

Assignment 4 - Camera

This assignment practices topics related to camera, as one of the most important components in computer graphics. For your in-depth understanding of this area, it also covers a few related concepts that might new to you. You should use any resources (e.g., books, search engines, calculators, and etc.) that can help you accomplish it.

There are code skeletons for programming tasks, please check our GitHub repository¹.

Task 1: View Transformation

Suppose we are using a right-handed coordinate system. As shown in Figure 1, a camera at position $\mathbf{p} = (p_1, p_2, p_3, 1)^\top$, looking in the direction $\mathbf{l} = (l_1, l_2, l_3, 0)^\top$, with an up-vector $\mathbf{u} = (u_1, u_2, u_3, 0)^\top$ is given.

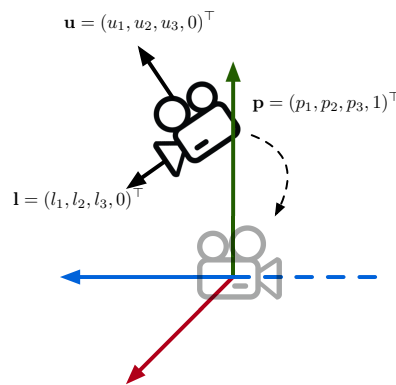


Figure 1: Camera view transformation

- Show the translation matrix T_{view} that transforms the position of the camera to the origin.
- After the translation, show the rotation matrix R_{view} that transforms the camera look-at to the direction $(0, 0, -1, 0)^\top$ and up-vector to the direction $(0, 1, 0, 0)^\top$.

Include your answers in a Markdown file called “task01.md”.

Task 2: Projection Transformation

Let’s continue along with Task 1. After the view transformation, the camera is now at the origin, the look-at is $\mathbf{l} = (0, 0, -1, 0)^\top$ and the up-vector is $\mathbf{u} = (0, 1, 0, 0)^\top$. Assume, a point $\mathbf{x} = (x, y, z, 1)^\top$ is inside the view frustum of the given camera with near plane $z = n, n < 0$ and far plane at $z = f, f < n$.

- What are the projected coordinates for \mathbf{x} on the x - y plane in orthographic projection?

¹<https://github.com/mimuc/cg1-ss20>

- b) Assume, the orthographic view frustum is $[l, r] \times [b, t] \times [f, n]$ where l, r are left and right planes, and b, t are bottom and top planes. Deduce the projection matrix T_{ortho} that transforms the given view frustum $[l, r] \times [b, t] \times [f, n]$ to a cube $[-1, 1]^3$.

Now, let's switch from orthographic to a perspective camera. The perspective projection can be seen as a combination transformation of two transformations: a) an orthographic transformation, and b) perspective-to-orthographic transformation. As illustrated in Figure 2, the view frustum of a perspective camera can be squashed as a cube-like orthographic view frustum.

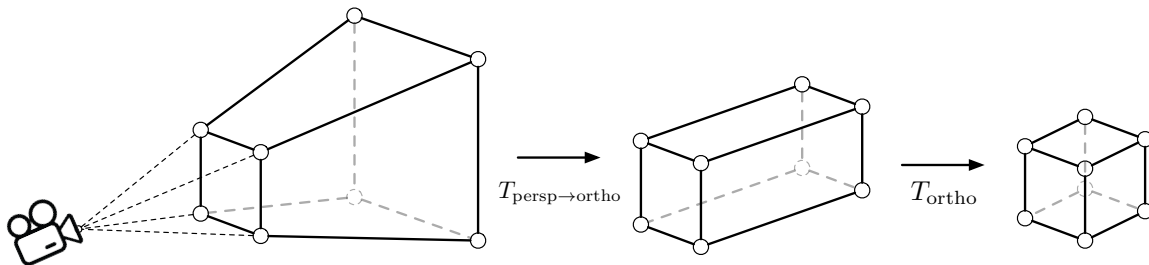


Figure 2: Perspective view frustum to orthographic view frustum

- c) Let the field of view be θ and the aspect ratio λ . Deduce the transformation matrix $T_{persp \rightarrow ortho}$ that transforms the given perspective view frustum to a orthographic view frustum.
- d) Use T_{ortho} you derived from b) and calculate the combined transformation $T_{ortho}T_{persp \rightarrow ortho}$.

Include your answers in a Markdown file called "task02.md".

Task 3: Viewport Transformation

Let's continue along with Task 2. After the projection transformation, your view frustum is transformed to a cube $[-1, 1]^3$. Show the transformation matrix $T_{viewport}$ that projects all points from the cube to $x-y$ plane in a rectangle $[0, w] \times [0, h]$ where $w, h > 0$, as shown in Figure 3.

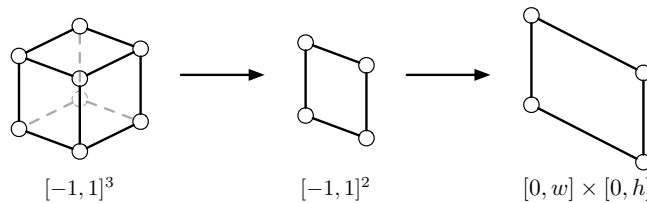


Figure 3: Project cube view frustum to 2D plane

Include your answers in a Markdown file called "task03.md".

Task 4: Hitchcock Zoom

In photography, *Hitchcock zoom*, also known as *dolly zoom*², is an in-camera effect that appears to undermine normal visual perception. This technique was invented by Alfred Hitchcock while filming *Vertigo*³. There is a video for you to learn more about how Hitchcock zoom applied in industrial filming⁴. This task is going to walk you through the implementation of the Hitchcock zoom effect in `three.js`.

From the code skeleton, your initial scene is a bunny in the Sponza⁵, as shown in Figure 4a. As always, there is a demo⁶ for your to interact with it.



(a) The initial scene from code skeleton.



(b) A possible projection when applying Hitchcock zoom.



(c) Another possible projection when applying Hitchcock zoom.

Figure 4: Hitchcock zoom: the background the bunny suddenly resizes and overwhelms the foreground.

- Read the links (Wikipedia and YouTube) provided in the previous description and try to understand how the Hitchcock zoom works. What are the two camera-related factors of the Hitchcock zoom?
- In the code skeleton, you can find three files: `main.js`, `renderer.js`, and `dolly.js`. Look for `// TODO:` and implement the Hitchcock zoom in `dolly.js`.
- What do you need for updating the camera projection in `three.js`?
- Where is the center of the perspective projection in `three.js`?

Answer text questions in the `README.md` of the given code skeleton, then include your answers and implementation in a folder called “task04”. **Exclude** the installed dependencies (folder `node_modules`) in your submission.

²https://en.wikipedia.org/wiki/Dolly_zoom

³[https://en.wikipedia.org/wiki/Vertigo_\(film\)](https://en.wikipedia.org/wiki/Vertigo_(film))

⁴<https://www.youtube.com/watch?v=u5JBlwlnJX0>

⁵<https://github.com/KhronosGroup/glTF-Sample-Models/tree/master/2.0/Sponza>

⁶<http://www.medien.ifi.lmu.de/lehre/ss20/cg1/demo/4-proj/hitchcock/>

Submission

- Participation in the exercises and submission of the weekly exercise sheets is voluntary and not a prerequisite for participation in the exam. However, participation in an exercise is a good preparation for the exam (the content is the content of the lecture and the exercise).
- For non-coding tasks, write your answers in a Markdown file. Markdown is a simple mark-up language that can be learned within a minute. A recommended the Markdown GUI parser is typora (<https://typora.io/>), it supports parsing embedded formula in a Markdown file. You can find the syntax reference in its Help menu.
- **Please submit your solution as a ZIP file** via Uni2Work (<https://uni2work.ifi.lmu.de/>) before the deadline. We do not accept group submissions.
- Your solution will be corrected before the discussion. Comment your code properly, organize the code well, and make sure your submission is clear because this helps us to provide the best possible feedback.
- If we discover cheating behavior or any kind of fraud in solving the assignments, you will be withdrawn for the entire course! If that happens, you can only rejoin the course next year.
- If you have any questions, please discuss them with your fellow students first. If the problem cannot be resolved, please contact your tutorial tutor or discuss it in our Slack channel.