Lecture Notes for CMU 15-462: Computer Graphics

Emulie Chhor

May 12, 2021

1 Introduction

Courses Notes for CMU 15-462: Computer Graphics by Keenan Crane

2 Course Overview

Overview

1. What is Computer Graphics
2. What do we learn in Computer Graphics
3. What is Modelization
4. What is Rendering
5. Projects throughout the course

2.1 What is Computer Graphics

The goal of computer graphics is to simulate sensory information by turning digital information into sensory stimuli (visual, sound, texture). Its "inverse" would be the field of computer vision, whose goal is to translate sensory information in our environment into digital information.

2.2 What do we learn in Computer Graphics

There are two phases to learning computer graphics:

1. Theory: What are the techniques used to simulate sensory stimuli
2. Systems: How can we simulate those stimuli efficiently

Theory

1. Representation: How do we encode shape and motion?

2. Sampling and Aliasing: How can we acquire and reproduce signal
3. Numerical Methods: How can we modify signal
4. Radiometry and Light Transport: How can we reproduce light behavior
5. Perception: What can we learn from psychology and physiology to enhance human perception

Systems

1. Parallel and Heterogenous Processing
2. Graphics-Specific Programming Languages

2.3 What is Modelization

The first concept to understand in computer graphics is that we can produce simuli in two steps:

1. Modelization: How can we describe object to a computer
2. Rasterization: How will the computer translate our model into sensory stimuli

Usually, modelization is done by

1. Listing a bunch of points
2. Linking these points with a list of edges

However, we often have to perform some transformations to accurately represent the object. For instance, we need to find a way to represent 3D object onto a 2D shape.

How to convert 3D points into 2D shapes

One solution is to use perspective to our advantage. We know that object further from us are smaller, so we can use the "pinhole camera" from optical physics to realize that we have rectangular triangle, which allow us to use pythagoras to form a ration between the original points onto our 2D shape. If we have a 3D point (x,y,z) that we want to translate onto a 2D shape (u,v), we

1. substract the camera from the 3D point
2. Divide (x,y) by z to get our 2D point (u,v)

2.4 What is Rendering

To display our points on the computer, we first have to understand how the computer screen works. A computer screen is made of pixels, which are little square that contains a value which tells us how much green, red and blue should be in the square.

Often, we talk about raster display because we take a continuous object and represent it with a discrete mapping (pixels grid)

Which Rasterization Techniques to use: diamond rule

Since our shapes aren't alway rectangular, we must decide which pixels to turn on and off. There exist a bunch of rasterization techniques, but we often make up our own based on our needs:

1. Turn on all pixels that the vector touches
2. Diamond Rules: Turn pixel only if it passes through the diamond shape inside the pixel

How to color the pixel

Like with rasterization, there are several ways to color each pixel, but we often like to color it using the ratio of the vector inside the pixel: the more a vector is inside a pixel, the more we make the color thicker

How does the computer find vector: incremental rasterization

A naive solution to find which pixels to color would be to check each pixel individually, which takes $O(n^2)$ for O(n) pixels. A better method would be to use incremental rasterization. We

1. Calculate the slope of the line and add it to the y-value to find the pixels inside the x-range of the vector
2. Compute the pixel value
3. Draw

2.5 Projects throughout the course

The course consists of building our own library

1. Rasterization
2. Geometric Modeling
3. Photorealistic Rendering
4. Motion and Animation

3 Linear Algebra

NEXT