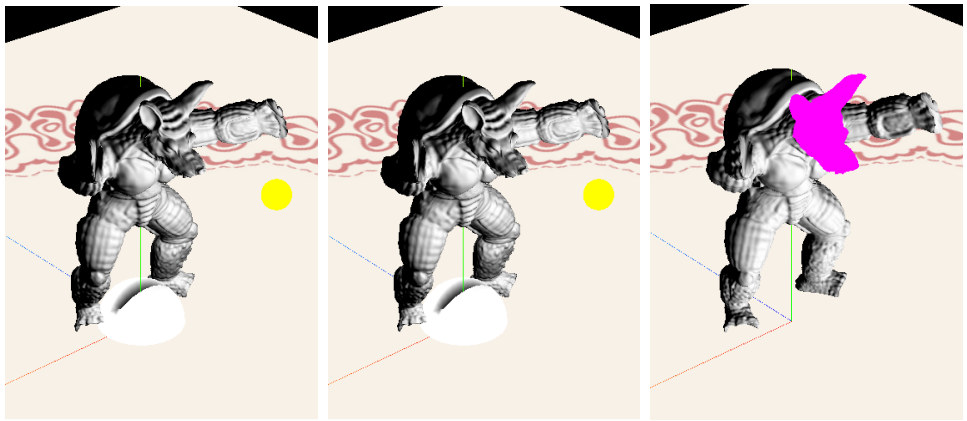
CPSC 314
Assignment 2: The Armadillo Awakens.
Transformations

1 Introduction

The main goal of this assignment is to create an understanding of rotation, translation, and scaling transformations. This will be done by giving the armadillo eyes, make them shoot lasers at the lightbulb and finally nod in satisfaction.



Initial template setup. Press '1' '2' '3' to change scene.

The template code provides a starting base for your work. The code is structured as three separate scenes, that you can access with the respective variables: `scenes[Part.EYES]` `scenes[Part.LASERS]` `scenes[Part.DEFORM]`.
Read **carefully** the questions and go through the source code before doing any work to familiarize yourselves with it, comments are provided to help guide you.

1.1 Template

- The file `A2.html` is the launcher of the assignment. Open it in your preferred browser to run the assigment, to get started.
- The file `A2.js` contains all the Javascript code for the assignments, you will add your objects and setup shaders here.

- The folder `glsl` contains the vertex and fragment shaders for all the objects in the scene. You will have to add your own shaders and this is where you will do most of your coding.
- The folder `js` contains the required JavaScript libraries. You do not need to change anything here.
- The folder `obj` contains the geometric models loaded in the scene.
- The folder `images` contains the texture images used.

1.2 Execution

The assignment can be run by opening the file `A2.html` in any modern browser. However, most browsers will prevent pages from accessing local files on your computer. If you simply open `A2.html`, you may get a black screen and an error message on the console similar to this:

```
XMLHttpRequest cannot load armadillo.vs.glsl. Cross origin
  requests are only supported for protocol schemes: http,
  data, https.
```

Please see this web page for options on how to run things locally:

`https://threejs.org/docs/#manual/introduction/How-to-run-thing-locally`

2 Work to be done (100 pts)

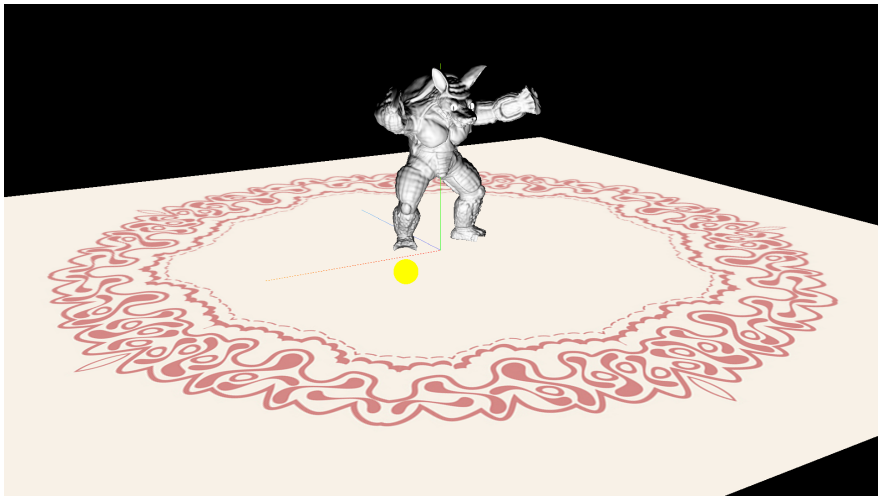
Study the template to get a sense of how it works before diving in.

1. (70 points)

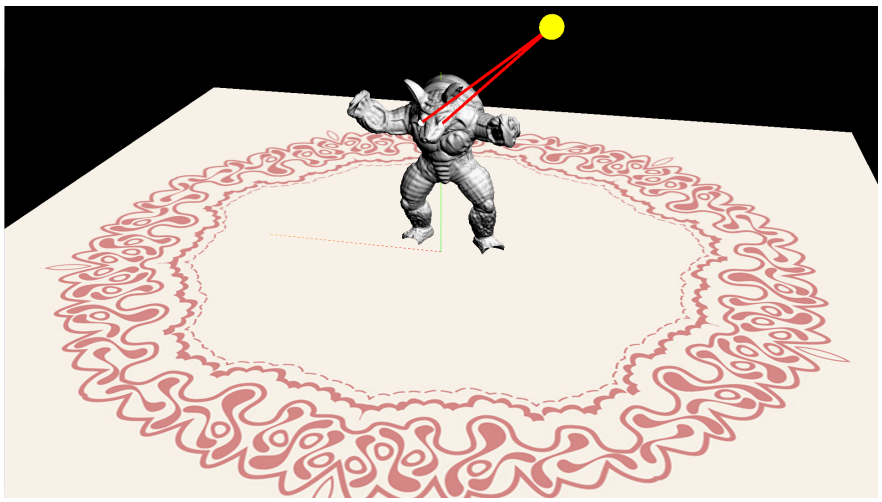
   (a) **Eyes** (15 pts) For this question you have to make the armadillo's eyes track the lightbulb as it moves. In the template you will find the lightbulb and the armadillo eyes loaded for you as well as the lightbulb controls. It's your responsibility to scale both eyes to an appropriate size, rotate them to point at the lightbulb and make sure they continue to do so as the lightbulb moves. Both eyes share the same shader files (`eye.vs.glsl` and `eye.fs.glsl`) but different materials: `leftEyeMaterial` `rightEyeMaterial`.

   Important: **do not** use Three.JS functions, must modify the shaders for credit.

   (b) **Lasers** (25 pts) Create the required shaders and materials to display the laser beams centered in an eye, shooting towards the lightbulb. Do this for **both eyes**. You have to transform the lasers in the same way as you did the eyes so that they are located and oriented properly. You will also have to apply the appropriate scaling in order for the lasers to always point to the center of the lightbulb as it moves.



Eyes



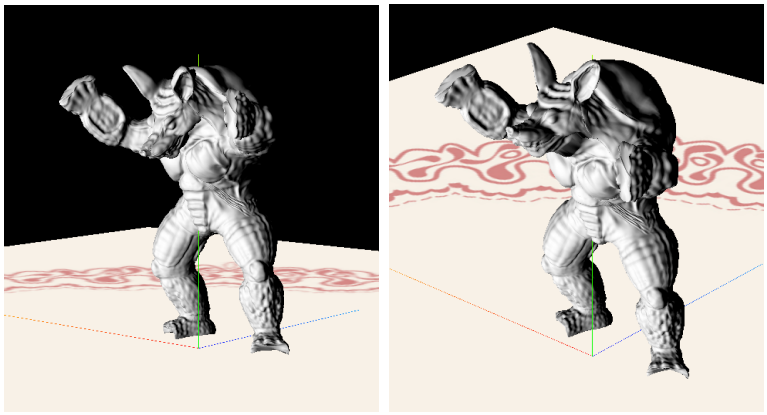
Lasers

(c) **Deform** (30 pts) Now you have to make the armadillo nod by simply rotating the head around the x-axis. You will have two coordinate frames, the Neck and Head frame that initially will be perfectly aligned. The Neck frame is fixed to the body, while the Head is fixed to the armadillo's head and moves with it. As the armadillo doesn't exactly have a neck, an approximation such as (0.0, 1.0, -0.3) for the origin will suffice, the axis are aligned to the world frame.

You first have to determine if a specific vertex is part of the Head or the Neck, one way is by finding an axis-aligned box for the head, thus find a range for each dimension where the vertex have to lie in. The `nodding_armadillo.vs.glsl` shader already gets you started on the task.

The angle at which the Head frame is rotated relative to the Neck should be controllable by the keyboard. Make sure to add a new variable and correctly pass it to the shaders.

Dont panic if some significant artifacts pop up around the boundary between the head and neck for larger rotations.



Deform

2. **Part 2:** (30 pts) Creative License

   For this part we want to see what you can do. Your ideas should use at least one new shader, and should be of a similar complexity to the previous tasks. If you have any doubts, make sure to OK it with a prof or TA. Some possible suggestions might be:

   - Add rotations to arms or other body parts.
   - Linear blend skinning (also known as Skeletal Subspace Deformation) is a simple way to reduce, but not eliminate, the artifacts when parts of a mesh are deformed. This is a common technique used to animate characters in video games and other real time applications. The idea is to define a vertexs position as a linear combination of its position before head rotation and its position after rotation, that you computed in the previous part. To make this work, the key is to pick the weights so that the blending is limited to a small region around the neck, and change continuously from one end of the region to the other.
   - Animate the awakening!

   Bonus marks may be given at the discretion of the marker for particularly noteworthy explorations.

2.1 Hand-in Instructions

You do not have to hand in any printed code. Create a README.txt file that includes your name, student number, and login ID, and any information you would like to pass on to the marker. Create a folder called "a2" under your "cs314" directory. Within this directory have two subdirectories named "part1," and "part2", and put all the source files and your README.txt file for each part in the respective folder. Do not use further sub-directories. The assignment should be handed in with the exact command:

```
handin cs314 a2
```