

# Reporte de documentación: Acuario

CC3501-1 Modelación y Computación Gráfica para Ingenieros

Sebastián Olmos Hernández, 22 de julio de 2020

Primero se resuelve una EDP de tres dimensiones en el programa *aquarium-solver.py* con los parámetros entregados por el archivo *problem-setup.json* del cual para discretizar el problema se ocupa un  $h$  que depende de las dimensiones del acuario entregada, de este manera siempre se obtienen soluciones de tamaños similares. Luego se crea un método para resolver la ecuación similar a la trabajada en clases y auxiliares pero agregando una dimensión más, lo que equivale a trabajar con un stencil de 7 puntos, aplicando las condiciones de borde indicadas resulta que al construir la matriz que resuelve la ecuación hay que ponerse en 27 casos (interior, vértices, aristas, caras, etc.) en los que se puede ubicar el stencil y que requieren diferente tratamiento. Para finalizar esta parte se resuelve la ecuación matricial con ayuda de una matriz sparse.

Después de resolver la ecuación, se tiene la implementación del acuario, del cual se puede ver una versión simplificada en el diagrama de flujo de la figura 1. Para hacerlo más entretenido se ocuparon pokémon de tipo agua para los modelos de los peces, estos modelos fueron trabajos en blender para agregarle texturas, separarles la cola e implementar el movimiento de esta.

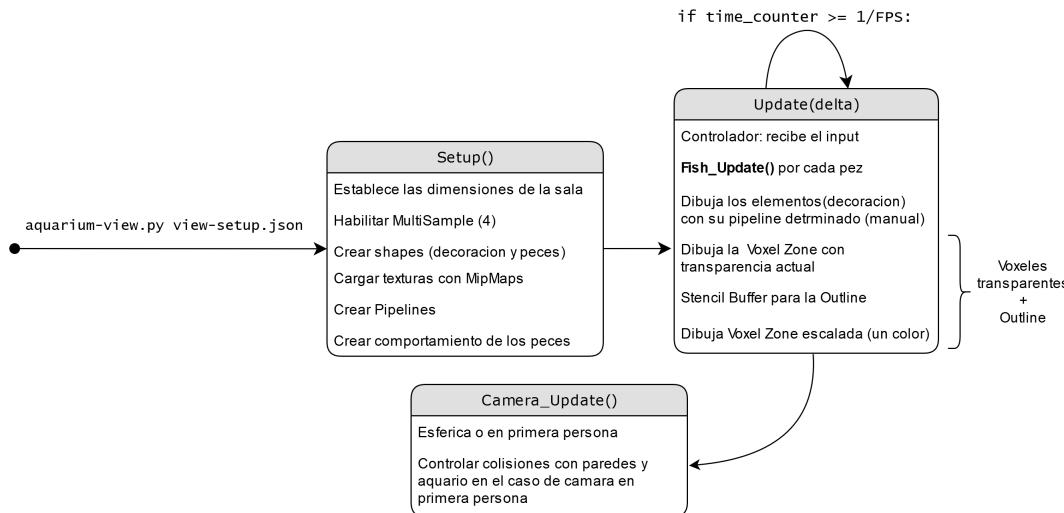


Figura 1: Diagrama de flujo simple.

Para la visualización de la escenas, se implementó una cámara en primera persona con colisiones y una cámara esférica centrada en el centro del acuario. También se trabajó en el Anti Aliasing para lo cual se implementaron texturas con MipMap para las texturas de la sala y se habilitó el multiSample (con glFW y openGL) que realiza cuatro muestra por fragmento a la hora de renderizar un píxel, resultando una gran mejora en la definición de la escena comparado con trabajos anteriores.

En cuanto el método para resaltar los voxels en los que pueden estar los peces, se crea un shape que contiene solamente los cuadrados/caras exteriores de los voxels, para que al momento de usar un shader con transparencias solo se vea la cara exterior (y no un cumulo de cubos) además de activar el *CullingFace* para las caras traseras en toda la escena. Con el fin de resaltar bien estas zonas se implementó una outline con la técnica del stencil buffer (explicado mejor en el código) que en simples palabras permite no renderizar una sección de alguna shape. Junto a una pequeña función que ordena los colores de las outlines según su temperatura, siendo la más fría de color morado, la más caliente de color naranja y la intermedia de color rojo.

Además es importante señalar que para esta implementación no se utilizaron SceneGraphs como tal, ya que al utilizar varios Pipelines y transparencias, el orden de dibujo de las shapes se hizo de forma manual

para no tener problemas con las transparencias.

Para darle más "vida." a la escena se implemento un movimiento para los peces en sus determinadas zonas, como se ve de manera simplificada en la Figura 2.

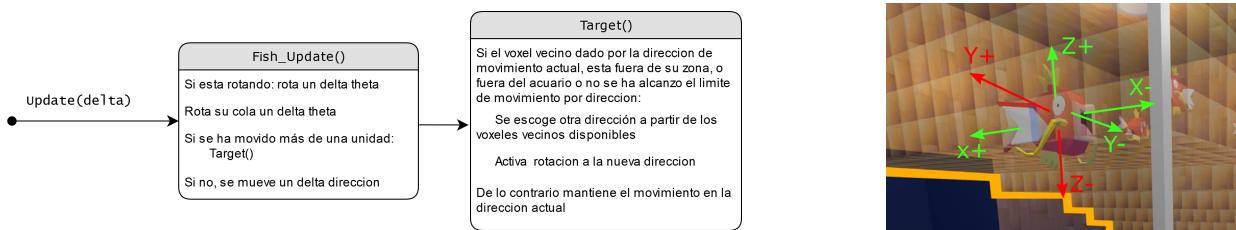


Figura 2: Diagrama del comportamiento de los peces y una representación de las direcciones disponibles de un pez, siendo los vectores verdes direcciones en la que si se puede mover y las rojas no.

Básicamente cada pez se encuentra moviéndose de voxel a voxel, comprobando si el siguiente esta en su rango de temperatura y no este fuera del dominio del acuario, en caso contrario cambia de dirección.

Por ultimo esta la construcción de la sala que se representa en la escena y tiene como centro al acuario. Para que todo quede de manera consistente, todas las dimensiones de la sala y de los distintos elementos están realizadas a partir de las dimensiones del acuario, Como se ve en la Figura 3. La velocidad del usuario y de los peces también están relacionadas con estas dimensiones, con el objetivo de tratar de modelar cualquier dimensión que tenga el acuario.

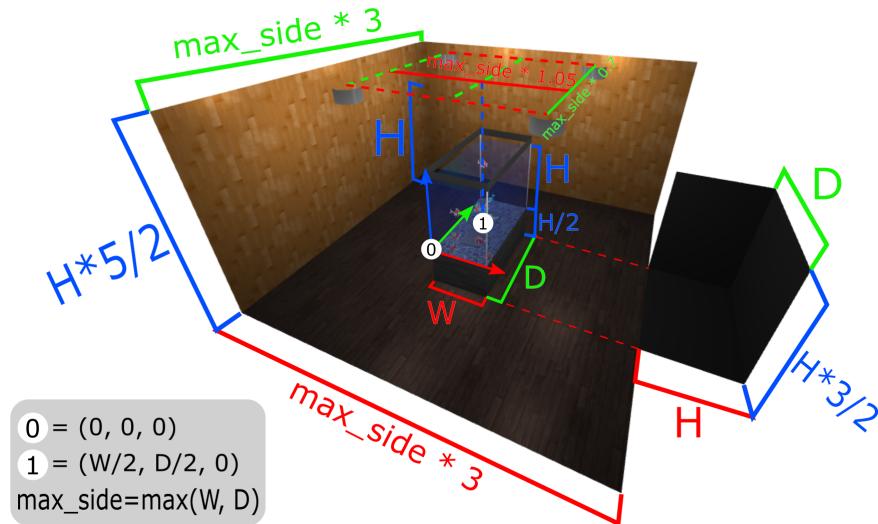


Figura 3: Esquema para indicar como esta construido la sala alrededor del acuario.

Además mediante la implementación de un cubemap de la escena hecha en blender y un pipeline, se logró crear reflejos en la cara exterior de los vidrios del acuario, mientras que para la cara interior se dibuja un color sólido.

## Instrucciones de Ejecución

Primero se tiene que resolver la EDP, por lo que se necesita ejecutar `python aquarium-solver.py problem-setup.json` con los parámetros deseados en el archivo `problem-setup.json`. Luego para la visualización del

acuario se ejecuta `pythonaquarium - view.pyview - setup.json` con la cantidad de peces y su temperatura preferida especificados en el archivo `view - setup.json`.

Si se da una temperatura que no coincide con algún voxel, se mostrara un mensaje en la consola, no se pueden crear los peces respectivos y no se mostrara dicha zona en el acuario.

Con la tecla *V* se cambia el tipo de cámara, la cámara esférica se controla con las flechas, donde las flechas arriba y abajo mueven la cámara hacia arriba y abajo (*theta*) y las teclas izquierda y derecha rotan la cámara en estas direcciones (*phi*), mientras que con las teclas *Z* y *X* controlan el zoom. Para la cámara en primera persona se controla con las teclas: *E*, *D*, *S*, *F* para ir hacia adelante, atrás, izquierda y derecha respectivamente, con las flechas direccionales rota la cámara en esas direcciones, y con la tecla Control izq. puede agacharse/pararse.

Por ultimo la visualización de los voxels se activa/desactiva con la tecla *Q*, si está activada se puede cambiar los voxels mostrados con las teclas *A*, *B*, *C* que muestran sus zonas respectivas, esto se hizo así para poder resaltar las zonas individualmente. con la outline.

## Resultados



Figura 4: Los tres modelos de peces/pokemon implementados

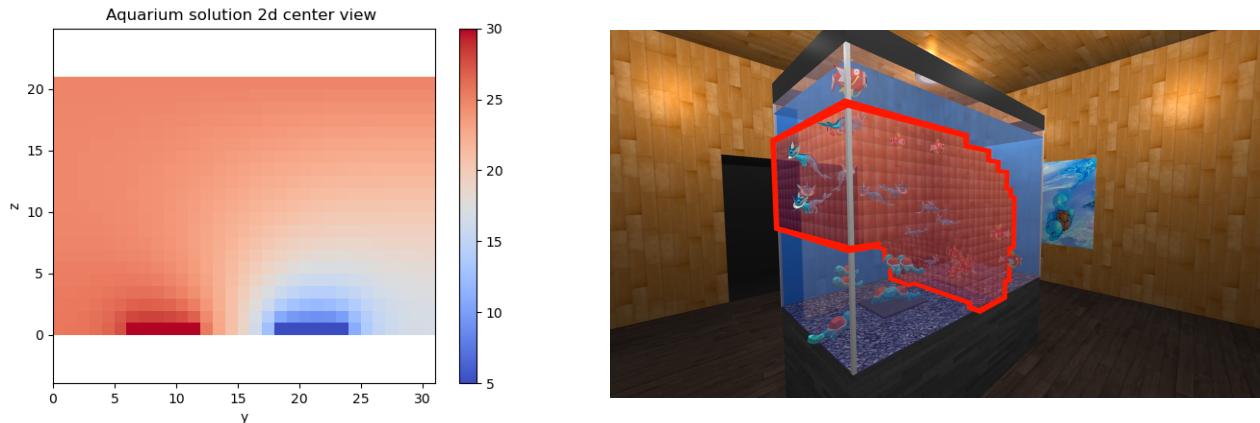


Figura 5: Solución de la EDP y su implementación en el acuario con los parámetros del enunciado.

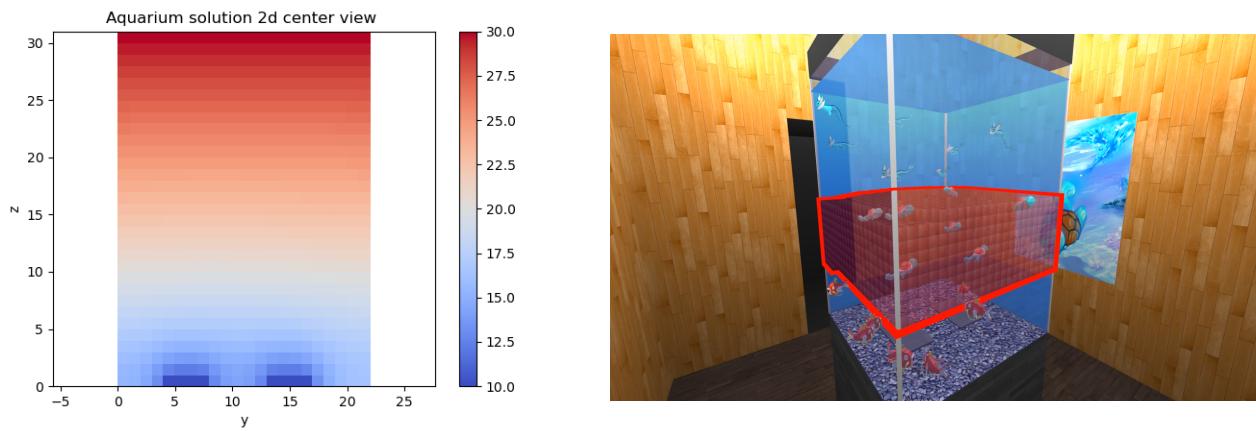


Figura 6: Solución de la EDP y su implementación en el acuario con otros parámetros.

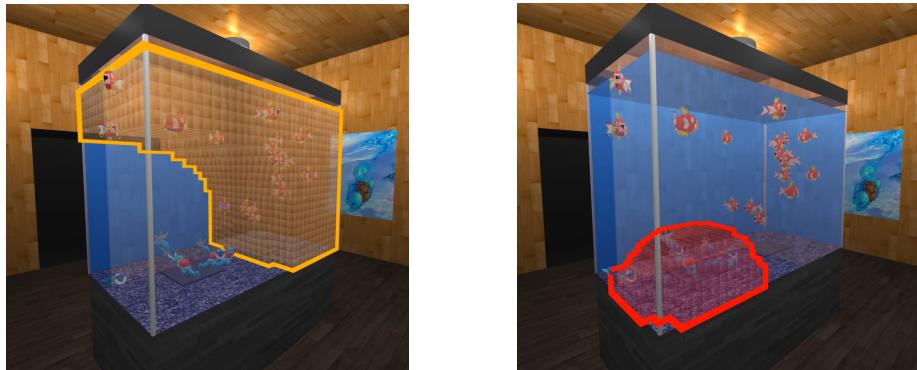


Figura 7: Vista de dos zonas de temperatura para el acuario con los para metros del enunciado (25C y 15C respectivamente).



Figura 8: Vista de la zona de 10C para el acuario con los para metros del enunciado.



Figura 9: Captura en el que se ven los distintos elementos decorativos de la escena.

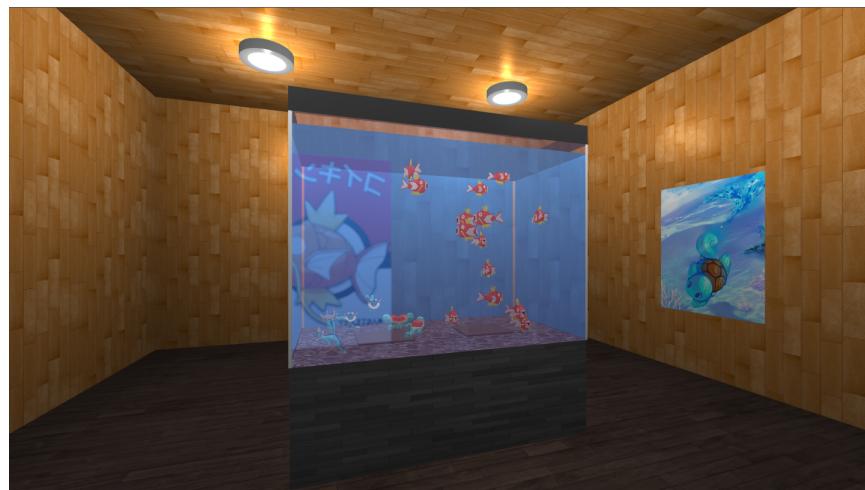


Figura 10: Captura en la que se ve el reflejo del vidrio del acuario(imagen del magikarp se encuentra detrás del usuario).