Team: Mykhailo-Taras Sobko, Anna-Alina Bondarets, Oleksandra Stasiuk

- Describe the research area of your project and why it is interesting/important (e.g., the project is on face recognition ...)

Our project is about 3D graphics rendering, 3D objects representation and their geometric transformations. The linear algebra related to these tasks is an integral part of the game developing and physical simulations.

- Describe the aim of the project and tasks you want to achieve (e.g. develop an automated face recognition system using LA techniques)

We want to apply linear algebra knowledge to develop a 3D engine as a Python library, though the main aim is not to program as many features as we can, but rather use in practice the mathematical understanding and linear algebra concepts to render and transform 3D objects.

The task combines many sub-topics, such as: linear and affine transformations, orientation representation, viewing and projection, programmable shading and lighting, ray tracing, asterization, and interpolation. We will not implement all the features, yet choose the most fundamental concepts and ideas of linear algebra with the ability for the user to extend the details and topics.

- Describe possible approaches and available solutions (literature review) and explain which method you'll use and why

The literature we will use:

1. https://canvas.projekti.info/ebooks/Mathematics%20for%203D%20Game%20Programming%20and%20Computer%20Graphics,%20Third%20Edition.pdf

   It is a great book describing every aspect of game development in detail with respect to its mathematical requirements, providing C++ source code implementation for some features using OpenGL.

2. A tutorial-blog http://blog.wolfire.com/2009/07/linear-algebra-for-game-developers-part-1/ (as well as the next parts)

   It summarizes the main and the most essential ideas should be implemented with a nice visualization and sometimes pseudocode, yet we will explore for some details each of the listed topic.

3. Tutorials on PyOpenGL:

   https://dev.to/suvink/graphic-designing-using-opengl-and-python-beginners-1i4f https://pythonprogramming.net/opengl-rotating-cube-example-pyopengl-tutorial/

Official documentation:
http://pyopengl.sourceforge.net/documentation/manual-3.0/index.html

- Describe the pipeline of the implementation (if any)

Our plan on implementing the features:

1. Rendering of simple shapes (just interconnected vertices);
2. Make it possible to rotate and move relatively to the object;
3. Correctly display the object's faces;
4. Simple texturing;
5. Shadows and lightning (extra)

- How will you test your implementation? Do you have the necessary data?

Assuming that 3D engines are not required to output the exact bit-to-bit image, it is hard to automate the testing. Still, we will use them for simple sanity checks on a simple geometry.

We will implement rendering objects using *.obj* files (or similar) which contain geometric information about the object. Since there are a lot of sample *.obj* files on the internet, we have plenty of files to do the testing on.

- Briefly outline the plan of future research (tasks to perform and time schedule)

The next step for us is to apply in practice the linear algebra concepts we already have learnt on the course, so we will mainly focus on the PyOpenGL documentation and implementation. As we will reach stage 3 from our pipeline plan, we will research the information specialized on the texturing of the objects. If time permits, we will also try to learn how to implement the shading and lightning.

- Are there any challenges that might hamper your work?

From a technical point of view, we will have to spend some time learning technical things, not related to linear algebra, like using PyOpenGL (or similar tools if something goes wrong with this one). Then, there may be difficulties with calculating object's faces and other things, as some edge cases may introduce artifacts, not reproducible in other ways. Again, from the technical perspective, viewing large (in the sense of the number of polygons) objects may have low performance as we use Python for simplicity.

🐝🐝🐝bzzz bzzz I'm a bee bzzz 🐝🐝🐝