Volumetric Image Visualization

Alexandre Xavier Falcão

LIDS - Instituto de Computação - UNICAMP

afalcao@ic.unicamp.br

From 2D to 3D

- In the previous lecture, we introduced a rigid geometric transformation of a scene followed by projection.

From 2D to 3D

- In the previous lecture, we introduced a rigid geometric transformation of a scene followed by projection.

- This approach, however, requires an old technique, named *voxel splatting*, to guarantee that the rendition will not present unpainted voxels (“holes”).

From 2D to 3D

- In the previous lecture, we introduced a rigid geometric transformation of a scene followed by projection.
- This approach, however, requires an old technique, named *voxel splatting*, to guarantee that the rendition will not present unpainted voxels ("holes").
- In this lecture, we will introduce the *ray casting* algorithm, which assumes the inverse of the scence transformation applied to the visualization plane (i.e., image $\hat{J}$), followed by the tracing of one ray per transformed pixel towards the scene.

From 2D to 3D

- In the previous lecture, we introduced a rigid geometric transformation of a scene followed by projection.

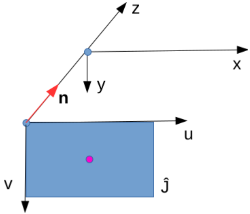
- This approach, however, requires an old technique, named *voxel splatting*, to guarantee that the rendition will not present unpainted voxels ("holes").

- In this lecture, we will introduce the *ray casting* algorithm, which assumes the inverse of the scence transformation applied to the visualization plane (i.e., image $\hat{J}$), followed by the tracing of one ray per transformed pixel towards the scene.

- One may also translate the visualization plane after rotations to obtain cuts of the scene.

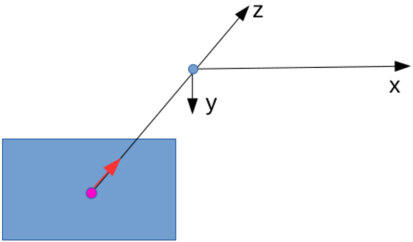
Transformation of the visualization plane

- Let the visualization plane be at $z = -\frac{d}{2}$, d being the diagonal of scene, then $(\frac{d}{2}, \frac{d}{2}, \frac{-d}{2})$ is the center of image $\hat{J}$.
- We must apply the inverse of the scene transformation — i.e., translate the center of $\hat{J}$ of $(\frac{-d}{2}, \frac{-d}{2}, \frac{-d}{2})$, apply the inverse of the rotations of the scene, and then translate it of $c = (x_c, y_c, z_c)$, being c the center of the scene.



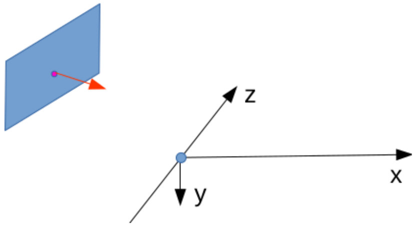
Transformation of the visualization plane

- Let the visualization plane be at $z = -\frac{d}{2}$, d being the diagonal of scene, then $(\frac{d}{2}, \frac{d}{2}, \frac{-d}{2})$ is the center of image $\hat{J}$.
- We must apply the inverse of the scene transformation — i.e., translate the center of $\hat{J}$ of $(\frac{-d}{2}, \frac{-d}{2}, \frac{-d}{2})$, apply the inverse of the rotations of the scene, and then translate it of $c = (x_c, y_c, z_c)$, being c the center of the scene.



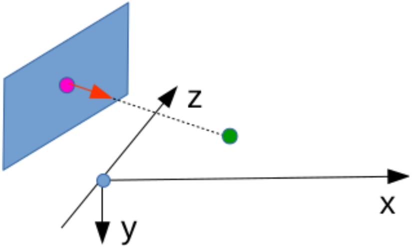
Transformation of the visualization plane

- Let the visualization plane be at $z = -\frac{d}{2}$, d being the diagonal of scene, then $(\frac{d}{2}, \frac{d}{2}, \frac{-d}{2})$ is the center of image $\hat{J}$.
- We must apply the inverse of the scene transformation — i.e., translate the center of $\hat{J}$ of $(\frac{-d}{2}, \frac{-d}{2}, \frac{-d}{2})$, apply the inverse of the rotations of the scene, and then translate it of $c = (x_c, y_c, z_c)$, being c the center of the scene.



Transformation of the visualization plane

- Let the visualization plane be at $z = -\frac{d}{2}$, d being the diagonal of scene, then $(\frac{d}{2}, \frac{d}{2}, \frac{-d}{2})$ is the center of image $\hat{J}$.
- We must apply the inverse of the scene transformation — i.e., translate the center of $\hat{J}$ of $(\frac{-d}{2}, \frac{-d}{2}, \frac{-d}{2})$, apply the inverse of the rotations of the scene, and then translate it of $c = (x_c, y_c, z_c)$, being c the center of the scene.



Transformation of the visualization plane

If ϕ is the scene transformation, then its inverse ϕ^{-1} is the visualization-plane transformation

$$\begin{bmatrix} x_2 \\ y_2 \\ z_2 \\ 1 \end{bmatrix} = \mathbf{T}(x_c, y_c, z_c)\mathbf{R}_x(-\alpha)\mathbf{R}_y(-\beta)\mathbf{T}(\frac{-d}{2}, \frac{-d}{2}, \frac{-d}{2}) \begin{bmatrix} x_1 \\ y_1 \\ z_1 \\ 1 \end{bmatrix}$$

which must be applied to each pixel $p \in D_J$, before casting a ray from the transformed p towards the scene. For $p_2 = (x_2, y_2, z_2, 1)$ and $p_1 = (x_1, y_1, z_1, 1)$, $(x_1, y_1) \in D_J$, we may rewrite the above equation as

$$p_2 = \phi^{-1}(p_1).$$

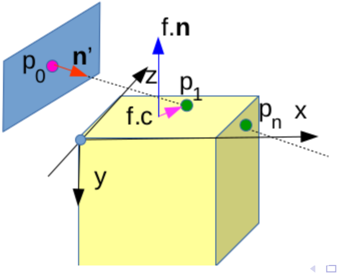
Transformation of the visualization plane

- Note, however, that (u_p, v_p) are the coordinates of p in the image domain D_J. In the homogeneous $(x, y, z, 1)$ coordinates system, a pixel p has coordinates $(u_p, v_p, \frac{-d}{2}, 1)$.

Transformation of the visualization plane

- Note, however, that (u_p, v_p) are the coordinates of p in the image domain D_J. In the homogeneous $(x, y, z, 1)$ coordinates system, a pixel p has coordinates $(u_p, v_p, \frac{-d}{2}, 1)$.

- Similarly, $\phi_r^{-1}(\mathbf{n})$ must be $\mathbf{R}_x(-\alpha)\mathbf{R}_y(-\beta)$, which maps $\mathbf{n} = (0, 0, 1, 0)$ into a new vector $\mathbf{n}'$, such that $p' = p_0 + \lambda\mathbf{n}'$, $p_0 = \phi^{-1}(p)$ and $\lambda > 0$, is the equation of a ray that starts from p_0 towards the scene by following the direction and orientation of $\mathbf{n}'$.



Casting a ray towards the scene

- When casting a ray $p' = p_0 + \lambda \mathbf{n}'$ towards the scene, we wish to find the first point p_1 and the last point p_n at which the ray intersects the scene.

Casting a ray towards the scene

- When casting a ray $p' = p_0 + \lambda \mathbf{n}'$ towards the scene, we wish to find the first point p_1 and the last point p_n at which the ray intersects the scene.

- The ray casting algorithm is then reduced to the DDA algorithm in 3D from p_1 to p_n.

Casting a ray towards the scene

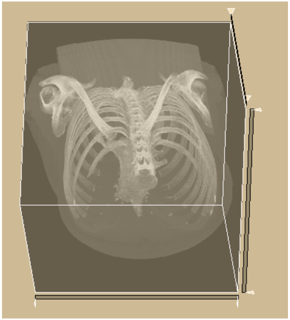
- When casting a ray $p' = p_0 + \lambda \mathbf{n}'$ towards the scene, we wish to find the first point p_1 and the last point p_n at which the ray intersects the scene.

- The ray casting algorithm is then reduced to the DDA algorithm in 3D from p_1 to p_n.

- It can be used in a variety of applications, such as the detection of the intersection between the ray and an object's surface and the extraction of scene's attributes along the ray.

Casting a ray towards the scene

A popular example is the maximum intensity projection (MIP) — a rendering technique that assigns to each $p \in D_J$ the maximum intensity along the ray $p' = p_0 + \lambda \mathbf{n}'$ inside the scene region.



That is, $J(p) = \max_{k=1,2,\ldots,n}\{I(p_k)\}$, where $p_1, p_2, \ldots, p_n$ are obtained by the DDA algorithm in 3D.

Casting a ray towards the scene

- For each $p' = p_0 + \lambda \mathbf{n}'$, the intersection points p_1 and p_n belong to two face planes of the scene.

Casting a ray towards the scene

- For each $p' = p_0 + \lambda \mathbf{n}'$, the intersection points p_1 and p_n belong to two face planes of the scene.
- Let then $f.\mathbf{n}$ and $f.c$ be the unit normal vector and center of a face $f \in \mathcal{F}$ (the set of faces).

Casting a ray towards the scene

- For each $p' = p_0 + \lambda \mathbf{n}'$, the intersection points p_1 and p_n belong to two face planes of the scene.
- Let then $f.\mathbf{n}$ and $f.c$ be the unit normal vector and center of a face $f \in \mathcal{F}$ (the set of faces).
- In the intersection, one should expect that

$$\langle p' - f.c, f.\mathbf{n} \rangle \;=\; 0.$$

Casting a ray towards the scene

- For each $p' = p_0 + \lambda \mathbf{n}'$, the intersection points p_1 and p_n belong to two face planes of the scene.
- Let then $f.\mathbf{n}$ and $f.c$ be the unit normal vector and center of a face $f \in \mathcal{F}$ (the set of faces).
- In the intersection, one should expect that

$$\langle p' - f.c, f.\mathbf{n} \rangle = 0.$$

- The solution $p' \in \{p_1, p_n\}$ comes from the lowest and highest finite values of λ, such that $p' = (\lceil x_{p'} \rceil, \lceil y_{p'} \rceil, \lceil z_{p'} \rceil) \in D_I$.

Casting a ray towards the scene

- For each $p' = p_0 + \lambda \mathbf{n}'$, the intersection points p_1 and p_n belong to two face planes of the scene.
- Let then $f.\mathbf{n}$ and $f.c$ be the unit normal vector and center of a face $f \in \mathcal{F}$ (the set of faces).
- In the intersection, one should expect that

$$\langle p' - f.c, f.\mathbf{n} \rangle = 0.$$

- The solution $p' \in \{p_1, p_n\}$ comes from the lowest and highest finite values of λ, such that $p' = (\lceil x_{p'} \rceil, \lceil y_{p'} \rceil, \lceil z_{p'} \rceil) \in D_I$.
- Let now DDA3D($\hat{I}, \mathcal{P}$), $\mathcal{P} = \{p_1, p_n\}$, be the function that returns $\mathcal{P} = \{p_1, p_2, \ldots, p_n\}$ — a set of points visited by the DDA algorithm in 3D, using $sign(X) \in \{-1, 1\}$.

The DDA algorithm in 3D

Input : Scene $\hat{I} = (D_I, I)$ and set $\mathcal{P} = \{p_1, p_n\}$ of points inside the scene region.
Output: Set $\mathcal{P} = \{p_1, p_2, \ldots, p_n\}$ with the visited points.

If $p_1 = p_n$ then set $n \leftarrow 1$.
Else
  Set $D_x \leftarrow x_{p_n} - x_{p_1}$, $D_y \leftarrow y_{p_n} - y_{p_1}$, $D_z \leftarrow z_{p_n} - z_{p_1}$.
  If $|D_x| \geq |D_y|$ and $|D_x| \geq |D_z|$ then
    Set $n \leftarrow |D_x| + 1$, $d_x \leftarrow sign(D_x)$, $d_y \leftarrow \frac{d_x D_y}{D_x}$, and $d_z \leftarrow \frac{d_x D_z}{D_x}$.
  Else
    If $|D_y| \geq |D_x|$ and $|D_y| \geq |D_z|$ then

The DDA algorithm in 3D

8 Set $n \leftarrow |D_y| + 1$, $d_y \leftarrow sign(D_y)$, $d_x \leftarrow \frac{d_y D_x}{D_y}$, and $d_z \leftarrow \frac{d_y D_z}{D_y}$.

9 Else

10 Set $n \leftarrow |D_z| + 1$, $d_z \leftarrow sign(D_z)$, $d_x \leftarrow \frac{d_z D_x}{D_z}$, and $d_y \leftarrow \frac{d_z D_y}{D_z}$.

11 Set $(x_{p'}, y_{p'}, z_{p'}) \leftarrow (x_{p_1}, y_{p_1}, z_{p_1})$.

12 For each $k = 2$ to $n - 1$, do

13 Set $(x_{p'}, y_{p'}, z_{p'}) \leftarrow (x_{p'}, y_{p'}, z_{p'}) + (d_x, d_y, d_z)$ and $\mathcal{P} \leftarrow \mathcal{P} \cup \{(x_{p'}, y_{p'}, z_{p'})\}$.

The MIP algorithm

Input : Scene $\hat{I} = (D_I, I)$ and angles α and β.
Output: MIP image $\hat{J} = (D_J, J)$.

1 $\mathbf{n}' \leftarrow \phi_r^{-1}(\mathbf{n})$, where $\mathbf{n} = (0, 0, 1, 0)$.
2 For each $p \in D_J$ do
3 $\quad p_0 \leftarrow \phi^{-1}(p)$.
4 $\quad$ Find $\mathcal{P} = \{p_1, p_n\}$ by solving $\langle p_0 + \lambda \mathbf{n}' - f.c, f.\mathbf{n} \rangle = 0$ for each face $f \in \mathcal{F}$ of the scene, whenever they exist.
5 $\quad$ if $\mathcal{P} \neq \emptyset$ then
6 $\quad\quad \mathcal{P} \leftarrow \mathrm{DDA3D}(\hat{I}, \mathcal{P})$
7 $\quad\quad J(p) \leftarrow argmax_{p'=(x_{p'}, y_{p'}, z_{p'}) \in \mathcal{P}} \{I(p')\}$

where the intensities $I(p')$ for $p' \in \mathcal{P}$ are found by interpolation.