void CObjectExhibitCtrl::OnSphereErosionROIMod()

{

m\_bIsROIErosion = TRUE;

unsigned char chGSDir = 1;

//Create sphere for erosion

OnCreateMeshObjectsSphere();

if(m\_pRSelectedFrame == NULL || m\_pRSelectedFrame->GetFrameRealType() != C3DBaseFrame::MARCHINGCUBE ||

m\_pCurrentProcessMesh == NULL || m\_pCurrentProcessMesh->GetMeshType() != CD3DMESH::MSPHERE)

return;

m\_pSelectCutFrame = m\_pRSelectedFrame;

m\_pRSelectedFrame = NULL;

m\_bEnableScale3DObjects = FALSE;

FLOAT fRadius;

UINT nSlice, nStack;

((CSPHEREMESH\*)(m\_pCurrentProcessMesh))->GetSphereInfo(fRadius, nSlice, nStack);

//Set up Cut Tool

GUID guid;

CoCreateGuid(&guid);

CTube\_XML \*pCutTube = new CTube\_XML(guid);

CreateCutToolSphere(pCutTube, (int)(fRadius));//create a sphere

//m\_pRSelectedFrame=pCutTube; //assign to right select object

D3DXVECTOR3 v3TubeCenter = pCutTube->GetCentrum(); // Get center of Cut Tool

//Step 2. Move Matrix of Cut Tool to Sphere

D3DXMATRIX matSphere = m\_pCurrentProcessMesh->GetMatrixByType(C3DBaseFrame::COMBINEMATRIX); //Combine Matrix of sphere

pCutTube->SetMoveMatrix(matSphere.\_41 - v3TubeCenter.x, matSphere.\_42 - v3TubeCenter.y, matSphere.\_43 - v3TubeCenter.z);

m\_pTubeManager->AddFrame(pCutTube);

//Step 3. Set Cut Tool invisible

pCutTube->SetVisible(FALSE);

//Step 4. Set Mesh invisible and transparent

m\_pCurrentProcessMesh->SetSelectState(FALSE);

m\_pCurrentProcessMesh->SetTransparency(0.8f);

pCutTube->AddChild(m\_pCurrentProcessMesh);

//Cut Tool information

m\_pCutMCOTube = pCutTube;

if(m\_pMCOCutToolInfo == NULL)

m\_pMCOCutToolInfo = new MCO\_CUTTOOL\_INFO;

//Transform between the Cut Tool and Target

D3DXMATRIX mat = m\_mSceneScale \* pCutTube->GetMatrixByType(C3DBaseFrame::COMBINEMATRIX);

D3DXMATRIX matBeCut = m\_mSceneScale \* m\_pSelectCutFrame->GetMatrixByType(C3DBaseFrame::COMBINEMATRIX);

D3DXMATRIX matTarget, matBeCutInverse;

D3DXMatrixInverse( &matBeCutInverse, NULL, &matBeCut);

matTarget = mat \* matBeCutInverse;

D3DXVECTOR4 TransPosPoint;

D3DXVec3Transform(&TransPosPoint, &v3TubeCenter, &(matTarget));

m\_pMCOCutToolInfo->mco\_id = m\_pSelectCutFrame->GetFrameName();

m\_pMCOCutToolInfo->center[0] = (int)(TransPosPoint.x);

m\_pMCOCutToolInfo->center[1] = (int)(TransPosPoint.y);

m\_pMCOCutToolInfo->center[2] = (int)(TransPosPoint.z);

m\_pMCOCutToolInfo->nRadius = (int)(fRadius);

m\_pMCOCutToolInfo->nState = 1;

m\_pMCOCutToolInfo->nOperation = 3; //ROI Mod

m\_pMCOCutToolInfo->nOperationState = 0;

m\_pMCOCutToolInfo->fCameraVector[0] = 0;

m\_pMCOCutToolInfo->fCameraVector[1] = 0;

m\_pMCOCutToolInfo->fCameraVector[2] = 1;

FireChangeAssistViewTab(0);

//MCO Surface Move

m\_PreviousPickPoint.x = 0.0;

m\_PreviousPickPoint.y = 0.0;

m\_fPickRange = fRadius\*fRadius;

m\_pMCPickInfo = new PickTestINFO;

m\_pMCPickInfo->fAlphaAngle = 0.0;

m\_pMCPickInfo->fBetaAngle = 0.0;

m\_pMCPickInfo->fMinDist = FLT\_MAX;

m\_pMCPickInfo->bIsCalRealPos = false;

m\_pSurfaceVertex = new DPOINT;

if (!m\_pErosionROI\_MCOGS\_CSDlg)

{

CreateGetGSInfo();

CreateGetBSInfo();

if(m\_pMCCGSObjINFO->pImageArray)

{

m\_pErosionROI\_MCOGS\_CSDlg = new CDialogCSServer(g\_pMouseClickSubject);

//同步更新globla local cserver

if(!m\_pLocalCSObsPack)

m\_pLocalCSObsPack = new FunctionPack<CDialogCSServer>;

m\_pLocalCSObsPack->p = m\_pErosionROI\_MCOGS\_CSDlg;

m\_pLocalCSObsPack->pmf = &CDialogCSServer::UpdateClickPos;

g\_pMouseClickSubject->Attach(intermediary<CDialogCSServer>,m\_pLocalCSObsPack);

LONGLONG pImage=reinterpret\_cast<LONGLONG>(m\_pMCCGSObjINFO->pImageArray);

m\_pErosionROI\_MCOGS\_CSDlg->Create(IDD\_DIALOG\_CSSERVER);

m\_pErosionROI\_MCOGS\_CSDlg->ShowWindow(SW\_SHOW);

m\_pErosionROI\_MCOGS\_CSDlg->SetGSProject(pImage,m\_pMCCGSObjINFO->nImageWidth,m\_pMCCGSObjINFO->nImageHeight,m\_pMCCGSObjINFO->nImageCount,m\_pMCCGSObjINFO->dWindowCenter,m\_pMCCGSObjINFO->dWindowWidth,1,m\_pMCCGSObjINFO->nImgBitAlloc);

m\_pErosionROI\_MCOGS\_CSDlg->SetCutSphereRadius(m\_pMCOCutToolInfo->nRadius);

m\_pErosionROI\_MCOGS\_CSDlg->SetInterpolatedInfo(m\_pMCCGSObjINFO->m\_bIsInterpolated,m\_pMCCGSObjINFO->m\_fZoom,m\_pMCCGSObjINFO->m\_nOffset);

if(m\_pMCCGSObjINFO->m\_bIsInterpolated)

{

m\_pErosionROI\_MCOGS\_CSDlg->SetID("Local");

m\_pErosionROI\_MCOGS\_CSDlg->SetWindowText("Local");

}

else

{

m\_pErosionROI\_MCOGS\_CSDlg->SetWindowText("Global");

m\_pErosionROI\_MCOGS\_CSDlg->SetID("Global");

}

m\_pErosionROI\_MCOGS\_CSDlg->SetPosition((long)m\_pMCCGSObjINFO->nImageWidth/2,(long)m\_pMCCGSObjINFO->nImageHeight/2,(long)m\_pMCCGSObjINFO->nImageCount/2);

m\_pErosionROI\_MCOGS\_CSDlg->SetParentOECtl(this);

m\_pErosionROI\_MCOGS\_CSDlg->EnableShow3DPlan(true);

m\_pErosionROI\_MCOGS\_CSDlg->EnableErosionROI(true);

CreateGSBSInfoForCServer();

}

}

else

{

m\_pErosionROI\_MCOGS\_CSDlg->ShowWindow(SW\_SHOW);

m\_pErosionROI\_MCOGS\_CSDlg->SetCutSphereRadius(m\_pMCOCutToolInfo->nRadius);

CreateGSBSInfoForCServer();

}

if (!m\_pMCOManipulation)

{

m\_pMCOManipulation= new CMCOManipulation(this,CMCOManipulation::TextureMapping);

m\_pMCOManipulation->SetCSDialog(m\_pErosionROI\_MCOGS\_CSDlg);

m\_pMCOManipulation->SetGSDirection(chGSDir);

m\_pMCOManipulation->SetGSInfo(m\_pMCCGSObjINFO);

}

m\_matInitWorld = m\_mSceneScale \* m\_pSelectCutFrame->GetMatrixByType(C3DBaseFrame::COMBINEMATRIX) \* m\_matInverseYAxis;

ROIViewDir();

SetFocus();

return;

}

void CObjectExhibitCtrl::ROIViewDir()

{

//View direction of ROI

if(m\_pErosionROI\_MCOGS\_CSDlg)

{

if(!m\_pSelectCutFrame) return;

D3DXVECTOR3 ViewDir,RightDir;

D3DXMATRIX tempWorld;

D3DXVECTOR3 vtPickRayOrig, vtPickRayDir;

tempWorld = m\_mSceneScale \* m\_pSelectCutFrame->GetMatrixByType(C3DBaseFrame::COMBINEMATRIX) \* m\_matInverseYAxis;

CalculateMouseDownRay(1,1, tempWorld, vtPickRayOrig, vtPickRayDir);

D3DXVec3Normalize(&vtPickRayOrig,&vtPickRayOrig);

vtPickRayOrig\*=1.2;

if(m\_pDynamicCamera)

{

vtPickRayOrig = m\_pDynamicCamera->m\_VTCameraView;

vtPickRayOrig.z = -vtPickRayOrig.z;

D3DXVec3Normalize(&vtPickRayOrig,&vtPickRayOrig);

vtPickRayOrig\*= 1.2;

}

vtPickRayOrig.x = (!m\_pDynamicCamera) ? -vtPickRayOrig.x : vtPickRayOrig.x;

ClickPosition ClickData = g\_pMouseClickSubject->GetData();

ClickData.ClickVector = vtPickRayOrig;

g\_pMouseClickSubject->SetData(m\_sOberverID,ClickData);

}