

PROGRAMACIÓN GRÁFICA AVANZADA

EJERCICIO PRÁCTICO DE SHADERS

Profesor: Jordi Arnal

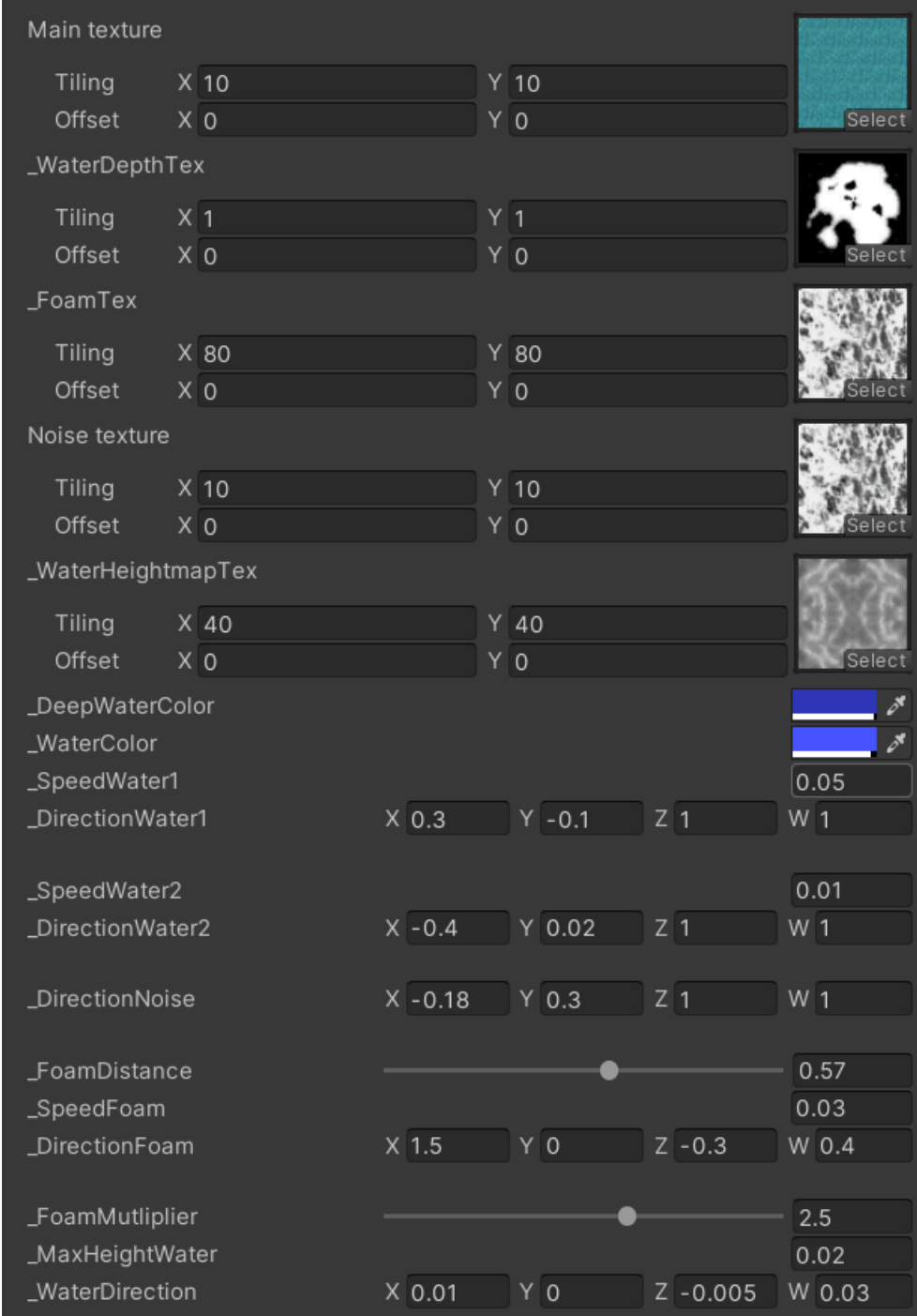
Fecha límite: 5 de marzo de 2023

SOBRE LA PRÁCTICA

Para la realización de la práctica deberéis implementar un shader que simule un efecto de agua con movimiento, olas y espuma.

A partir del proyecto que se os entrega deberéis modificar el shader de agua dado.

- Podéis crear propiedades como las siguientes para implementar el shader.



The image shows a Unity Shader Inspector window with the following properties and values:

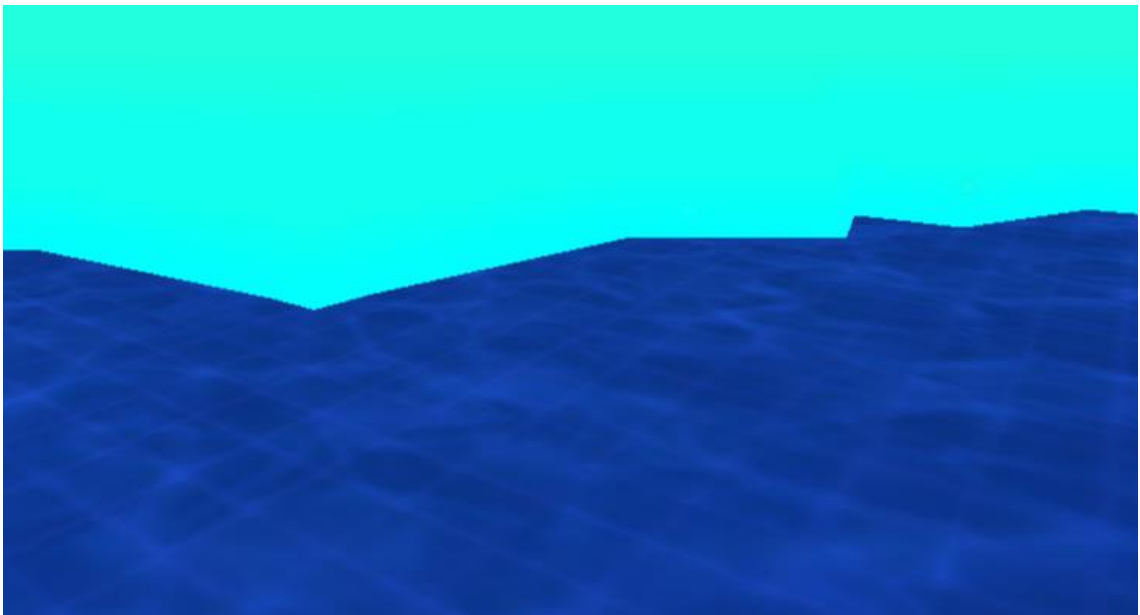
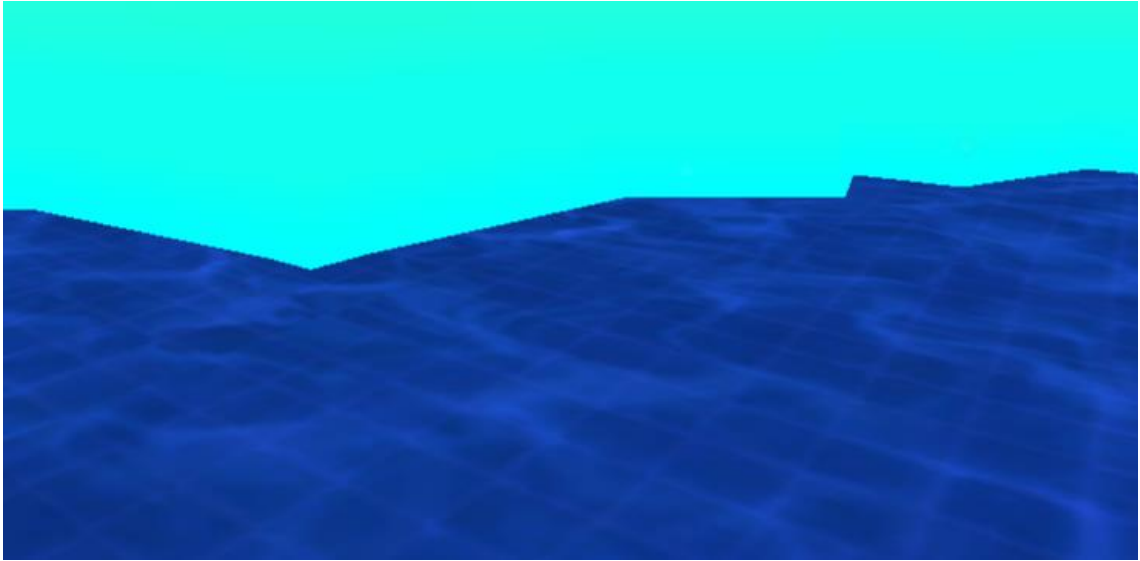
- Main texture**
 - Tiling: X 10, Y 10
 - Offset: X 0, Y 0
- _WaterDepthTex**
 - Tiling: X 1, Y 1
 - Offset: X 0, Y 0
- _FoamTex**
 - Tiling: X 80, Y 80
 - Offset: X 0, Y 0
- Noise texture**
 - Tiling: X 10, Y 10
 - Offset: X 0, Y 0
- _WaterHeightmapTex**
 - Tiling: X 40, Y 40
 - Offset: X 0, Y 0
- _DeepWaterColor** (Color field)
- _WaterColor** (Color field)
- _SpeedWater1**: 0.05
- _DirectionWater1**: X 0.3, Y -0.1, Z 1, W 1
- _SpeedWater2**: 0.01
- _DirectionWater2**: X -0.4, Y 0.02, Z 1, W 1
- _DirectionNoise**: X -0.18, Y 0.3, Z 1, W 1
- _FoamDistance**: 0.57 (Slider)
- _SpeedFoam**: 0.03
- _DirectionFoam**: X 1.5, Y 0, Z -0.3, W 0.4
- _FoamMutliplier** (Note the typo): 2.5 (Slider)
- _MaxHeightWater**: 0.02
- _WaterDirection**: X 0.01, Y 0, Z -0.005, W 0.03

- Lo primero que deberéis realizar será mover el agua, para ello deberéis utilizar dos vectores de dirección (**_DirectionWater**) que transformarán las coordenadas UV según el tiempo y la velocidad (**_SpeedWater**).

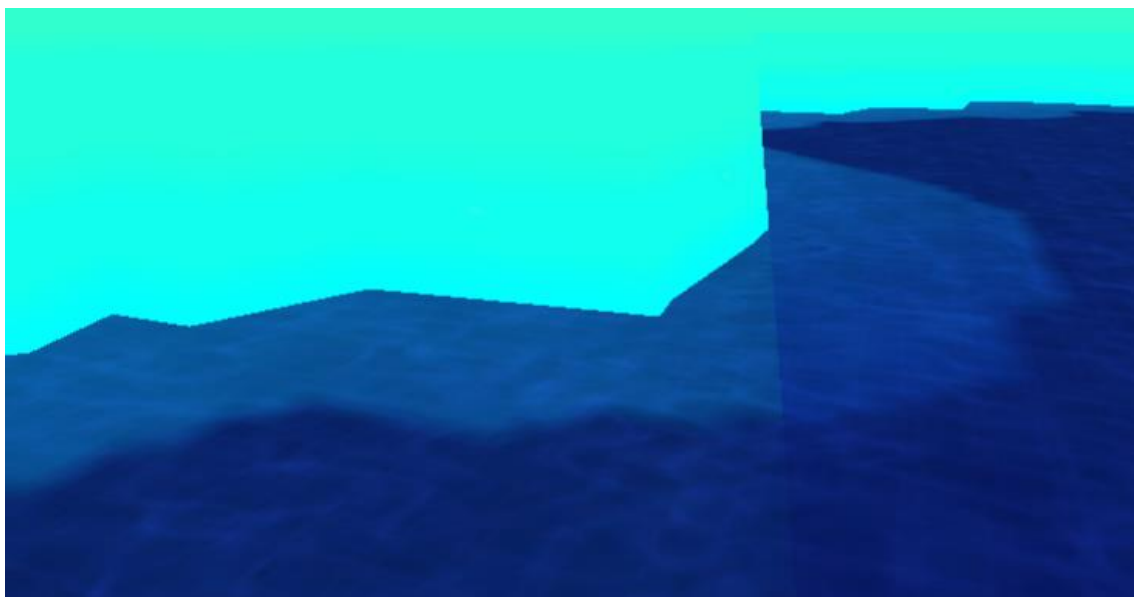
- A continuación añadiremos un efecto de aguas con poca o mucha profundidad, para ello utilizaremos una textura prebakeada que nos dará la distancia del agua al suelo (`_WaterDepthTex`), dependiendo si el color de la textura es blanco o negro utilizaremos el color de aguas profundas o aguas de poca profundidad.
- Para terminar con la parte del pixel shader vamos a añadir un poco de espuma a nuestra agua, para ello utilizaremos de nuevo el color de las texturas de profundidades (`_WaterDepthTex`) y si el valor del componente Red es superior a un `threshold _FoamDistance` entonces aplicaremos espuma, para ello utilizaremos las coordenadas UV del plano tileado según la textura `_FoamTex`, y sumaremos el color de la espuma sobre el resultado final de nuestro Pixel Shader. Deberíamos hacer algún tipo de operación como desplazar las coordenadas UV para que la espuma no esté quieta y meterle alguna operación de ruido para que el agua se vaya disolviendo, podéis utilizar una textura de noise o buscar alternativas.
- Por último nos centraremos en la transformación de los vértices para dar un efecto de olas. Basándonos en como realizamos el shader de terrain, modificaremos la altura del vértice del agua según la propiedad de altura máxima (`_MaxHeightWater`), utilizaremos una textura de ruido como heightmap de las olas (`_WaterHeightmapTex`), añadiremos un movimiento en las UV según el tiempo y una o dos direcciones para conseguir movimiento en las olas.
- Deberéis modificar la estructura del vértice que le llega al pixel shader según las propiedades que necesitéis, podéis crear diferentes coordenadas de UV para la profundidad del agua, espuma, ruido o lo que necesitéis.

RESULTADO PASO A PASO

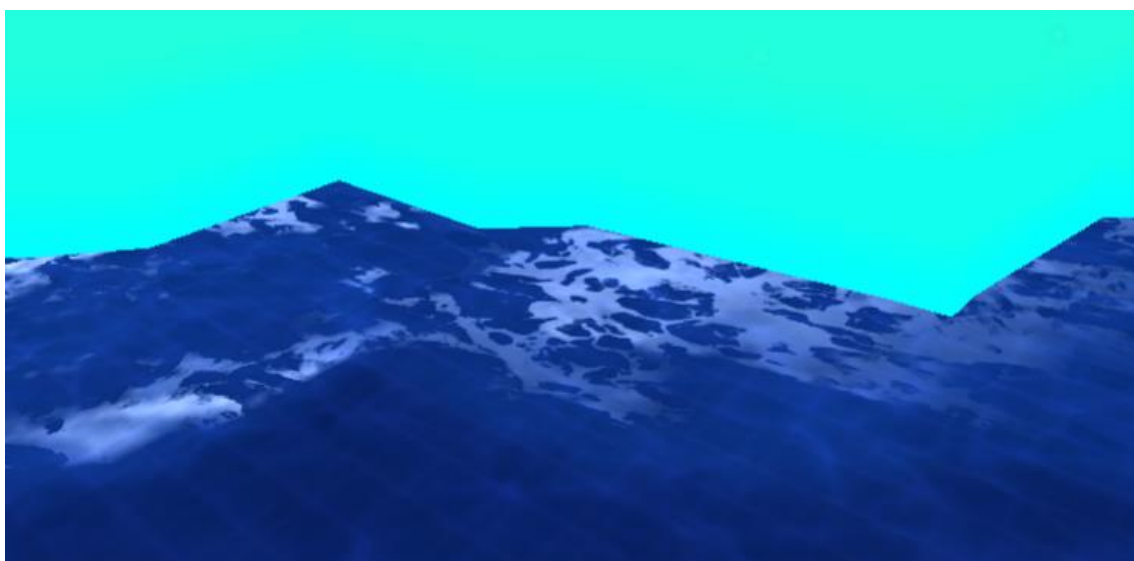
Moviendo el agua



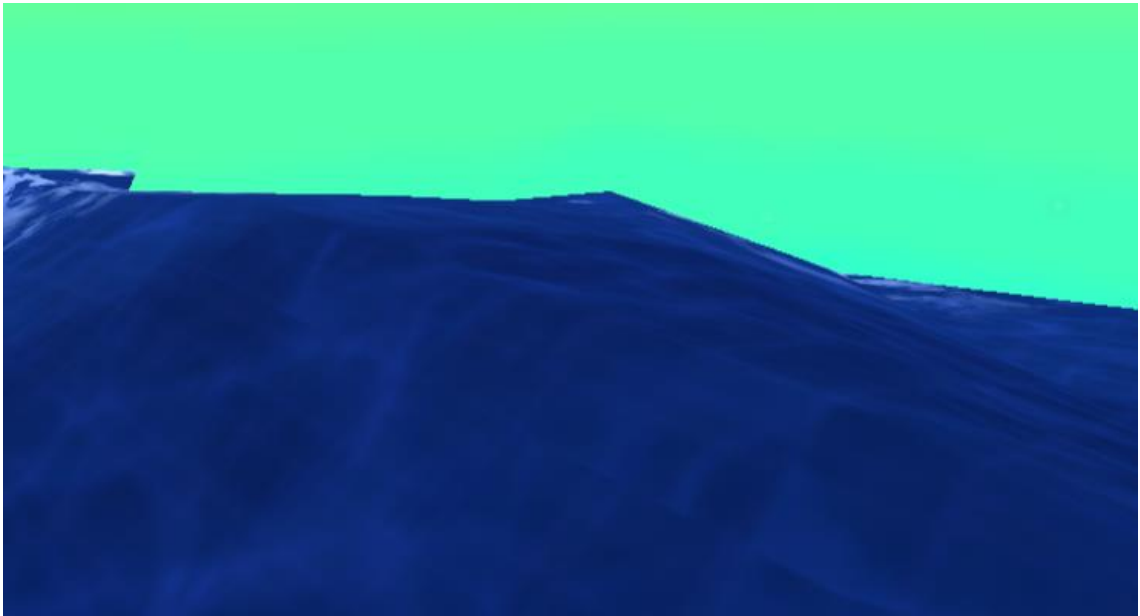
Dos colores de agua



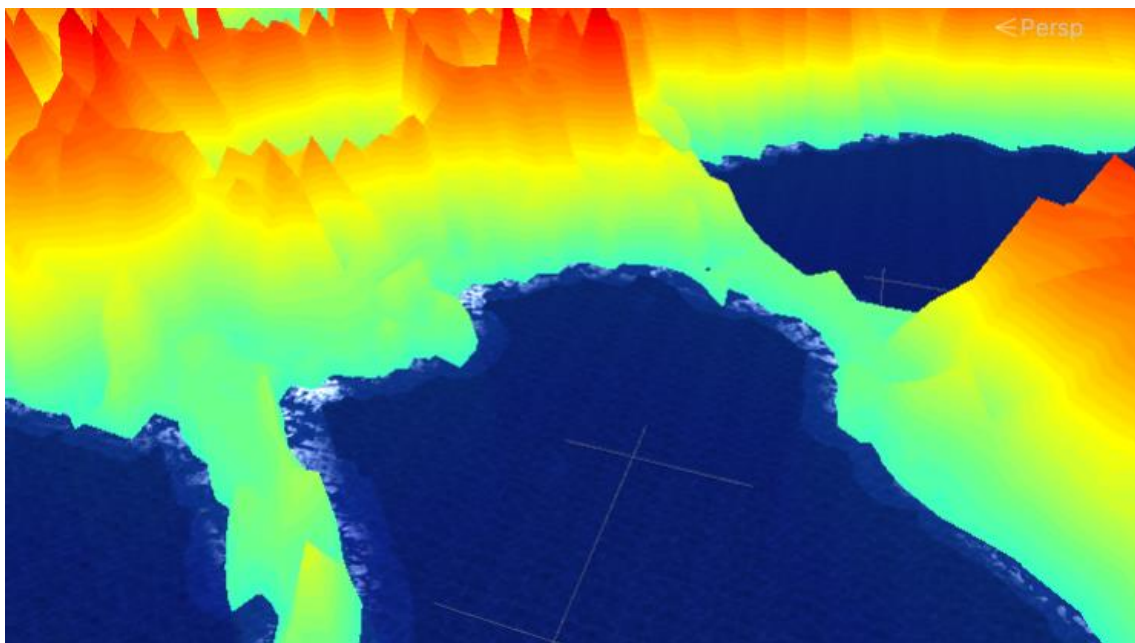
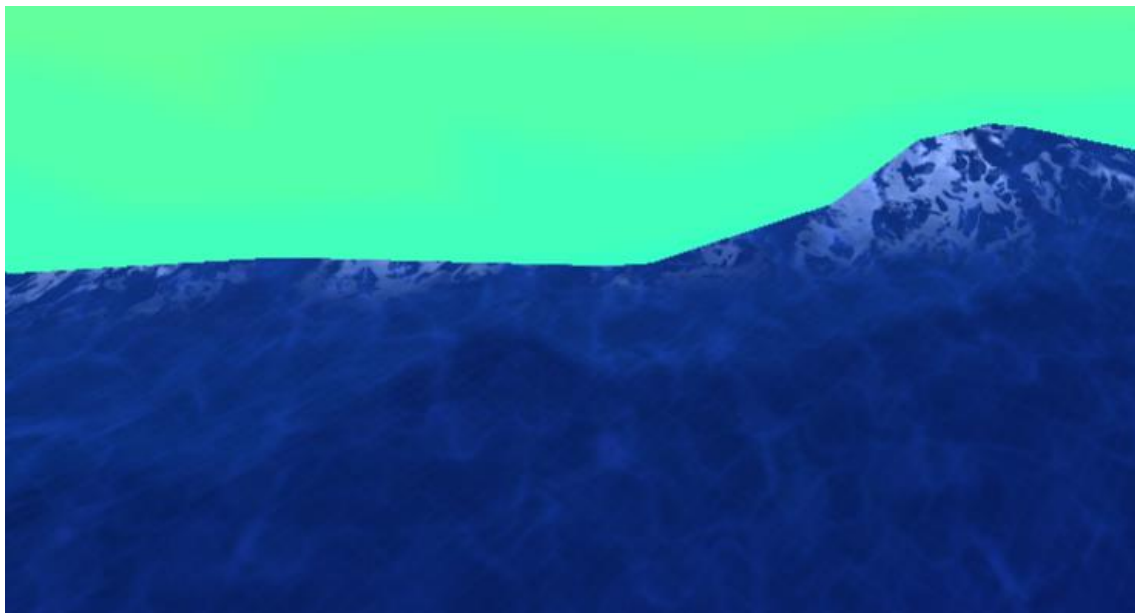
Añadir espuma



Añadir olas



Resultado final



Enlace a youtube

<https://youtu.be/-Ki0WtHEaq4>

FORMATO DE ENTREGA

La práctica se deberá realizar por parejas.

La entrega se deberá realizar a través del aula virtual antes del fin del día de la fecha límite.

Para entregar el proyecto eliminar las carpetas Library y VS, comprimir todo el proyecto.

Se deberá entregar dentro del archivo comprimido un txt con los nombres de los alumnos que han participado.

Además deberá enviarse un mail a jarnal@tecnocampus.cat con la nota de pares del compañero.

La rúbrica de la práctica será la siguiente.

Implementación shader agua	7 puntos
Implementación extra	2 puntos
Nota de pares	1 punto