Computer image generation overview

Example of computer generated image

<IMAGE: synthetic image which motivates this lecture. We will break image into parts which make pillars of computer image generation>

Goal of computer graphics

- Computer graphics is solving a **problem of generating (synthesizing) images**.
- As in computer science, computer graphics problems are solved using **simulation and approximation.**
  - The real world which is simulated is continuous.
- Computer graphics simulations and its results (generated image) are **discrete**.
  - Computer generated image and display device are 2D array of **pixels** – **raster** graphics and display (grid of x and y coordinates on a display device).

Pillars of computer graphics

- To synthesize an image:
  - We need a **3D scene** which contains objects and phenomena we want to simulate
  - A way to transform a 3D scene into 2D array of pixels – **rendering** – another simulation as we will see
  - A way of storing 2D array of pixels for raster display – **image**

- Computer graphics R&D is this concerned with:
  - Modeling of 3D scene
  - Development of rendering algorithms for creating image out of 3D scene
  - Storing and displaying image.

- Image generation software can be thus split into:
  - **Modeling** – process in which 3D scene is defined
  - **Rendering** – process in which image of 3D scene is generated

- Tasks of computer graphics are intertwined:
  - Rendering process must “understand” how 3D scene is represented
  - Modeling of 3D scene relies on representations which are understood by rendering algorithm

3D scene modeling – a start of image generation journey*

- In order to generate an image (to render it) we need something to render → a 3D scene.
- 3D scene is simulated 3D world from which images can be taken - similarly as photographing a real world. Thus, we also need to simulate a device which is needed for taking images – a **camera** – which also defines from where image is taken.
- The real world is made of wide range of phenomena and objects. For modeling of a 3D scene we would like to simplify those. Therefore we describe **objects** by **shape and appearance**.
- The reason why we see objects around us is due to light reflecting in our eyes. Therefore, for purposes of simulation, we can separate objects in those which generate light - **light sources** and those which only reflect light.

* As discussed, 3D scene modeling and rendering in order to obtain an image is highly intertwined process. For learning purposes and as well as in research in development this process is separated. Depending on application and teaching the description of whole process can start from rendering as well.

Objects in 3D scene

- Real world objects exhibits wide range of shapes and appearances; for example cloud, chair and river stream.
  - In graphics, we simplify, and categorize different objects being **solid (volume)** or **surface**. Volume objects are used to represent interior since we can see through it (e.g., water). For surface objects, we only care about exterior (e.g., tree)*

    <IMAGE: DIFFERENT OBJECTS AND REPRESENTATIONS>

* Representing objects is not only important for visualizing purposes, it is also important for modeling them. Therefore, even some objects for which we only care about the surface appearance (e.g., car rim) we would like to represent them as solids in certain modeling tools which would enable simulation of cutting and modling the surface). For example: https://www.autodesk.com/products/fusion-360/features#3d-modeling or https://pixologic.com/.

Concept of 3D space

- Before we go to further, we need to understand that all objects in 3D scene are "living" in a 3D space.
- To define any object we need to introduce a concept of 3D space.
- 3D space is represented with 3 coordinate axis. This main coordinate system is called **world.**
  - Coordinate system representation which we are using is **Cartesian coordinate system*.**
- All objects have a position in this world coordinate system.

<IMAGE: coordinate system as concept of space>

* Other representations for 3D space are also possible, one very commonly used is spherical coordinate system.

Surface shape representation 1

- To define object's shape in a 3D scene, we need to define a concept of points in 3D space.
- **Point** is defined as three floating point numbers for each of Cartesian coordinate system **(x,y,z)**.
- To define a surface, simplest way is to connect points to form a **polygon**. In computer graphics, for computation tractability, we often use co-planar – all points lying on the same plane - polygons and especially **triangles**.
  - Triangle is almost widely used surface shape representation **primitive.** This is because it is very simple and holds great properties for easy calculation thus very much researched and used for efficient rendering purposes. Different shape representations are also introduced for easier modeling, but it is very often that all representations are turned to triangles before/during rendering stage - using very elaborated method called **triangulation**.
- Triangle is basic building block for creating more complex shapes. And modeling is all about creating complex shapes using basic building blocks.

<IMAGE: POINTS, TRIANGLES, COMPLEX SHAPES>

Surface shape representation 2

- Some shapes do not have flat surfaces! Polygonal shape representation will always have flat surfaces*.
- If we would like to represent curved surface, we would need smaller triangles which would fit the curved surface better. In this process we are placing more points on to surface (sampling). Generally, converting smooth surface to triangulated polygon representation – a **triangulated mesh** – is called **tessellation**.
- Main point to take away is that we can represent any object using triangle polygons. Those objects will never be perfect representations of a real world, but triangles of which they consists of can be small enough to display objects in high quality taking in account restrictions of raster display.
  - Amount of details increase realistic representation but also complexity of rendering. Computer graphics often deals with trade-off between visual quality and speed**.

* There are methods in rendering which make surface looks smooth although is represented using polygons. This method is called smooth shading and we will discuss it later.

Surface shape representation 3

- Although polygonal meshes are widely used and popular (e.g., games and film) there are other representations that are more suitable for modeling purposes
- One of these are **curved surfaces** and **subdivision surfaces** used to design manufactured objects and often used in CAD software.
- Foundation of those representations are still **points**, but those points define a **control mesh** from which perfect curved surface can be generated using **analytical description**.
  - Note that this kind of representation is much more compact than polygonal mesh.
- Modeling using control points is very beneficial for curved objects. When it comes to rendering, this representation can not be rendered directly. As discussed, process called **tessellation** must be performed prior rendering.

<IMAGE: CURVED AND SUBDIVISIONS SURFACES>

Note: How are objects created?

- Lot of 3D objects are created by hand
  - Lot of mesh and curve modifiers as well as techniques and methods are developer for faster and easier design
- Some objects are digitized from real world
  - 3D scanning can be used to “import” real object into digital world.
- Some objects are simulated
  - Some objects are very hard to simulate and capture from a real world. For example water stream. For these purposes physical simulations are also employed to generate shape.

<IMAGES: manual modeling, capturing from real world, simulating>

Surface shape representation 4

- For generating shape, simulation can be used, e.g., smoke simulation. For this purposes, it is required to represent 3D space in which simulation is performed as grid of cells which are called **voxels**.
- Each voxel (a cell) is a small volume of space on which computation is performed. Simulation is performed in series of steps. Each step is recorded and transformed to triangulated mesh for rendering.
  - This way, **animated** mesh can be generated.
- Voxels are also widely used for any kind of modeling purposes where it is required to describe object's interior – e.g., digital sculpting.

Shape representation for modeling

- Constructing complex shapes can be also done using basic primitives (box, sphere, etc.) and series of operations (add, substract, etc.). This kind of modeling is called Constructive Solid Geometry.
- An representation for such basic primitives is called implicit since they are completely analytically defined.
- This enables fast and flexible modeling system, but when it comes to rendering, they need to be converted to triangulated mesh – similarly as for curved surfaces. Algorithm in this case is called marching cubes.

<CSG AND IMPLICIT SHAPES MODELING>

Static and animated objects

- Introducing a time component into 3D scene and moving/rotating objects and their parts as time goes gives foundations for animation.
- Animation can be predefined, generated at render-time or influenced by interaction
- TODO

Modeling and representing 3D objects: keypoints

- From previous discussion it is important to note that different representations for 3D models exists and that different representation are better for rendering (triangulated mesh) and other are better for modeling (e.g., implicit surfaces).
- In professional software, different representations for modeling will be available, but also, those are converted to triangulated mesh for efficient rendering.
  - Having one representation for rendering makes rendering process highly efficient since all effort is put into efficient methods for rendering one representation.
  - Converting all objects to same representation (triangulated mesh) is also convenient for applying various rendering effects. For example, techniques for adding more details during rendering time can be only developed to work with triangulated mesh.
  - Lot of research and hardware development was focused onto efficient rendering of triangles as we will see*.

* It is easy to imagine that different rendering primitive is used. But due to historical events this turned out to be a triangle. What is important to note is that mapping of various representations to triangle mesh is possible and thus it is not important which is the rendering primitive as long as it supports various representations.

Other important information on 3D objects

- Until now we discussed shape representation of 3D object.
- Shape is important for determining how object will appear, but also its appearance depends on material and surface details.
- TODO

Light in 3D scene

- As briefly mentioned, light enables us to see objects.
- Some objects emit light – light sources.
- Some objects reflect/refract light.
- What we see is light falling falling in our eyes/camera.
- TODO

Camera in 3D scene

- Camera defines point of view and it is used to simulate effects of real-world cameras.
- Camera is a window to a 3D scene.
- TODO

Recap

- First step to synthesized image is to create a **3D scene**, that is, to define:
  - 3D objects
  - Lights
  - Cameras
- Second step is to generate image using 3D scene information using **rendering**.

Rendering

- Rendering is general term for generating images from a 3D scene.
- Often, goal is to create **photo-realistic** images – images that look like a photograph* of a real world. This is one spectrum
- On the other hand, creating **non-photo-realistic** (NPR) images** – images that contain wide range of expressive styles, is also often desired.
- In both cases, certain rendering method is used. In order to generate an image, a simulation of appearance of objects is needed***.
- For photo-realistic images, to make objects appear as they do in real world, we need to simulate laws of physics that are related to appearance – optics, that is, simulating light transport and interaction with objects in 3D scene and camera.

* Note that there is also distinction between images that look like a real photo and that are "correct" like a real photo. In graphics, almost always it enough to create images that look like a real photo but the way they are produced doesn't necessary be correct in terms of how real world works. Again, this is trade-off between quality and speed and decision depends on application.
** Specific look of NPR images often comes from exaggerating some characteristics of 3D scene or rendering algorithm. Therefore, it is better to start with photo-realistic rendering and then altering this method to obtain required style in NPR.
*** Note that the realism/style of image, next to rendering, also depends on the way how 3D scene is simulated as well as its complexity. This is another example showing how rendering and modeling of 3D

Rendering: simulating visual system

- As we are simulating images that resemble ones that we take with a real camera or see with our eyes – we need to take in account the **foreshortening effect**.
- This effect is important for photo-realistic rendering since it simulates how our visual system works – it defines shape and size with respect to the distance to the eye.
  - Objects that are further appear smaller than those which are closer
- Method for achieving foreshortening effect is to trace rays from corners of objects to eye and intersecting them with imaginary canvas that lies in between <IMAGE>. https://en.wikipedia.org/wiki/Perspective_machine
  - Direct result of such method is used to simplify the 3D scene while modeling and it is called **wireframe** rendering.
- This example brings the key idea used in every rendering algorithm – **perspective projection**.

Rendering: a visibility problem

- Looking with a camera from particular point of view, we can only see portion of the scene and only some objects.
- Determining which objects and which of their parts are visible is called **visibility problem***
- Visibility problem is used for both determining what is visible from camera and which surfaces are visible to each other in 3D scene (this will be clear later).
- Solution to visibility problem in computer graphics can be solved using two main methods:
  - **Rasterization***
  - **Ray-tracing**

* Also known as: hidden surface elimination, hidden surface determination, hidden surface removal, occlusion culling and visible surface determination
* This is umbrella term and some popular methods are painter's algorithm and z-buffer. Almost all GPUs use an algorithm from this category.

Rendering: appearance calculation

- Once visibility problem for camera view is solved (which inherently takes in account perspective projection giving us information of size of the objects) – once we determine which shapes we see and how large, we would like to calculate their appearance.
  - Note that for solving the visibility we need shape information of the objects.
- Appearance determines the look of objects in terms of color, texture and brightness.

  <EXAMPLE: light traveling from light source, falling on surface, interacting with material, reflecting into eye>
- Appearance is determined with light falling on the object surface and interacting with surface material. What we see is the result of this interaction. Therefore, we can divide appearance computation in:
  - Light interaction with surface - **shading**
  - Gathering light onto surface – **light transport**

Rendering: shading

- When light comes in contact with an object, simply speaking, two things can happen:
  - Light is absorbed
  - Light is reflected into scene
- Absorption gives objects their unique color. If white light (containing all colors) falls onto object which absorbs all colors except red – which is reflected, we perceive this object as red.
  - Color of object is one foundational material information of object as we will see later. It can be defined directly or indirectly using physically-based models which calculate color.
- Reflection. Light which is not absorbed is reflected. We know amount and color of reflection using information on absorption. Direction of reflection depends on:
  - Surface orientation on which light falls – this is described with geometrical property known as **normal** vector.
  - **Scattering** – a model which describes what happens with light as small scale. This model can have uniform parameters or varying parameters – giving textured look.

Rendering: light transport

Rendering: recap