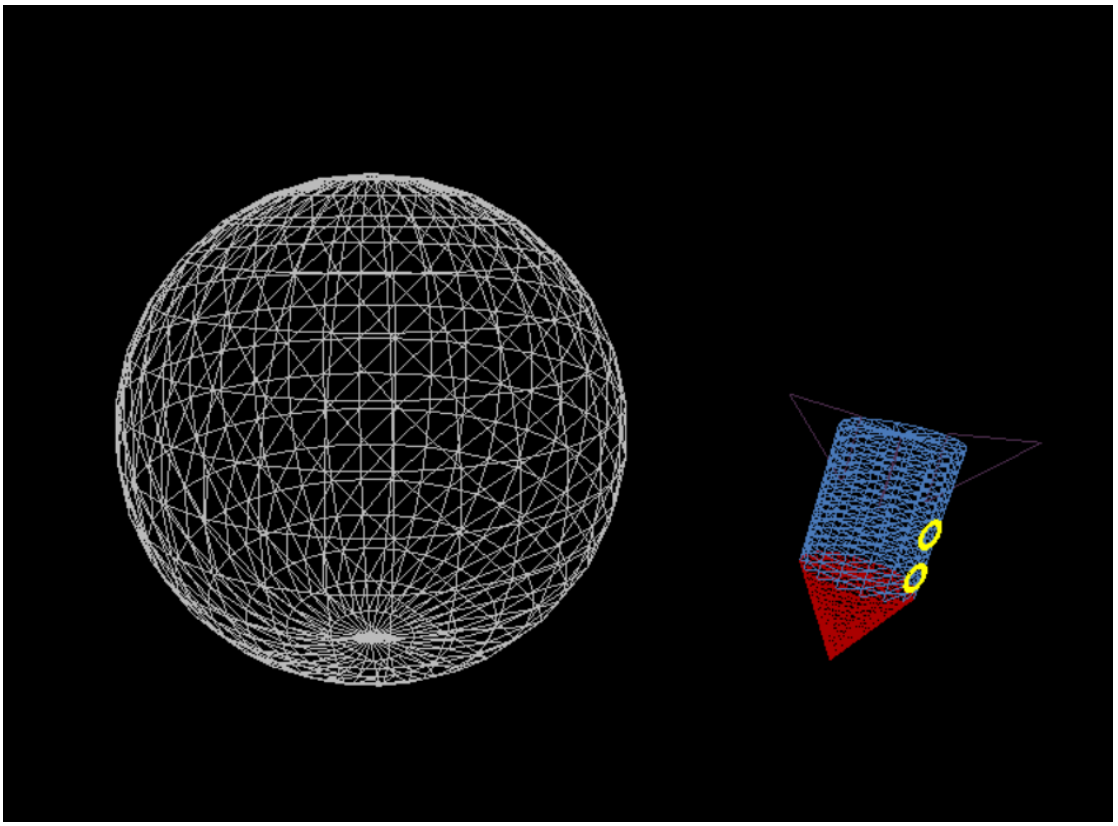
The Interdisciplinary Center: Computer Graphics Course 2022
Instructor: Prof. Ariel Shamir
Submission due: 29th of May 2022 (23:55)

# 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 at least one model – a spaceship, and a planet for it to fly around.

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

The Interdisciplinary Center: Computer Graphics Course 2022
Instructor: Prof. Ariel Shamir
Submission due: 29th of May 2022 (23:55)

# Usage

- Pressing ‘w’ toggles wireframe view.
- Pressing ‘o’ toggles the Orbit Controls. (already implemented)
- Pressing ‘1’ / ’2’ / ’3’ will toggle the trajectory of the spaceship.

# Modeling

There are endless ways to model a spaceship. 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 spaceship 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:

- Cylinder for the ship’s hull
- Cone for the ship’s head
- A two-sided mesh plane for the ship’s wing
- A ring for the ship’s windows.
- A sphere for a planet.

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

## Structure

Your model must be modular and constructed as a group of primitives under a single object. 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.

- Ship:
    - Hull: The main cylinder of the ship. Has two sub-components
        - Windows: Two identical ring-shaped windows one above the other.
        - Wings: Three triangles all positioned in identical distances around the hull bottom.
    - Head: A cone resting on top of the hull of the ship.
- Planet: A sphere in gray colors.

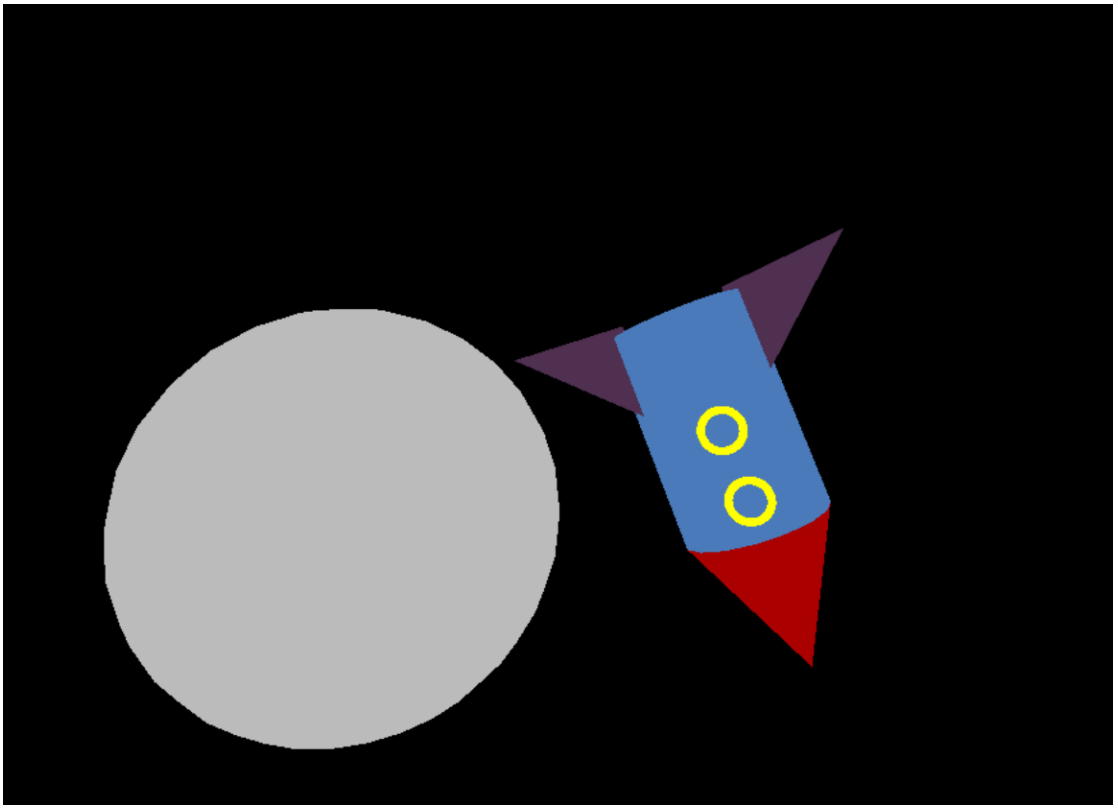
The Interdisciplinary Center: Computer Graphics Course 2022
Instructor: Prof. Ariel Shamir
Submission due: 29th of May 2022 (23:55)

## Model Specifications and Unit-measure

- **Planet:**
  - Planet must be a sphere object
  - Ship to Planet scale ratio: 1:5 (The planet is 5 times bigger than the ship)
  - The planet should **not** be placed at the center of the scene. You may decide where will it be placed, though.
- **Ship:**
  - **Hull:**
    - Hull radius is identical to the ship head radius.
    - Hull radius to hull length ratio**:** 1:3
    - **Wings:**
      - One of the wing's edges must be perpendicular and tangent to the ship's hull.
      - There must be at least 3 wings, all in identical distances from another in relation to the ship hull.
      - The wing must be rendered in any direction.
      - There is no limitation for the shape of the wing.
    - **Windows**:
      - The windows must be aligned with the ship hull length.
      - There must be at least two windows.
  - **Head**:
    - The head radius to head height ratio must not exceed 1:3.

**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.

The Interdisciplinary Center: Computer Graphics Course 2022
Instructor: Prof. Ariel Shamir
Submission due: 29th of May 2022 (23:55)



### 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 matrix application and multiplication**.

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.

The Interdisciplinary Center: Computer Graphics Course 2022
Instructor: Prof. Ariel Shamir
Submission due: 29th of May 2022 (23:55)

## Surface Color

In this exercise you need 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.

## 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*.

## Simple Animations

After creating the models, we will need to animate them. You are required for 3 types of animations, each named after the keyboard key that toggles them on and off.
**'1'** - The ship will orbit the planet in a single axis of your choice. The distance from the planet is for your consideration.

**'2'** - The ship will orbit the planet in a **different** axis of your choice, but not the same one from animation '1'.

**'3'** - The ship will increase its orbit radius from the planet. You are not required to have a maximum radius height, therefore it might just drift away to the great unknown if you let it go far too much..

Note that the animations **are not exclusive, meaning you should be able toggle all of them at once.**

## 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:

The Interdisciplinary Center: Computer Graphics Course 2022
Instructor: Prof. Ariel Shamir
Submission due: 29th of May 2022 (23:55)

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 spaceship model, you can design additional features (such as back lights, exhausts) or anything else you see fit. There will be up-to 5 points bonuses, for creative additional features.

## 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 hull of the spaceship:
    a. Start with the hull itself.
    b. Design a single wing, then duplicate it so you won't need to rewrite each wing separately. Make sure that they are correctly attached.
    c. Design the windows in a similar way to the way you approached the wings.
4. Complete the spaceship with its head.
5. Add the planet and place it correctly besides the ship.
6. Start working on both animation '1' and '2' together, as they are very similar.
7. Add the final animation and make sure it works correctly with the other animations.

### Tips

- Pay attention to the order in which you present the polygons vertices.
- As we hint quite heavily, the spaceship model should be individual and independent. This will make your work in HW6 much simpler.
- 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 spaceship 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