

**شبیه سازی بازدید از یک موزه با طراحی یک محیط گرافیکی**

**پروژه دوره کارشناسي**

**در رشته کامپیوتر گرایش نرم افزار**

**ارائه شده به:**

**دانشکده فنی مهندسی**

**موسسه آموزش عالی نبی اکرم (ص)**

**استاد راهنما:**

**آقای مهندس سعدالله سبحانی**

**توسط:**

**سامان حبیبی**

**نام ماه و دو رقم آخر سال ارائه :**

**بهمن 90**



**چکیده**

شبيه سازي علم و هنر ساختن نمايشي (مدلي) از يک پروسه يا سيستم  ، به منظور ارزيابي و آزمايش راهبردها مي باشد، ويا شبيه سازي روشي براي آگاهي از نتايج ايده هاي پيشنهادي قبل از اجراي آنها

به روشی ساده تر است .

در این شبیه سازی با استفاده از کتابخانه گرافیکی DirectX یک محیط گرافیکی ارایه خواهد شد که

بازدید از یک موزه را شبیه سازی میکند در این محیط از تابلوها و اجسام سه بعدی مثل میز و صندلی و

پخش موزیک ملایم همراه با صدای قدم های شخص هم وجود خواهد داشت.

حال میتوان این سوال را پرسید که

چه زمان **بايد** از شبيه سازي استفاده کرد : شرايطي که تجزيه تحليل جبري ميسر نمي باشد :

* سيستم هاي  غير قطعي
* سيستم هاي  پويا
* سيستم هاي  پيچيده

شرايطي که امکان آزمايش در دنياي واقعي وجود ندارد :

* سيستم هنوز ايجاد نشده است.
* ريسک هاي زياد و خطرناکي وجود دارد.
* هزينه آزمايش بالاست.

**فهرست مطالب**

فصل1 مقدمه ....................................................................................................................................................... 5

فصل 2 معرفی ابزارها و امکانات استفاده شده ................................................................................................. 7

2-1 زبان برنامه نویسی C# ................................................................................................................................ 7

2-2 Autodesk 3DS Max ........................................................................................................................ 8

فصل 3 تعریف و بیان مفاهیم کلی .................................................................................................................... 9

3-1 سه دیدگاه ...................................................................................................................................................... 9

3-2 تعریفی که Microsoft از DirectX ارایه میکند ........................................................................................... 10

3-3 اجزای اصلی DirectX............................................................................................................................. 11

3-4 پیکر بندی کارت گرافیکی .......................................................................................................................... 11

3-5 آشنایی با اصطلاح Real-time ................................................................................................................ 15

3-6 فیلتر ها و انواع بافر ........................................................................................................................................ 17

تاریخچه .......................................................................................................................................................................22

ضمیمه ....................................................................................................................................................................... 21

فصل 5 جمع بندی و نتیجه گیری ...................................................................................................................... 27

فهرست منابع ............................................................................................................................................................ 28

پیوست ....................................................................................................................................................................... 29

**فصل 1**

**مقدمه**

اولین کامپیوترها شامل سطرهای بسیاری از چراغها و کلیدها بودند. متخصصین فنی و مهندسین برای ساعتها و روزها و هفته ها کار میکردند تا این ماشینها را برنامه ریزی کنند و نتایج محاسباتشان را بخوانند. الگوی روشن سازی چراغها اطلاعات مفیدی را نصیب کاربران کامپیوتر کرد. ممکن است شما بگویید اولین چیز در گرافیک کامپیوتری پانلی از لامپهای چشمک زن بود که این عقیده درست است.

اما زمان تغییر کرد. از آن دستگاههایی که ماشین تفکر نامیده بودندش دستگاههای قابل برنامه نویسی جدیدی پدید آمد که میتوانستند بر روی لوله کاغذ با مکانیزمی شبیه ماشین های تحریر از راه دور چاپ کنند. داده ها باید بصورت موثر بر روی نوارهای مغناطیسی یا دیسک یا سطرهای کاغذ بشکل پانچ شده نگهداری میشدند. به علت این که هر کاراکتری از حروف الفبا اندازه و شکل ثابتی داشت برنامه نویسان خلاق بسیار خوشحال شدند چون میتوانستند تصاویری خلق کنند که تنها از علامت ستاره تشکیل میشد.

**CRT** وارد می شود :

کاغذ به عنوان یک خروجی متوسط برای کامپیوترها مفید است و استفاده از آن تا امروز هم تداوم یافته پرینتر های لیزری و جوهر افشان امروزه هنر چاپ حروف اسکی را بخاطر کیفیت بالا و نزدیک به واقیت شان از بین برده است.بهر حال مصرف کاغذ میتواند ولخرجی در منابع طبیعی باشد مخصوصا به این دلیل که در بیشتر اوقات ما نیاز به یک کپی چاپی از محاسبات اطلاعاتی مان نداریم.

لامپ اشعه کاتدی یک افزودنی لازم برای کامپیوترها بود. مانیتور های CRT اولیه در آغاز تنها خروجی هایی برای تصاویری بودند که تنها حروف اسکی را نمایش می داد. اما CRT ها بطور کلی توانایی بالای در رسم نقاط و خطوط داشتند درست مانند کارکترهای حروف ، به زودی نشانه ها و تصاویر دیگری شروع به اضافه شدن به مجموعه کاراکترهای خروجی شدند. برنامه نویسان از کامپیوترها و مانیتورهایشان برای خلق تصاویری که به متون و جدول ها ضمیمه میشدند استفاده کردند. اولین الگوریتم ها برای خلق خطوط و منحنی ها توسعه یافت و منتشر شد. گرافیک کامپیوتری به چیزی بیشتر از سرگرمی تبدیل شد ، اولین گرافیک های کامپیوتری که بر روی این خروجی ها نمایش داده میشد دو بعدی بودند. این خطوط و دایره ها و چندضلعی ها برای طراحی گرافیکی برای مقاصد گوناگون استفاده میشدند. نمودارها و رسم ها میتوانستند اطلاعات علمی و آماری را نمایش دهند ، به صورتی که جداول و اشکال نمیتوانستند. بسیاری از برنامه نویسان ماجراجو توانستند بازیهای ساده ای نظیر Lunar Lander و یا Pong را طراحی کنند که با خطوط ساده ای طراحی شده بودند که چندین بار در ثانیه ریفرش می شد ند .

**فصل 2**

**معرفی ابزارها و امکانات و اصطلاحات استفاده شده**

**2-1 زبان برنامه نویسی C# 5**

سی شارپ یک زبان برنامه نویسی سطح بالاست . این زبا بسیار شبیه به زبان های java , c++ می باشد .

با توجه به نقش محوری این زبان از آن به عنوان مادر زبنن های برنامه نویسی در دات نت نام برده می شود.

انجمن تولید کنندگان کامپیوتر اروپا (ECMA) زبان C# در سوم اکتبر سال 2001 به عنوان یک استانداردپذیرفته (ECMA-334) و به دنبال ان تلاششهای وسیعی برای کسب گواهی ISO نیز انجام شده است .

از جمله قابلیت های ورژن 5 سی شارپ می توان به مدیرت بهینه پردازش های موازی (Multi-Thread)

و مدیریت منابع بدون آدرس (زباله روبی) توسط (CLR) دات نت اشاره کرد .



ایجاد پروژه برای برنامه نویسی در C# شکل 2-1

**2-2 Autodesk 3DS Max**

نرم افزار [**3DS Max**](http://mihandownload.com/cat_39/software_engineering/)  از جمله نرم افزار هاي مشهور و بسيار پيشرفته طراحي و انيميشن سازي مي

باشد. اين برنامه ساخت کمپاني معروف Autodesk بوده که قطعآ ساير نرم افزارهاي معروف اين

شرکت مانند اتوکد را مي شناسيد. شما با استفاده اين نرم افزار به راحتي مي توانيد از طرح هاي ساده

ي گرافيکي خود يک مدل سه بعدي بسيار زيبا بسازيد. اين برنامه به دليل داشتن ابزارهاي بسيار

پيشرفته در ساخت بازي هاي رايانه اي و همچنين طراحي و خلق جلوه هاي ويژه در فيلم ها و بسياري

از پروژه هاي عظيم ديگر که به خلق يک مدل سه بعدي حرفه اي نياز است استفاده مي شود. همچنين

از اين برنامه مي توان براي ساخت انيميشن، ميکس و تدوين و متحرک سازي نيز بهره گرفت. در واقع

تمامي ابزارهايي که شما براي ايجاد تصاوير شفاف و با کيفيت جهت شبيه سازي مدل هايي همچون مو،

لباس و ساير قسمت ها نياز داريد در اين برنامه قرار دارد. از ديگر امکانات اين نرم افزار میتوان به امکان

ساخت و طراحي نماي سه بعدي از يک ساختمان اشاره نمود در واقع مهندسان رشته ي عمران به

راحتي ميتوانند نقشه ي يک ساختمان را که با ديگر نرم افزار اين شرکت يعني اتوکد طراحي کرده اند

وارد نرم افزار [3DS Max](http://mihandownload.com/cat_39/) کنند و سپس اقدام به طراحي مدل سه بعدي از ساختمان مورد نظر خود

نمايند. يکي ديگر از قابليت هاي اين برنامه طراحي ماتريال ها يا همان اجسام مشبک با وضوح بسيار

بالا با استفاده از نگاشتهاي جا به جايي می باشد. علاوه بر این با استفاده از ويراشگر منحصر به فرد

اجسام مي توان مدل هايي با انواع بافت ها و اندازه هاي واقعي و پيچيدگي زياد به دست آورد.

**فصل سوم**

**تعاریف و بیان مفاهیم کلی**

**3-1 سه دیدگاه :**

تعریف جامعی از DirectX وجود دارد که هر کدام از دیدگاه خاصی آن را تعریف می کنند . .( شکل 1)

**Hardware**

**DirectX**

**Windows**

**Applications**

شکل 3-1

**دیدگاه Developer** :

DX به عنوان مجموعه ای از API Application Programing Interface های گرافیکی است که شامل توابع و روال هایی از پیش تعریف شده است که می توان آنها را در برنامه نویسی به کار گرفت .

**دیدگاه سیستم عاملی** :

DX مجموعه ای است از توابع و روال های RunTime سیستم عامل در قالب فایل های .dll و .lib که در زمان اجرا در اختیار Application قرار می گیرد .

**دیدگاه سخت افزاری :**

DX مجموعه ای از توابع Low-Level میباشد که برای اجرا شدن احتیاج به سخت افزار شتاب دهنده گرافیکی یا Accelerator Graphics مناسب با توان اجرای دستورات گرافیکی پایه را دارد .

**3-2 تعریفی که Microsoft از DirectX ارایه میکند :**

**DirectX**

**Microsoft DirectX is a collection of industry-leading graphics technologies designed to deliver the most advanced, stable, and visually impressive graphics experience on Microsoft platforms.**

**DirectX is comprised of best-in-breed runtime technologies, developer tools,**

**graphics programming interfaces and languages. The DirectX standard is supported by a wide variety of consumer and workstation-class graphics software and hardware solutions.**

مجموعه ای است از برترین تکنولوژی های گرافیکی که با هدف ارتقا سطح کیفی و پایداری و تاثیر گذاری

تصاویر و کارهای گرافیکی در پلتفرم های مبتنی بر محصولاتMicrosoft طراحی و ایجاد شده است .

DirectX شامل بهترین توابع و تکنولوژی های زمان اجرا ، ابزارهایی برای سازندگان و طراحان و همینطور API های گرافیکی برای برنامه نویسان گرافیکی می باشد .

**3-3 اجزای اصلی DirectX :**

DirectX 10.1 جدیدترین ورژن از DX در حال حاضر می باشدو دارای 7 مولفه ی اصلی Direct3D ,DirectDraw ,DirectPlay, DirectShow ,DirectMusic , DirectSound ,DirectInput می باشد .

**3-4 پیکر بندی کارت گرافیکی**

بحث خواهد شد . Device در این بخش نحوه پیکربندی کارت گرافیکی و راه اندازی یک

رابط بین برنامه نویس و کارت گرافیکی متصل به کامپیوترش است . این شی به برنامه نویس Device

امکان استفاده از توانایی های سخت افزاری کارت گرافیکی اش را می دهد . جهت پیکر بندی کارت راDirectX SDKگرافیکی وساخت بازی های سه بعدی توسط سی شارپ قبل از هر چیزابتدا باید

بر روی کامپیوتر نصب کرده و در محیط ویژوال استدیو رفرنس هایDirectX را به پروژه اضافه کنید .

با افزودن این رفرنس ها تمام کلاس های DirectX در محیط برنامه نویسی سی شارپ قابل استفاده خواهد بود . برای انجام این کار کافیست پس از نصب نسخه ی مناسبی از DirectX در

VisualStudio.Net یک پروژه ی جدید ایجاد کرده ، در پنجره ی solutionExplorer بر روی برگه ی رفرنس کلیک راست کرده و گزینه ی AddReference را انتخاب کنیم.(شکل 3-1)



شکل 3-2

با انجام این کار پنجره ی کوچکی باز می شود که لیست تمام مولفه های نرم افزارهای مختلف در آن نشان داده می شود . از این لیست باید مولفه های Microsoft.DirectXو Microsoft.DirectX.Direct3D

را به پروژه ی خود اضافه کنید . البته در برنامه هایی که در بخش های بعد خواهیم نوشت مولفه های دیگر از قبیل DirectSound ، Direct3DX و DirectInput نیز باید به پروژه اضافه گردد.



لیست مولفه های DirectX در پنجره ی Add Reference شکل 3-3

بعد از افزودن رفرنس های DirectX به پروژه ، دو دستور زیر باید به ابتدای تمام فایل هایی که از کلاس های DirectX استفاده می کند درج شود .

Using Microsoft.DirectX ;

Using Microsoft.DirectX.Direct3D;

پس از افزودن دستورات using گام بعدی اتصال به کارت گرافیکی وپیکربندی آن است برای این منظور از دستورات زیر استفاده می کنیم .

**class InitializeGraphics**

**{**

**private PresentParameters pp;**

**public InitializeGraphics()**

**{**

**pp = new PresentParameters();**

**pp.Windowed = true;**

**pp.SwapEffect = SwapEffect.Discard;**

**pp.AutoDepthStencilFormat = DepthFormat.D16;**

**pp.EnableAutoDepthStencil = true;**

**}**

**public Device getDevice(Form frm)**

**{**

**Caps DevCaps = Manager.GetDeviceCaps(0, DeviceType.Hardware);**

**DeviceType DevType = DeviceType.Reference;**

**CreateFlags DevFlags = CreateFlags.SoftwareVertexProcessing;**

**if (DevCaps.PixelShaderVersion >= new Version(2, 0))**

**{**

**DevType = DeviceType.Hardware;**

**if (DevCaps.DeviceCaps.SupportsHardwareTransformAndLight)**

**{**

**DevFlags = CreateFlags.HardwareVertexProcessing;**

**if (DevCaps.DeviceCaps.SupportsPureDevice)**

**{**

**DevFlags |= CreateFlags.PureDevice;**

**}**

**}**

**}**

**Device device3d = new Device(0, DevType, frm, DevFlags, pp);**

**return (device3d);**

**}**

**}**

**3-5 آشنایی با اصطلاح Real-time**

اصطلاح Real-time اولین بار به تصاویری که متحرک بودند اطلاق شد. یک استفاده گسترده از کلمه در

علم کامپیوتر که به این معناست که کامپیوترها میتوانند اطلاعات را با همان سرعتی که وارد میشوند و حتی

سریعتر پردازش نمایند. برای مثال تلفن زدن یک عمل real-time است که انسانها در آن شرکت دارند.

شما صحبت میکنید و شنونده فورا صدای شما را میشنود و به آن واکنش نشان میدهد. و به شما این اجازه

را میدهد که حرفهای او را شنیده و واکنش نشان دهید و همینطور الی آخر. در حقیقت یک تاخیری به

علت مسائل الکترونیکی وجود دارد اما این تاخیر معمولا برای کسانی که مشغول مکالمه هستند جزیی

می باشد. در مقام مقایسه نوشتن نامه یک عمل real-time نیست.

بکارگیری اصطلاح real-time برای گرافیک کامپیوتری بدین معناست که کامپیوتر در حال ارائه کردن یک

انیمیشن یا رشته ای از تصاویر است که بی درنگ به بعضی از ورودی ها واکنش نشان میدهد. مانند حرکت

دادن جوی استیک و یا ضربه زدن به صفحه کلید و الی آخر. گرافیک کامپیوتری بلادرنگ

Real-time می تواند خروجی های قابل خواندن عددی یا بازیهای اینتراکتیو و شبیه سازیهای مجازی را

نمایش دهد.

پیش به سوی سه بعدی : اصطلاح سه بعدی یا 3D به این معناست که یک شئ در حال نمایش سه بعد

قابل اندازه گیری دارد طول و عرض و عمق. یک مثال برای یک شئ دو بعدی یک قطعه کاغذ است بر روی

میزتان که عمق (ضخامت) چندانی ندارد. و مثالی برای یک شئ سه بعدی یک قوطی کنسرو است که

پیرامون آن طول و عرضش را نشان میدهد و بلندی آن نمایشگر عمق است. بسته به پرسپکتیو شما میتوانید

تعیین کنید که کدام طرف قوطی طول و یا عرض باشد اما در نهایت قوطی سه بعد دارد.

گرافیک سه بعدی غیر همزمان : Non-Real-Time :

برای برنامه هایی که از گرافیک سه بعدی بلادرنگ استفاده میکنند قانونی و.جود دارد. با دادن فرصت

بیشتری برای پردازش تصاویر شما میتوانید گرافیک های سه بعدی با کیفیت بالاتری ایجاد نمایید. به طور

مثال بعضی از نرم افزارهای مدل سازی از گرافیک سه بعدی بلادرنگ برای تقابل با هنرمند برای خلق

محتوای مورد نظرش استفاده میکنند. سپس تصاویر به برنامه دیگری فرستاده میشوند (ray tracer) که

تصاویر را رندر میکنند. رندر کردن یک فریم تنها برای انیمیشنی مانند داستان اسباب بازی به ساعتها زمان

بر روی یک کامپیوتر سریع نیاز دارد. این پروسه رندر و ذخیره سازی صدها فریم یک انیمیشن را میسازد که

بطور رشته متوالی قابل پخش مجدد میباشد. اگرچه پخش تصاویر انیمیشن ممکن است یک عمل بلادرنگ

به نظر برسد اما اینطور نیست. چون آن اینتراکتیو نیست در نتیجه آن بلادرنگ نیست بلکه بیشتر یک سری

تصاویر از پیش رندر شده میباشد.

با پیشرفت چشم گیر تنکنولوژی در صنعت کارت های گرافیک امکانات فوق العاده ای برای تنظیم کیفیت عکس ها ایجاد شده که همگی در بهبود کیفیت تصاویر و جلوه های بازی های رایانه ای کمک شایانی کرده اند . حال در این مقاله قصد توضیح و معرفی تکنیک های مختلف استفاده شده در این بازی های سه بعدی

داریم که شما را در ایجاد تصاویر بهتر گرافیکی یاری می کند . البته لازم به ذکر است که بیشتر این تنظیمات بسته به قدرت کارت گرافیک شما داره و باید بسته به قدرت آن این تنظیمات را انجام دهید تا سرعت کاهش پیدا نکند .

**3-6 فیلتر ها و انواع بافر**

**Anti-aliasing**  : این تکنیک که به اختصار AA گفته می شود برای نرم و هموار کردن زاویه های تیزی که

خط های مورب ایجاد می کنند ، استفاده می شود که معمولا در تنظیمات 3d تصویر وجود داره و معمولا

مقدار قابل تنظیم آن ‌ Disable فعال نبودن AA ) " ، 2X ، 4X یا " (None: 6X است و در برخی

موارد در بازی های رایانه ای به جای مقدار آن از Low پایین ، Medium متوسط و High بالا

استفاده می شود . هر چه مقدار AA بالاتر باشد کیفیت تصاویر بهتر و در عوض سرعت کند تر می شود که

اندازه این کندی بسته به کارت گرافیک شما متغیر است .

**Bilinear Filtering** : این تکنیک بهینه کننده برای نرم تر کردن مکان بر خورد رنگ های مختلف است . برای

مثال زمانی که شما تغییر رنگ مکان زرد رنگ به قرمز رنگ را دارید این $$$$$ینگ با ایجاد یک ناحیه

تغییر رنگ ، ناحیه برخورد این دو رنگ را صاف تر و نرم تر می کند .

**Trilinear filtering** : این تکنیک تقریبا مشابه " Bilinear Filtering " کار می کند با این تفاوت که این

تکنیک قوی تر است اما سرعت را کند تر می کند .

**Anisotropic filtering** : این تکنیک برای بالا بردن کیفیت تصاویری است که طرح قراردادی ندارند و در

کل برای واضح تر کردن و نمایان تر کردن تصاویر و اجزای آن هاست . برای مثال بعضی اوقات شده که شما

در بازی ها شاهد حروف و جمله هایی هستید که در هنگامی جریانی در پشت آن در حال اجراست تکان می

خورند اصلا واضح نیستند و نمایش بهتر اجزا و تیره نشدن آن ها . معمولا این تنظیم با مقدار 2X ، 4X ،‌8X

و16X مشخص می شود و هر چه این مقدار بیشتر شود تصویر واضح تری خواهید داشت اما سرعت کند تر

خواهد شد .

این تکنیک باعث می شود زمانی که یک تکستچر در سراسر یک محیط چندین بار تکرار **:MIP Mapping**

می شود با سایز های کوچک تر از کیفیت تصویر کم نشود . زیرا زمانی که یک تکستچر باید کوچک تر شود

باعث له شدن تکستچر می شود تا آن را بتواند در شی مورد نظر قرار

باعث له شدن تکستچر می شود تا آن را بتواند در شی مورد نظر قرار " squeeze " با انجام عمل VGA

دهد . که این کار باعث ایجاد نویز های تصویری و تصاویری رنج آور و زشت می شود . در حقیقتMIP

است . Anti-Aliasing یک تکنیک از نوع Mapping

**: Z-buffering**

به طور عادی حافظه گرافیکی رنگ های مربوط به هر پیکسل را در خودشان ذخیره می کنند .

ها Z حافظه کارت گرافیک با ذخیره کردن موقعیت ها در محور Z-Buffering اما با استفاده از

در حال شناختن و تعبیه کردن مکان VGA( عمق تصویر ) سرعت رندر کردن تصاویر را در زمانی که

ها نسبت به هم Object های صفحه یا به عبارت واضح تر در حال شناختن موقعیت Object

Z-Buffering و چگونگی پشت هم قرار گرفتنشان است ، بیشتر می کند . و بر عکس در صورتی که از

و طرز قرار گرفتشان Object استفاده نشود محاسبات زیادی برای پیدا کردن موقعیت دقیق و درست

نسبت به یکدیگر صرف کند .



شکل 2-2

بازی های سه بعدی معمولا حافظه کارت گرافیک را به دو قسمت تقسیم می کنند . با **: Frame buffer**

استفاده از این تکنیک زمانی که یک تصویر در حال نمایش است تصویر بعدی در قسمت دیگر حافظه در

است . البته بعضی از کارت های گرافیکی Double Buffering حال ترسیم است . نام دیگر این تکنیک

Triple Frame Buffer اجازه تقسیم به سه قیمت را نیز می دهند که این تکنیک در آن ها با نام

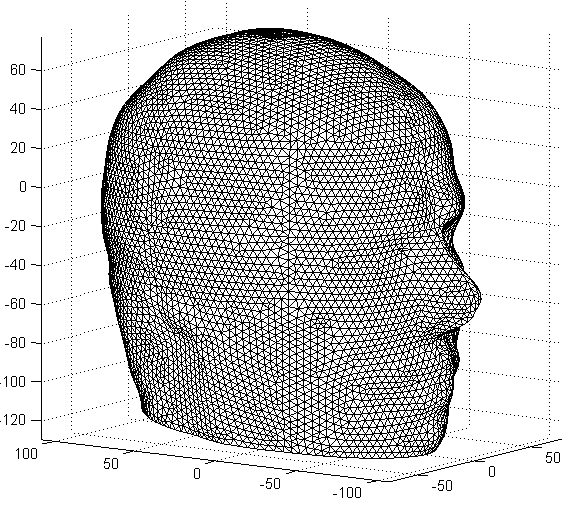
شناخته می شود و زمانی قابل استفاده است که کارت گرافیک شما سریعتر از مانیتور شما باشد یعنی زمانی

که مانیتور شما دارد تصویر اول را نمایش می دهد کارت گرافیک شما تصویر دوم را هم ترسیم کرده و در این صورت است که کارت گرافیک شروع به ترسیم تصویر سوم می کند . که باز این قابلیت به میزان حافظه

کارت گرافیک شما بستگی دارد .

**:Mesh**

توری و مشبکه که از مثلث های کوچکی تشکیل شده و می تواند شکل خاصی را نمایش دهد . شکل 2-3

****

شکل 2-3

ضمیمه الف : مقایسه بین ورژن های مختلف DX







**تاریخچه :**

سال 1995 همراه با ظهور ویندوز 95 شرکت Microsoft اولین نسخه از DirectX و DX 1.0 را به بازار نرم افزار عرضه کرد .یک سال بعد یعنی 1996 نسخه ی جدیدتری از DX 2.0 معرفی شد. در همان سال DX 3.0 به سرعت منتشر شد .روند توسعه DX سیر صعودی خود را به سرعت طی کرد تا اینکه در سال 2002 نسخه 9.0 منتشر شد .این نسخه برای اولین بار همراه با ویندوز xp از سوی Microsoft عرضه شد . 2 سال بعد 2004 با ورود کنسول XBOX 360 نسخه جدیدتری از DX 9.c تولید شد . در سال 2006 نسخه ی فوق العاده قوی DX 10.0همراه با ویندوز Vista انتشار یافت . سرانجام در سال 2008 آخرین نسخه DX10.1 از سوی Microsoft تولید وانتشار یافت .

در شکل زیر توسعه DX به خوبی نشان داده شده است .



شکل 5

**تاریخچه ی شرکت  Autodesk**

Autodesk بزرگترین شرکت طراح نرم افزار اتوماسیون, با محصولات متنوع نرم افزاری در بیش از 85 کشور و 18 زبان در دنیاست. Autodesk دارای مراکزی در سوئد, اتریش, اسپانیا, آلمان, انگلستان, ژاپن, استرالیا و جمهوری چک است. دفاتر مرکزی در ساسالیتوی کالیفرنیا, چندمایل دورتر از شمال سان فرانسیسکو قرار دارد.

Autodesk در سال 1982 توسط جان واکر, برنامه نویس کامپیوتری و بنیانگذار شرکت تأسیس شد. کسی که جسی برست, مقاله نویس *پی سی ویک* او را به عنوان " با استعدادترین و عجیب غریب ترین شخصی که تا بحال دیده ام " توصیف کرده است. واکر نرم افزاری برای طراحی کامپیوتری به نام Autocad از مایکل ریدل و درمقابل 10 میلیون دلار حق الامتیاز  بدست آورد. سال بعد, شرکت او  Autodesk, برنامه ی Autocad را در اختیار عموم قرار داد.

با گسترش بازار کامیپوترهای شخصی و نرم افزار , Autodesk به موفقیت زود هنگامی دست یافت.

در سال 1985, واکر Autodesk را به اندازه کافی بزرگ کرده بود تا در سال بعد مقام مدیریت شرکت را ترک کند و به علاقه اش به برنامه نویسی برسد . درنتیجه, واکر برنامه مکملی برای Autocad طراحی کرد که مخصوصاً برای صنعت ساختمان طراحی شده بود. برنامه مکمل که از طریق Autodesk به بازار آمد, به طراحان اجازه میداد تا از طریق اطلاعات موجود در طراحی هایشان , پیشنهاد قیمت و زمان بندی ساخت را ارائه دهند.

در همین زمان , 40 هزار بسته ی Autocad برای فروش ارسال شده بود.

در تابستان سال 1987, Autodesk دومین دور سهام خود را به جریان انداخت که 5/2 میلیون سهم هرکدام به قیمت 2400 دلار را عرضه می کرد. سرمایه ی تولید شده از عرضه ی این سهام , شرکت را قادر ساخت تا بعضی از بدهی هایش را بپردازد. در سال 1988, دارایی Autodesk بر 100 میلیون دلار نقد و اوراق بهادار بالغ می شد در حالی که سود سهام 40 درصد نسبت به سال قبل افزایش یافته بود.

در سال 1989, Autodesk 60 درصد سهم بازار را برای کامیپیوتر شخصی و نرم افزار طراحی خودکار, با فروشی به ارزش 117 میلیون دلار به خود اختصاص داد.

در اوایل دهه ی 1990, رشد در Autodesk و وابستگی شرکت به Autocad , نیاز به بازسازی عملکرد شرکت را می طلبید. پنج بخش پشتیبانی جداگانه به وجود آمد و هرکدام بر پنج خط عمده ی تولید شرکت که همگی برای استفاده از Autocad طراحی شده بودند, نظارت می کردند. بعلاوه , شرکت 20 درصد از سود سهم متعارف نرم افزار ایزاکا, تولید کننده ی سیستم گرافیکی هوپس را نیز خریداری نمود. سیستم گرافیکی هوپز زمانی که با Autocad تلفیق می شد بسیار خوب عمل می کرد و منجر به محصولی می شد که کاربر به راحتی با آن کنار می آمد و  Autodesk, نرم افزار ازاکا را در آگوست 1993 به طور کامل بدست آورد.

در آوریل  1992, کارول ای. بارتز به مقام ارشد اجرائی در Autodesk منصوب گردید.

بارتز که دارای مدرک مهندسی در علوم کامیپوتری از دانشگاه واشنگتن بود و معاون سان میکروسیستم, یکی از دو زنی شد که ریاست شرکت عمده ی ایالات متحده را در صنعت عالی تکنولوژی برعهده داشت. علاوه بر وظایفش در Autodesk, بارتز از اعضای هیئت مدیره ی شرکت سیستمهای طراحی کادنس, یک شرکت ارتباطات ایرتاچ, یک شرکت به مالکیت پسیفیک تلسیز نیز می باشد.

بارتز پس از انتصاب در Autodesk, سه هدف عمده را برای شرکت در نظر گرفت:

1. تبدیل Autodesk به یک شرکت 1 میلیارد دلاری تا سال 1999
2. کاهش وابستگی شرکت به Autocad به عنوان منبع اولیه ی سود
3. حرکت شرکت به سوی تولیداتی که علاوه بر طراحی مبتنی بر کامپیوتر باشند.

بارتز همچنین به سازماندهی مجدد شرکت پرداخت و ساختار سنتی تری به بخش مدیریت داد و یک تیم اجرایی نیز تشکیل داد تا با مهندسی نرم افزار برای تسهیل انطباق Autocad با نیازهای عمومی کار کنند.

برای وحدت بخشیدن به تمرکز روی اتوماسیون طراحی , Autodesk سهامش را در AMIX  یک شبکه ی فروش اینترنتی و Xanadu  یک شرکت نرم افزاری پایگاه داده ها فروخت و شرکتی به نام راه حل های مهندسی میکرو را خریداری کرد که به نرم افزار شبیه سازی کامیپوتری می پرداخت. در سال 1993, Autodesk دارائی های کل شرکت وودبورن را خریداری نمود که باعث شد تکنولوژی شبیه سازی سه بعدی وودبورن به Autodesk منتقل گردد.

محصولات Autodesk حاوی برنامه های اجرایی در زمینه های بسیار متنوع شامل معماری , مهندسی , ساختمان سازی , سیستمهای اطلاعات جغرافیایی , طراحی مکانیکی و ویدئوگرافی است. از بین مشتریان عمده ی Autodesk, می توان شرکتهای چورون, کوهلر, سونی پیکچرز و شرکت تلفن و تلگراف ژاپنی به نام نیپون را نام برد.

چورون از بسته ی تلفیقی نسخه ی 12 Autocad و نرم افزار ADE و برنامه ی سیستم اطلاعات زمین شناختی Autodesk استفاده کرد تا زمینها و املاک اجاره ای را روی زمین و دور از دریا, نظارت نماید.

کوهلر از Autocad , اتوسرف و اتومیل برای طراحی لوله کشی بخشهای ساختمانی نظیر وان حمام, توالت و سینکها استفاده کرد.

سونی پیکچرز از استودیوی تریدی برای ترسیم زوایای دوربین قبل از فیلمبرداری عملی استفاده کرد. بعلاوه Autodesk یک قرارداد 550 میلیون دلاری با فرماندهی مهندسی تسلیحات دریایی جهت تأمین برنامه ی CAD 2 دارد.

Autodesk به همکاری خود با تأمین کنندگان سایر نرم افزارها نیز همکاری داشته و با آنان اتحادیه تشکیل می دهد تا شرکتهای عضو برنامه ها را تولید نمایند و درعین حال آزادی عمل خود را نیز حفظ کنند. در سال 1993, Autodesk ابزاری برای استفاده از نسخه ی 12 Autocad روی سیستم عامل IBM OS/22.0 طراحی کرد و سال بعد همکاری با شرکت ابزارهای Xaos را اعلام کرد. هدف از این کار, تولید نرم افزار پردازش تصویر بود که با نسخه ی 3 استودیو تری دی Autodesk سازگاری داشته باشد. Autodesk همچنین پخش اتواسکچ2 برای ویندوز مایکروسافت را اعلام نمود.

سایر سرمایه گذاری های مشترک حاوی توافق با مایکروسافت برای طراحی برنامه های CAD سازگار با مایکروسافت بودند تا با مایکروسافت آفیس تلفیق شوند. شکل گیری توافق نامه ی شوارایی با شرکت مشاوره ای UGC, شرکتی که درباره ی بازار سیستم اطلاعات زمین شناختی اطلاعات خوبی دارد. و یک توافق نامه با شرکت سیلیکون گرافیک تا مشترکاً برنامه ای را طراحی و پخش نمایند که ویژگی طراحی نسخه ی 3 استودیو تری دی Autodesk را با برنامه ی سیلیکون گرافیک تلفیق نموده و همچنین نسخه ای از برنامه ی اتوویژن شرکت Autodesk را که با برنامه ی سیلیکون گرافیک سازگار است, طراحی نماید.

در سال 1994, Autodesk حدود 750 مرکز آموزشی در سطح دنیا با هدف آموزش کاربران در زمینه ی برنامه های اجرایی Autocad و سایر نرم افزارهای Autodesk به مرحله ی اجرا درآورد. Autodesk به شبکه ای از فروشندگان, پخش کنندگان و فروش مستقیم برای پخش محصولاتش وابسته است و سیستمی از همکاری با طراحان نرم افزاری ثالث را نیز به کار می گیرد. این سیستم که سیستم توسعه ی Autocad یا ADS نامیده می شود در سال 1990جهت تشویق طراحان مستقل نرم افزار برای ایجاد برنامه های ضمیمه برای محصولات Autodesk و برنامه های اجرایی جدید برای تکنولوژیهای Autodesk, طرح ریزی شد.

در سال 1994 بیش از 2 هزار طراح مستقل برای طراحی سیستمهای ویژه که از برنامه های Autodesk استفاده کنند, کار می کردند

در اواسط دهه ی 1990, Autodesk هیچگونه دارایی غیرمنقول نداشت. تمامی تجهییزات مدیریتی, طراحی محصول, بازاریابی و فعالیتهای تولید در سرتاسر ایالات متحده, اروپا و آسیا- شامل تجهییزات در نوفل شاتل و سوئد که در سال 1993 باهزینه ی 1.4 میلیون دلار به منظور تمرکز بر روی محصول اروپائی Autodesk گشایش یافته بود- به اجاره داده شده بودند. با اینکه Autodesk مالک بسیاری از تجهیزات استفاده شده در کارش بود, حقوق معنوی شرکت و شبکه ی برنامه نویسی و استعداد تجاری به عنوان دارایی های عمده ی آن محسوب می گردید.

Autodesk نظیر بسیاری از شرکتهای نرم افزاری دیگر, تحت تأثیر بهره برداری غرمجاز در اوایل دهه ی 1990 قرار گرفت. برای رویارویی با این مسئله, Autodesk گروههایی را با شرکتهایی که تکنولوژی بالایی دارند متحد کرده است تا سازمانهای منطقه ای مختلفی تشکیل دهند شامل اتحادیه ی کانادایی علیه سرقت نرم افزار (CAAST) و اتحادیه ی تجارت نرم افزار (BSA). شرکت دعوای حقوقی را جهت حمایت از قانون کپی رایت اینشرکت اقامه کرده است که شامل دعوای حقوقی سال 1994 علیه شرکت کادسیس می باشد. نتیجه ی این دعوا 100 هزار دلار غرامت به نفع Autodesk در 23 می سال 1994 بود.

با تغییر تکنولوژی در صنعت نرم افزار کامپیوتر, Autodesk همچنان منعطف باقی مانده و امیدوار است که تکنولوژیها و امکانات تجاری جدیدی را ابداع نماید تا رتبه ی خود را بین شرکتهای نرم افزاری پیشرو, حفظ کند. توسعه ی Autodesk به صورت یک مالتی مدیا همراه با نیازهای بازار برای ابداعات نو در این زمینه, پیش می رود.

**فصل 5**

**جمع بندی و نتیجه گیری**

در سال های اخیر پیشرفت های چشم گیر در زمینه گرافیک کامپیوتری باعث تولید انیمیشن و بازی های سه بعدی فوق العاده زیبایی شده است ، بدون شک یکی از عوامل مهم این پیشرفت ، تکنولوژی DirectX

ماکروسافت می باشد .

DirectX مجموعه ای از واسط های گرافیکی قدرتمند است که برنامه نویس میتواند از آنها برای ساخت صحنه های دو بعدی و سه بعدی استفاده کند ، از طرف دیگر پیدایش زبان C# وپشتیبانی کامل این زبان از مفاهیم شی گرا ، سبب شد تا قدرت برنامه نویسان برای ساخت انیمیشن های زیبا و بازی های بسیار بزرگ و پیچیده ، با تصاویر گرافیکی حیرت انگیز افزایش یابد .

در این شبیه سازی با استفاده از کتابخانه گرافیکی DirectX یک محیط گرافیکی ارایه خواهد شد که

بازدید از یک موزه را شبیه سازی میکند در این محیط از تابلوها و اجسام سه بعدی مثل میز و صندلی و

پخش موزیک ملایم همراه با صدای قدم های شخص هم وجود خواهد داشت.

**فهرست منابع**

[1]وحید خلیل پور اکرم ومحرم چلنگر ، مقدمه ای بر گرافیک سه بعدی و DirectX ، دانشگاه آزاد اسلامی واحد شبستر .

[2] Harrison , Lynn ,T[2004] introduction to 3D game Engine design using DirectX9.0 ,Prentice Hall Inc , New jersey .

[3] Microsoft DirectX SDK [2008] Software & Documentation .

[4] MSDN Library Documentation [2008] .

[5] Stan Trujillo , Cutting Edge Direct3D Programming , The Coriolis Group Publishing – 1996 .

**پیوست**

**کدهای برنامه با توضیح :**

public class GraphicEngine

{

public Device device3d;

public Vector3 CameraPosition, CameraTarget, Upvector;

public int trainWidth = 6, trainHeight = 6, trainDepth = 6;

private int width, height, depth = 5000;

private InputToolBox GItb;

private Material material;

public int drawStep = 50;

textureWall[] tW;

textureWall[] roadTextures;

Door \_door;

window \_window;

Mesh3D chairMesh3d;

Mesh3D dinerTableMesh3d;

Mesh3D pottryMesh3d;

Mesh3D extLump;

Mesh3D roadBarriers;

SkyBox Sky;

SplashScreen sps;

SplashButton[] btnStart;

SoundBox doorOpen\_Sound;

SoundBox doorClose\_Sound;

SoundBox background\_Music;

SoundBox walk\_Sound;

SoundBox windowOpen\_Sound;

SoundBox windowClose\_Sound;

public GraphicEngine(Device d)

{

d.DeviceReset += new EventHandler(device3d\_DeviceReset);

device3d = d;

material = new Microsoft.DirectX.Direct3D.Material();

material.Diffuse = Color.White;

device3d.Material = material;

width = device3d.PresentationParameters.BackBufferWidth;

height = device3d.PresentationParameters.BackBufferHeight;

device3d.RenderState.CullMode = Cull.None;

device3d.RenderState.Lighting = false;

device3d.RenderState.FillMode = FillMode.Solid;

//

// Declaration Camera Information

//

CameraPosition = new Vector3(drawStep \* trainWidth / 2, 52, drawStep \* (3 / 2));

CameraTarget = new Vector3(drawStep \* trainWidth / 2, 58, trainDepth \* drawStep);

Upvector = new Vector3(0, 1, 0);

//

// Note:

// InputToolBox must be define after declaration of camera Inforamation!

GItb = new InputToolBox(this);

createObjects();

background\_Music.playBackMusic();

GItb.SetWalkingSound(walk\_Sound);

List<textureWall> lstTextureWall;

List<Mesh3D> lstMesh3D;

setObjectsToCollision(out lstTextureWall, out lstMesh3D);

GItb.SetCollistionObjects(lstTextureWall, lstMesh3D);

initialize\_text(device3d);

}

private void setObjectsToCollision(out List<textureWall> lstTextureWall, out List<Mesh3D> lstMesh3D)

{

lstMesh3D = new List<Mesh3D>();

lstMesh3D.Add(chairMesh3d);

lstMesh3D.Add(dinerTableMesh3d);

lstMesh3D.Add(pottryMesh3d);

lstMesh3D.Add(Sky);

//

//

//

lstTextureWall = new List<textureWall>();

lstTextureWall.AddRange(tW);

lstTextureWall.Add(\_door);

lstTextureWall.Add(\_window);

}

private void device3d\_DeviceReset(object sender, EventArgs e)

{

device3d.RenderState.Lighting = false;

device3d.RenderState.FillMode = FillMode.Solid;

device3d.RenderState.CullMode = Cull.None;

SetupCamera();

createObjects();

}

private void SetupCamera()

{

device3d.Transform.View = Matrix.LookAtRH(CameraPosition, CameraTarget, Upvector);

device3d.Transform.Projection = Matrix.PerspectiveFovRH((float)Math.PI / 2, width / height, 1f, depth);

}

public void OpenCloseDoor()

{

if (\_door != null)

\_door.openClose(doorOpen\_Sound, doorClose\_Sound);

}

public void OpenCloseWindow()

{

if (\_window != null)

\_window.openClose(windowOpen\_Sound, windowClose\_Sound);

}

public void DrawWorld()

{

SetupCamera();

device3d.Clear(ClearFlags.Target | ClearFlags.ZBuffer, Color.DarkSlateBlue, 1.0f, 0);

if (!btnStart[0].Enable)

{

device3d.BeginScene();

//

//

//

device3d.Transform.World = Matrix.Scaling(2, 2, 2);

#region textures

// Floor

tW[0].drawWall();

//

// Front Wall

tW[1].drawWall();

tW[2].drawWall();

tW[3].drawWall();

//

// Right Wall

tW[4].drawWall();

//

// Left Wall

tW[5].drawWall();

//

// Back Wall

tW[6].drawWall();

//

// Top

tW[7].drawWall();

//

// Door

//\_door.drawWall();

//

// Carpet

tW[8].drawWall();

//

// right Tableau

tW[9].drawWall();

//

// left Tableau

tW[10].drawWall();

//

// window Frame

tW[11].drawWall();

//

// window

\_window.drawWall();

#endregion

#region road Textures

roadTextures[0].drawWall();

roadTextures[1].drawWall();

roadTextures[2].drawWall();

#endregion

device3d.Transform.World = Matrix.Identity;

#region Mesh3Ds

//

// Mesh3Ds

if (chairMesh3d != null)

{

chairMesh3d.Draw(new Vector3(280, 25, 150), -4.75f);

chairMesh3d.Draw(new Vector3(280, 25, 80), -4.75f);

}

if (dinerTableMesh3d != null)

{

dinerTableMesh3d.Draw(new Vector3(210, 0, 130), -1.5f);

}

if (pottryMesh3d != null)

{

pottryMesh3d.Draw(new Vector3(210, 40.8f, 130));

}

if (Sky != null)

{

Sky.Draw(new Vector3(0, 600, 500));

}

if (extLump != null)

{

extLump.Draw(new Vector3(250, 0, 750));

extLump.Draw(new Vector3(-100, 0, 450), 1.57f);

}

if (roadBarriers != null)

{

// one Barriers \_

// |<-------------->| |

// | | | 15 px = depth.

// |<-------------->| |

// 130 px -

// width.

//

// |a a|

// 2 |b b| 1

// o |c c|

// \_\_\_\_|d d|\_\_\_\_\_

// hgfe efghi

// \_\_\_\_ \_\_\_\_\_

// 4 edcb|a a|bcdef 3

// .------ /------.

// | / |

// | HOUSE |

//

//

//

// 1

//

roadBarriers.Draw(new Vector3(-30, 0, 667)); // e

roadBarriers.Draw(new Vector3(-160, 0, 667)); // f

roadBarriers.Draw(new Vector3(-290, 0, 667)); // g

roadBarriers.Draw(new Vector3(-320, 0, 667)); // h

roadBarriers.Draw(new Vector3(-450, 0, 667)); // i

roadBarriers.Draw(new Vector3(51, 0, 713), 1.57f); // d

roadBarriers.Draw(new Vector3(51, 0, 843), 1.57f); // c

roadBarriers.Draw(new Vector3(51, 0, 973), 1.57f); // b

roadBarriers.Draw(new Vector3(51, 0, 1103), 1.57f); // a

//

// 2

//

roadBarriers.Draw(new Vector3(282, 0, 667)); // e

roadBarriers.Draw(new Vector3(412, 0, 667)); // f

roadBarriers.Draw(new Vector3(542, 0, 667)); // g

roadBarriers.Draw(new Vector3(672, 0, 667)); // h

roadBarriers.Draw(new Vector3(217, 0, 713), 1.57f); // d

roadBarriers.Draw(new Vector3(217, 0, 843), 1.57f); // c

roadBarriers.Draw(new Vector3(217, 0, 973), 1.57f); // b

roadBarriers.Draw(new Vector3(217, 0, 1103), 1.57f); // a

//

// 3

//

roadBarriers.Draw(new Vector3(-30, 0, 501)); // b

roadBarriers.Draw(new Vector3(-160, 0, 501)); // c

roadBarriers.Draw(new Vector3(-290, 0, 501)); // d

roadBarriers.Draw(new Vector3(-320, 0, 501)); // e

roadBarriers.Draw(new Vector3(-450, 0, 501)); // f

roadBarriers.Draw(new Vector3(51, 0, 433), 1.57f); // a

//

// 4

//

roadBarriers.Draw(new Vector3(282, 0, 501)); // b

roadBarriers.Draw(new Vector3(412, 0, 501)); // c

roadBarriers.Draw(new Vector3(542, 0, 501)); // d

roadBarriers.Draw(new Vector3(672, 0, 501)); // e

roadBarriers.Draw(new Vector3(217, 0, 433), 1.57f); // a

}

//

//

//

device3d.SamplerState[0].MinFilter = TextureFilter.Anisotropic;

device3d.SamplerState[0].MagFilter = TextureFilter.Anisotropic;

device3d.SamplerState[0].AddressU = TextureAddress.Mirror;

device3d.SamplerState[0].AddressV = TextureAddress.Mirror;

//

//

//

#endregion

//

// runTime Method for Text

//

draw\_text();

//

device3d.EndScene();

device3d.Present();

//

GItb.Poll(); // setCurrent State for Mouse and Keyboard

}

else

{

//

// SplashScreen

//

sps.Draw();

}

}

private void createObjects()

{

try

{

#region textures

//

// Define Array for 6 face of a Cube (my Room)

// tW[0] = Floor

// tW[1] = Front Wall

// tW[2] = Front Wall

// tW[3] = Front Wall

// \_door = Door

// tW[4] = Right Wall

// tW[5] = Left Wall

// tW[6] = Back Wall

// tW[7] = Top

// tW[8] = Carpet

// tW[9] = right Tableau

// tW[10] = left Tableau

// tW[11] = window Frame

// \_window = window

tW = new textureWall[12];

//

// Floor

tW[0] = new textureWall(device3d, false, new threeAxisRectangle(new Vector3(0, 0, 0),

(trainWidth \* drawStep), 0, (trainDepth \* drawStep)), drawStep, 0, drawStep,

@"myTextures/ConcreteFloor.jpg");

//

// Front Wall

tW[1] = new textureWall(device3d, false, new threeAxisRectangle(new Vector3(150, 0, (trainDepth \* drawStep)),

(2 \* 75), (4 \* 75), 0), 75, 75, 0,

@"myTextures/TilesOrnate\_Front.jpg");

tW[2] = new textureWall(device3d, false, new threeAxisRectangle(new Vector3(0, 0, (trainDepth \* drawStep)),

(1 \* 75), (4 \* 75), 0), 75, 75, 75,

@"myTextures/TilesOrnate\_Front.jpg");

tW[3] = new textureWall(device3d, false, new threeAxisRectangle(new Vector3(75, 225, (trainDepth \* drawStep)),

(1 \* 75), (1 \* 75), 0), 75, 75, 75,

@"myTextures/TilesOrnate\_Front.jpg");

//

// Door

\_door = new Door(device3d, true,

new threeAxisRectangle(new Vector3(74, 0, (trainDepth \* drawStep)), 77, 226, 0),

77, 226, 0, @"myTextures/door.jpg", 1.8f);

\_door.speed = 0.04f;

//

// Right Wall

tW[4] = new textureWall(device3d, false, new threeAxisRectangle(new Vector3(0, 0, 0),

0, (trainHeight \* drawStep), (trainDepth \* drawStep)), drawStep, drawStep, drawStep,

@"myTextures/TilesOrnate.jpg");

//

// Left Wall

tW[5] = new textureWall(device3d, false, new threeAxisRectangle(new Vector3((trainWidth \* drawStep), 0, 0),

0, (trainHeight \* drawStep), (trainDepth \* drawStep)), drawStep, drawStep, drawStep,

@"myTextures/TilesOrnate.jpg");

//

// Back Wall

tW[6] = new textureWall(device3d, false, new threeAxisRectangle(new Vector3(0, 0, 0),

(trainWidth \* drawStep), (trainHeight \* drawStep), 0), drawStep, drawStep, drawStep,

@"myTextures/TilesOrnate.jpg");

//

// Top

tW[7] = new textureWall(device3d, true, new threeAxisRectangle(new Vector3(0, (trainHeight \* drawStep), 0),

300, 0, 300), 300, 0, 300,

@"myTextures/OrnamentsRound.jpg");

//

// Carpet

tW[8] = new textureWall(device3d, true,

new threeAxisRectangle(new Vector3(25, 0.1f, 50), 250, 0, 150),

250, 0, 150, @"myTextures/PersianCarpets.jpg");

//

// right Tableau

tW[9] = new textureWall(device3d, true,

new threeAxisRectangle(new Vector3(1, (trainHeight / 2) \* drawStep, (trainDepth / 3) \* drawStep), 0, 90, 100),

0, 90, 100, @"myTextures/rightTableau.jpg");

//

// left Tableau

tW[10] = new textureWall(device3d, true,

new threeAxisRectangle(new Vector3((trainWidth \* drawStep) - 1, ((trainHeight / 2) \* drawStep) - 20, (trainDepth / 3) \* drawStep),

0, 100, 75), 0, 100, 75, @"myTextures/leftTableau.jpg");

//

// window Frame

tW[11] = new textureWall(device3d, true,

new threeAxisRectangle(new Vector3(((trainWidth / 2) \* drawStep), ((trainHeight / 2) \* drawStep) - 20, 0.1f),

100, 75, 0), 100, 75, 0, @"myTextures/view\_out\_our\_window.png");

//

// window

\_window = new window(device3d, true, openTranslationVector.Right, 40f,

new threeAxisRectangle(new Vector3(((trainWidth / 2) \* drawStep) + 50.8f, ((trainHeight / 2) \* drawStep) - 15, 0.2f),

46, 67, 0), 46, 67, 0, @"myTextures/window.png", true);

#region road Textures

roadTextures = new textureWall[3];

roadTextures[0] = new textureWall(device3d, false,

new threeAxisRectangle(new Vector3(50, 1, 305), 150, 0, 900),

150, 0, 150, @"myTextures/FloorHerringbone.jpg");

roadTextures[1] = new textureWall(device3d, false,

new threeAxisRectangle(new Vector3(200, 1, 500), 600, 0, 150),

150, 0, 150, @"myTextures/FloorHerringbone.jpg");

roadTextures[2] = new textureWall(device3d, false,

new threeAxisRectangle(new Vector3(-550, 1, 500), 600, 0, 150),

150, 0, 150, @"myTextures/FloorHerringbone.jpg");

#endregion

#endregion

#region Mesh3Ds

//

// Mesh3Ds

chairMesh3d = new Mesh3D(device3d, "MALLETCH.x", new Vector3(0.025f, 0.025f, 0.025f),

CustomVertex.PositionTextured.Format);

dinerTableMesh3d = new Mesh3D(device3d, "DinerTable.x", new Vector3(1.2f, 1.2f, 1.2f),

CustomVertex.PositionTextured.Format);

pottryMesh3d = new Mesh3D(device3d, "POTTRY2.x", new Vector3(3f, 3f, 3f),

CustomVertex.PositionTextured.Format);

Sky = new SkyBox(device3d, "skybox.x", new Vector3(300.5f, 200.5f, 300f),

CustomVertex.PositionTextured.Format);

extLump = new Mesh3D(device3d, "Lump1.x", new Vector3(3f, 2.5f, 3f),

CustomVertex.PositionTextured.Format);

roadBarriers = new Mesh3D(device3d, "Barriers.x", new Vector3(1f, 1f, 1f),

CustomVertex.PositionTextured.Format);

#endregion

#region SplashScreen objects

//

// SplashScreen

//

btnStart = new SplashButton[1];

btnStart[0] = new SplashButton((Form)device3d.CreationParameters.FocusWindow,

device3d, new Point((width / 2) - 125, height - 300), new Size(250, 100),

@"SplashScreen\START\_Normal.png", @"SplashScreen\START\_MouseOver.png",

@"SplashScreen\START\_MouseDown.png", @"SplashScreen\START\_Disable.png");

sps = new SplashScreen(device3d, @"SplashScreen\SplashScreen.png", btnStart);

#endregion

#region SoundBox

//

// sounds

//

doorOpen\_Sound = new SoundBox((System.Windows.Forms.Form)device3d.CreationParameters.FocusWindow,

@"mySounds\door\_open.wav");

doorClose\_Sound = new SoundBox((System.Windows.Forms.Form)device3d.CreationParameters.FocusWindow,

@"mySounds\door\_close.wav");

windowOpen\_Sound = new SoundBox((System.Windows.Forms.Form)device3d.CreationParameters.FocusWindow,

@"mySounds\folding-door-open.wav");

windowClose\_Sound = new SoundBox((System.Windows.Forms.Form)device3d.CreationParameters.FocusWindow,

@"mySounds\folding-door-close.wav");

walk\_Sound = new SoundBox((System.Windows.Forms.Form)device3d.CreationParameters.FocusWindow,

@"mySounds\walk.wav");

background\_Music = new SoundBox((System.Windows.Forms.Form)device3d.CreationParameters.FocusWindow);

background\_Music.SetBackMusic(@"mySounds\backMusic.mp3");

background\_Music.backMusicVolume = -700;

#endregion

//

//

//

device3d.SamplerState[0].MinFilter = TextureFilter.Anisotropic;

device3d.SamplerState[0].MagFilter = TextureFilter.Anisotropic;

device3d.SamplerState[0].AddressU = TextureAddress.Mirror;

device3d.SamplerState[0].AddressV = TextureAddress.Mirror;

//

//

//

}

catch (System.Exception ex)

{

System.Windows.Forms.MessageBox.Show(ex.Message, "Textures or Mesh3Ds are not OK.");

System.Environment.Exit(0);

}

}

#region Text

private Microsoft.DirectX.Direct3D.Font text\_Resulation;

private Microsoft.DirectX.Direct3D.Font text\_Framerate;

private Microsoft.DirectX.Direct3D.Font text\_Position;

private Microsoft.DirectX.Direct3D.Font text\_Target;

private Microsoft.DirectX.Direct3D.Font text\_exit;

/// <summary>

/// initialize tex

/// </summary>

/// <param name="device"></param>

public void initialize\_text(Device device)

{

//

// Create a Font for any Text in Display

//

System.Drawing.Font sysFont = new System.Drawing.Font("tahoma", 14f, FontStyle.Bold);

//

//initialize text Resulation

text\_Resulation = new Microsoft.DirectX.Direct3D.Font(device, sysFont);

//initialize text Framerate

text\_Framerate = new Microsoft.DirectX.Direct3D.Font(device, sysFont);

//initialize text Camera Position

text\_Position = new Microsoft.DirectX.Direct3D.Font(device, sysFont);

//initialize text Camera Target

text\_Target = new Microsoft.DirectX.Direct3D.Font(device, sysFont);

//initialize text exit ESC

text\_exit = new Microsoft.DirectX.Direct3D.Font(device, sysFont);

}

/// <summary>

/// draw text

/// </summary>

/// <param name="device"></param>

public void draw\_text()

{

//

// Set Graphic Resulations

//

text\_Resulation.DrawText(null, string.Format("Resulation ({0} , {1})", this.width, this.height), new Point(5, 5), Color.Red);

//

// Display ESC key's

//

text\_exit.DrawText(null, "Please press ESC key's for exit.", new Point(5, 30), Color.LightBlue);

//

// Display FPS number

//

text\_Framerate.DrawText(null, string.Format("Framerate : {0:0.00} fps", Framerate.UpdateFramerate()), new Point(5, 55), Color.LightGreen);

//

// Display Camera Position and Target

//

text\_Position.DrawText(null, string.Format("Camera Position = ({0:0.0}x , {1:0.0}y , {2:0.0}z)",

CameraPosition.X, CameraPosition.Y, CameraPosition.Z), new Point(5, 80), Color.Aqua);

text\_Target.DrawText(null, string.Format("Camera Target = ({0:0.0}x , {1:0.0}y , {2:0.0}z)",

CameraTarget.X, CameraTarget.Y, CameraTarget.Z), new Point(5, 105), Color.Aqua);

}

#endregion

}

}

namespace GraphicTools

{

/// <summary>

/// Stores a set of four integers that represent the location and size of a rectangle in 3D-Space.

/// For more advanced region functions, use a System.Drawing.Region object.

/// </summary>

public class threeAxisRectangle

{

/// <summary>

/// Initializes a new instance of the threeAxisRectangle class with the

/// specified location and size.

/// </summary>

/// <param name="rightLowerPos">The x & y & z coordinate of the lower-right corner of the rectangle.</param>

/// <param name="width">The width of the rectangle.</param>

/// <param name="height">The height of the rectangle.</param>

/// <param name="depth">The depth of the rectangle.</param>

public threeAxisRectangle(Microsoft.DirectX.Vector3 rightLowerPos, int width, int height, int depth)

{

this.x = rightLowerPos.X;

this.y = rightLowerPos.Y;

this.z = rightLowerPos.Z;

this.Width = width;

this.Height = height;

this.Depth = depth;

\_rightLowerPos = rightLowerPos;

}

/// <summary>

/// Gets or sets the width of this threeAxisRectangle structure.

/// </summary>

public int Width;

/// <summary>

/// Gets or sets the height of this threeAxisRectangle structure.

/// </summary>

public int Height;

/// <summary>

/// Gets or sets the depth of this threeAxisRectangle structure.

/// </summary>

public int Depth;

private float x;

/// <summary>

/// Gets or sets the x-coordinate of the lower-right corner of this

/// threeAxisRectangle structure.

/// </summary>

public float X

{

get { return x; }

set

{

x = value;

\_rightLowerPos.X = x;

}

}

private float y;

/// <summary>

/// Gets or sets the y-coordinate of the lower-right corner of this

/// threeAxisRectangle structure.

/// </summary>

public float Y

{

get { return y; }

set

{

y = value;

\_rightLowerPos.Y = y;

}

}

private float z;

/// <summary>

/// Gets or sets the z-coordinate of the lower-right corner of this

/// threeAxisRectangle structure.

/// </summary>

public float Z

{

get { return z; }

set

{

z = value;

\_rightLowerPos.Z = z;

}

}

/// <summary>

/// The x & y & z coordinate of the lower-right corner of the rectangle.

/// </summary>

private Microsoft.DirectX.Vector3 \_rightLowerPos;

/// <summary>

/// The x & y & z coordinate of the lower-right corner of the rectangle.

/// </summary>

public Microsoft.DirectX.Vector3 RightLowerPos { get { return \_rightLowerPos; } } // ReadOnly

/// <summary>

/// Tests whether two threeAxisRectangle structures differ in location.

/// </summary>

/// <param name="left">The threeAxisRectangle structure that is to the left of the inequality operator.</param>

/// <param name="right">The threeAxisRectangle structure that is to the right of the inequality operator.</param>

/// <returns>This operator returns true if any of the threeAxisRectangle.X, threeAxisRectangle.Y,

/// threeAxisRectangle.Z, threeAxisRectangle.Width, threeAxisRectangle.Height or threeAxisRectangle.Depth

/// properties of the two threeAxisRectangle structures are unequal; otherwise false.</returns>

public static bool operator !=(threeAxisRectangle left, threeAxisRectangle right)

{

return ((left.X != right.X) || (left.Y != right.Y) || (left.Z != right.Z) ||

(left.Width != right.Width) || (left.Height != right.Height) || (left.Depth != right.Depth));

}

/// <summary>

/// Tests whether two threeAxisRectangle structures have equal location.

/// </summary>

/// <param name="left">The threeAxisRectangle structure that is to the left of the equality operator.</param>

/// <param name="right">The threeAxisRectangle structure that is to the right of the equality operator.</param>

/// <returns>This operator returns true if the two threeAxisRectangle structures

/// have equal threeAxisRectangle.X, threeAxisRectangle.Y, threeAxisRectangle.Z,

/// threeAxisRectangle.Width, threeAxisRectangle.Height and threeAxisRectangle.Depth properties.</returns>

public static bool operator ==(threeAxisRectangle left, threeAxisRectangle right)

{

return ((left.X == right.X) && (left.Y == right.Y) && (left.Z == right.Z) &&

(left.Width == right.Width) && (left.Height == right.Height) && (left.Depth == right.Depth));

}

/// <summary>

/// Tests whether obj is a threeAxisRectangle structure with the same location

/// of this threeAxisRectangle structure.

/// </summary>

/// <param name="obj">The System.Object to test.</param>

/// <returns>This method returns true if obj is a threeAxisRectangle structure and

/// its threeAxisRectangle.X, threeAxisRectangle.Y, threeAxisRectangle.Width,

/// and threeAxisRectangle.Height properties are equal to the corresponding

/// properties of this threeAxisRectangle structure; otherwise, false.</returns>

public override bool Equals(object obj)

{

if (obj is threeAxisRectangle)

{

return ((this.X == ((threeAxisRectangle)obj).X) &&

(this.Y == ((threeAxisRectangle)obj).Y) &&

(this.Z == ((threeAxisRectangle)obj).Z) &&

(this.Width == ((threeAxisRectangle)obj).Width) &&

(this.Height == ((threeAxisRectangle)obj).Height) &&

(this.Depth == ((threeAxisRectangle)obj).Depth));

}

else return false;

}

/// <summary>

/// Converts the attributes of this threeAxisRectangle to a human-readable

/// string.

/// </summary>

/// <returns>A string that contains the position, width, height and depth of this threeAxisRectangle

/// structure ¾ for example, {X=20, Y=20, Z=30, Width=100, Height=50, Depth=90}</returns>

public override string ToString()

{

return string.Format(System.Globalization.CultureInfo.CurrentCulture,

"{X={0}, Y={1}, Z={2}, Width={3}, Height={4}, Depth={5}}",

this.X, this.Y, this.Z, this.Width, this.Height, this.Depth);

}

/// <summary>

/// Returns the hash code for this System.Drawing.Rectangle structure. For information

/// about the use of hash codes, see System.Object.GetHashCode() .

/// </summary>

/// <returns>An integer that represents the hash code for this rectangle.</returns>

public override int GetHashCode()

{

return base.GetHashCode();

}

}

}