1 Introduction to Graphics

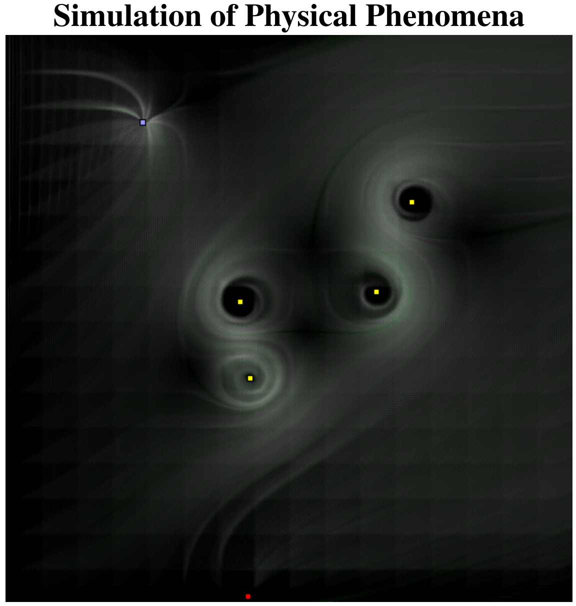
1.1 What is Computer Graphics?

Computer graphics is the area of computer science that deals with the representation, storage, generation, and display of visual information. While the term can be understood to include the hardware used to display information, for the purpose of this course we will focus on the software aspects of computer graphics. In particular, we will examine the principles and algorithms that underlie the rendering of visual information in modern graphics applications.
A few of the main areas of application for computer graphics include:

Information Visualization

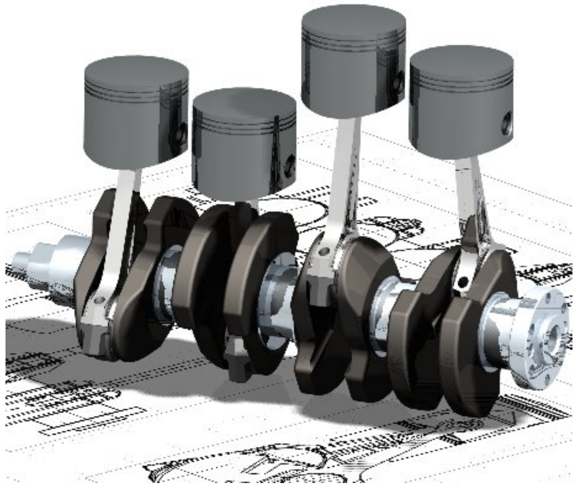
Visualization of a social network. [Source: Wikipedia, author: MartinGrandjean]

Simulation of Physical Phenomena



Visualization of flow patterns. [Source: Wikipedia, author: AndersSandberg]

Computer Aided Design

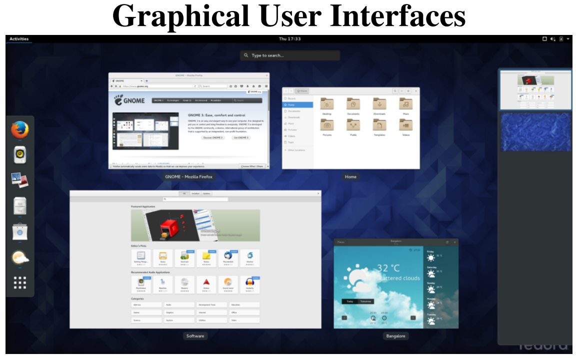


A 3D CAD model of an engine part. [Source: Wikipedia, author: Freeformer]

Virtual environments for training

A flight simulator used for training future pilots. [Source: Wikipedia, author: SuperJet International]

Graphical User Interfaces



The user interface for the Gnome desktop environment. [Source: Wikipedia, author: Irkshnan]

Video Games

A screenshot from 'Sintel the Game'. [Source: Wikipedia, author: Sintel The Game]

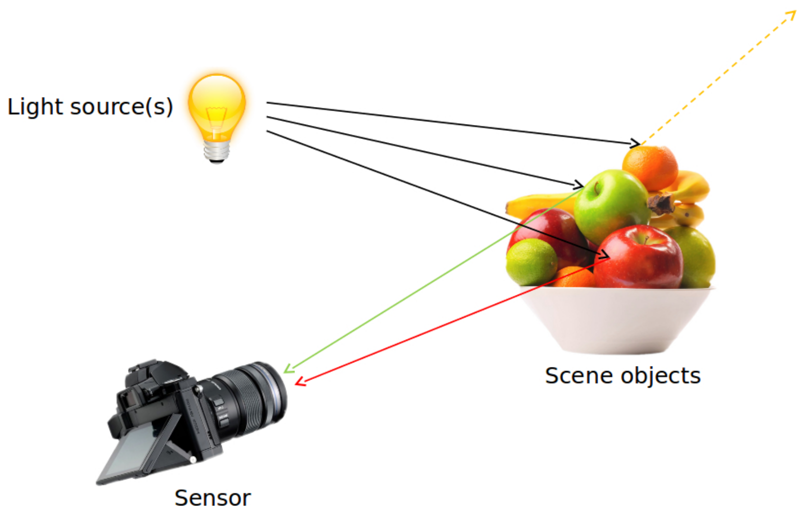
Computer Generated Imagery and Visual Effects

A computer generated scene of a house. [Source: Wikipedia, author: Mayqel]

Each of the areas above could well be developed into a course on its own. In this course, we will talk about specific applications only briefly. However, the principles we will cover apply to each and every of the application areas mentioned above.

1.2 The components of computer graphics

To understand how computer graphics are generated, we first need to understand how regular images are created - that includes both the images we perceive using our eyes, as well as any photographs we can record either in film or digitally.
The process of image formation is fairly simple:



The process of image formation. What creates the image on the camera's sensor (or your eyes as you look at the world) is all the light bounced off from objects in the scene.

- Light is emitted by one or multiple light sources (lamps, the sun, stars, fire, or anything else that generates and emits light).
- Light rays hit objects in the world and are bounced off from them
- Some of the bounced light ends up making its way into a sensor (an eye, or a camera) and is recorded there, forming an image of the objects in the scene

The process itself, though very simple to describe, turns out to be fairly complex to simulate inside a computer. For one thing, light may bounce multiple times before hitting the sensor - from one object to another, to another, and so on. This happens for all types of objects, not just for shiny things such as mirrors. Figuring out the paths light takes to get to the sensor is computationally expensive.

What happens to light when it hits an object is also complex. In general, a ray of light hitting an object will find its colour is changed in some way which depends on the surface properties of the object: The objects colour, how reflective or shiny it is, whether it is partially transparent, and what kind of material it is made of. Simulating this process correctly is essential to capturing the 'look' of the object on computer generated images.

Finally, simulating the camera or sensor itself needs attention. All sensors use lenses of some type to focus on objects of interest, and this introduces visual effects (such as objects in the background being blurry and out of focus) that need to be simulated for rendering convincing images. While we don't usually worry about simulating the sensor itself, if we were interested in reproducing a specific type of medium (for example if we were attempting to create an image that looks as if it had been captured on photographic film), we would need to simulate the particular characteristics of the recording medium as well.

The three elements of image formation described above: Light sources and their light rays, objects and their surface properties, and sensors that capture and record light will be the central components of our course.

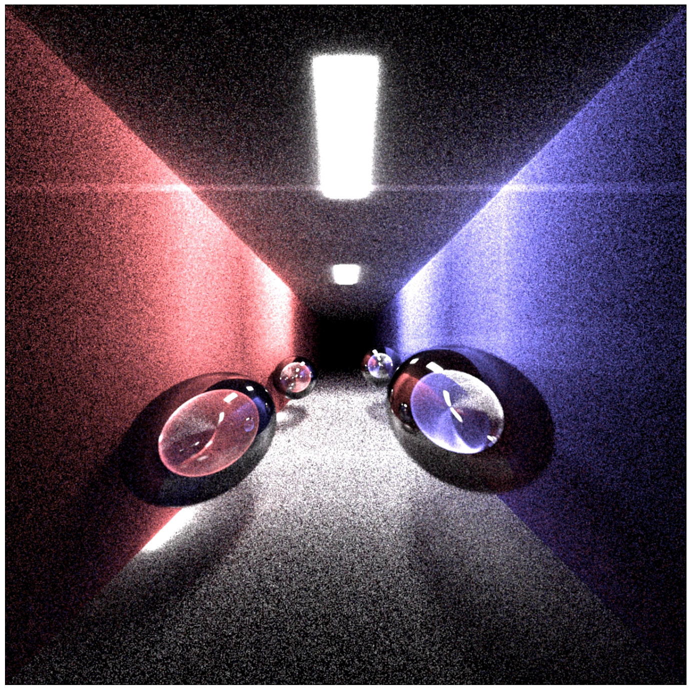
We will study the properties of light sources, what gives their light rays colour, and how the type, size, and shape of the light source constrains the directions of emitted light rays. We will study how to represent and manipulate simulated light rays, and how to determine when they have hit an object in the scene we are rendering.

We will study objects and all of their properties, starting with figuring out how to represent them within a computer, how to study their shape, and how to manipulate them so that we can build the scene we want to render. We will then look at surface properties, how to model them, and how to use them to figure out what happens to light rays when they hit a particular object.

Finally, we will look at cameras and how they form an image of a scene. This will involve figuring out the specific details of what parts of a scene can contribute to the colour recorded at each specific point in the image, and modeling the complete process of image formation from the moment a light ray is emitted to the time it hits the camera's sensor and is recorded as a specific colour.

Throughout the course, we will make use of linear algebra and calculus to help us model, represent, and manipulate the different entities required in the rendering process: light sources, light rays, objects (their shape, size, location, and surface characteristics), cameras, and the images themselves that we are rendering.

Having studied these fundamental components of computer graphics, we will be able to write from scratch a software renderer capable of simulating the image formation process described above. Such a renderer, although very slow in practice, is able to fully account for the most complex visual phenomena. A rendering engine that directly simulates the image formation process is called a *forward ray tracer*, and though not used in practice - we will find out there are much more efficient ways of rendering realistic images - it illustrates well the principles involved in the generation of realistic images with a computer. A simple scene rendered by a forward ray tracer is shown below.



A very simple scene rendered using a forward ray tracer - about 50 billion light rays were simulated in order to yield the final image. While the geometry is simple, the image shows complicated lighting effects such as transparency, reflections, colour bleeding, soft shadows, and caustics; all of which are difficult to simulate accurately.

1.3 Organization of the course

The course is organized so that you will be able to start creating images fairly early. We will first spend some time studying objects and how to represent and model them within the computer.
This will include learning about simple 2D and 3D curves and surfaces, transforming and manipulating objects, representing objects with polygon meshes, and using homogeneous coordinates to easily handle object point coordinates as well as for managing transformations.
We will then move on to the study of cameras and image formation, we will understand the complete image formation process, and we will discuss in detail the process of ray tracing.
Next we will study light sources and light rays, we will look at how to model the appearance of objects, and we will study how to determine when light rays hit objects and in which directions it is bounced as a result. Having covered the fundamental components of image formation, you will write your own simple ray tracer.
From that point on, we will focus on advanced rendering techniques that will enable your ray tracer to produce realistic images with complex lighting, objects with interesting surface characteristics (including transparency and texture), and lens effects. And then we will move on to state-of-the-art algorithms for rendering all of these advanced effects at a much lower computational cost. By the

end of the course you will have written your very own path tracer, an advanced rendering engine capable of dealing with realistic scenes and of generating very high quality images in a reasonable amount of time.