# Tarea12

## *Water Shader*

Material Recomendado:

https://docs.gameloft.org/3d-training/#Your_Tasks -> **More exercise with the learned techniques: Implementing a water shader**

**Instrucciones**:

Los recursos para este shader serán:

- El geometry file: Utilizar en que está dentro de WaterGeometry.rar
- El normal map texture: tomarlo de WaterNormal.rar
- Rock.tga para el fondo del agua

El agua se simula con un solo objeto con dos superficies (superior y fondo). Como es un solo objeto, la componente luz (ambiental, difusa y especular) solo se aplicara una sola vez, decidiremos en cuál de las dos superficies será necesario aplicarla.

El agua se debe obtener de la combinación de los colores de estas dos superficies:

Superior: reflectionColor

Inferior/Fondo: refracción.

La combinación es realizada con ayuda de un Fresnel, el fesnel actuara como peso:

```
gl_FragColor = vec4(mix(refractionColor, reflectColor, fresnelTerm), 1.0);
```

Dónde:

**fresnelTerm** = pow((1 – abs( dot( N, normalize(ToEye )) )), FresnelPower)

→ **FresnelPower** – Valores más altos hacen que el agua sea más transparente (prueba valores entre 1.0 y 2.0 y has una comparación)

➔ **N** – Es la normal sacada del normal map.

Básicamente cuando vemos perpendicularmente sobre el agua, tendrás la refracción al máximo y tendrás casi nula reflexión. Cuando veas tangencialmente veras al máximo la reflexión y tendrás cero refracción.

**Fresnel**, el reflejo que percibimos de una superficie depende del ángulo de visión. En la siguiente imagen puedes observar que en la imagen de la izquierda estamos viendo desde arriba directo hacia el agua y hay muy poca reflexión y puedes ver el fondo de la piscina. Si vemos al nivel del agua veras más luz especular y reflexiones sobre la superficie del agua, y no serás capaz de ver el fondo de la piscina.


**A. Superficide inferior/Fondo (Color de refraccion)**

Esta es una superficie mate, por lo tanto será iluminada por la luz ambiental y difusa del Phong (como es mate, no es muy reflectivo, por lo tanto no tendremos especular).

El refractionColor será calculado como la combinación entre el color inferior y el color del agua. La combinación será hecha en base a un depth factor.

refractionColor = `mix(bottomColor, waterColor, depthFactor);`

Donde:

**watercolor** es una unirform

**depthFactor** = clamp(u_depthAdjust * v_uv.x, 0.0, 1.0);

- ➔ v_uv.x – asumimos que la profundidad (profundidad del agua) para un vertex es dado por uv.x, porque no hay especificación en el archivo de geometría para esto.
- ➔ u_depthAdjust – es mandado como uniform, y será igual a 1 / max(uv.x) de tu archivo de geometría. Usa valores pequeños como por ejemplo 0.1.

bottomColor, es el sampling de la textura para el fondo (Rock.tga), el sampling es hecho con un desplazamiento como el utilizado en el fuego, la cual tiene que ser mayor cuando haya más profundidad ( porque la refracción es mayor para aguas más profundas y nosotros usaremos el desplazamiento uv para simular el desplazamiento).

Utilizaras una fórmula similar para la refracción con desplazamiento como lo hiciste con el fuego, pero dMax ya no será una constante.

```
offsetRefraction = dMaxRefraction * (2.0 * disp - 1.0);
```

Donde:

```
dMaxRefraction = v_uv.x * u_depthAdjustDisplacement
```

→ u_depthAdjustDisplacement – es una uniform para hacer el ajuste

Ahora, necesitaras:

- Hacer sampling de Rock.tga para obtener el bottomColor con el desplazamiento de uv.
- Calcular el refractionColor por medio de un mix
- Aplicar iluminación (Phong: ambiental y difusa) :

**refractionColor** = refractionColor * light

**B. Superficie superior (reflexión)**

Te recomiendo leer de nuevo la guía para hacer el Phong shader (clasificación de los objetos según su grado de reflexión).

La parte superior debe tener mucha reflexión, por lo tanto aplicaremos solamente la reflexión desde el cielo. Prácticamente, la reflexión desde el cielo nos dará una iluminación especular. No se harán otros cálculos para la iluminación (por lo tanto, **no usaremos el iluminación Phong**).

Sigue los siguientes pasos:

1. Para obtener el efecto de ondas en el agua, necesitaras usar un normal mapping. Pero necesitaremos ondas que se muevan por lo que haremos el sampling con algo de desplazamiento uv utilizando la normal de WaterNomal.tga.

   ```
   offsetReflection = dMaxReflection * (2.0 * disp - 1.0);
   ```

   Donde:

   dMaxReflection – es una constante (uniform), como lo hiciste con el fuego, porque en la superior del agua no importa la profundidad.

   Tu sampling será de WaterNormal.tga para obtener la normal, tendrás que hacer los ajustes necesarios (repasa la guía para obtener el normal mapping shader), construye y aplica la matriz TBN. Obtén la normal en World Space.

2. Usa la normal obtenida anteriormente para calcular el vector reflexión y utiliza ese vector reflexión para hacer el sampling de la textura cubo (evnMap.tga). Lee de nuevo como realizar la reflexión del cielo en la documentación del skybox.

**C. Combinación de las 2 superficies**

- La combinación es hecha en la superficie 'superior', por lo tanto utiliza la normal calculada en el punto B ( la normal obtenida del normal map) para obtener el Fresnel
- Haz un mix de las dos superficies (superior e inferior) usando el Fresnel.

**Sigue las siguientes pasos para hacer este shader:**

- **stage 1**: simple reflection on the water plane
- **stage 2**: reflection combined with normal mapping
- **stage 3**: reflection + normal mapping + Fresnel (in the movie you will see that we used first a Fresnel power of 1 which creates an ice impression –not ok-, so we changed the power to 2 to become more transparent, so more water like)
- **stage 4**: the complete water reflection implementation: reflection + normal mapping + Fresnel + sliding displacement map
- **stage 5**: simple refraction with constant depth. In the movie we chose first a dMax of 0.025 to show you the refraction in the case of a shallow water and, after that, we changed it to 0.1 to show you the refraction for a deeper water
- **stage 6**: refraction with variable depth, computed from the u texture coordinate
- **stage 7**: refraction with variable depth and water color
- **stage 8**: the complete solution (stage4 + stage7).

**Resultado**: