

Computer Graphics & OpenGL

Sam Sartor

February 2, 2017

Mines Linux Users Group

Introduction

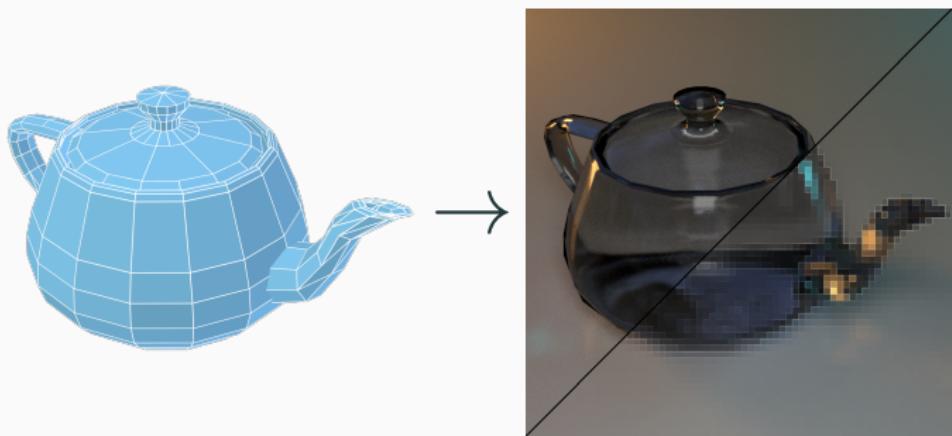
Uses

Computer graphics is everywhere!

- Your terminal
- Web browsers
- Video games
- CAD software
- Movies, TV Shows
- Virtual reality
- Your bootloader
- QT, GTK+, wxWidgets
- Vim, Emacs, Notepad
- Embedded devices



Definition



Computer graphics is the science of turning *shapes* into *pixels*.¹

¹Kindof, it can get more interesting than that

Behind the Scenes

Realtime

Realtime graphics use OpenGL or Direct3D to rasterize and shade triangular geometry on a graphics card/chip. Performance is very important due to the high framerate that is required for smooth gameplay/interactivity/animation. Lighting and materials focus on being "good enough" rather than on being truly accurate.



UIs

While they look different, UIs generally use OpenGL or Direct3D as well. Everything is still made of textured & shaded triangles. Anti-aliasing, text fidelity, etc. are all more important while lighting effects are generally absent. Responsiveness is key, but the frame can be updated as needed, not every 30th of a second.



Offline

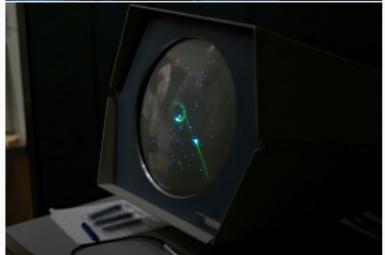
Offline graphics are used when the medium is non-interactive (movies, advertisements, etc). Because the available resources are limited only by budget and patience, offline graphics have unmatched fidelity. CPUs are often used instead of GPUs because this allows for more advanced calculations. However, this comes at a cost. Individual frames may take days to render.



History

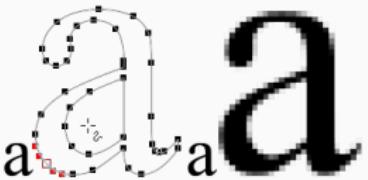
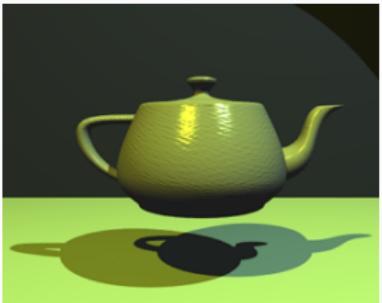
1950s & 1960s

- Military used computer controlled oscilloscopes to display strategic information
- Very simple graphical CAD programs and visualizers created
- Very first computer games
- Research into elementary 3D wireframe graphics
- Very early raster displays



1970s & 1980s

- Basic lighting models such as Phong developed
- Low-res, 2D games become commercially available
- CGI starts to be used in Movies such as 1982's *Wrath of Khan* and 1985's *Young Sherlock Holmes*
- Modern GUIs are developed
- High-quality digital typesetting becomes commonplace



PostScript type

Bitmap type

1990s & 2000s

- Fidelity and performance are immensely increased
- Personal computers, 3D video games, and GUIs become ubiquitous
- OpenGL and Direct3D standardize hardware graphics support
- CGI becomes commonplace in Movies, advertisements, and TV
- Global illumination and physically based rendering (PBR) techniques developed



Today

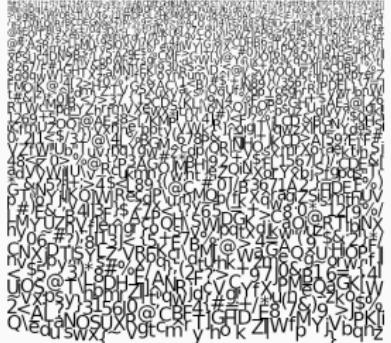
- Given enough time, budget and expertise, offline graphics are photorealistic
- Particle and fluid simulations are extremely fast and accurate
- Realtime graphics make extensive use of shaders and PBR techniques
- UIs and offline graphics are increasingly GPU accelerated
- Linux and Mac have improved support for games and graphical software



Easy on the Eyes

Text

The most common thing shown on screens is probably text. It is no wonder then that displaying text is one of the most sophisticated areas of computer graphics.



A dense grid of characters, primarily lowercase letters, forming the word "Text". The characters are arranged in a pattern where 'T' is at the top left, 'e' is below it, 'x' is to the right of 'e', and 't' is to the right of 'x'. The grid is composed of approximately 100 columns and 100 rows of characters.



Which is better?

Standard Rendering

Render

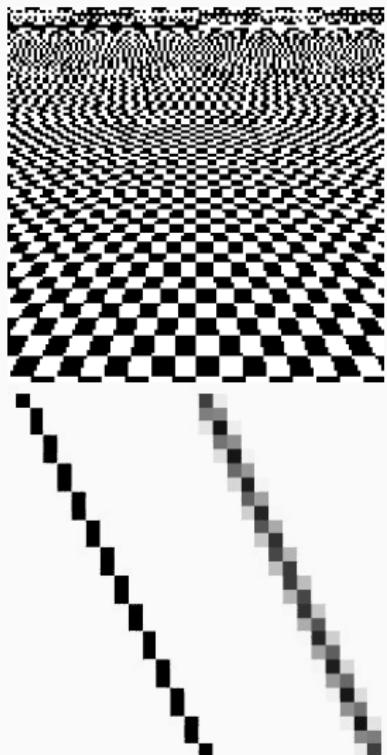
Subpixel Rendering

Render

Anti-aliasing

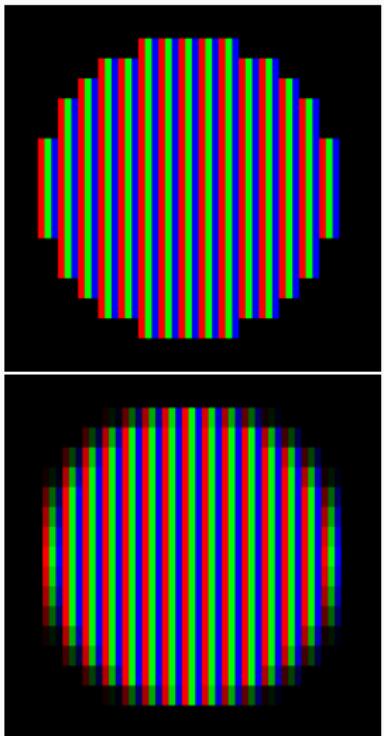
When small shapes (such as letters) are rasterized, details can find themselves squished into single pixels. In these cases, the color of the pixel is chosen arbitrarily depending on how the rasterization is performed. This creates all sorts of strange effects and hard edges, called "aliases".

The quality of the image can be greatly improved in some cases by using "anti-aliasing" algorithms and techniques.



Sub-pixels

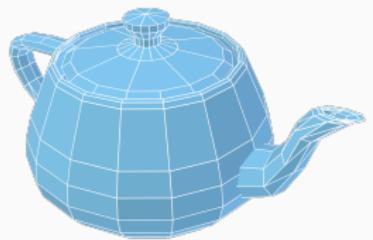
Anti-aliasing can be taken further. After all, each pixel is really 3 small red, green, and blue rectangles joined together. Clever people have figured out how to use these extra rectangles to further smooth the edges of letters and lines. This is why blown-up text (in an image editor) ends up being slightly colored on the edges.



3D Shapes & Geometry

Meshes

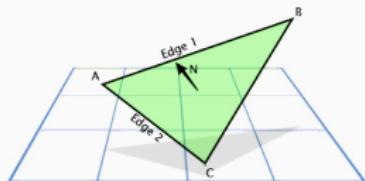
3D shapes are often stored as a jumble of points (called vertices), scattered in space. These points are linked together into polygons, which form the surface of the model. Other methods of storing 3D shapes exist, such as the parametric forms used by Solid Works, but "meshes" are by far the most common. In fact, even when a different form is used, the model is almost always converted into a mesh for rendering anyway.



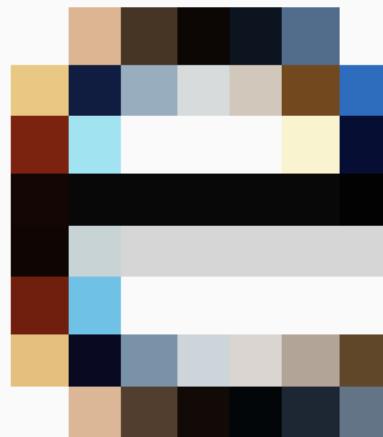
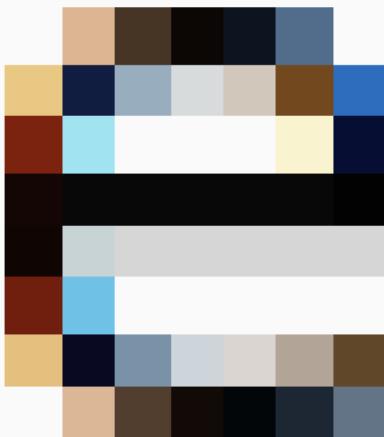
Triangles

While any polygon could be used in a mesh, modern computer graphics deal almost exclusively in triangles. This is because triangles have some nice geometric properties that other shapes don't:

- It is very easy to test if a point is within a triangle (barycentric coordinates!)
- Any 3 points can be haphazardly connected together and still form a flat triangle
- It is easy to calculate the direction a triangle is facing



3D Modeling

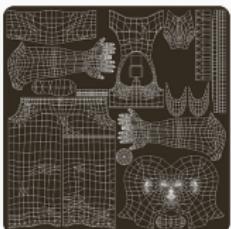


Making it Pretty

Textures

Textures are images (though they can contain more than colors) that are wrapped around 3D models. The exact method for this varies, but most of the time the wrapping is defined by a second set of coordinates attached to each vertex/point. This specifies where on the texture that point (and by extension, attached triangles) should lie.

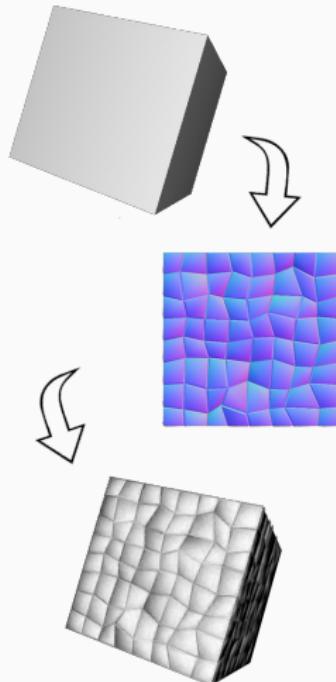
This second set of positions is called the "UV map".



Mapping

Textures can be used to cover objects in all sorts of information. Not just color, but also material properties such as shininess and even precalculated lighting information.

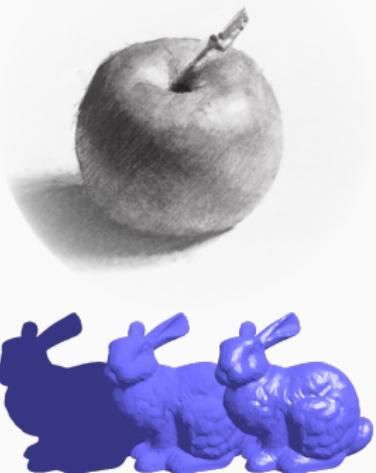
More advanced techniques (such as bump, normal, and parallax mapping) can be used to fake bumps and other minute, shaded details on technically flat surfaces. Games use this extensively to make low-resolution objects appear highly detailed (think screws, buckles, bricks, tree bark, ground texture, etc.)



Phong

Phong is the simplest common shading model. It mixes diffuse shading (areas facing a light are brighter) and specular shading (areas that reflect light towards the camera are brighter) together. Think of how you would shade something with a pencil.

While this kind of works, it massively simplifies how real-world materials and lights interact. As a result, it looks downright fake.

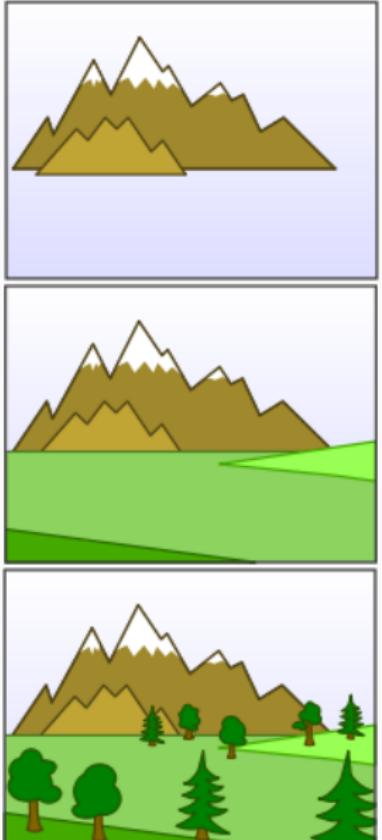


Painter's Algorithm

It goes without saying that things that are farther away should be hidden by things that are closer. The simplest way to do this in a computer is to draw the farther stuff first, so that when nearer stuff is drawn, it covers the previous stuff up.

However, this requires all the objects in the scene to be constantly resorted. It also can't deal with objects that mutually cover each other (like the Olympic rings).

That being said, there are still cases where the Painter's Algorithm is the only viable method.



Depth

A more common way to hide objects behind each other is to use a depth buffer. A depth buffer is a sort of distance image. In addition to keeping track of the color of each pixel, the distance from the camera to that pixel is also tracked. Then, whenever a new pixel is calculated, the computer checks whether it is in front of or behind what is already there.



Effects

Transparency

Transparency can put a wrench in things.



Post-Processing

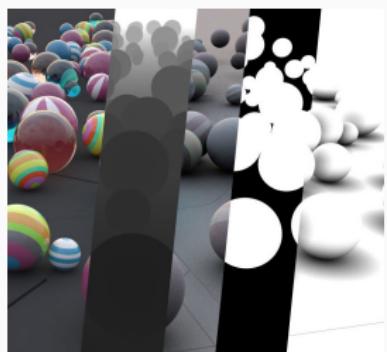
Texturing and shading is not always enough to make an image look good. Any photoshop guru can tell you how much a little tweaking can improve things. These "post-processing effects" aren't just used to make realtime graphics nicer, they are used to style-up photographs and videos as well.



Multipass Rendering

Post-processing can be taken even further. Instead of lighting and textured being performed as the shapes are rasterized, data from along the way can be saved in buffers and stapled together after the fact.

This has numerous benefits depending on how it is used. It is useful in realtime graphics because data from a single pass can be recycled to create multiple effects. Artists working on offline graphics like it because it lets them tweak shading and lighting without re-rendering the whole scene.



Bluring

Transparency can put a wrench in things.



Compositing

- ...



Realism

Reflection

- ...



Refraction

• ...



Raytracing

• ...



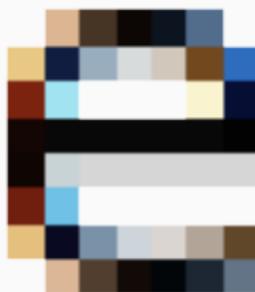
Fresnel & PBR

• ...



Local Illumination

- ...



Global Illumination

- ...



Pathtracing

- ...

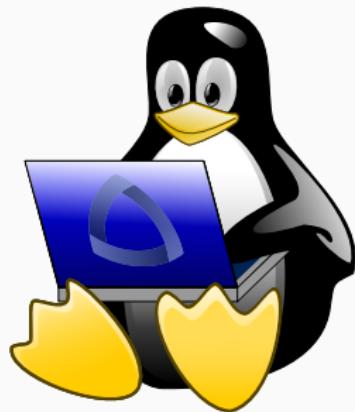


Wrapping it Up

Copyright Notice

This presentation was from the **Mines Linux Users Group**. A mostly-complete archive of our presentations can be found online at
<https://lug.mines.edu>.

Individual authors may have certain copyright or licensing restrictions on their presentations. Please be certain to contact the original author to obtain permission to reuse or distribute these slides.



Colorado School of Mines
Linux Users Group