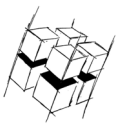

**Prof. Dr.-Ing. Ralf Steinmetz**
Multimedia communications Lab

Dipl. Inf. Robert Konrad
Polona Caserman, M.Sc.

TECHNISCHE UNIVERSITÄT DARMSTADT

**„Game Technology“**
**Winter Semester 2017/2018**

**Solution 3**

## General Information

- The exercises may be solved by teams of up to three people.
- The solutions have to be uploaded to the Git repositories assigned to the individual teams.
- **The submission date (for practical and theoretical tasks) is noted on top of each exercise sheet.**
- If you have questions about the exercises write a mail to game-technology@kom.tu-darmstadt.de or use the forum at https://www.fachschaft.informatik.tu-darmstadt.de/forum/viewforum.php?f=557

# 1. **Practical Tasks: Triangle Mesh Rendering (5 Points)**

Implement a basic software triangle renderer. Specifically, implement keyboard camera controls by carrying out rotation, translation and projection as described in lecture 3. You may use the Kore::vec3/4 classes to store data, but not the Kore matrix or quaternion classes. As an ungraded bonus, you can try out mouse camera control.

You will find comments in the code that direct you to the correct positions to add your code.

The suggested order is to start with translation first, then the perspective transform, and finally rotations. Note the effect each transformation has on the result. In case the model is not visible anymore after a change you made, check your calculations with some example data, e.g. the point (0, 0, 0), and find out at which position on the screen it ends up in.

https://github.com/TUDGameTechnology/Exercise3.git contains code for triangle rasterizing and mesh loading and a mesh to get you started. You can either copy the code changes manually or just pull them into your own repository using
git pull https://github.com/TUDGameTechnology/Exercise3.git

**Please remember to push into a branch called "exercise3".**

You can find the source code for the solution at https://github.com/TUDGameTechnology/Solution3.

# 2. Theoretical Tasks (5 Points)

## 2.1 Geometric interpretations (2 Points)

a) Write down the definition of the cross product. What for can we use cross product in games? Give a small example.

*Cross product gives us the direction the normal is facing. In a game, we can use cross product when we do Backface Culling. If the vector points in the direction of the camera, we render the triangle. Additionally, we can use cross product to calculate the angle between two vectors.*

b) Write down the definition of the dot product. What for can we use dot product in games? Give a small example.

*When using dot product, we project one vector onto the other. In a game, we can use dot product to look, if both vector point in the same direction.*

## 2.2 Angle calculations (1 Point)

Rotate a Triangle with points $p_1 = (1,1)$, $p_2 = (2,3)$ and $p_3 = (3,0)$ by $\alpha = \frac{\pi}{3}$. Show your calculations.

*First, we calculate sinus and cosinus values:*

$$\cos\left(\frac{\pi}{3}\right) = \frac{1}{2}, \sin\left(\frac{\pi}{3}\right) = \frac{\sqrt{3}}{2}$$

*For each point, we calculate:*

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x\cos(\alpha) - y\sin(\alpha) \\ x\sin(\alpha) + y\cos(\alpha) \end{pmatrix}$$

*We become:*

$$p_1^* = \begin{pmatrix} -0.366 \\ 1.366 \end{pmatrix}$$
$$p_2^* = \begin{pmatrix} -1.598 \\ 3.232 \end{pmatrix}$$
$$p_3^* = \begin{pmatrix} 1.5 \\ 2.598 \end{pmatrix}$$

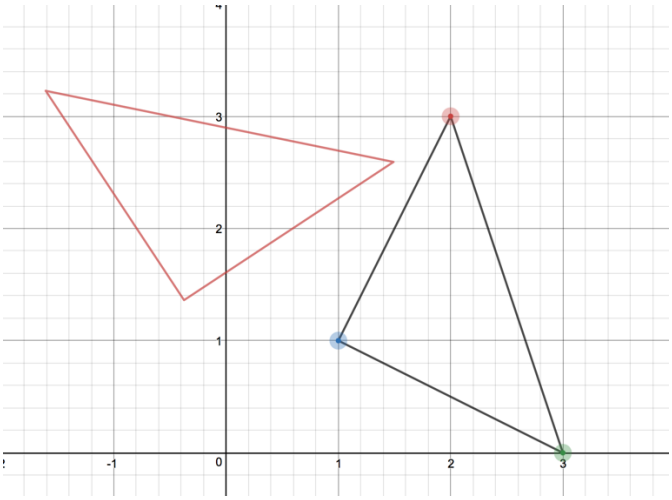


*Figure 1: Rotation about a point.*

## 2.3 Raytracing and Rasterization (2 Point)

Explain in your own words the differences between raytracing and rasterization.

*Raytracing*

- *We send out a ray for each pixel of the image*
- *We color the pixel by following the ray and calculating the color at which it intersects the world*

*Rasterization*

- *We transform from 3D to 2D*
- *We draw primitives (triangles) in 2D*
- *We fill the 2D image by drawing each point where the triangle is visible*