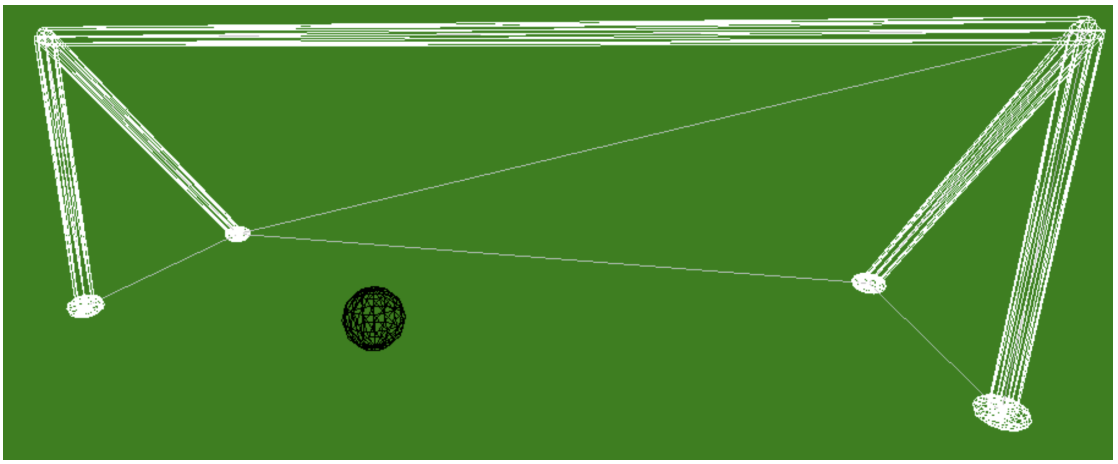
Reichman University: Computer Graphics Course, Spring ‘23
Instructor: Prof. Ariel Shamir
Submission due: XXth of May 2023 (23:59)

Exercise 5 – Modeling with WebGL



Overview

In this exercise you will practice basic WebGL techniques in modeling, viewing and projecting. Your goal (😉) is to create an application that allows a user to view an WebGL rendered model. You will have to create a model of a football goal, and a ball that moves near it.

The model you create here will serve you for the next exercise as well.

Reichman University: Computer Graphics Course, Spring ‘23
Instructor: Prof. Ariel Shamir
Submission due: XXth of May 2023 (23:59)

Usage

- Pressing ‘w’ toggles wireframe view.
- Pressing ‘o’ toggles the Orbit Controls. (already implemented)
- Pressing ‘-’ / ‘+’ modifies the speed of the ball. (partially implemented)
- Pressing ‘1’ / ’2’ will toggle the trajectory of the ball.
- Pressing ‘3’ will shrink the goal while preserving its proportions.

Modeling

There are multiple ways to model a football goal. Here we require a minimum level of details. You may however embellish the model with as many additional details you see fit, or alternatively, you could build a 3D goal model that is at least as impressive as our model **as long as they follow the mode specifications**.

Building Blocks

The minimum set of primitives you need to use for modeling are:

- Cylinders for the two front goalposts, the two back supports, and the crossbar.
- Rings or toruses at the edges of the goalposts and back supports.
- Double-sided mesh planes for two side nets and the back net.
- A sphere for a ball.

Our model is basically built from these 3D primitives, along with the necessary affine transformations.

Structure

Your goal model must be modular and constructed as a group of primitives. Following is a list of the parts in our model. These are only a suggestion, but you do have to use the building blocks mentioned above.

- Goal:
  - Skeleton: The main components of the goal. Must be white. Has these sub-components:
    - Crossbar: Long horizontal cylinder.
    - Goal posts: Two identical vertical cylinders next to each other. Each goal post also has a torus or a ring at its edge.
    - Back supports: same as goal posts, but notice that the cylinders are placed diagonally, yet the toruses\rings are perpendicular to the ground just like the goalposts’ toruses\rings.
  - Nets: a rectangular in the back of the goal, and two triangles at the sides of the goal. The nets must be light gray.
- Ball: A black sphere.

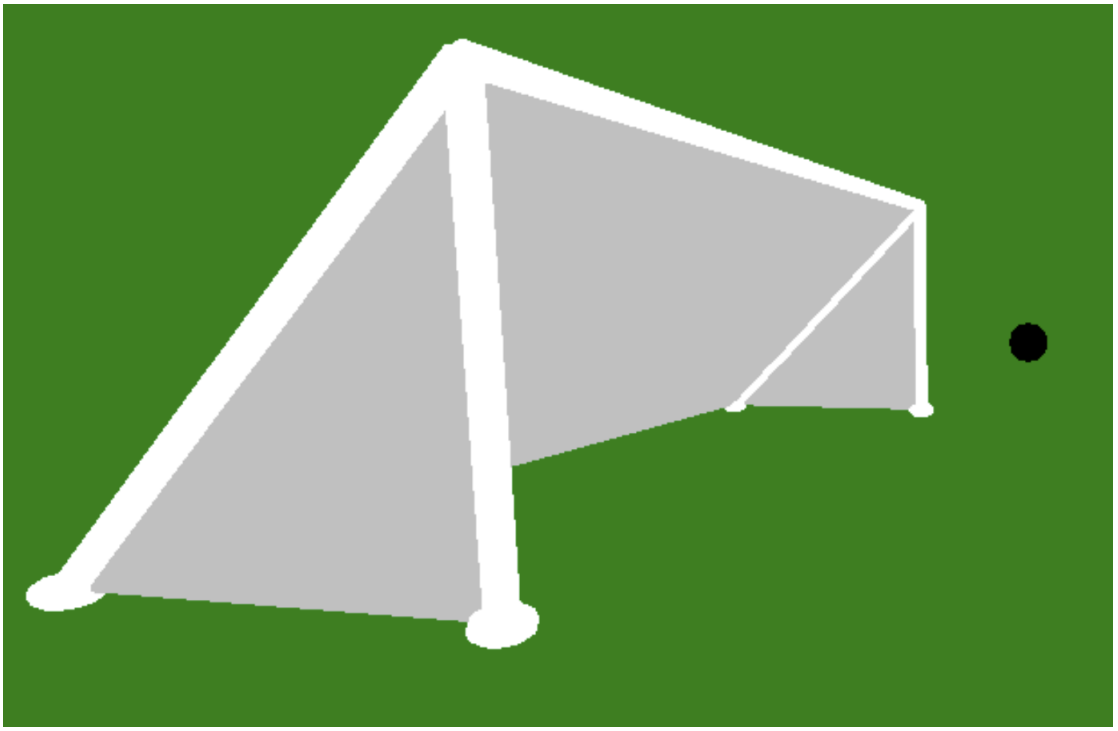
Reichman University: Computer Graphics Course, Spring '23
Instructor: Prof. Ariel Shamir
Submission due: XXth of May 2023 (23:59)

Model Specifications and Unit-measure

- **Ball:**
  - Ball must be a sphere object
  - Ball to goal vertical scale ratio: 1:8 (eight balls on top each other are of the same height as the goal)
  - Vertically, the goal, in its entirety (remember that it has a radius), must be somewhere between the top and bottom of the goal
  - Depth-wise, the ball, in its entirety, must be far from the goal's entry by some distance of your choice
- **Goal:**
  - **Skeleton:**
    - The width\height ratio of the net must be 3:1.
    - The angle between the goalposts and the back supports must be between 30° and 45°.
  - **Nets**:
    - The three nets must be wrapped between the skeleton parts.

If we did not define any limitation or specification, you are free to do as you please! We want you to have some creative freedom.

Reichman University: Computer Graphics Course, Spring '23
Instructor: Prof. Ariel Shamir
Submission due: XXth of May 2023 (23:59)



Positioning and rotating the objects

As we've learnt in class and you might also see online, there are a lot of ways to manipulate a specific object pose. However, we require you to manipulate the pose **only by applying matrices onto objects and multiplying matrices.**

This means you **cannot** use commands like *cube.position.x = 6* etc.

However, you may use the already implemented basic Matrix4 commands, such as makeRotation, makeScale and everything that appears in the official documentation.

Viewing

The user should be able to rotate the model using the mouse (specifically, dragging the mouse while clicking it). The viewing module involves projecting the positions of the mouse before and after the click on a sphere, and then computing the rotation needed to transform between the two. This transformation is then performed on the model. This module is already implemented, you may read the implementation details here.

Reichman University: Computer Graphics Course, Spring '23
Instructor: Prof. Ariel Shamir
Submission due: XXth of May 2023 (23:59)

Surface Color

In this exercise it is recommended to use **MeshPhongMaterial** for your objects. In the next exercise we will add light sources and add on the definitions of these materials, so it is important you will have the framework prepared for it.
Note: For working with Phong material, as you already know, you need a light source. If you wish to use the **MeshPhongMaterial**, you are required to add any kind of lighting (your choice). If you rather change it in HW6, you may use the **MeshBasicMaterial** for now.

Wireframe

As you saw in the recitation, we can change how the material works using the wireframe attribute set to **true** or **false**. You should implement a way to turn on all the wireframe materials on and off at the same time.
Hint - you can change any attribute during runtime using *material.attribute = value*.

Interactivity

You are required for 2 types of animations, each named after the keyboard key that toggles them on and off.
'1' - The ball will do a rotational movement in a single axis of your choice, and by that, it will enter the goal. Remember to consider the speedFactor variable.

'2' - The ball will enter the goal by doing a rotational movement in a **different** axis of your choice, not the same one from animation '1'. Remember to consider the speedFactor variable.

'Up arrow' - If animated, the speed of the ball increases (partially implemented).

'Down arrow' - If animated, The speed of the ball increases (partially implemented).

You are also required to implement this:

'3' - The goal, in its entirety, will shrink by 5% in each of its axes. Each click applies shrinking again (e.g. after two clicks, the goal's length in each dimension is 90.25% of its original length).

Note that the animations 1 and 2 are not exclusive, meaning you should be able toggle all of them at once (in that case it's okay if the ball doesn't enter the goal).

Reichman University: Computer Graphics Course, Spring '23
Instructor: Prof. Ariel Shamir
Submission due: XXth of May 2023 (23:59)

Keyboard Trigger

In order to create keyboard triggers, we can use the **EventListener** implemented in javascript. You would want to use the "keydown" event type. You can see how to implement this kind of logic in our framework for toggling the orbit controls. Note that the argument is a callback, so program accordingly!

Framework

You are given a framework that will set for you a small server in your computer for you to work on your project. We made this in order for you to be able to work with multiple javascript files at the same webpage without invoking CORS errors.

In order to work with the server, you should download node.js (the LTS version is recommended).
After installing, you need to set up your environment:

1. Use your terminal to navigate to the folder in which the environment is prepared.
2. write *npm install* to install the server framework.
3. In order to run the server, write *node index.js* in your terminal and it will start hosting a server.
4. In default, your server will be at http://localhost:8000/

Improve the appearance or add animation to the Model (up to 8 points bonus)

In addition to the basic goal and ball models, you can design additional features (such as seats with fans, a trophy, or a flag of your favorite team) or anything else you see fit. There will be up to 8 points bonuses, for creative additional features. If you want to be eligible for the bonus points, you must add a README file to your submission, describing what you did.

Grading

- Goal model - 50 points
  - Goal posts and crossbar - 15 points
  - Back supports - 20 points
  - Nets - 15 points
- Ball model - 5 points
- Wireframe mode - 10 points
- Animations - 20 points (2 x 10)
- Speed change - 10 points
- Goal size scale - 5 points
- Bonus - up to 8 points

Reichman University: Computer Graphics Course, Spring '23
Instructor: Prof. Ariel Shamir
Submission due: XXth of May 2023 (23:59)

Recommended Milestones

Design the 3D model in a bottom-up fashion. We suggest the following implementation milestones:

1. Start with adding only a single geometry and render it. Make sure that you understand how the orbit controls work . Note that we already created a box for you - you should remove it and replace it with a different geometry.
2. Implement the toggle for the wireframe, as it will help you with your designing and might be more complex to add later on when there are many materials.
3. Complete the goal's skeleton:
    a. Start with the goal posts and crossbar.
    b. Continue with the diagonal back supports.
    c. Add the toruses\rings at the ends of the goal posts, back supports, and the crossbar.
4. Add the goal nets.
5. Add the ball and place it correctly besides the goal.
6. Start working on both animation '1' and '2' together, as they are very similar. Make sure the speed is affected by the up and down arrows.
7. Add the goal scaling action '3'.

Tips

- Pay attention to the order in which you present the polygons vertices.
- Use the wireframe mode (by pressing "W") to see how different parts are modeled in the provided jar file.

Submission

- Submission is in pairs
    - Zip file should include only a single script - hw5.js
    - **A short readme document if you decide to add more elements to your model or add new animations explaining what you've implemented.**
- Zip file should be submitted to Moodle.
    - Name it:
    - <Ex##> <FirstName1> <FamilyName1> <ID1> <FirstName2> <FamilyName2> <ID2>
- Before submission be sure to check for updates on moodle

- **There will be a point reduction of up to 10 points for submitting incorrectly.**