Práctica Sólidos deformables

ANIMACIÓN 3D

ROBERTO HERENCIAS MUÑOZ

Universidad
Rey Juan Carlos

Instrucciones

Ejecución de la simulación

Seleccionamos el “3D Object” AmongUs del apartado de Jerarquía de objetos de la interfaz de Unity, y en el Inspector vemos que hay un apartado para vincular un Script al objeto.

Añadimos el ElasticSolid.cs, GizmoExample.cs y Parser.cs al apartado script del Inspector y saldrán todos los valores modificables public de los scripts.

En el Parser.cs deberemos colocar el amongus.1.node.txt (en el campo Node File) y el amongus.1.ele.txt (en el campo Ele File); estos dos documentos se encuentran en la carpeta “tetgen.txt” dentro de “Assets > Soruce > P2”.

De manera similar, en el objeto Sujección debemos añadir al apartado de script, el Fixer.cs. Esto hará que podamos asignar un “Objeto” que es el objeto que será Fixeado al entrar en contacto cube que contiene el script Fixer; ponemos el objeto AmongUs en el apartado “Objeto” y de esta forma AmongUs tendrá fixeados aquellos nodos que están en contacto con el cube Sujección.

Finalmente, tras colocar los parámetros deseados, se da al play de la escena.

Selección de parámetros

En la selección de parámetros del objeto AmongUs, todos vienen por defecto colocados para una simulación óptima, pero puede cambiarse la masa de los nodos, el stiffness de los springs y la aceleración de la gravedad.

También puede cambiarse entre el método de integración “Symplectic” o “Explicit”.

Por último, se quita la pausa para que pueda dar comienzo la simulación, desmarcando el bool “Paused”.

Componentes implementados

Primer requisito

Para el primer requisito se ha creado un script ElasticSolid.cs donde se crean los nodos y springs de un objeto de una manera muy similar a la práctica anterior, solo que esta vez no se le asignará ningún plano, y tampoco tendremos que sacar los nodos de una mesh. En este caso se crea sobre un Object Empty y los vértices se ponen a mano para formar un tetraedro, después, ahora ya si de la misma forma que anteriormente, se crean los nodos y los springs en función a esos vértices y a las aristas que unen estos.

A continuación, el script GizmoExample, recibe la lista de nodos y la lista de springs con un Getter, y los utiliza como parámetros para el pintado de las esferas y líneas respectivamente, que forman los gizmos.

Segundo requisito

Para el segundo requisito se ha utilizado tetgen para crear un txt que contenga las posiciones de los vértices de un objeto 3D, en mi caso, el recubrimiento de un muñeco de AmongUs. Posteriormente creamos en unity un script Parser, que se encarga de pasar esos txt con coordenadas en forma de string, a la lista de vértices y de tretaedros, con la que posteriormente en ElasticSolid.cs, crearemos la lista de nodos y springs (6 springs por cada tetraedro), de la misma forma que hacíamos con la mesh en la práctica de la tela, o con los vértices a mano, en el primer requisito de esta práctica.

Al igual que en el apartado 1, una vez están creados los nodos y los springs, se accede a ellos desde GizmoExample.cs, para pintar las esferas y líneas que forman los gizmos y poder mostrarlo por pantalla.

Tercer requisito

Para el tercer requisito se mete en unity el modelo HighPoly del que se hizo previamente el recubrimiento, y se obtienen sus vértices mediante la mesh, (similar a como se cogía la mesh de los planos en la práctica de telas).

Además, creamos tetraedros en una nueva clase Tetrahedrons, del recubrimiento con la clase parser, solo que esta vez antes de crear los springs con ellos, creamos tetraedros propiamente dichos, en los cuales podremos calcular el volumen.

Ahora para cada vértice del modelo highPoly tendremos que calcular el tetraedro al que pertenece (tetraedro que contiene las coordenadas en las que se encuentra el punto que actúa como vértice del highPoly), y las coordenadas baricéntricas, para saber cuánto “peso” ejerce cada vértice del tetraedro sobre el vértice del modelo.

Vamos actualizando la posición de esos vértices en cada updarte para que se mueva junto con los tetraedros del recubrimiento que ya se movían previamente, de esta forma dotamos de movimiento también al modelo highPoly.

Cuarto requisito

Para el cuatro requisito necesitaremos acabar con las aristas duplicadas, para de esta forma crear únicamente los springs compartidos por los tetraedros una única vez.

Por ello cuando estamos creando un tetraedro con sus correspondientes 6 aristas, comprobamos si estas ya son un spring creado, y en caso de no serlo, creamos un spring con las coordenadas de la arista que estamos tratando.

Para este apartado necesitaremos calcular la densidad del modelo, y asignarle un volumen a cada tetraedro, para poder sustituir la masa de los nodos con la correspondiente a la masa de densidad y el volumen del tetraedro asignado a esos vértices (cada nodo, tendrá una masa diferente que dependerá de la densidad y volumen del tetraedro del cual forma parte).