Pipeline Process :

Stage 1:

The pipeline process starts with an "Application/Scene" stage, also known as the workload-reduction trick. This stage is devoted to deciding which particular object will be rendered in the three dimensional(3D) environment. The way that 3D environments are created is through a Cartesian Coordinate Systems (an x, y, and z axis) in which objects are placed to create a scene.

فرایند پایپ لاین :

مرحله اول : اولین مرحله روند Pipeline مرحله درخواست / نمایش می باشد. لازم به ذکر است که این تکنیک در حقیقت یک نوع فن اجرای دستورالعمل می باشد. این مرحله از Pipeline به فضای سه بعدی و رندر کردن آنها تمایل بیشتری دارد.

می دانیم که فضای سه بعدی از سه عنصر X , Y , Z تشکیل می شود که هر کدام از این اجزا برای ایجاد یک نمایش یا کاراکتر در فضای سه بعدی استفاده می شود.

The view space is determined by objects and angles depending on how the creator programmed the scene.

یک نمایش یا کاراکتر می تواند چندین زاویه برای دیدن یا View داشته باشد که هرکدام از آنها از یک نقطه مرجع ایجاد می شوند و فضای دید توسط سه عنصر و زوایا ، البته خلق کننده کاراکتر بر می گردد ، تعیین می شود.

The first process of the graphics pipeline is to only render and produce the images of the view space and to skip over unnecessary instructions that will not

be displayed even if it was processed. This system allows the graphics card to produce scenes and graphics efficiently.

اولین مرحله از پایپ لاین گرافیک ها فقط برای رندر کردن و تولید کردن Image از فضای دیداست. این سیستم و این قابلیت به کارت گرافیک اجازه می دهد تا صفحه هایی با کیفیت و کارایی بالا ایجاد کند.

Stage 2:

The second stage involves the scene's geometry. Objects mainly get moved from frame to frame to give an illusion that the object is moving in a real time setting. Objects can both be moved and manipulated in a scene depending on the application in which it is running. This manipulation of objects is generally called transformation. The objects can be stretched, skewed, moved or moved about an axis, or scaled differently. It is in the second stage of processing that the objects in the environment are altered.

مرحله دوم :

دومین مرحله از پایپ لاین در بر گیرنده علم هندسه یک صفحه می باشد که یکی از این تکنیک هایی که می تواند عناصر را در زمان واقعی حرکت دهد ایجاد فریم ها و حرکت فریم به فریم یک موضوع می باشد. همچنین بر حسب این که عملیات (پردازش ها) در صفحه به چه شکلی باشد می توانیم موضوع را با هم حرکت دهیم یا با هم دستکاری کنیم که به این دستکاری ها transformation می گویند. همچنین این عناصر می توانند با هم کشیده یا به صورت اریب حرکت داده شوند یا حرکت آنها بر حسب محور باشد.

Geometric lightning also occurs in the second stage after the objects are in their proper place, and once the figures receive their shape through the geometric transform process. Different types of lightning, depending on the application being run, will be processed to give the graphics a realistic appearance.

مرحله دوم از پردازش عناصر در یک محیط سه بعدی می تواند متناوب باشند. رعد و برق هندسی هم در مرحله دوم رخ می دهد که بعد از آن مکان عناصر که شاید و باید در یک مکان قرار گرفته شده اند و یکباره شکل از میان پردازش هندسی به اشکال دیگر در می آیند.

After the lighting is calculated the scene needs to get rid of unnecessary triangles that are only partially shown through the view space. This process is similar to the process which occurs in the first stage and also includes the process of “clipping.” “Clipping is the operation to discard only the parts of triangles that in some way partially or fully fall outside the view volume”

بعد از این که رعد و برق ها یا برق دادن ها محاسبه شدند یک نمایش در حقیقت نیازمند این است که مثلث هایی که زاویه هایی را بخشی از نمایش در فضای دید هستند را حذف کنیم. این روند در حقیقت شبیه به همان فرآیند در مرحله اول است که شامل فرآیند Clipping یا تکه تکه کردن است. Clipping به عملیاتی گفته می شود که مثلث هایی که در فضای سه بعدی نیاز نیست را حذف و برش می دهد تا حجم کار پائین بیاید.

Stage 3 :

The third stage implies an algorithm called the digital differential analyzer (DDA) which calculates the position of each part of all the triangles, and determines if the triangles are connected to other triangles. This process is done by computing the slope of each triangle’s edge in hope to improve the quality of the image being produced and by allowing more detailed information to be assigned to the triangles Sometimes when two triangles are touching, or even overlapping each other, a rough pixel “stair-step” occurs in which the edges between the two triangles create non realistic images of edges extruded surfaces that would not normally occur.

مرحله سوم:

سومین مرحله از پایپ لاین به الگوریتمی به نام DDA (Digital Differential Analyzer) دلالت می کند که این الگوریتم مکانی از هر بخش از کل مثلث ها را محاسبه کرده و تعیین می کند که این مثلث ها به سه گوش های دیگر متصل می باشند.

این فرایند توسط کامپیوتر انجام می شود و برای بهبود کیفیت تصویر لبه هر مثلث را شیب دارتر می کند که این را می توان توسط اطلاعات تعیین شده سه گوش ها انجام دهد. بعضی اوقات که دو مثلث با هم تماس داده می شوند یا حتی Overlap می شوند در این صورت پیکسل ها به صورت درشت و ضخیم نشان داده می دهند که هر کدام از این لبه ها بین دو مثلث ایجاد تصویر غیر واقعی می کند و لبه ها از قالب رویه بیرون می زند که می توان گفت یک تصویر غیر نرمال رخ داده است.

Another thing that occurs in the triangle setup phase is the assignment of color and depth values for each pixel. Since the edges of the triangles were calculated, the color and depth values may be interpolated using each triangle's vertex vales of color and depth. Along with the color and depth, the texture coordinates of each pixel is also interpolated in which they will be processed in the fourth stage.

مورد بعدی در مورد سه گوش ها ، وجه سه گوش است که تعیین کننده ی رنگ و عمق برای همه پیکسل ها می باشد. زمانی که لبه های یک مثلث محاسبه می شود مقادیر رنگ ها و عمق ها ممکن است داخل داده شود و با استفاده از مقدار گره هر لبه رنگ ها و عمق ها و بافت های تشکیل شده هر پیکسل شامل رنگ و عمق می باشد که این عملیات در حقیقت در مرحله چهارم از پایپ لاین انجام می شود.

Stage 4 :

The last stage of the pipeline is considered the rendering / rasterization stage. “To fill the frame buffer the drawing primitives are subdivided into pixels, a process known as scan-conversion or rasterization” (Schneider, Benyt-Olaf, pg 245). After all the processes of computing location, color, geometric values, etc., this last stage puts it all together and produces the 3D environment onto a 2D screen.

مرحله چهارم:

آخرین مرحله پایپ لاین شامل مرحله render و Raster می باشد. برای پر کردن بافر فریم بـه طراحـی های قبلی به زیر پیکسل هایی تقسیم می شود که به این روند راستر می گویند و بعد از این کـل فراینـد ها ، رنگ ها و مقادیر هندسی را محاسبه می کنند. این مرحله از پایپ لاین به این شکل عمل می کند کـه محیط سه بعدی را در صحنه دو بعدی قرار می دهد.

After the triangle setup is completed in the third stage, the next step is to provide shading values. Shading values are similar to the color and depth values contained in stage 3 and add the finishing touches on the scene.

بعد از این که تنظیمات مثلث ها در سومین مرحله به اتمام رسید مرحله بعدی در حقیقت مقادیر شـکل هاراآماده می کند. مقادیر سایه ها مانند رنگ ها و عمق ها در مرحله ۳ محاسبه و انجام می شـود و در آخـر در صفحه نمایش نشان داده می شود.

There are three common shading methods: Flat, Gouraud, and Phong shading.

سه تکنیک برای سایه زنی وجود دارد ، که هر کدام از آنها را توضیح می دهیم:

۱- تخت

۲- سایه های Phong
۳- کدویی

Flat Shading - operates per triangle and provides a quick render of the scene that does not involve extensive computations. This type of shading does not produce a high quality image.

۱- سایه های تخت : عملیات مثلث ها و رندر های سریع را در صفحه انجام می دهند ولی از محاسبات اضافی اجتناب می کنند. این نوع از سایه ها تصویری با کیفیت را نمی تواند ایجاد کند.

Phong Shading - operates per pixel and is the most computation demanding shading process compared to flat and Gouraud shading. Phong shading incorporates the Gouraud Shading idea of taking the average shading of the vertices and also implies its own process that includes other triangle's shading as well. This makes the object blend together easier for more complex designs and results in a higher quality image making it more realistic.

۲- سایه های Phong : عملیات این نوع سایه توسط پیکسل و همچنین توسط سایه هایی که درخواست می شود محاسبه می گردد. این فرایند به این شکل با فرایندهای تخت یا کدویی مقایسه می شوند. این کار با مخلوط کردن عناصر با هم طراحی را کامل کرده اما کیفیت تصویر واقعاً طبیعی و واقعی است.

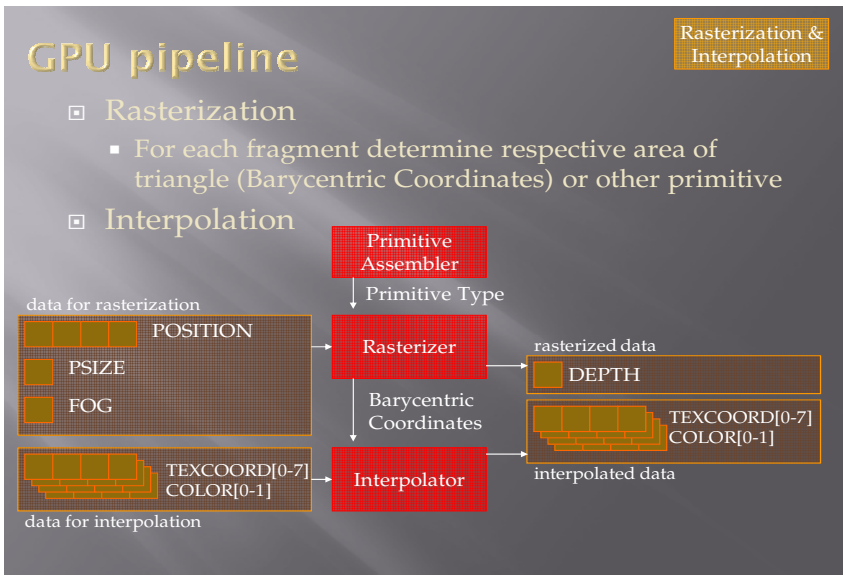
Gouraud Shading - operates per vertex of the triangles. Compared to flat shading, Gouraud shading produces a higher quality image while sacrificing render speed. Because of Gouraud shading takes the lighting values of each vertex of the triangle and interpolates the values across the surface of the triangle, the object will appear smother and not as rigid as flat shading.

۳- سایه های کدویی : عملیات بر حسب گره هر مثلث می باشد که مقایسه آن با سایه های تخت می تواند کیفیت بالای تصویر نام برد .

Raster Operations :

Inputs to this stage include the pixel locations and fragments depth and color values. This performs a series of tests on the fragment: scissor test, alpha test, stencil test, depth test. Then the fragment information is used to update the pixel's value according to the blend mode.

عملیات raster:



(شکل ۱۰)

ورودی ها در این بخش شامل موقعیت پیکسل ها و عمق قطعات و ارزش رنگی می باشند (شکل۱۰).
انجام این مرحله مشمول تست های زیر بر روی قطعه می باشد :

تست پراکندگی ، آلفا تست ، تست الگو ، تست عمق .

مطالب اصلی :

Computational functions :

Modern GPUs use most of their transistors to do calculations related to 3D computer graphics. They were initially used to accelerate the memory-intensive work of texture mapping and rendering polygons, later adding units to accelerate geometric calculations such as translating vertices into different coordinate systems. Recent developments in GPUs include support for programmable shaders which can manipulate vertices and textures with many of the same operations supported by CPUs, oversampling and interpolation techniques to reduce aliasing, and very high-precision color spaces.

توابع محاسباتی :

Gpu های جدید از ترانزیستورهایشان بیشتر برای محاسبه عملیات مربوط به گرافیک کامپیوتر ۳ بعدی استفاده می کنند. آنها در آغاز برای شتاب دادن به کار حافظه اصلی از نقشه بافت و اجرا چند ضلعی ها و بعدها برای شتاب دادن محاسبات هندسی از قبیل تعیین رئوس در سیستم های مختصاتی متفاوت استفاده کردند. توصعه های اخیر در Gpuها شامل
پشتیبانی برای سایه زن برنامه پذیر که می تواند رئوس وعملیات مشابه که با Cpu پشتیبانی می شود را ساده کرده و ماهرانه به کار ببرد .

Shader :

A shader is commonly said when referring to a custom program that resides on the graphics processing unit. A shader is a set of instructions that can be executed either for each vertex or each pixel. If it is for a vertex then it is called

a vertex shader, and if it for a pixel it is called a pixel shader or fragment shader.

سا یه زن :

یک سایه زن مجموعه ای از دستورات است که برای حتی هر راس یا هر بردار می تواند اجرا شود. اگر برای بردار باشد، سایه زن بردار و اگر برای پیکسل باشد سایه زن پیکسل یا سایه زن قطعه نامیده می شود.

Vertex shader :

Vertex shaders are being executed whenever a vertex is transformed. Most common transformations done using vertex shaders are surface deformations.

سایه زن بردار :

سایه زن های برداری هنگامی که یک بردار به وجود می آید اجرا می شوند. بیشتر تبدیلات رایج که از سایه زن های برداری استفاده می کنند تغییییر شکل ها را صاف می کنند.

Pixel Shader :

Pixel shader are being executed whenever a pixel is going to be rendered. They provide last opportunity to modify transformed, lit, and textured pixel color.

سایه زن پیکسل:

یک سایه زن پیکسل زمانی که یک پیکسل آماده اجرا (رندر شدن)است اجرا می شود. آنها آخرین فرصت را برای تغییر شکل روشنایی و رنگ بافت پیکسل تولید می کند.

Texture :

A texture (technically: texture map) is an image that is loaded from a file and applied to a surface in the 3D game world. Textures provide additional detail to what would otherwise be flat surfaces.

Texture :

یک بافت تصویری است که از یک فایل بار می شود و در یک فضا در جهان بازی های سه بعدی به کار می رود. بافت ها جزئیات اضافی را برای این که چطور سایر فضاها صاف شوند را تولید می کند.

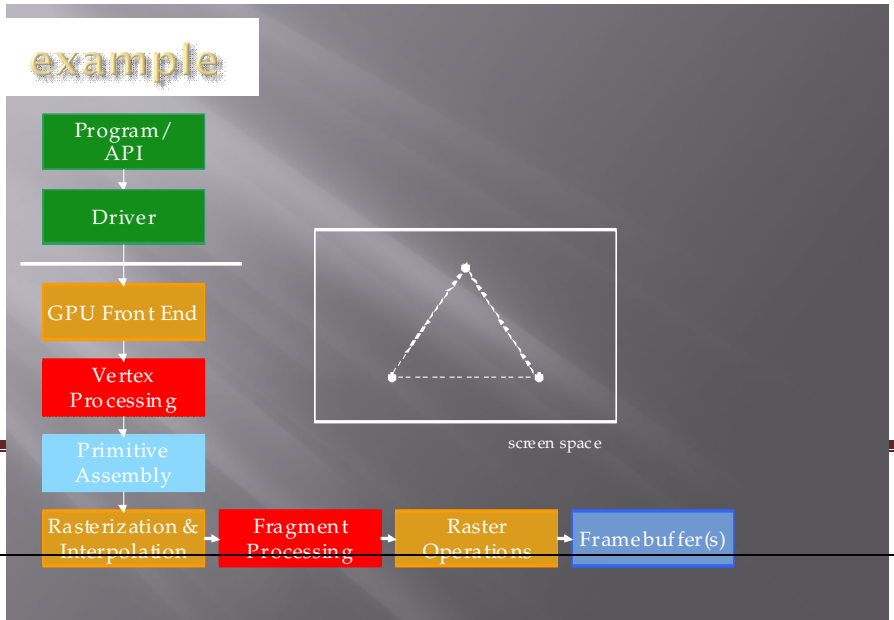
Normal Vector :

A normal vector is a vector that is perpendicular (i.e. points straight up out of a surface at a 90-degree angle) to something, such as a triangle or other surface. Normal vectors are very important for lighting calculations, as well as many other special effects, and for physics calculations. You can calculate the normal of a flat surface by taking any three points on the surface (for a triangle, these could be its three vertices) and then compute:

$$edge1 = point2 - point1$$
$$edge2 = point3 - point1$$
$$normal = edge1 \times edge2$$

Normal Vector :

یک بردار عادی برداری است که بر چیزهایی مانند مثلث یا سطوح دیگر عمود می شود. بردارهای نرمال برای محاسبات روشنایی و محاسبات فیزیکی به کار می روند. شما می توانید یک فضای صاف نرمال را با گرفتن سه نقطه در فضا محاسبه کنید.(این سه نقطه مطابق شکل ۱۱ سه راس برای یک مثلث هستند)



(شکل ۱۱)

پس محاسبه می کنید :

Edge1 = point 2 - point1

Edge2 = point3 - point1

Normal = edge1 * edge2

So in summary, you get the two edges that are formed by the three points, and take their cross product. The resulting vector will be the normal vector that is perpendicular to the surface that is formed by the three points.

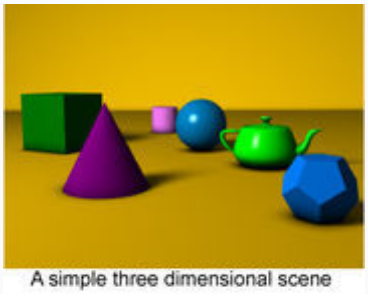
بنابراین به طور خلاصه دو لبه داریم که با سه نقطه شکل گرفته است و نرمال از ضرب خارجی آن ها را حاصل می گردد.

Z-buffer data :

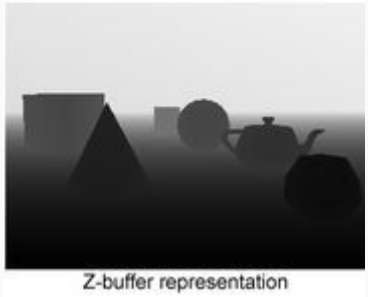
In computer graphics, z-buffering is the management of image depth coordinates in three-dimensional (3-D) graphics, usually done in hardware, sometimes in software. It is one solution to the visibility problem, which is the problem of deciding which elements of a rendered scene are visible, and which are hidden.

Z-Buffering :

در گرافیک رایانه ، z-buffering مدیریت مختصات عمق تصویر در گرافیک سه بعدی است. معمولا در سخت افزار وگهگاه در نرم افزار انجام می شوند. این یک راه حل برای روشن شدن مشکلات است ، مشکل تصمیم گیری که کدام المان ها از یک صفحه رندر شده ، روشن یا پنهان است. (شکل ۱۲)



A simple three dimensional scene



Z-buffer representation

(شکل ۱۲)

When an object is rendered by a 3D graphics card, the depth of a generated pixel (z coordinate) is stored in a buffer (the z-buffer or depth buffer). This buffer is usually arranged as a two-dimensional array (x-y) with one element for each screen pixel.

z-buffering همچنین با بافر عمق شناخته می شود. زمانی که یک کار توسط کارت گرافیک سه بعدی ترجمه می شود عمق یک پیکسل تولید شده در یک بافر ذخیره می شود (بافر عمق یا z-buffering). این بافر معمولا مانند یک آرایه دو بعدی (x , y) با یک المان برای هر پیکسل نمایش مرتب می شود.

If another object of the scene must be rendered in the same pixel, the graphics card compares the two depths and chooses the one closer to the observer. The chosen depth is then saved to the z-buffer, replacing the old one. In the end, the z-buffer will allow the graphics card to correctly reproduce the usual depth perception: a close object hides a farther one. This is called z-culling.

اگر هدف دیگری از صحنه در پیکسل مشابه رندر شده باشد کارت گرافیک دو عمق را مقایسه می کند و آنی را انتخاب می کند که به مشاهده کننده نزدیکتر است. سپس عمق انتخاب شده در z-buffering به جای نوع قدیمی ذخیره می شود. در پایان z-buffering به کارت گرافیک برای تولید مجدد از عمقی که به درستی درک کرده اجازه خواهد داد. یک هدف نزدیک هدف دورتر را پنهان می کند.این کار z-buffering نامیده می شود.