Assignment 2: Transformations

BK7084 Computational Simulations (2020/2021)

1 Introduction

In this assignment, we will dive deeper into transformations. It's important that you get how to apply transformations to objects in 3D scenes and how to compose transformations, as you will use these operations a lot in the final assignment of this course. In this assignment, we slowly build up complexity: first, you will create transformation matrices from scratch to control a virtual car. Then you will compose transformation matrices to construct a lamp. Finally, you can apply these lessons to build an animated solar system. Follow the comments in *ex01.py*, *ex02.py*, and *ex03.py* to finish the assignments. You can use this document as a reference. There are questions in this document. Don't skip over them, but think about them. If you can't answer them, feel free to ask one of the TAs.

2 Translation matrices

You have learned about translation matrices and how they can be applied in order to move points in space. Below you can see how a point is moved by multiplying with a translation matrix

$$\begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} v_x \\ v_y \\ v_z \\ 1 \end{bmatrix} = \begin{bmatrix} v_x + t_x \\ v_y + t_y \\ v_z + t_z \\ 1 \end{bmatrix}. \tag{1}$$

⟶ Why are we using four dimensions in the matrix and coordinates of the point?
⟶ Can you compute the result yourself?
⟶ What would happen to the result if the last entry in the matrix was a 4?

In the *translate* function you see that we create a matrix *mat* which will become your translation matrix. You must fill in the rows and columns of this matrix appropriately to form such a matrix. It starts off as the identity matrix

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}. \tag{2}$$

3 Rotation matrices

We use a rotation matrix to rotate points. In 3D, a point can be rotated around three possible axes (the x, y and z axes). Therefore, we need three rotation matrices to support all these rotations.

$$R_x(\theta) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta \\ 0 & \sin\theta & \cos\theta \end{bmatrix} \tag{3}$$

$$R_y(\theta) = \begin{bmatrix} \cos\theta & 0 & \sin\theta \\ 0 & 1 & 0 \\ -\sin\theta & 0 & \cos\theta \end{bmatrix} \tag{4}$$

$$R_z(\theta) = \begin{bmatrix} \cos\theta & -\sin\theta & 0 \\ \sin\theta & \cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{5}$$

4 Composing transformations

Let's say you want to rotate a point and then translate it. You have created the rotation matrix **R** and translation matrix **T**. The point you want to transform is called **p**. You could first compute the rotation

$$\mathbf{p}' = \mathbf{R}\mathbf{p} \tag{6}$$

and then the translation

$$\mathbf{p}'' = \mathbf{T}\mathbf{p}'. \tag{7}$$

Another way to write this is

$$\mathbf{p}' = \mathbf{T}\mathbf{R}\mathbf{p}. \tag{8}$$

You could also first compute the product of the matrices **T** and **R** and then multiply the result with **p**.

$$\mathbf{M} = \mathbf{T}\mathbf{R} \tag{9}$$

$$\mathbf{p}' = \mathbf{M}\mathbf{p}. \tag{10}$$

This is possible, because matrix products are *associative*, a fancy word to say that $(\mathbf{AB})\mathrm{C} = \mathbf{A}(\mathbf{BC})$. You know this property from regular old numbers (we call them *scalars*). For example, you know that $(3 \times 4) \times 5 = 3 \times (4 \times 5)$. This is probably so obvious to you that you never thought about it, but we'll see that not all properties of scalars hold for matrices. The associativity property is nice, because it means we can compose all our transformations into *one* 4x4 matrix, which is then multiplied with all the points in our scene.

We already mentioned that matrix multiplications don't share all properties of scalars. One such property is *commutativity*. Commutativity means that you can swap the order of operations and the result will be the same: $a \times b = b \times a$. Try it out with numbers: $3 \times 4 = 4 \times 3$.

⟶ Why is this true? Hint: Can you think of a picture for multiplication?

Matrix multiplications are not commutative. That means you cannot change the order of matrices and expect the same result. Try it out yourself:

$$\begin{bmatrix} 1 & 3 \\ 4 & 2 \end{bmatrix} \begin{bmatrix} 20 & 10 \\ 40 & 60 \end{bmatrix} = ? \tag{11}$$

$$\begin{bmatrix} 20 & 10 \\ 40 & 60 \end{bmatrix} \begin{bmatrix} 1 & 3 \\ 4 & 2 \end{bmatrix} = ? \tag{12}$$

⟶ What is the difference between first rotating and then translating and first translating and then rotating?