Project Documentation:

This project allows the visualisation of two graphic's effects using openFrameworks.

The 3D mesh manipulation code consists of two classes: **myEffectVBO & lavaBall**.

"myEffectVBO" is placed in the middle of the scene, dynamically moving across the bordered area within the planes. This effect utilises the Vertex Buffer Object in order to process the pixels and turn them into a vertex data on the GPU. You create multiple copies of the same object but this would drastically effect the performance of your computer.

If you would like to change the video file, you need to chose a different file in your ("bin/data" folder) and update <#std::string name#>. The update method should not be changed, this is where we calculate the position of the vertex based on the video's pixel RGB brightness. However in the draw method of this class, there is an option to position **myEffectVBO** with X,Y,Z and glPointSize with the float value: pixelSize. You can scatter the pixels vertically by using the Z value.

"lavaBall" is a class where I chose to use a sphere primitive for my additional effect. I sourced the mesh from the primitive itself: mesh = icoSphere.getMesh(); which simplifies the manipulation of the vertices. There are two floats in lavaBall::update: amplitude & time. These can be set up when we call the class in ofApp to control the overall animation of the sphere. Positioning is simply done with X,Y,Z coordinates.

"myShader" primarily uses the fragment shader to distort the image of the storm image as it's rotating around the centre on the planes. The vertex shader of **"myShader"** is set to the default parameters as the geometry isn't changed in any way.