Pol Salvia 217698
Tomàs Sanchez 216810

Lab 3 Part 1&2 : Volume Rendering

Ray setup and loop

Per fer aquest apartat primer de tot hem creat la classe VolumeMaterial que és una rèplica de StandardMaterial on a més a més inicialitzem els shaders a la propia funció. A part, també hem canviat les variables uniformes i hem afegit a les que ja havien la matriu inversa de la model perquè d'aquesta manera en el shader podrem canviar de world coordinates a local coordinates. Aquest pas serà necessari ja que per computar els punts del volum serà molt mes facil si estan en variables locals.

Un cop en el shader, hem inicialitzat els vectors tal i com s'especificava a la guia de la pràctica. Primer hem calculat el step vector en world coordinates i l'hem convertit a coordenades locals amb la matriu inversa de la model. A part, em definit una variable que es diu MAX ITERATIONS per limitar els cops que s'executa el loop per conèixer el color del volum.

```
//Ray setup
vec3 step_vec = normalize(v_world_position-u_camera_position);

//canviem a coordenades locals

vec3 step_vector = (u_model_inv*vec4(step_vec,0.0)).xyz*u_length;
vec3 sample_position = v_position;
```

Inicialitzem el step vector i canviem de world coordinates a locals

Per últim, hem afegit les restriccions per acabar el loop abans d'executar totes les iteracions, en el cas que l'alpha fos igual a 1 i en cas que el sample point estigués fora de la mesh, és a dir que el valor absolut fos més gran que 1.

```
if (abs(sample_position.x)>1 || abs(sample_position.y)>1 || abs(sample_position.z)>1){
    break;
}

if (finalColor.a == 1.0){
    break;
}
```

Restriccions per aturar el loop

En el cas del Imgui, a la part de Entities/Volum node/ Material hem afegit un slider que controla la longitud del step vector. Per fer-ho hem hagut de declarar una altre uniform a la classe VolumeMaterial que fos el resultat del slider del step vector.

```
shader->setUniform("u_length", length);
shader->setUniform("u_brie", brightness);
shader->setUniform("u_model_inv", model_inv);
```

Variables uniformes afegides al shader

Volume Sampling

Per les textures hem hagut de passar el sample position a coordenades de textura ja que estava en coordenades locals, que ha sigut una operació simple de sumar 1 i dividir el vector entre dos, d'aquesta manera els límits passen a ser de [-1,-1,-1],[1,1,1] a [0,0,0],[1,1,1].

Pol Salvia 217698
Tomàs Sanchez 216810

Classification and Composition

Finalment, per conseguir el valor del color, hem seguit els pasos de les slides y hem creat una variable uniforme que es diu u_brie que correspon al valor de un slider creat per multiplicar el valor del color final i modular la brillantor del color del volum.

A més hem afegit també una condició que descarta tots aquells punts que tenen una alpha al color final inferior a 0.05 perquè d'aquesta manera treiem brutícia de la imatge que pràcticament no te densitat.

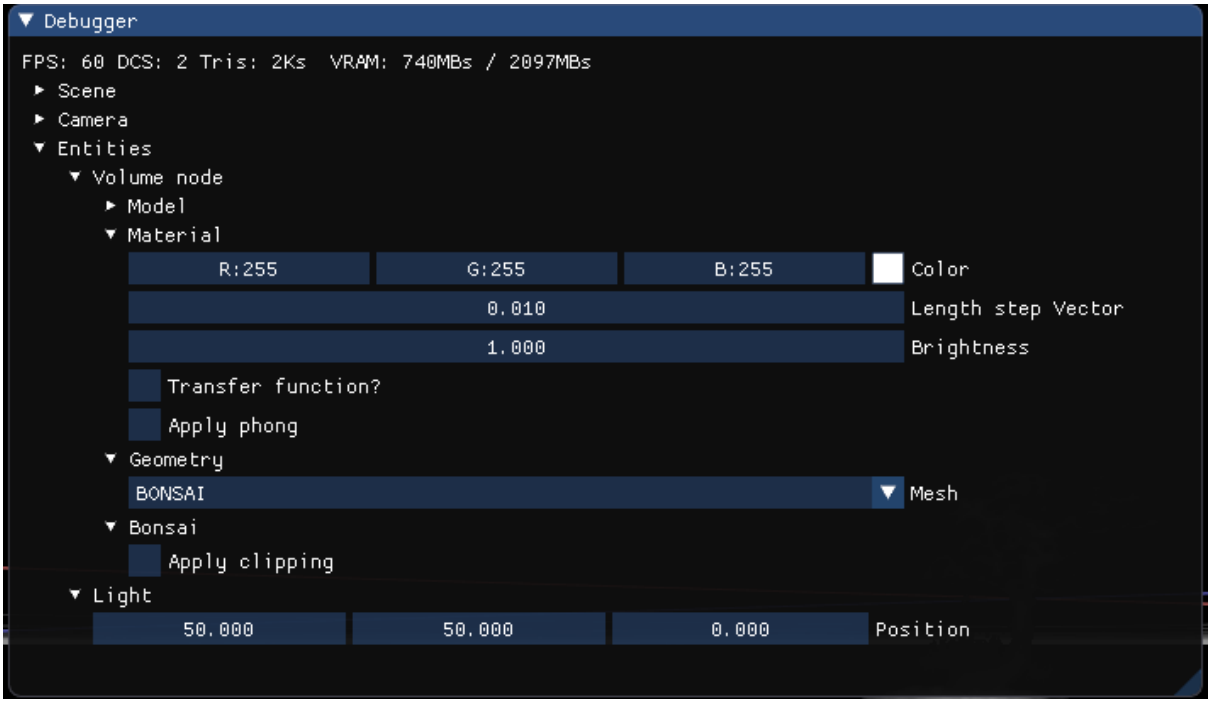
```
if (finalColor.a<0.05){
    discard;
}

gl_FragColor = finalColor*u_brie;
```

Restricció per alfes més petites que 0.05 i control de la brillantor per u_brie

Pol Salvia 217698
Tomàs Sanchez 216810

ImGUI

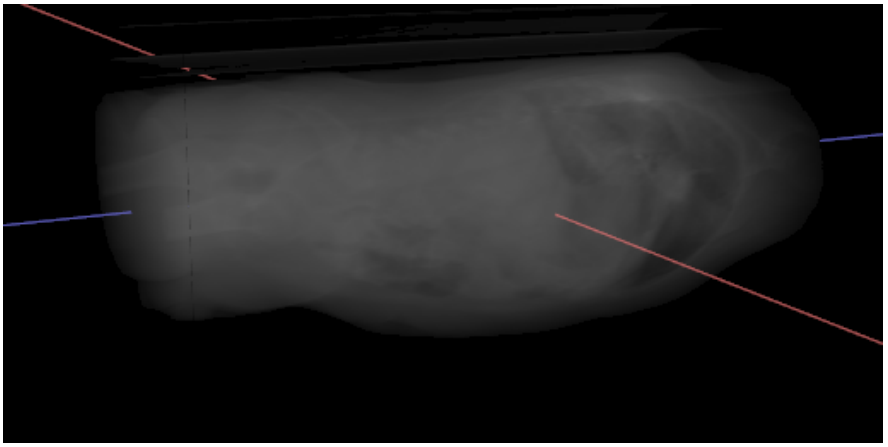
En ImGUI definim diferents paràmetres, amb els quals el usuari pot interactuar i veure en pantalla a temps real els canvis.



En l'apartat de Geometry, l'usuari pot escollir el volum entre les diferents opcions que son: Abdomen, Bonsai, Teapot i Foot.

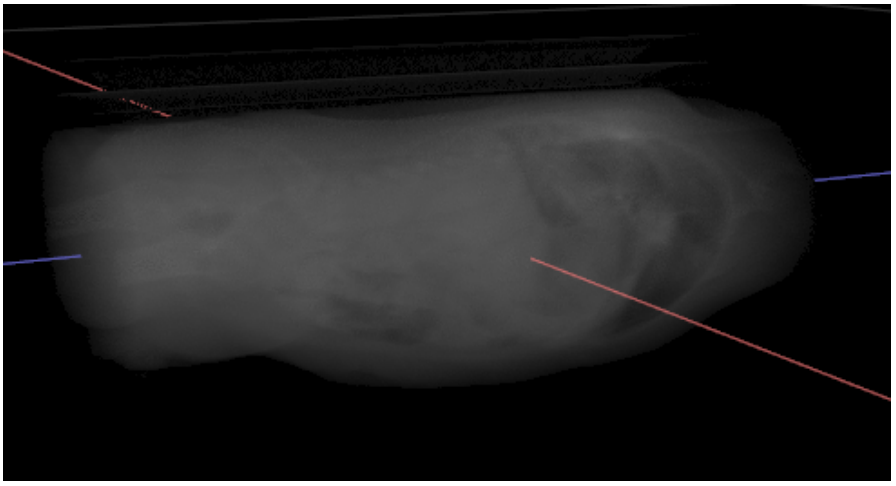
Per cada volum, tenim les següents opcions:

- Canviar Lenght step Vector, aquest paràmetre té un efecte important en com veiem el volum. En les següents imatges, podem veure un exemple.



Lenght step Vector 0.001

Pol Salvia 217698
Tomàs Sanchez 216810



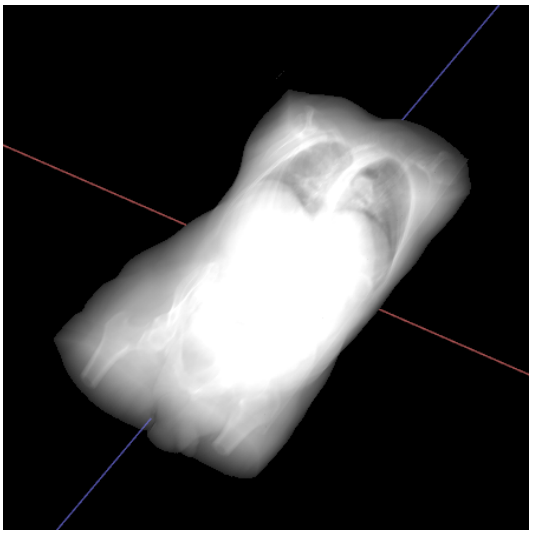
Lenght step Vector 0.05

Com podem observar, en la imatge de damunt (Lenght step Vector 0.001) veiem més detall, perquè la iteració del bucle està feta moltes més vegades i obtenim una millor resolució.

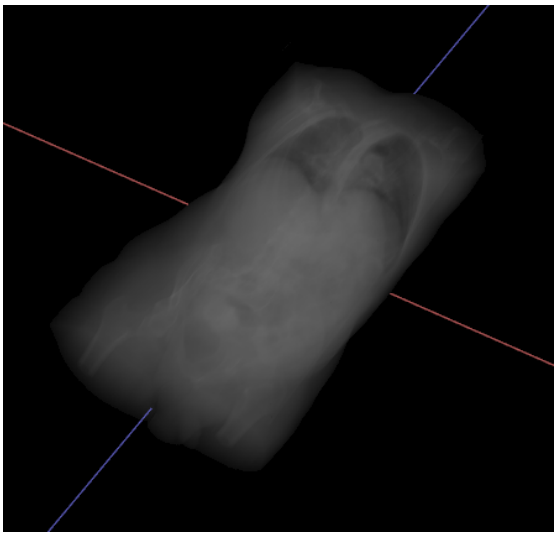
I en la imatge de davall (Lenght step Vector 0.05), que té un Lenght step Vector més alt, podem veure una resolució menor. No obstant això, com que hem aplicat Jittering, la diferència no és tan notable al no perdre molta informació entre els espais..

- Modificar la brillantor (Brightness), inicialitzem a 1, i definim que pot anar de 0.01 a 1, com podem veure a continuació.

```
ImGui::DragFloat("Brightness", (float*)&brightness, 0.01, 0.1, 10);
```



Brightnes 1.0

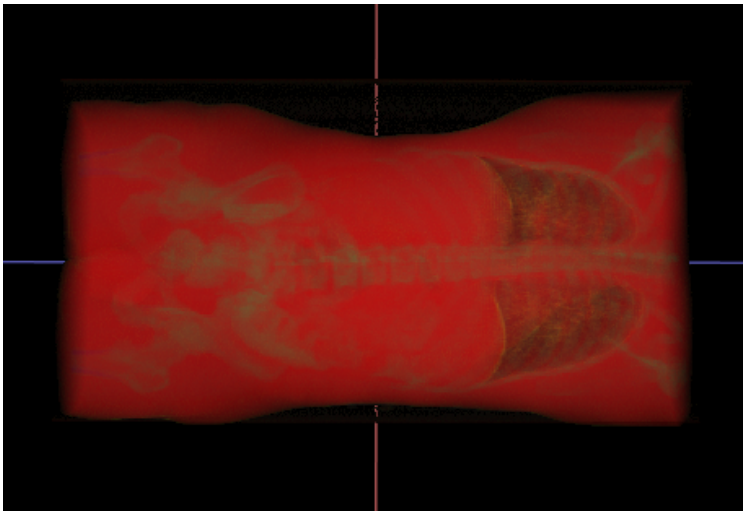


Brightnes 3.0

Pol Salvia 217698
Tomàs Sanchez 216810

- Funció de transferència (Transfer function): En aquest cas, l'usuari pot afegir l'opció de mostrar la funció de transferència del volum. Per a cada tipus de volum, hem creat una LUT específica que facilita el visualització dels diferents objectes en termes de valors de densitat.

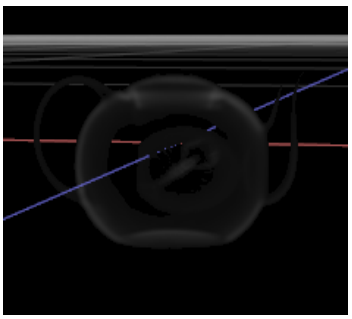
A la part de Classification creem el sample color, que associa el valor del volum a un color. I en el cas que la Transfer Function estigui seleccionada, diferents textures de LUT per cada volum s'implementen per tal d'observar els punts de densitat més alta i més baixa en la nostra mesh.



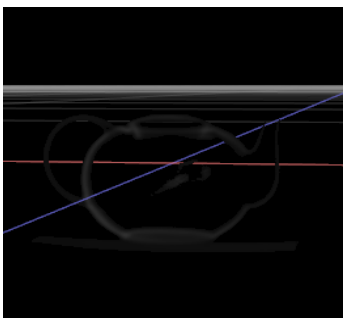
Transfer function aplicat en Abdomen

- Aplicant diferents opcions de clipping: per a cada volum diferent, podem seleccionar si volem aplicar clipping o no. En el cas del volum de Teapot , l'usuari pot triar entre amagar la taula de Teapot o tallar per la meitat Teapot per veure què hi ha dins.

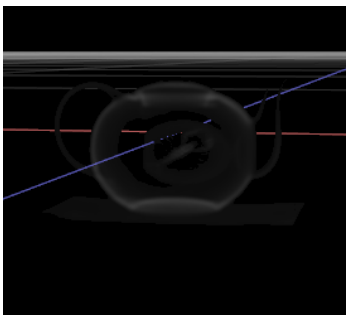
En la part de composition, volume clipping es pot calcular o bé eliminant les parts no desitjades del volum, o observant algun punt concret. Per fer-ho, implementem l'equació d'un pla per "talla" el volum i només mostrar les parts desitjades. Per tant, cada volum necessita diferents valors que son passats com a uniformes des del framework.



Complete Teapot



DeleteTable

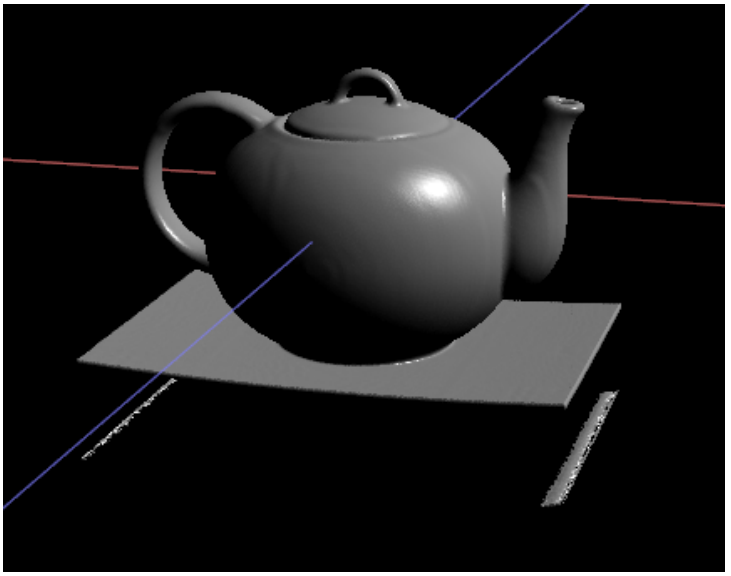


See inside

Pol Salvia 217698
Tomàs Sanchez 216810

- Visualització isosuperficies (isosurfaces): seleccionant aquesta opció, podem veure el Phong aplicat a una isosuperfície específica amb una certa densitat. Aquest llindar de densitat també pot ser modificat des de la GUI. També podem modificar el color del volum que és renderitzarà en aplicar phong.

En la part de composition, per visualitzar isosurfaces, s'aplica una funció de gradient per tal d'obtenir les diferents
normals, i després aquestes normals s'utilitzen per calcular la il·luminació phong. Això es fa amb una condició if (ja que el gradient és un càlcul pesat) que el l'usuari pot gestionar mitjançant l'ImGUI anomenada "treshold". La condició és que si el la densitat del píxel és més gran que el llindar, el color final utilitzant el Phong es computarà i es pintarà.



Phong aplicat a Teapot

- Canviar la posició de la llum (Light position): Per fer aquesta part de la pràctica hem creat un nou Standard Material anomenat Light que s'utilitza per calcular el color del volum quan s'aplica phong. En aquest cas, des de GUI també podem modificar la posició d'aquesta llum.

Gràfics per Ordinador Avançats
Lab 3 Part 1&2

Pol Salvia 217698
Tomàs Sanchez 216810

Coses a tenir en compte

Hi havia alguns aspectes importants que havíem de tenir en compte en el nostre projecte per aconseguir el resultat desitjat:

- Afegir un ‘threshhold’ per eliminar el soroll: Quan es mostra el volum, hi ha parts no desitjades que s'han d'eliminar abans de renderitzar-lo. En cas contrari, és molt difícil veure el volum correctament. En aquest cas, és possible canviar aquest llindar per adaptar-lo a cada volum.

- Un pas important que calia fer en el Framework era escalar la model matrix utilitzant les dimensions del volum i l'espai entre elles. Per tal d'evitar renderitzar un gran volum, vam reajustar les tres dimensions establint l'amplada a 1 i dividint la resta de dimensions per l'amplada del volum (fer-les proporcionals).