Assignment 3

Vertex Normals

This exercise was pretty straight-forward. As indicated in the PDF file of the assignement, we loop through each triangle, and multiply each vertex normal by the weights (computed with `angleWeights`). Then, we loop again through each Vertex to normalize the normals.

Ray-Triangle intersection + normal calculations

To compute the intersection between our ray and each triangle efficiently, we will use Cramer's rule. For that, we need to arrange the triangle equation so that we get the form $\mathbf{A}\vec{\mathbf{x}} = \vec{\mathbf{b}}$, where **A** is a *3 x 3* matrix, **x** and **b** are vectors of dimension *3*. In other words, we want to find an equation of the form :

$$x_1 \cdot \vec{\mathbf{a_1}} + x_2 \cdot \vec{\mathbf{a_2}} + x_3 \cdot \vec{\mathbf{a_3}} = \vec{\mathbf{b}}$$

where $\vec{\mathbf{a_1}}, \vec{\mathbf{a_2}}, \vec{\mathbf{a_3}}$ are respectively the first, second and third columns of $\mathbf{A}$ and x_1, x_2, x_3 the components of the vector **x**.

We then need the 3 vertices of the triangle, $\vec{\mathbf{A}}, \vec{\mathbf{B}}, \vec{\mathbf{C}}$, the origin of the ray $\vec{\mathbf{o}}$ and its direction $\vec{\mathbf{d}}$:

$$\begin{gathered}
\vec{\mathbf{o}} + t \cdot \vec{\mathbf{d}} = \alpha \cdot \vec{\mathbf{A}} + \beta \cdot \vec{\mathbf{B}} + \gamma \cdot \vec{\mathbf{C}} \\
\vec{\mathbf{o}} + t \cdot \vec{\mathbf{d}} = (1 - \beta - \gamma) \cdot \vec{\mathbf{A}} + \beta \cdot \vec{\mathbf{B}} + \gamma \cdot \vec{\mathbf{C}} \\
\vec{\mathbf{o}} + t \cdot \vec{\mathbf{d}} = \vec{\mathbf{A}} - \beta \cdot \vec{\mathbf{A}} - \gamma \cdot \vec{\mathbf{A}} + \beta \cdot \vec{\mathbf{B}} + \gamma \cdot \vec{\mathbf{C}} \\
\vec{\mathbf{o}} + t \cdot \vec{\mathbf{d}} - \vec{\mathbf{A}} + \beta \cdot \vec{\mathbf{A}} + \gamma \cdot \vec{\mathbf{A}} - \beta \cdot \vec{\mathbf{B}} - \gamma \cdot \vec{\mathbf{C}} = 0 \\
\vec{\mathbf{o}} + t \cdot \vec{\mathbf{d}} + \beta \cdot \vec{\mathbf{A}} + \gamma \cdot \vec{\mathbf{A}} - \beta \cdot \vec{\mathbf{B}} - \gamma \cdot \vec{\mathbf{C}} = \vec{\mathbf{A}} \\
\Longrightarrow \\
\begin{pmatrix}
t \cdot \mathbf{d}_1 + \beta \cdot (\mathbf{A}_1 - \mathbf{B}_1) + \gamma \cdot (\mathbf{A}_1 - \mathbf{C}_1) = \mathbf{A}_1 - \mathbf{o}_1 \\
t \cdot \mathbf{d}_2 + \beta \cdot (\mathbf{A}_2 - \mathbf{B}_2) + \gamma \cdot (\mathbf{A}_2 - \mathbf{C}_2) = \mathbf{A}_2 - \mathbf{o}_2 \\
t \cdot \mathbf{d}_3 + \beta \cdot (\mathbf{A}_3 - \mathbf{B}_3) + \gamma \cdot (\mathbf{A}_3 - \mathbf{C}_3) = \mathbf{A}_3 - \mathbf{o}_3
\end{pmatrix} \\
\Longrightarrow \\
\begin{pmatrix}
\mathbf{d}_1 & \mathbf{A}_1 - \mathbf{B}_1 & \mathbf{A}_1 - \mathbf{C}_1 \\
\mathbf{d}_2 & \mathbf{A}_2 - \mathbf{B}_2 & \mathbf{A}_2 - \mathbf{C}_2 \\
\mathbf{d}_3 & \mathbf{A}_3 - \mathbf{B}_3 & \mathbf{A}_3 - \mathbf{C}_3
\end{pmatrix}
\begin{pmatrix} t \\ \beta \\ \gamma \end{pmatrix}
=
\begin{pmatrix} \mathbf{A}_1 - \mathbf{o}_1 \\ \mathbf{A}_2 - \mathbf{o}_2 \\ \mathbf{A}_3 - \mathbf{o}_3 \end{pmatrix}
\end{gathered}$$

Then we implemented Cramer's algorithm to solve this equation following this Wikipedia page.

To compute the solution using Cramer's algorithm we modularized our code with three utility functions that:

1. Computes the determinant of a matrix
2. Solves one equation
3. Solves the whole system

Bounding box intersection

The code for this part was inspired by the algorithm found on this website. We mainly changed some variable names for clarity but the algorithm is the same.

Because we are looking to solve $bb_{min} \leq o + t \cdot \vec{d} \leq bb_{max}$ we manually solve this for x, y and z in the bounding case (i.e. $bb_{min} = o + t \cdot \vec{d}$ and $bb_{max} = o + t \cdot \vec{d}$). That means we check for each component that we are intersecting with the box, swapping the variables when needed to ensure the minimum is always less than the maximum.

This algorithm seems to work fine, as you can see on the degub_aabb images in the folder.

Workload

- Lucien Michaël Iseli, 274999: **33.33%**
- Lucas Strauss, 272432: **33.33%**
- Joachim Dunant, 262314: **33.33%**