey(window, GLFW_KEY_RIGHT) == GLFW_PRESS)
766     modelMatrixBuzz = glm::rotate(modelMatrixBuzz, -0.02f, glm::vec3(0, 1, 0)); //gitar al otro lado
767 if (modelSelected == 3 && glfwGetKey(window, GLFW_KEY_UP) == GLFW_PRESS)
768     modelMatrixBuzz = glm::translate(modelMatrixBuzz, glm::vec3(0, 0, 0.02)); //avanzar
769 else if (modelSelected == 3 && glfwGetKey(window, GLFW_KEY_DOWN) == GLFW_PRESS)
770     modelMatrixBuzz = glm::translate(modelMatrixBuzz, glm::vec3(0, 0, -0.02)); //retroceder
771 if (modelSelected == 4 && glfwGetKey(window, GLFW_KEY_LEFT_SHIFT) == GLFW_RELEASE && glfwGetKey(window, GLFW_KEY_1) == GLFW_PRESS)
772     rotBuzzLeftArm += 0.2;
773 else if (modelSelected == 4 && glfwGetKey(window, GLFW_KEY_LEFT_SHIFT) == GLFW_PRESS && glfwGetKey(window, GLFW_KEY_1) == GLFW_PRESS)
774     rotBuzzLeftArm -= 0.2;
775 if (modelSelected == 4 && glfwGetKey(window, GLFW_KEY_LEFT_SHIFT) == GLFW_RELEASE && glfwGetKey(window, GLFW_KEY_2) == GLFW_PRESS)
776     rotBuzzRightArm += 0.2;
777 else if (modelSelected == 4 && glfwGetKey(window, GLFW_KEY_LEFT_SHIFT) == GLFW_PRESS && glfwGetKey(window, GLFW_KEY_2) == GLFW_PRESS)
778     rotBuzzRightArm -= 0.2;
779 if (modelSelected == 4 && glfwGetKey(window, GLFW_KEY_LEFT_SHIFT) == GLFW_RELEASE && glfwGetKey(window, GLFW_KEY_3) == GLFW_PRESS)
780     rotBuzzLeftLeg += 0.2;
781 else if (modelSelected == 4 && glfwGetKey(window, GLFW_KEY_LEFT_SHIFT) == GLFW_PRESS && glfwGetKey(window, GLFW_KEY_3) == GLFW_PRESS)
782     rotBuzzLeftLeg -= 0.2;
783 if (modelSelected == 4 && glfwGetKey(window, GLFW_KEY_LEFT_SHIFT) == GLFW_RELEASE && glfwGetKey(window, GLFW_KEY_4) == GLFW_PRESS)
784     rotBuzzRightLeg += 0.2;
785 else if (modelSelected == 4 && glfwGetKey(window, GLFW_KEY_LEFT_SHIFT) == GLFW_PRESS && glfwGetKey(window, GLFW_KEY_4) == GLFW_PRESS)
786     rotBuzzRightLeg -= 0.2;
```

Se agregaron variables para cada extremidad:

```
147 float rotBuzzHead = 0.0, rotBuzzLeftArm = 0.0, rotBuzzLeftLeg = 0.0, rotBuzzRightArm = 0.0, rotBuzzRightLeg = 0.0;
```

Tambien se articularon cada una de las extremidades:

```
1119 //----->Renderizando a Buzz
1120 glm::mat4 modelMatrixBuzzBody = glm::mat4(modelMatrixBuzz);
1121 modelMatrixBuzzBody = glm::scale(modelMatrixBuzzBody, glm::vec3(4.0, 4.0, 4.0));
1122 modelBuzzHead.render(modelMatrixBuzzBody);
1123 modelBuzzHip.render(modelMatrixBuzzBody);
1124
1125 glm::mat4 modelMatrixBuzzLeftArm = glm::mat4(modelMatrixBuzzBody); //----->
1126 modelMatrixBuzzLeftArm = glm::translate(modelMatrixBuzzLeftArm, glm::vec3(0.179319, 0.580728, -0.025066)); //----->
1127 modelMatrixBuzzLeftArm = glm::rotate(modelMatrixBuzzLeftArm, glm::radians(-60.0f), glm::vec3(0.0, 0.0, 1.0)); //----->
1128 modelMatrixBuzzLeftArm = glm::rotate(modelMatrixBuzzLeftArm, rotBuzzLeftArm, glm::vec3(0.0, 1.0, 0.0)); //----->Articulando el brazo
1129 modelMatrixBuzzLeftArm = glm::translate(modelMatrixBuzzLeftArm, glm::vec3(-0.179319, -0.580728, 0.025066)); //----->Izquierdo
1130 modelBuzzLeftArm.render(modelMatrixBuzzLeftArm); //----->
1131 modelBuzzLeftForearm.render(modelMatrixBuzzLeftArm); //----->
1132 modelBuzzLeftHand.render(modelMatrixBuzzLeftArm); //----->
1133
1134 glm::mat4 modelMatrixBuzzLeftLeg = glm::mat4(modelMatrixBuzzBody); //----->
1135 modelMatrixBuzzLeftLeg = glm::translate(modelMatrixBuzzLeftLeg, glm::vec3(0.061653, 0.358356, 0.010275)); //----->
1136 modelMatrixBuzzLeftLeg = glm::rotate(modelMatrixBuzzLeftLeg, rotBuzzLeftLeg, glm::vec3(1.0, 0.0, 0.0)); //----->Articulando Pierna
1137 modelMatrixBuzzLeftLeg = glm::translate(modelMatrixBuzzLeftLeg, glm::vec3(-0.061653, -0.358356, -0.010275)); //----->Izquierda
1138 modelBuzzLeftThigh.render(modelMatrixBuzzLeftLeg); //----->
1139 modelBuzzLeftCalf.render(modelMatrixBuzzLeftLeg); //----->
1140 modelBuzzLeftFoot.render(modelMatrixBuzzLeftLeg); //----->
```

```

1141 glm::mat4 modelMatrixBuzzRightArm = glm::mat4(modelMatrixBuzzBody);//----->
1142 modelMatrixBuzzRightArm = glm::translate(modelMatrixBuzzRightArm, glm::vec3(-0.17096, 0.579034, -0.032705));//----->
1143 modelMatrixBuzzRightArm = glm::rotate(modelMatrixBuzzRightArm, glm::radians(60.0f), glm::vec3(0.0, 0.0, 1.0));//----->
1144 modelMatrixBuzzRightArm = glm::rotate(modelMatrixBuzzRightArm, rotBuzzRightArm, glm::vec3(0.0, 1.0, 0.0));//----->Articulando Brazo
1145 modelMatrixBuzzRightArm = glm::translate(modelMatrixBuzzRightArm, glm::vec3(0.17096, -0.579034, 0.032705));//----->Derecho
1146 modelBuzzRightArm.render(modelMatrixBuzzRightArm);//----->
1147 modelBuzzRightForearm.render(modelMatrixBuzzRightArm);//----->
1148 modelBuzzRightHand.render(modelMatrixBuzzRightArm);//----->
1149
1150 glm::mat4 modelMatrixBuzzRightLeg = glm::mat4(modelMatrixBuzzBody);//----->
1151 modelMatrixBuzzRightLeg = glm::translate(modelMatrixBuzzRightLeg, glm::vec3(0.061653, 0.358356, 0.010275));//----->
1152 modelMatrixBuzzRightLeg = glm::rotate(modelMatrixBuzzRightLeg, rotBuzzRightLeg, glm::vec3(1.0, 0.0, 0.0));//----->Articulando Pierna
1153 modelMatrixBuzzRightLeg = glm::translate(modelMatrixBuzzRightLeg, glm::vec3(-0.061653, -0.358356, -0.010275));//----->Derecha
1154 modelBuzzRightThigh.render(modelMatrixBuzzRightLeg);//----->
1155 modelBuzzRightCalf.render(modelMatrixBuzzRightLeg);//----->
1156 modelBuzzRightFoot.render(modelMatrixBuzzRightLeg);//----->
1157
1158 modelBuzzLeftWing1.render(modelMatrixBuzzBody);
1159 modelBuzzLeftWing2.render(modelMatrixBuzzBody);
1160 modelBuzzRightWing1.render(modelMatrixBuzzBody);
1161 modelBuzzRightWing2.render(modelMatrixBuzzBody);
1162
1163 modelBuzzTorso.render(modelMatrixBuzzBody);
1164

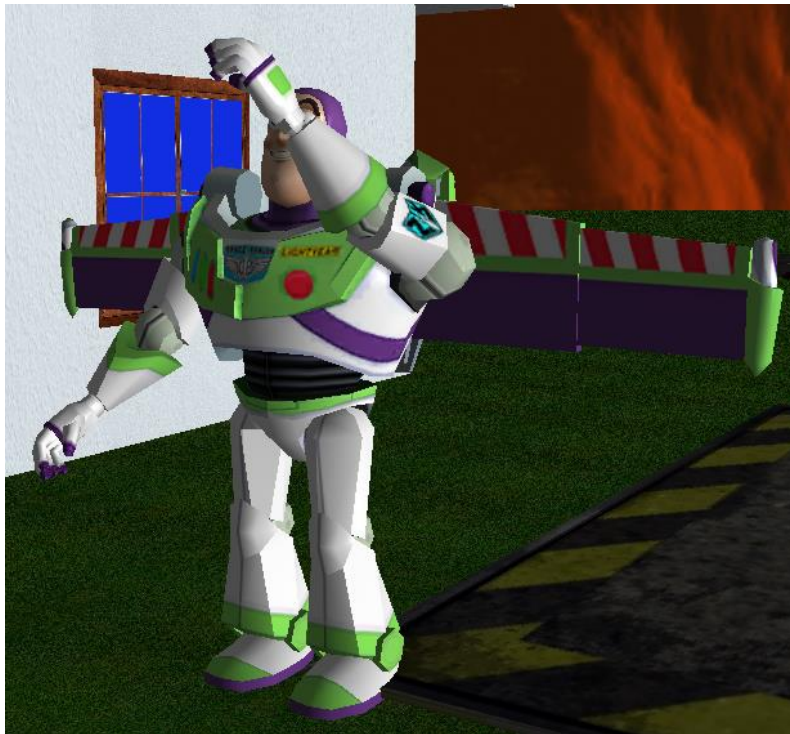
```

Al compilar, ejecutar el modelo de Buzz se puede ver de la siguiente forma:

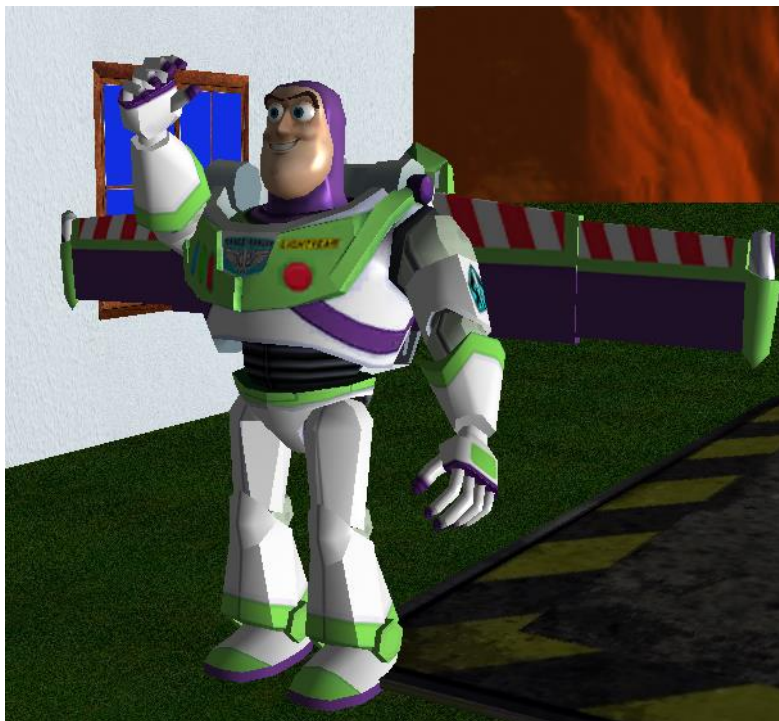


Al seleccionar el modelo Selected 4 podemos:

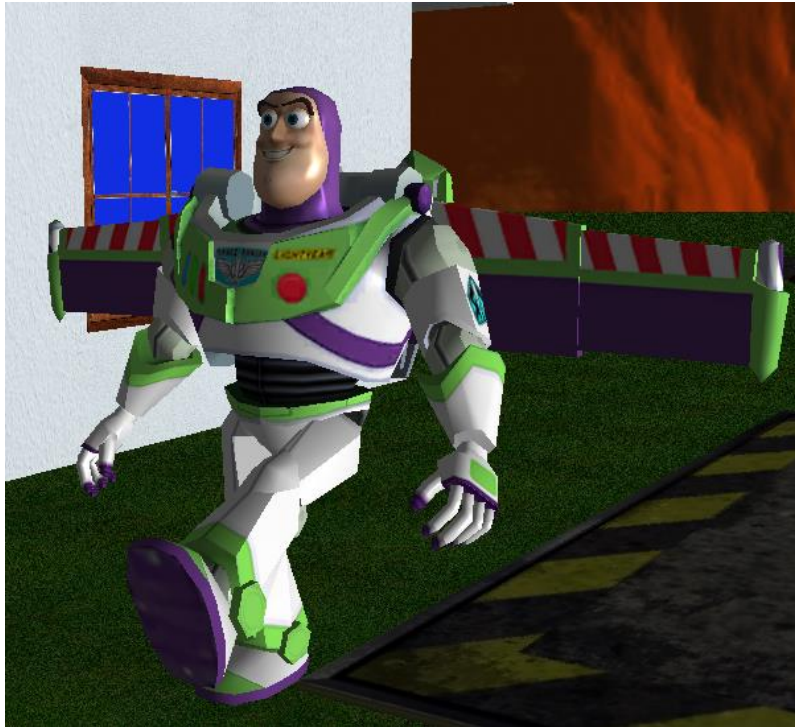
Mover el brazo izquierdo presionando 1 o shift+1



Mover el brazo derecho presionando 2 o shift+2



Mover la pierna izquierda presionando 3 o shift+3



Mover la pierna derecha presionando 4 o shift+4



Conclusiones.

Para poder animar un objeto por medio de maquinas de estados y keyframes es necesario tener en cuenta todas las transformaciones que se le aplicara a cada modelo, además de los pivotes y ejes de referencia que nuestros modelos usan para poder moverse.

Link a repositorio

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