Assignment 02 Theory Part

# Part 01: Mouse Input

## Question 01:

Explain what is wrong with each of them in how you’d expect them to behave (does it always rotate the way you’d expect? Does it always rotate at a sensible rate?)

## Answer:

The trackball code has some issues with it, it doesn’t rotates in the direction of mouse, it rotates opposite to the mouse direction. It is expected to rotate in the direction of mouse cursor.

The trackballQuanterion code is performing well as compare to the trackball, it rotates the object in the direction of the mouse cursor. This program has some additional functions in vertex shader like quanterion multiplier and quanterion inverse.

## Question 02:

Explain how you have tweaked the samples to make this feel more natural.

## Answer:

I have changed the shape, as it was required in the assignment, I make a square-based pyramid and controlling it by the mouse, I can rotate it and translate it with the mouse movement as you can see in the output.

# Part 04: Basic Utility

## Question 01:

Explanation of how/why you did things the way you do.

## Answer:

I just made simple changes in my lab 06 code files, I just add CTM matrix and pass the parameters, I tried so many times to get some output, I have checked my code but I couldn’t find the error but getting output was a problem.

# Part 05: Theory

## Question 01:

Why would it be advantageous to keep the transformation matrix on the GPU rather than copying it back and forth per object? (After all, the matrices are small)? What would be the disadvantages of that? If you’re not sure, think about how you would want to implement GPU-accelerated physics and AI.

## Answer:

There are many advantages when it comes to keeping the transformation matrix on the GPU rather than copying it back such as; the rendering process of the program becoming more efficient and easy when the transformation matrix is in the GPU. As the transformation matrix stores all of the information which is being used to transform the shapes, so the vertex shader can easily access the information from the GPU and doesn’t need to copy it from the CPU. It results in fast processing and efficient transformation.

The object’s physical and AI information is also stored in the transformation matrix, so while rendering the program gets all the position and rotation information of the object gets from the transformation matrix, so by the presence of matrix in the GPU, this process could also become faster.

The only disadvantage of keeping the transformation matrix in the GPU is just the storage problem, nowadays there are GPUs with a lot of storage to store multiple things while processing but, saving the transformation matrix in the GPU occupies memory. When the transformation matrix is in the GPU then the debugging would be challenging and difficult.