# **(I)****PROBLEM STATEMENT**

## **1. Goal:**

– Visualize a large environment with smooth camera motions along

trajectories (at least 3) inside the environment

## **2. Features**

– The trajectories are based on control points defining a parametric

curve that will control both the camera position and the target point

of view position. Basically, you will be applying a parametric curve

to interpolate/approximate the camera parameters, including the

camera orientation.

– The environment should have some complexity to it, not be too

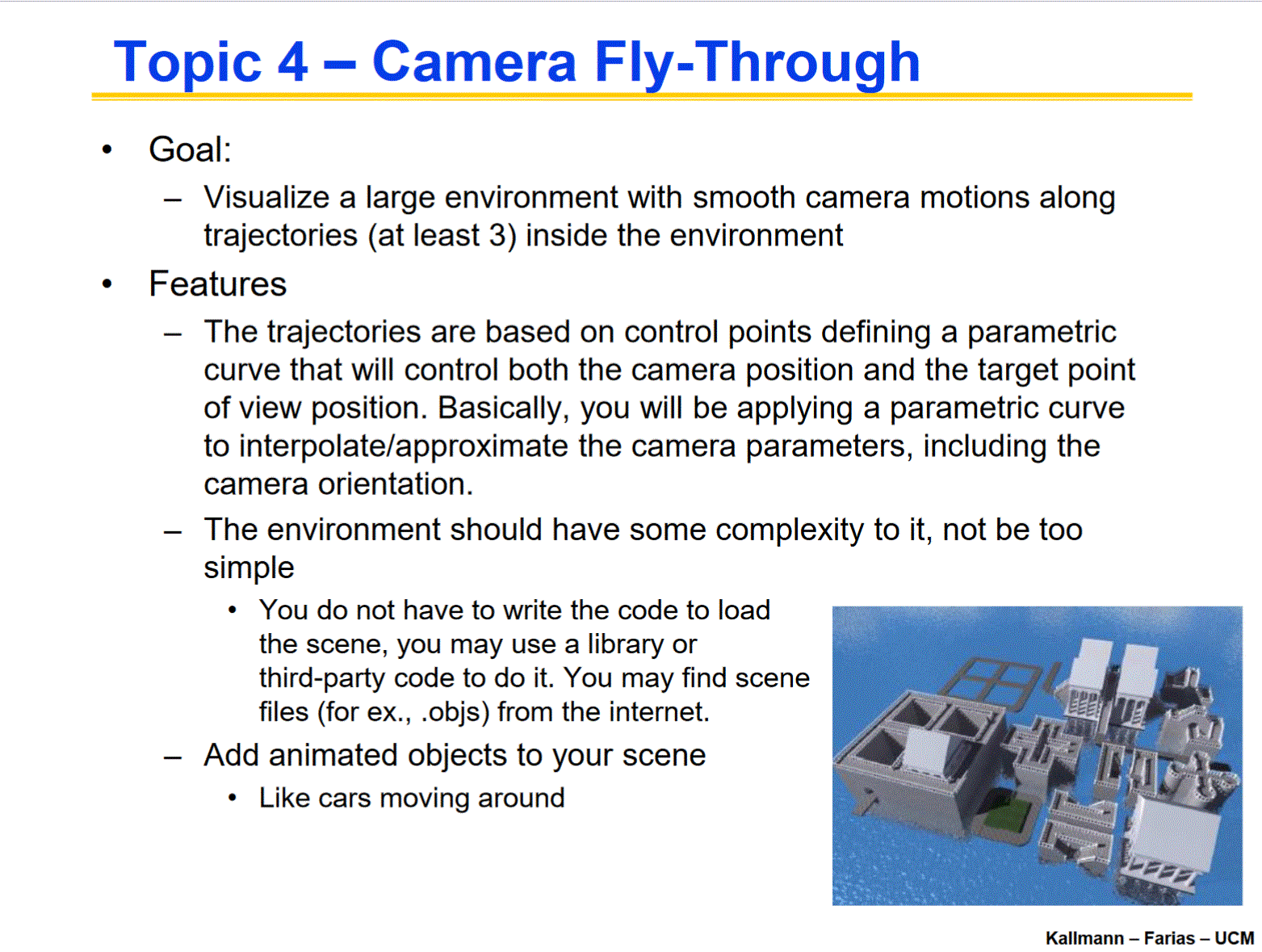
simple

• You do not have to write the code to load

the scene, you may use a library or

third-party code to do it. You may find scene

files (for ex., .objs) from the internet.

– Add animated objects to your scene

# 

# 

# 

# 

# **(II)CHECKPOINT:**

**Part 1: Show your initial results in a lab by the Checkpoint deadline: You have until your last lab**

**before the Checkpoint deadline to tell us the following:**

**a) define your chosen topic,**

**b) show some first results and**

**c) tell us what each member will do (if a group project).**

**More information follows below:**

1. **First results:** It is ok if you do not have much to show but you have to show something already done (some application running) and demonstrate that initial steps have already been implemented. Your project will most likely be in a group and each member will have to present some initial results and explain exactly what features each group member is expected to do. Note that the TA will have to agree that your proposal is reasonable before you move forward.
2. **Implementation:** Each member has to be responsible for at least one “significant feature” of the project. This “feature” has to be related to what we saw in class and has to be implemented completely by the member. External support code can be used but each member is expected to implement a significant feature of the project by himself/herself. Projects have to be implemented in C++ and OpenGL without immediate mode.
3. **Groups:** Groups can have 2 to 4 members. Individual projects are also possible.
4. **Changes:** If for any reason the plans for your project topic or group membership change, please communicate immediately to your TA, who will make sure we know what to expect from each project.

# **(II.2) CHECKPOINT -- SOLUTION:**

1. Topic 4- Camera Fly-Through

Abstract

The goal was to describe a 3-d space through a 4-d camera lens to create an effect of objects moving in a time at that space. Where we will have all objects in relation to a ground axis that would exhibit a range of random to pseudo-random movement patterns. The intention is to explore and create a rough draft or outline of our world to represent a natural flow as we exhibit in our daily lives.

1. Blah
2. S

|  |  |
| --- | --- |
| **Name** | **Objective/Component:** |
| Octavio Gonzalez | Helped define the overall structure, into finite amount of features to implement. Will help test the functionality and assist in implementation. |
| Milan Overholtzer | Transforming available space into various simple forms of objects, with their animations having patterns of movement that are predetermined, random or given simple AI. |
| David Hernandez | Stylings of the Scene; and its objects: in order to create the effect of a dynamic environment through color and lighting. |
| Mike Monokandilos | Structure Of Scene; finite bounds that contain objects in relation to ground axis |

Glfw3 not used deprecated for dlls upgraded to sdl2 because

Assimp object loading texture loading

GL create windows and scene of objs

Glm math libraries

SDL2 is for keyboard input and window creation coupled with GL

Libs -> to run librabraries

Shaders -> tentative place for camera definitions

Textures -> collection of pngs, jpegs, normals, diffuse, ambient, specular

Bonus features:

TBD