Computer Graphics

Dr./ Ahmed Mohamed Rabie

Chapter 4

Game Graphics

Computer graphics can be seen in many different areas of various media and entertainment fields. The topic of computer graphics is huge and can become very technical when dealing with advanced rendering topics. Having an understanding of computer graphics is the key to being able to create the types of visuals that are becoming commonplace across the various industries. Today, visuals are very important in the entertainment industry, whether these visuals are realistic, nonrealistic, or artistic in some manner.

Today's games require artistic talent as well as technical knowledge about the software and the hardware technology being utilized. What makes video games and other interactive applications special is that they are indeed interactive products that operate in real time. Video games are progressing in a direction that is getting them closer to the visuals in movies, but, unfortunately, the power of computer hard-ware still has a ways to go.

Video games have other resource consuming systems such as artificial intelligence, networking, sound, input detection and response, physics, and so forth, each having to operate in real time. This in itself makes creating impressive visuals difficult because games are required to render a number of times per second instead of one frame in X amount of time, which the movie and TV industries can afford when preprocessing visuals.

Computer graphics in video games is a huge area of study and will continue to grow as time goes on. Over the past few generations the graphics in games have increased considerably in technical terms, quality, and art style. With this increase in graphical quality also came an increase in what gamers expect from the games they buy. Every time a new game makes a leap in graphical quality, it raises the bar for the entire industry.

For many years video games have been sold based on their graphical representation. Today, this is accompanied by increased physics, artificial intelligence, and other areas that in the past took a back seat to the visuals. With games becoming higher in quality, they are starting to gain ground against other entertainment media such as the motion picture industry.

Some of the earliest video games were text games or x. Examples include MUDs (multi-user dungeons), where players could read or view depictions of rooms, objects, other players, and actions performed in the virtual world; and roguelikes, a subgenre of role-playing video games featuring many monsters, items, and environmental effects, as well as an emphasis on randomization, replay ability and permanent death. Some of the earliest text games were developed for computer systems which had no video display at all.

Text based Game

```
You are crawling over cobbles in a low passage. There is a dim light at
the east end of the passage.

? east

You are in a small chamber beneath a 3x3 steel grate to the surface.
A low crawl over cobbles leads inward to the west.

The grate is open.

? west

You are crawling over cobbles in a low passage. There is a dim light at
the east end of the passage.

? west

You are in a debris room filled with stuff washed in from the surface.
A low wide passage with cobbles becomes plugged with mud and debris
here, but an awkward canyon leads upward and west. A note on the wall
says

 "Magic word XYZZY".

A three foot black rod with a rusty star on an end lies nearby.

?
```

Text games are typically easier to write and require less processing power than graphical games. However, terminal emulators are still in use today, and people continue to play MUDs and explore interactive fiction. Many beginning programmers still create these types of games to familiarize themselves with a programming language, and contests are still held even today on who can finish programming a roguelike within a short time period, such as seven day.

Vector graphics refer to the use of geometrical primitives such as points, lines, and curves (i.e., shapes based on mathematical equations) instead of resolution-dependent bitmap graphics to represent images in computer graphics. In video games this type of projection is somewhat rare, but has become more common in recent years in browser-based gaming with the advent of Flash and HTML5 Canvas, as these support vector graphics natively.

Vector game can also refer to a video game that uses a vector graphics display capable of projecting images using an electron beam to draw images instead of with pixels, much like a laser show. Many early arcade games used such displays, as they were capable of displaying more detailed images than raster displays on the hardware available at that time. Many vector-based arcade games used full-color overlays to complement the otherwise monochrome vector images.

Full motion video (FMV) games are video games that rely upon pre-recorded television- or movie-quality recordings and animations rather than sprites, vectors or 3D models to display action in the game. FMV-based games were popular during the early 1990s as CD-ROMs and Laserdiscs made their way into the living rooms, providing an alternative to the low-capacity ROM cartridges of most consoles at the time. Although FMV-based games did manage to look better than many contemporary sprite-based games, they occupied a niche market; and a vast majority of FMV games were panned at the time of their release, with many gamers citing their dislike for the lack of interaction inherent in these games.

2D Game Graphics

2D graphics have been around in some form since the beginning of game graphics. In the early days of video games, 2D side-scrolling games were popular. Later, 3D games started to emerge and became the standard in the industry. Although 3D graphics have taken a huge role in video games, 2D gaming applications are still very popular.

Cuphead 2D Game

This can be seen on handheld gaming systems such as the Nintendo DS and Sony's PSP. It can also be seen on Microsoft's Xbox 360, Sony's PlayStation 3, and Nintendo's Wii video game consoles, where past arcade and console games are making a comeback as downloadable games. Along with bringing back old-school games, new 2D games are also being made available for download on each of these systems. One very popular 2D video game is Geometry Wars Evolved for the Xbox 360.

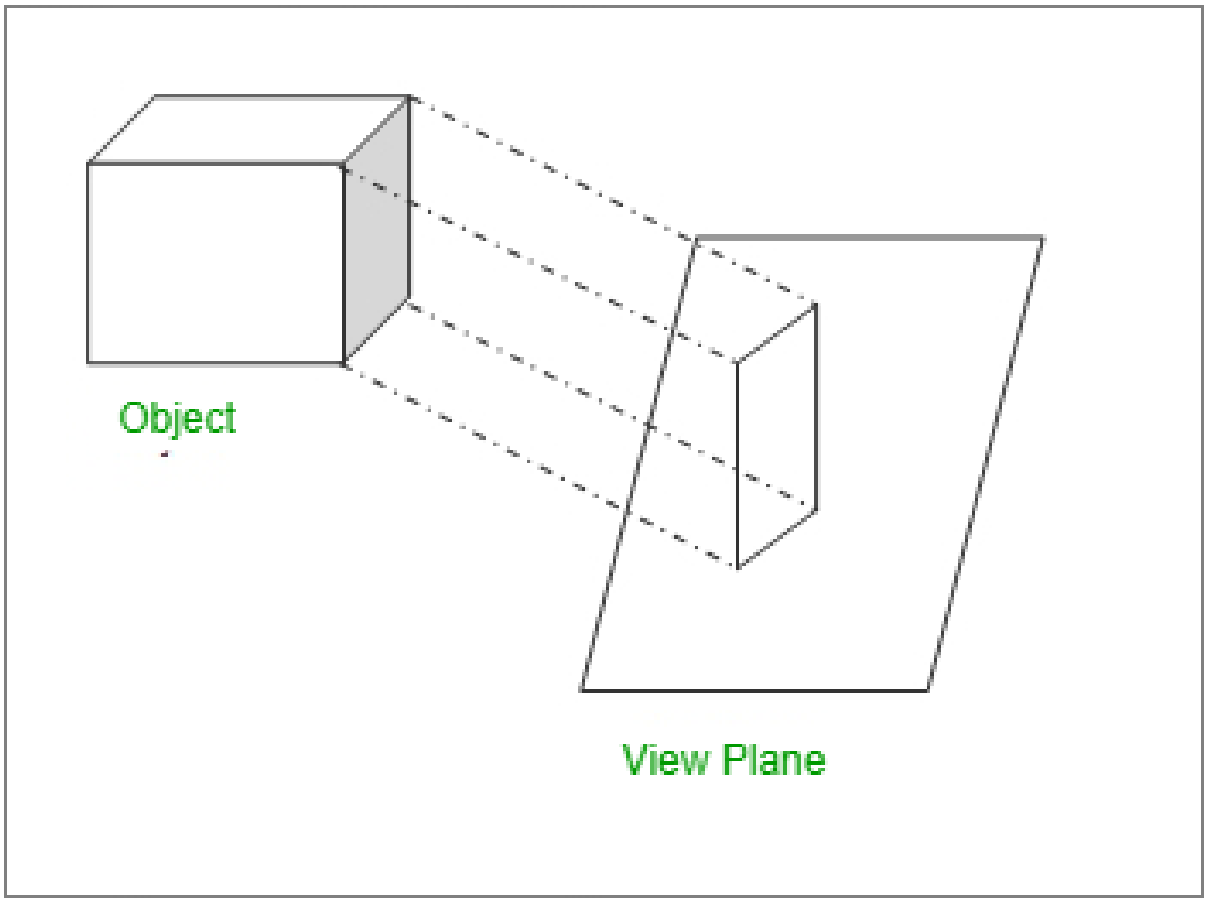
INSIDE 2D Game

There is still a place for 2D-based games in the marketplace, and they can prove to be a very valuable way into the games industry for enthusiasts, students, and those looking to make their way into professional game development. Not only are past arcade and console games doing quite well on the newer generation consoles, but new games based in or partially in 2D are also becoming very successful.

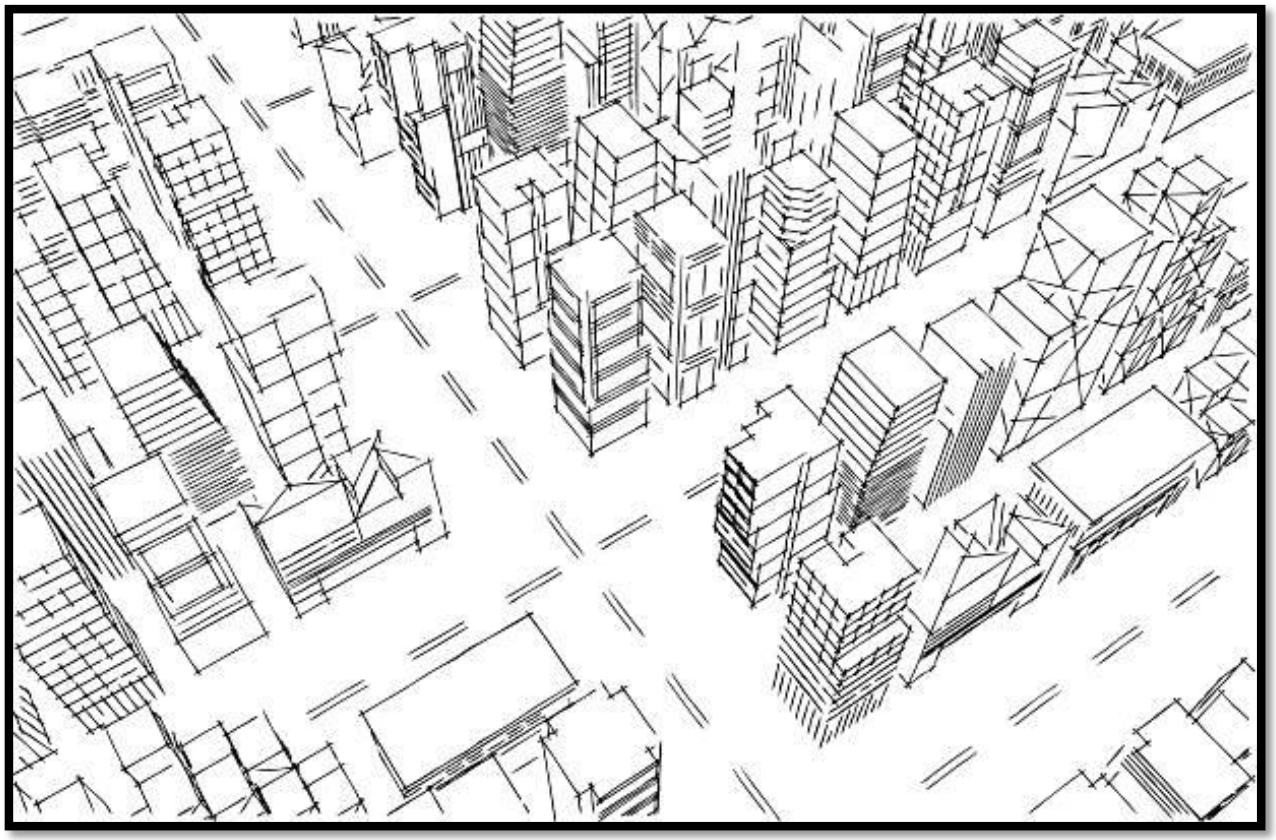

PS3
Vs
Wii
PLAYSTATION 3
SONY
Wii

2D graphics are essential and can be used as a stepping stone to moving toward the 3D realm for beginners. With the success 2D games enjoy today, it is important to discuss the topics that concern those types of applications since not all topics that apply to 2D games apply to 3D games and vice versa. Also, even in 3D games 2D elements are often used for user interfaces, menus elements, particles, and so forth.

Games utilizing **parallel projection** typically make use of two-dimensional bitmap graphics as opposed to 3D-rendered triangle-based geometry, allowing developers to create large, complex game worlds efficiently and with relatively few art assets by dividing the art into sprites or tiles and reusing them repeatedly (though some games use a mix of different techniques).


Object
View Plane

Top-down perspective, also sometimes referred to as bird's-eye view, Overworld, Godview, overhead view, or helicopter view, when used in the context of video games, refers to a camera angle that shows players and the areas around them from above. While not exclusive to video games utilizing parallel projection, it was at one time common among 2D role playing video games, wargames, and construction and management simulation games, such as SimCity, Pokémon, and Railroad Tycoon.



A side-scrolling game or side-scroller is a video game in which the viewpoint is taken from the side, and the onscreen characters generally can only move, to the left or right. Games of this type make use of scrolling computer display technology, and sometimes parallax scrolling to suggest added depth. In many games the screen follows the player character such that the player character is always positioned near the center of the screen.

In other games the position of the screen will change according to the player character's movement, such that the player character is off-center and more space is shown in front of the character than behind. Sometimes, the screen will scroll not only forward in the speed and direction of the player character's movement, but also backwards to previously visited parts of a stage. In other games or stages, the screen will only scroll forwards, not backwards, so that once a stage has been passed it can no longer be visited.

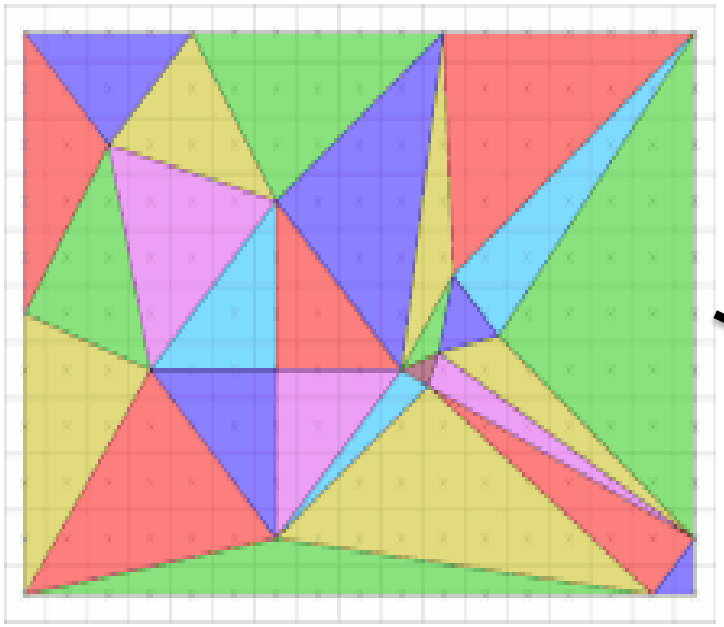
3D Game Graphics

Today's 3D games have evolved and now include not only 3D graphics but also 3D interactions through realistic physics, much more advanced and realistic artificial intelligence, and much more complex engineering designs, to name a few. Each of these areas are highly technical professions on their own. As computer processing power increases, so does our ability to create more complex gaming applications through many of these different areas.

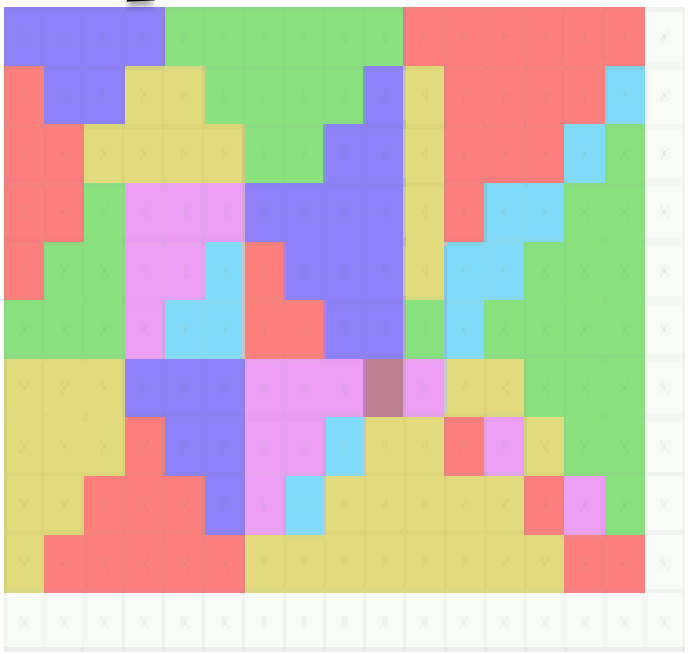
3D games offer a new level of realism. In 3D games, lighting and shadows can be simulated to appear more realistic, objects can have dynamic and complex animations that are physics based, reflections can appear on surfaces in real time, and the overall difference between interactive and no interactive entertainment, such as animated movies, is starting to de- crease greatly in terms of popularity and sales. The popularity of 3D games and the evolution of gaming as a whole have allowed video games to compete with all other entertainment industries, and they are worth billions.

Rasterization is the rendering technique used to display virtual scenes in real-time applications and is the technique commonly used in PCs, home consoles, and mobile video game systems. Rasterization is fast, whereas other techniques, such as ray tracing, are not as fast to process in real time using available hardware. Rasterization offers an efficient alternative for games.

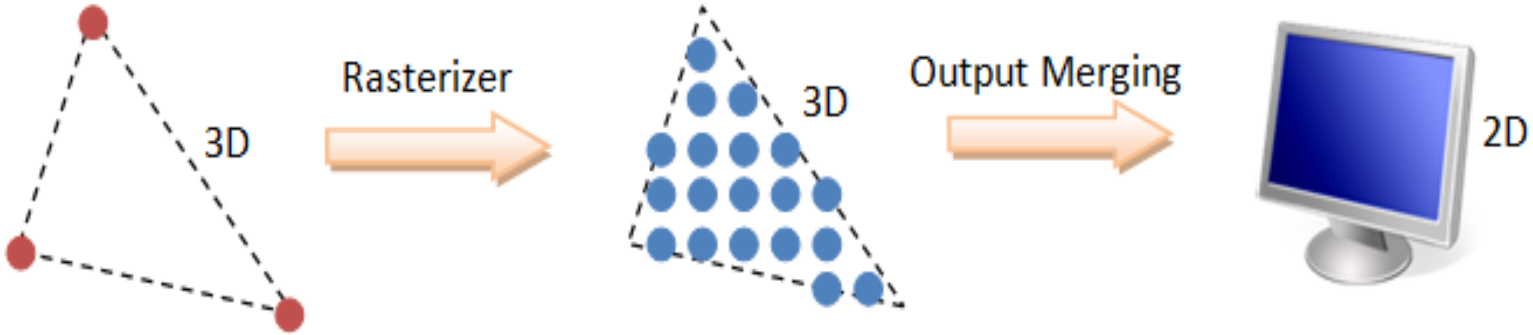
Rasterization is the task of taking an image described in a vector graphics format (shapes) and converting it into a raster image (a series of pixels, dots or lines, which, when displayed together, create the image which was represented via shapes). The rasterized image may then be displayed on a computer display, video display or printer, or stored in a bitmap file format. Rasterization may refer to the technique of drawing 3D models, or the conversion of 2D rendering primitives such as polygons, line segments into a rasterized format.



Rasterization



Rasterization works by taking a 3D scene made up of polygons and primitives and generating a 2D image out of it. Generally, a polygon in an application such as a video game describes a triangle made up of three points. A stream of triangles is passed to the graphics hardware, is transformed into 2D form, and is displayed to the screen. Transformations are done using various mathematical matrices.

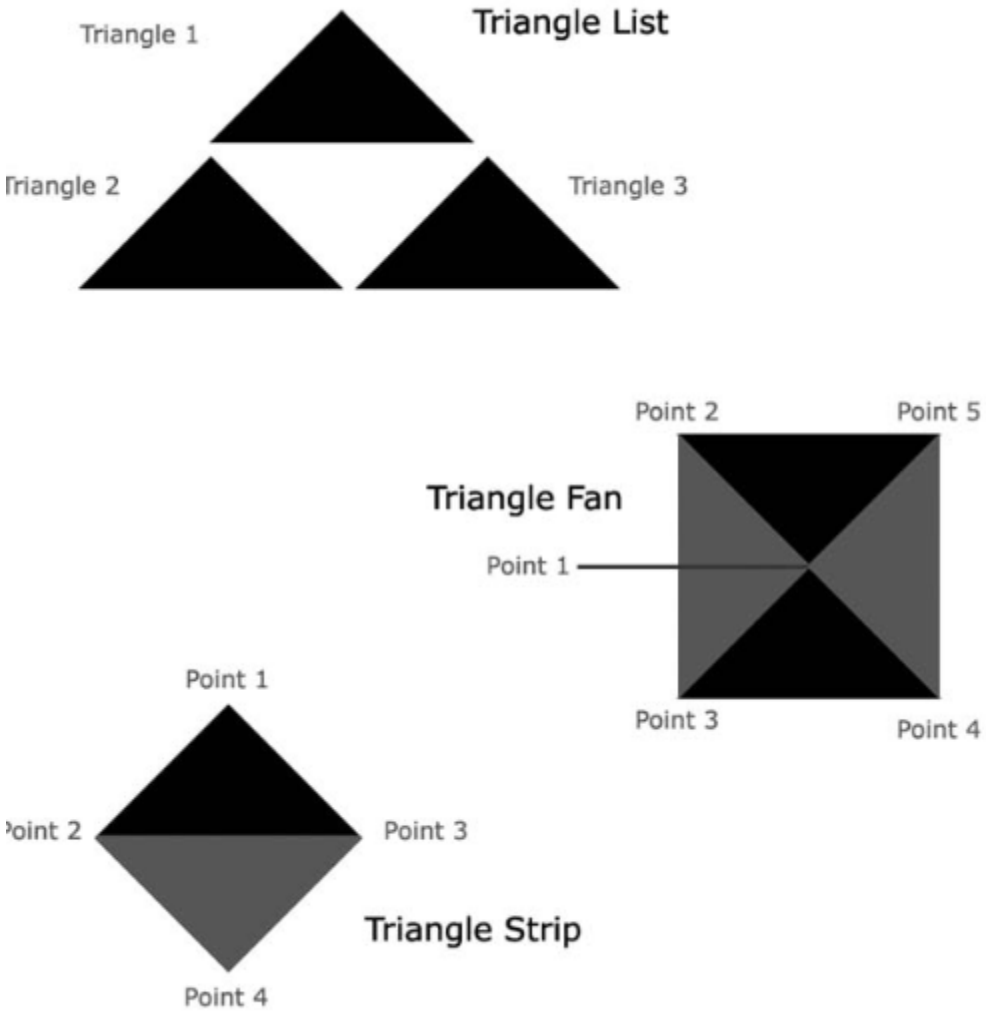


A *primitive* is formed by one or more *vertices*. Vertices are not grid-aligned

Grid-aligned *fragments* are interpolated from vertices.

All primitives are merged to produce 2D *pixels* on the display

The points of the polygons are known as vertices. Polygons that are transformed to 2D but are partially outside of the bounds of the screen are clipped to the rendering area. Clipping is a way to truncate polygons that are outside of the rendering area, which is often the window's screen, as shown. Once polygons have been transformed to 2D and have been clipped to the screen bounds, the color information is stored and the surfaces that make up the polygon are filled in line-by-line, where each line is known as a scan-line.



Various types of triangles.

Unclipped Polygon to the Viewport

Clipped Polygon to the Viewport

Clipping polygons.

A polygon is a geometric shape made up of three or more connected points. In games, polygons often refer to three-point polygons, also known as triangles. With rasterization, polygons are transformed and clipped, and then the volume that makes up the shape is filled in. This happens with all polygons, and to boost performance optimizations other techniques are usually used to prevent rendering of polygons that are not visible or are occluded by others pieces of geometry.

Fixed 3D refers to a three-dimensional representation of the game world where foreground objects (i.e. game characters) are typically rendered in real time against a static background. The principal advantage of this technique is its ability to display a high level of detail on minimal hardware. The main disadvantage is that the player's frame of reference remains fixed at all times, preventing players from examining or moving about the environment from multiple viewpoints.

Backgrounds in fixed 3D games tend to be pre-rendered two-dimensional images, but are sometimes rendered in real time (e.g. Blade Runner). Fixed 3D is also sometimes used to "fake" areas which are inaccessible to players. The Legend of Zelda: Ocarina of Time, for instance, is nearly completely 3D, but uses fixed 3D to represent many of the building interiors as well as one entire town (this technique was later dropped in favor of full-3D in the game's successor, The Legend of Zelda: Majora's Mask). A similar technique, the skybox, is used in many 3D games to represent distant background objects that are not worth rendering in real time.

First person refers to a graphical perspective rendered from the viewpoint of the player character. In many cases, this may be the viewpoint from the cockpit of a vehicle. Many different genres have made use of first-person perspectives, including adventure games, flight simulators, and the highly popular first-person shooter genre.

Games with a first-person perspective are usually avatar-based, wherein the game displays what the player's avatar would see with the avatar's own eyes. In many games, players cannot see the avatar's body, though they may be able to see the avatar's weapons or hands. This viewpoint is also frequently used to represent the perspective of a driver within a vehicle, as in flight and racing simulators; and it is common to make use of positional audio, where the volume of ambient sounds varies depending on their position with respect to the player's avatar.

Games with a first-person perspective do not require sophisticated animations for the player's avatar, and do not need to implement a manual or automated camera-control scheme as in third-person perspective. A first person perspective allows for easier aiming, since there is no representation of the avatar to block the player's view. However, the absence of an avatar can make it difficult to master the timing and distances required to jump between platforms, and may cause motion sickness in some players. Players have come to expect first-person games to accurately scale objects to appropriate sizes. However, key objects such as dropped items or levers may be exaggerated in order to improve their visibility.

Third person refers to a graphical perspective rendered from a view that is some distance away (usually behind and slightly above) from the player's character. This viewpoint allows players to see a more strongly characterized avatar, and is most common in action and action-adventure games. This viewpoint poses some difficulties, however, in that when the player turns or stands with his back to a wall, the camera may jerk or end up in awkward positions.


3RD PERSON VS 1ST PERSON

Developers have tried to alleviate this issue by implementing intelligent camera systems, or by giving the player control over the camera. There are three primary types of third-person camera systems: "fixed camera systems" in which the camera positions are set during the game creation; "tracking camera systems" in which the camera simply follows the player's character; and "interactive camera systems" that are under the player's control.

Shaders in Computer Graphics

Graphical application programming interfaces are used to talk directly to the graphics hardware attached to a computer or **gaming console**. The most popular two graphics APIs are Direct3D and OpenGL. In the past graphical APIs offered a set of algorithms and rendering states that a programmer could enable or disable any time in an application. This set of algorithms and rendering states is known as the **fixed-function pipeline**, and it provides developers a high-level point of access to the underlying hardware and features that are common in 3D video games.

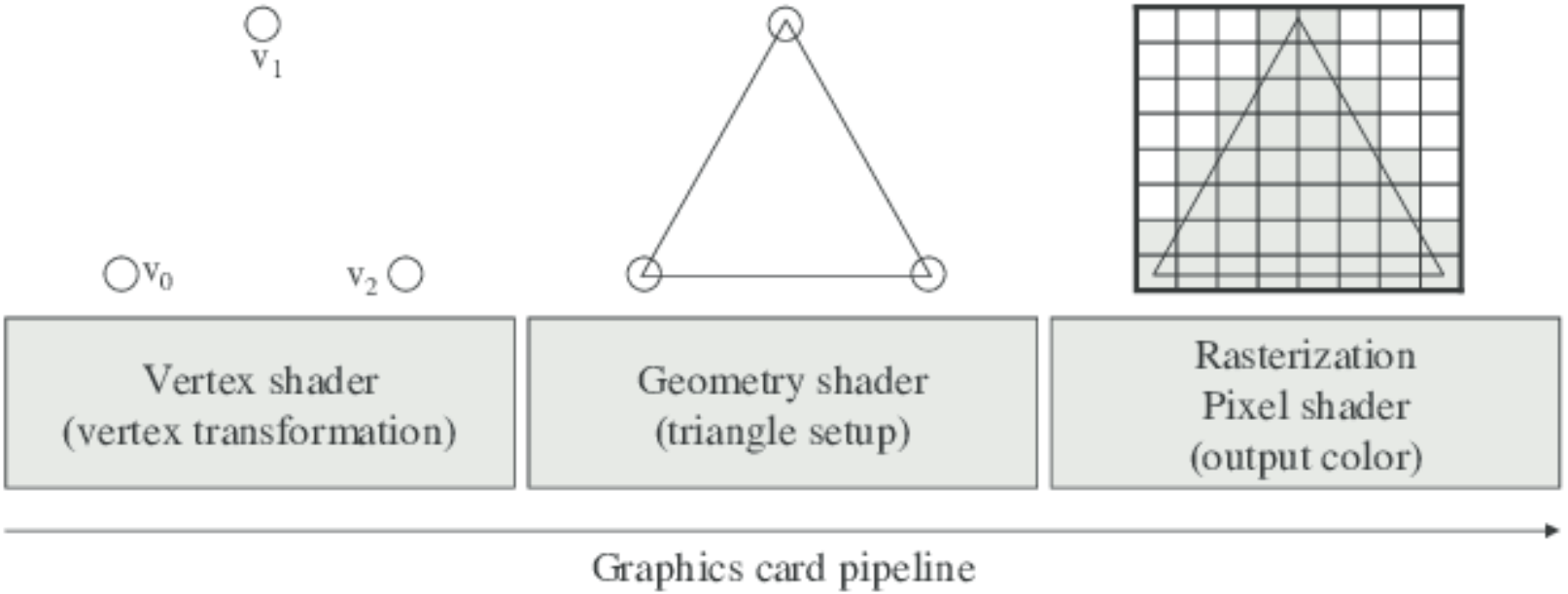
Although the fixed-function pipeline does offer some convenience when it comes to running specific algorithms that a particular API supports, its major downside is that it is restrictive in the number of features and the way developers talk to the hardware, and it lacks the customization of allowing developers to write their own algorithms and execute them in the graphics hardware.

Programmable shaders are the solution to the limitations and restrictions of a graphical API's fixed-function pipeline. Since the introduction of programmable hardware, the visuals seen in the games industry have grown in quality and complexity, among other things, by leaps and bounds. A **shader** is executable code that can be run on the graphics hardware. A **programmable shader** is a way for developers to write custom algorithms that can operate on the data that compose their virtual scenes. Shaders can be used to create just about any effect you can think of, which gives developers a high level of freedom and flexibility regardless of the API being utilized.

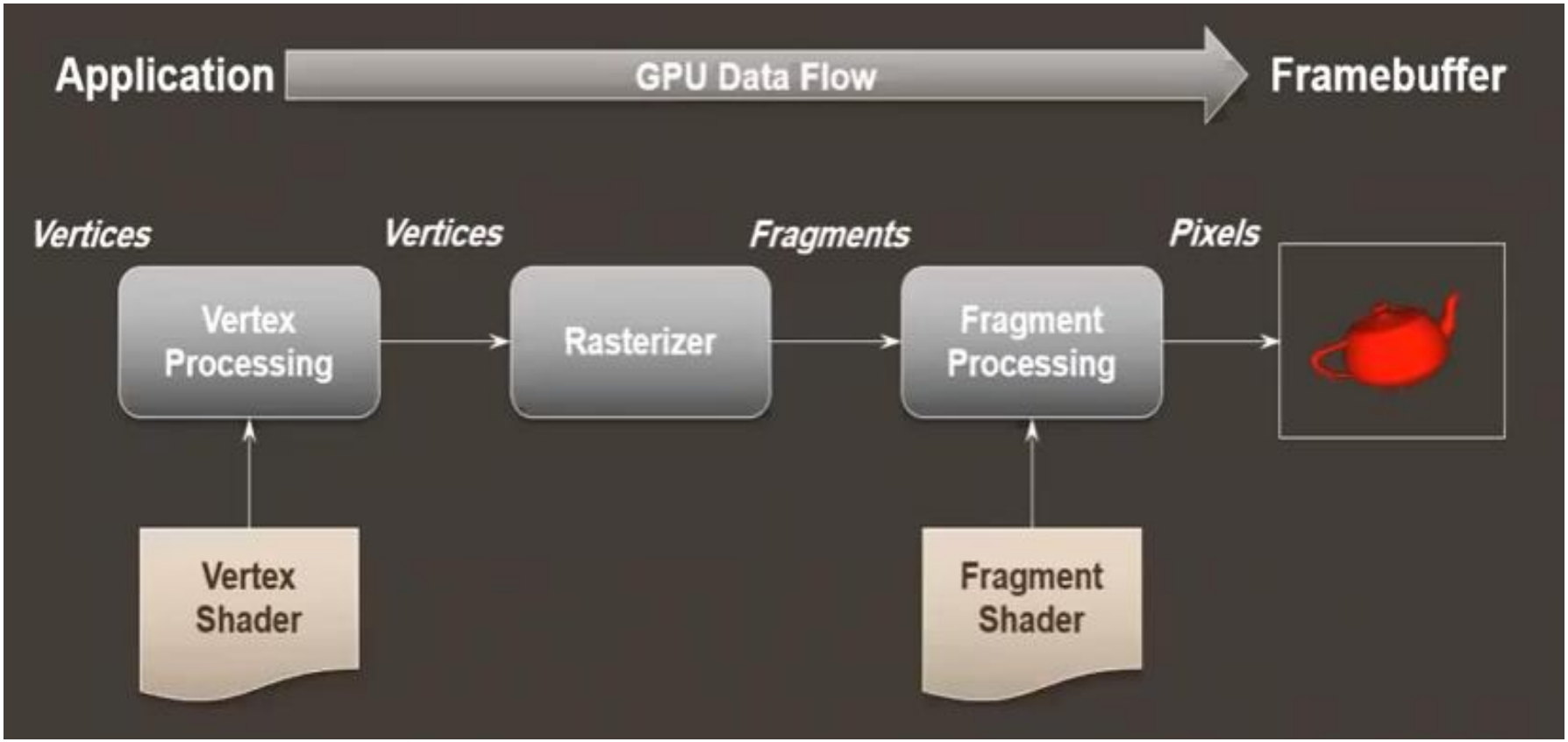
Today there are three different types of shaders that can be used to operate on the various pieces of information that compose a virtual scene. These shaders are vertex shaders, geometry shaders, and pixel shaders. When combined into one effect, a set of shaders is collectively called a shader program. Only one shader type can be active at a time. This means, for example, that it is not possible to enable two vertex shaders at the same time to operate on the same data.

Vertex shaders are code that is executed on each vertex that is passed to the rendering hardware. The input of a vertex shader comes from the application itself, whereas the other types of shaders receive their input from the shader that comes before it, excluding uniform and constant variables. Vertex shaders are often used to transform vertex positions using various matrices such as the model-view project matrix, and they are used to perform calculations that need to be performed once per vertex.

Geometry shaders sit between the vertex shader and the pixel shader. Once data have been operated on by the vertex shader, they are passed to the geometry shader, if one exists. Geometry shaders can be used to create new geometry and can operate on entire primitives. Geometry shaders can emit zero or more primitives, where emitting more than the incoming primitive generates new geometry and emitting zero primitives discards the original primitive that was passed to the geometry shader. Geometry shaders are a new type of shader that is available in Shader Model 4.0, which is currently supported by the Direct3D 10 and OpenGL 3.0 graphical APIs.


v1
v0
v2
Vertex shader
(vertex transformation)
Geometry shader
(triangle setup)
Rasterization
Pixel shader
(output color)
Graphics card pipeline

The third type of shader is the **pixel shader**, also known as the **fragment shader**. A pixel shader operates on each rasterized pixel that is displayed on the screen. "Rasterization," we saw that once a primitive has been transformed and clipped, the area that makes up the primitive is filled in (shaded). If you are using a pixel shader, each pixel that is shaded and that falls within the area of the primitive is operated on by the algorithm defined in the pixel shader. The input for the pixel shader can be either the output from the vertex shader or, if one exists, the output from the geometry shader.


Application
GPU Data Flow
Framebuffer
Vertices
Vertices
Fragments
Pixels
Vertex Processing
Rasterizer
Fragment Processing
Vertex Shader
Fragment Shader