# **Задача 2**

## **Морфинг геометрии**

### **Цель задания:**

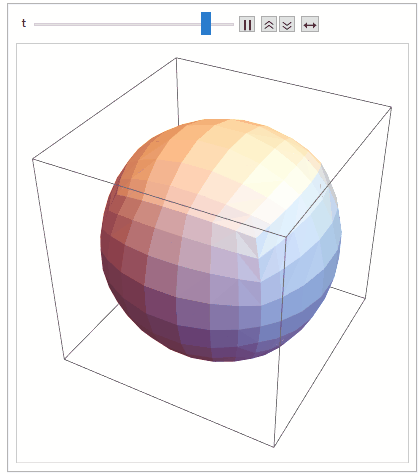
* более детально **освоить преобразования вершин** (vertex shader);
* научиться **«real-time анимировать объекты» от времени**, используя юниформы (uniforms).

### **Задача в общих словах**

**Используя вершинный шейдер**, необходимо реализовать один из вариантов **морфинга геометрии**:

* куба в сферу;
* закручивание цилиндра вокруг своей оси;
* произвольной модели в сферу/куб;
* любые другие ваши идеи :)

Визуальный пример морфинга куба:



### **Приблизительный алгоритм выполнения задания**

Опишем алгоритм, например, решения задачи **морфинга куба в сферу**.

* Взять приложения из **задачи 1**. Для него реализовать возможность создания геометрии куба (модели) из **«достаточно большого» количества вершин** (чтобы отдельные грани при трансформации вершин оставались выпуклыми).
* Поддержать возможность передачи **uniform переменной** в вершинный шейдер с **текущим временем или с некоторым параметром, варьирующимся от 0 до 1** (нужно будет для дальнейшего морфирования).
* **Нагуглить формулы для морфирования**, поддержать их в вершинном шейдере (учесть параметр выше). *Замечание: способов морфирования может быть несколько, выберите наиболее удобный вам и легко параметризующийся*.
* В основном коде **изменять параметр морфирования**, например варьировать значение параметра от 0 до 1.
* Нарисовать **«каркас модели»** — точки и полилинии**, поддержать его морфинг и посмотреть на результат.**

### **Дополнительно можно**

* **Данный вариант морфинга можно считать «устаревшим», т. к. не используется «параллелизм», обеспеченный вам графическим ускорителем и API**. Поэтому предлагается использовать дополнительные шейдеры — **тесселяционный и геометрический**, чтобы реализовать **морфинг на “упрощенной” исходной геометрии**.
* Получить **«визуально» похожую картинку** с морфингом, сделанным с помощью вершинного шейдера, **сравнить количество вершин в изначальной геометрии** для двух вариантов, а также **попробовать оценить производительность**.

### **Полезные ссылки**

* <https://www.khronos.org/registry/OpenGL/specs/gl/GLSLangSpec.3.30.pdf> — GLSL 3.30 specification.
* <https://www.khronos.org/opengl/wiki/GLSL_Object> — Program objects (shaders objects).
* <https://www.khronos.org/opengl/wiki/Uniform_(GLSL)> — GLSL uniforms.
* [http://www.opengl-tutorial.org](http://www.opengl-tutorial.org/ru/) — OpenGL tutorials.
* <http://mathproofs.blogspot.com/2005/07/mapping-cube-to-sphere.html> — Morphing cube to sphere.
* <http://mathproofs.blogspot.com/2005/07/mapping-square-to-circle.html> — Morphing square to circle.
* <https://mathematica.stackexchange.com/questions/174847/transform-sphere-into-a-cube> — Some formulas and explanations.