Departamento de Ciencias de la Computación
Facultad de Ciencias Físicas y Matemáticas
Universidad de Chile
CC3501-1 Modelación y Computación Gráfica

MIGRACIÓN DE LAS AVES

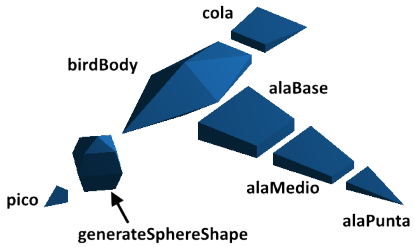
Tarea 2: OpenGL3D

Alumno: Alfredo Escobar
Profesor: Daniel Calderón
Auxiliares: Nelson Marambio Q.
Alonso Utreras
Ayudantes: Tomás Calderón R.
Nadia Decar
Beatriz Graboloza M.
Heinich Porro Sufan
Fecha: 14 de junio de 2020

1. Solución Propuesta

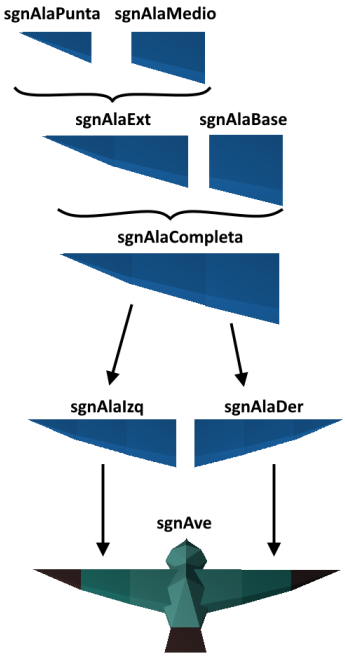
1.1. local_shapes.py

Primero se han definido funciones para la creación de las distintas GPUShapes correspondientes a las partes del ave. En cada una de estas funciones, se comienza por definir la posición de los vértices, y luego estos son entregados en orden a las funciones "createColorNormals—Indexation" para obtener nuevas listas de vértices, normales e índices.



También se definieron tres funciones para la creación de cubos con texturas: una de ellas no considera normales, otra usa normales interiores (hacia el centro del cubo) y la otra usa normales hacia el exterior.

Luego, en la función "crearAve", se unen todas las partes del ave mediante nodos de grafo de escena. Se comienza creando un nodo por cada GPUShape del ave. Luego, se crea un nodo que contenga como hijos tanto a la punta del ala como a la parte del medio del ala. Después, se crea otro nodo cuyos hijos son el nodo anterior y el nodo correspondiente a la sección más interna del ala.



A continuación, hacemos instancing del último nodo creado para ambas alas. Finalmente, creamos un nodo para el ave completa que tenga como hijos tanto a los dos nodos anteriores (las alas) como al cuerpo, la cabeza, el pico y la cola del ave.

El objetivo de esta estructura es el poder manejar la rotación de las distintas secciones del ala correctamente. Para simular el aleteo, se utilizarán transformadas de rotación en los nodos "sgnAlaPunta", "sgnAlaExt" y "sgnAlaCompleta".

Finalmente, tenemos funciones de instancing para aves y para palos de madera. La función de instancing para aves ordena automáticamente a las aves en forma de "V".

1.2. bird.py

Inicialmente se compilan distintos shaders de iluminación para objetos simples (seleccionables mediante las teclas Q, W y E). Se llama a la función "crearAve" para obtener un sólo pájaro. Luego, y durante toda la ejecución del programa, se obtienen los estados de las teclas de dirección, lo que permite cambiar la posición de la cámara.

Además, se obtiene la posición del cursor, y si este se encuentra dentro de la ventana del programa, será utilizado para calcular los valores de rotación de las alas. La posición vertical del mouse es usada para calcular directamente la rotación del nodo "sgnAlaCompleta". Luego, la rotación de "sgnAlaExt" variará progresivamente según la rotación del nodo previo, esto para generar un movimiento de aleteo fluido. Lo mismo sucede con la rotación del nodo "sgnAlaPunta": varía progresivamente según la rotación de "sgnAlaExt".

Al final de cada ciclo, se dibuja al pájaro con el pipeline seleccionado (por defecto se usa iluminación por pixeles o Phong)

1.3. bird-herd.py

El programa inicia obteniendo el nombre del archivo .csv a usar para la trayectoria de las aves.

Se han definido las funciones "generateT", "catmullRomMatrix" y "evalCurve" que, en conjunto, permiten obtener una lista de posiciones entre dos puntos, dados cuatro puntos en total. Estas posiciones formarán una curva de Catmull-Rom definida por los cuatro puntos entregados.

Luego, la función "caminoCompleto" toma una lista más larga de puntos, y utiliza a las funciones anteriores para obtener posiciones adicionales entre todos sus puntos (con excepción del primer y el último punto de la lista).

La siguiente función, "anguloVuelo", se encarga de retornar un ángulo entre dos puntos dados.

Se compilan distintos shaders de iluminación: 3 para cuerpos simples y 2 para cuerpos con texturas. Además, se compila un shader para objetos con textura y sin iluminación. Se llama a la función de instancing de aves (en local_shapes.py) para generar un conjunto de 5 pájaros. Luego, se crean los GPUShapes con texturas: el fondo, el mirador de aves y palos de madera:

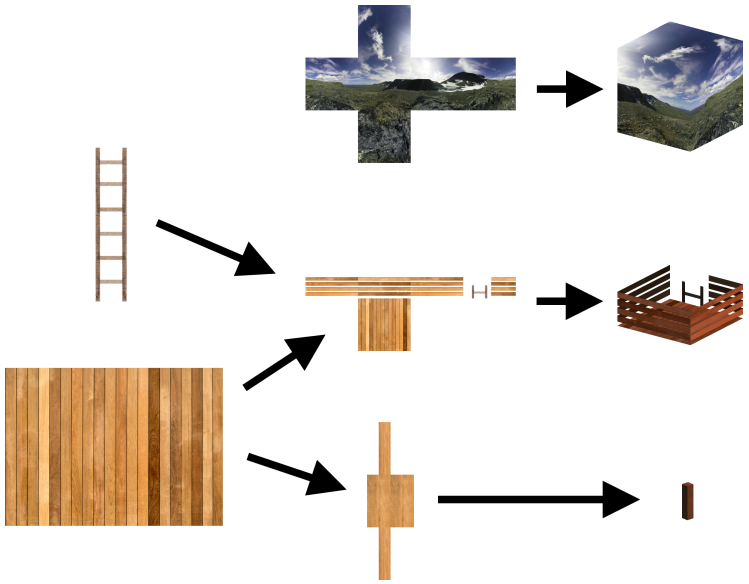


Figura 1: El cubemap del mirador de aves se hizo a partir de una imagen de tablones de madera y una imagen de una escalera. La imagen de los tablones también se usó para la textura de los palos. Las fuentes de las imágenes se encuentran al final del documento.

A continuación, se lee el archivo .csv, y se llama a la función “caminoCompleto” para generar una lista larga de posiciones (llamada “trayectoria”) que las aves deben seguir.

Ya en el loop principal, se obtiene la posición del cursor, y en base a esta se calculan los ángulos de barrido de la cámara, para luego construir la matriz de vista.

Luego, se aumenta el valor de la variable “alas_theta” en función del tiempo. Esta se usa para determinar la transformada de rotación de “sgnAlaCompleta”. Las rotaciones de “sgnAlaExt” y “sgnAlaPunta” se obtienen de la misma forma que se obtienen en bird.py

Después, hacemos que las aves avancen tomando un nuevo valor de la lista “trayectoria”, y hacemos que el grupo rote en base a la diferencia entre la posición actual y la posición anterior, llamando a la función “anguloVuelo”.

Finalmente, dibujamos todos los objetos. Las aves son dibujadas con uno de los 3 pipelines para cuerpos simples con iluminación, el mirador de aves y los palos son dibujados con uno de los 2 pipelines para cuerpos con textura con iluminación (se usa iluminación por caras o por vértices, pues la iluminación por pixeles provocó errores con las transparencias de la textura del mirador), y el fondo es dibujado con el pipeline para cuerpos con textura sin iluminación.

2. Instrucciones de Ejecución

2.1. Argumentos

El primer programa se ejecuta con la siguiente llamada:

```
python bird.py
```

El segundo programa se ejecuta con la siguiente llamada:

```
python bird-herd.py path.csv
```

En donde path.csv corresponde a un archivo de separador de comas que contiene distintos puntos 3D. Estos se utilizarán para formar una trayectoria (mediante splines de Catmull-Rom) que las aves recorrerán.

2.2. Control

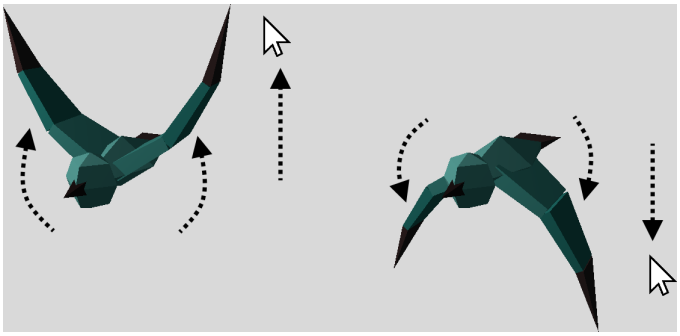
En bird.py se puede controlar el aleteo del ave moviendo cursor hacia arriba o hacia abajo **sobre la ventana**. Se puede cambiar el pipeline de sombreado con las teclas "Q" (Flat), "W" (Gouraud) y "E" (Phong). Además, se puede mover la cámara con las flechas de dirección.

En bird-herd.py se puede controlar la cámara moviendo el cursor vertical u horizontalmente, **sin importar si este está sobre la ventana o no**. Se puede cambiar el pipeline de sombreado con las teclas "Q" (Flat), "W" (Gouraud) y "E" (Phong para las aves y Gouraud para el mirador de aves).

Finalmente, se puede cerrar cualquiera de los dos programas mediante la tecla "ESCAPE".

3. Resultados

3.1. bird.py



En este programa se puede controlar el aleteo de un ave moviendo el mouse **sobre la ventana** arriba o abajo. También se puede mover la cámara mediante las teclas de flechas de dirección.

3.2. bird-herd.py

En este programa se puede observar a un grupo de 5 aves siguiendo una trayectoria. Esta es definida por el archivo .csv elegido al momento de ejecutar el programa. Se puede mover la cámara vertical u horizontalmente con el mouse, **independientemente de si el puntero está dentro de la ventana o no**.

4. Fuentes de imágenes

Imagen de fondo:
http://www.humus.name/index.php?page=Textures&ID=78

Tablones de madera:
https://www.deviantart.com/simoonmurray/art/Wooden-Planks-New-Texture-03-160257925

Escalera:
https://www.hiclipart.com/free-transparent-background-png-clipart-ijfzq