Main.cpp

1. 定义变量：Line:0-685

```
// Possible camera states when animating our scene:7
// Debugging Text:15
// Window and camera setting variables:18
// General polygon variables:28
// Reflection object variables:32
// Shadow variables:39
// 光照变量:46
// Animation variables:58
// 相机状态:62
// 键盘按键导致相机变化的步长:75
// 相机位置:128
// 相机视线:140
// 相机 Up 方向:152
// 相机旋转的角度:164
// 相机所绕旋转轴:176
// variables for orbits system;223
// variables for particle effect:226
// variables for girl camera action:234
// variables for girl:393
//slides variables:458
//ball variables:467
//door variables:476
//teapot variables:488
//nurbs variables:493
// Hills variables:503
// Roof variables:509
// Ceiling variables:513
// Fountain variables:517
// Trees outside variables:543
// Wall variables:554
// Chair variables:560
// Table variables:569
// Teapot in VG scene variable:580
// Default material:584
// Light material variables:591
// Floor material variables:614
// Glass material variables:619
// Bouncing ball material variables:626
// Define some colors:632
// display lists:662
```

2. 方法定义 Line:688-5016

- float VMagnitude(CVector3 vNormal): * This returns the magnitude of a vector
- CVector3 VNormalize(CVector3 vVector): * This returns a normalize vector (A vector exactly of length 1)
- void Init(HWND hWnd): * This function initializes the app window 这个相当于Assign0 的 display 函数
- WPARAM MainLoop():* The MAIN Loop 这个相当于 Assign0 的 Main 函数
- void CameraAction():* Controls the camera during scene animation.
- void RenderScene():* This function renders the entire scene.
- LRESULT CALLBACK WinProc(HWND hWnd,UINT uMsg, WPARAM wParam, LPARAM lParam): * This function handles the window messages.
- void DrawShadows()
- void DrawReflections()
- void RenderMuseumScene():* Render museum scene using occlusion culling.
- void DrawGirl()
- void GirlHead(float x, float y, float z, float hrx, float hry, float hrz, float rz)
- void GirlResetArmLegAngle()
- void GirlArm(float x, float y, float z, float rz)
- void GirlLeg(float x, float y, float z, float rz)
- void DrawCameraVector()
- void DrawOrbitz()
- void gluBox(float width, float length, float height)
- void DrawBouncingBalls()
- void DrawBall(int i)
- void DrawReflectiveTableTop()
- void DrawSlides()
- void DrawStepObject(float x, float y, float z, float radius, float r, float height, float color[])
- void DrawEntranceDoors()
- void DrawCylinder(float x, float y, float z, float radius, float height, float color[])
- void DrawFrame(float x, float y, float z, float width, float height, float innerWidth, float innerHeight, float angle, float color[])
- void DrawPainting(float x, float y, float z, float width, float height, float angle, int textureNum)
- void DrawWhiteBox(float x, float y, float z, float width, float length, float height, float color[], int textureNum, int blend, float ambient[], float diffuse[], float specular[], float emission[], float shine[]):* This function draws a white box, with xyz as the front lower left coordinate
- void DrawPlane(float xn, float yn, float zn, float x0, float y0, float z0, float x1, float y1, float z1, float x2, float y2, float z2, float x3, float y3, float z3, int textureNum, int blend, float color[], float ambient[], float diffuse[], float specular[], float emission[], float shine[]):* This function draws a

quadrilateral plane given the corner vertices

- void DrawPearlFloor()
- void DrawWalls(float x, float y): * This function draw the walls
- void DrawCeiling()
- void DrawALight(float rotX, float rotY, float rotZ, float x, float y, float z)
- void DrawingManyLights()
- void RenderVanGoghPaintingScene()
- void DrawVGGround()
- void DrawVGDoor()
- void DrawVGPatio()
- void DrawVGPatioRoof()
- void DrawVGFarBuilding()
- void DrawVGCafe()
- void DrawVGEndWall()
- void InitializeParticleSystem()
- void InitializeAParticle(int i, float idleTime)
- void DrawPraticleSystem()
- void ChooseGirlAct()
- void DrawGirlActs()
- void DrawShadowBall()
- void DrawReflectiveBall1()
- void DrawReflectiveBall2()
- void DrawReflectiveObjects()
- void DrawReflectiveFloor1()
- void DrawReflectiveFloor2()
- void DrawSidesofReflectivePlane1()
- void DrawSidesofReflectivePlane2()
- void DrawReflectiveCones()
- void shadowMatrix(float shadowMat[4][4], float groundPlane[4], float lightPos[4])
- bool IntersectAt(float y, float minX, float maxX)
- bool FacingXPos()
- bool FacingXNeg()
- bool FacingYPos()
- bool FacingYNeg()
- bool InRoomI()
- bool InRoomIIA()
- bool InRoomIIB()
- bool InRoomIII()
- bool InRoomIV()
- void RenderOutdoor()
- void RenderRoomI():// This is the room with the bouncing balls
- void RenderRoomIIA():// This is the room with the step object
- void RenderRoomIIB():// This is the room with the teapot
- void RenderRoomIII():// This is the room with the hanging wall

- void RenderRoomIV():// This is the room with the slideshow

Camera.cpp

1. 定义方法 Line:8-526
   - CVector3 Cross(CVector3 vVector1, CVector3 vVector2): * This returns a perpendicular vector from 2 given vectors by taking the cross product.
   - float Magnitude(CVector3 vNormal): * This returns the magnitude of a vector
   - CVector3 Normalize(CVector3 vVector): * This returns a normalize vector (A vector exactly of length 1)
   - void CalculateFrameRate()
   - CCamera::CCamera():* This is the class constructor
   - void CCamera::PositionCamera(float positionX, float positionY, float positionZ,float viewX,float viewY,float viewZ,float upVectorX, float upVectorY, float upVectorZ): * This function sets the camera's position and view and up vVector.
   - void CCamera::Update(): * This updates the camera's view and strafe vector
   - void CCamera::MouseInput():* This allows us to look around using the mouse
   - void CCamera::KeyboardInput():* This function handles the input faster than in the WinProc()
   - void CCamera::Look():* This updates the camera according to the
   - void CCamera::UpdateRightVector()
   - void CCamera::Zoom(float speed)
   - void CCamera::ForwardBackward(float speed)
   - void CCamera::LeftRight(float speed)
   - void CCamera::UpDown(float speed)
   - void CCamera::RotateH(float angle)
   - void CCamera::RotateV(float angle)
   - void CCamera::Panning(float angle): * This pans the camera view
   - void CCamera::Reset()
   - CVector3 CCamera::CalculateRotation(float angle, float x, float y, float z, CVector3 vRot)
   - void CCamera::TurnControlOn()
   - void CCamera::TurnControlOff()

Init.cpp

1. 方法定义
   - void CreateTexture(UINT textureArray[], LPSTR strFileName, int textureID): * This creates a texture in OpenGL that we can texture map
   - void ChangeToFullScreen():* This changes the screen to FULL SCREEN
   - HWND CreateMyWindow(LPSTR strWindowName, int width, int height, DWORD dwStyle, bool bFullScreen, HINSTANCE hInstance): * This function creates a window, but doesn't have a message loop
   - bool bSetupPixelFormat(HDC hdc): * This function sets the pixel format for OpenGL.
   - void SizeOpenGLScreen(int width, int height): * This function resizes the

viewport for OpenGL.

- void InitializeOpenGL(int width, int height): * This function handles all the initialization for OpenGL.
- void DeInit():* This function cleans up and then posts a quit message to the window
- int WINAPI WinMain(HINSTANCE hInstance, HINSTANCE hprev, PSTR cmdline, int ishow): * This function handles registering and creating the window.

Main.h

1. 变量定义
   - struct CVector3: // This is our basic 3D point/vector class
   - struct CParticle
   - extern bool g_bFullScreen;   // Set full screen as default
   - extern HWND g_hWnd; // This is the handle for the window
   - extern RECT g_rRect; // This holds the window dimensions
   - extern HDC   g_hDC;   // General HDC - (handle to device context)
   - extern HGLRC g_hRC;   // General OpenGL_DC - Our Rendering Context for OpenGL
   - extern HINSTANCE g_hInstance;   // This holds our window hInstance
   - extern bool animationOn; // Animation is either on or off
   - extern UINT g_Texture[MAX_TEXTURES]; // This is our texture data array
   - extern float lightPos1[4];// This is the position of our light
   - extern float lightSpotDir1[3];
   - extern float lightSpotCutoff1;
   - extern float lightSpotExp1;

Camera.h

1. 变量定义
   - class CCamera: // This is our camera class