```
// MeshEditorDoc.cpp : CMeshEditorDoc クラスの動作の定義を行います。
//

#include "stdafx.h"
#include "MeshEditor.h"

#include "MeshEditorDoc.h"
#include "MeshEditorView.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/////////////////////////////////////////////////////////////////////////////
// CMeshEditorDoc

IMPLEMENT_DYNCREATE(CMeshEditorDoc, CDocument)

BEGIN_MESSAGE_MAP(CMeshEditorDoc, CDocument)
    //{{AFX_MSG_MAP(CMeshEditorDoc)
    ON_COMMAND(ID_FILE_OPEN, OnFileOpen)
    ON_COMMAND(ID_WIRE, OnWire)
    ON_COMMAND(ID_VISUAL, OnVisual)
    ON_COMMAND(ID_AD_SMOOTH, OnAdSmooth)
    ON_COMMAND(ID_ANGLE, OnAngle)
    ON_COMMAND(ID_EDGE, OnEdge)
    ON_COMMAND(ID_ENHANCE, OnEnhance)
    ON_COMMAND(ID_FILE_SAVE, OnFileSave)
    ON_COMMAND(ID_FILE_SAVE_AS, OnFileSaveAs)
    ON_COMMAND(ID_FLAT, OnFlat)
    ON_COMMAND(ID_RIDGE, OnRidge)
    ON_COMMAND(ID_SIMPLIFY, OnSimplify)
    ON_COMMAND(ID_SMOOTH, OnSmooth)
    ON_COMMAND(ID_SUBDIVIDE, OnSubdivide)
    ON_COMMAND(ID_SMOOTH_SHADE, OnSmoothShade)
    ON_COMMAND(ID_PRINCIPAL, OnPrincipal)
    //}}AFX_MSG_MAP
END_MESSAGE_MAP()

static char file_ex[] = "All files(*.*)|*.*||";

/////////////////////////////////////////////////////////////////////////////
// CMeshEditorDoc クラスの構築/消滅

CMeshEditorDoc::CMeshEditorDoc()
{
    file_manager = new FileManager;
    feature_detector = new FeatureDetector;
    smoother = new Smoother;
    simplifier = new Simplifier;
    mesh = NULL;

    ridge_dialog = new RidgeDialog(NULL);
    smooth_dialog = new SmoothDialog(NULL);
```

```
    visual_dialog = new Visualization(NULL);
}

CMeshEditorDoc::~CMeshEditorDoc()
{
    if(mesh != NULL)
        delete mesh;
    delete file_manager;
    delete smoother;
    delete feature_detector;
    delete simplifier;

    delete ridge_dialog;
    delete smooth_dialog;
    delete visual_dialog;
}

BOOL CMeshEditorDoc::OnNewDocument()
{
    if (!CDocument::OnNewDocument())
        return FALSE;

    return TRUE;
}



/////////////////////////////////////////////////////////////////////////////
// CMeshEditorDoc シリアライゼーション

void CMeshEditorDoc::Serialize(CArchive& ar)
{
    if (ar.IsStoring())
    {
    }
    else
    {
    }
}

/////////////////////////////////////////////////////////////////////////////
// CMeshEditorDoc クラスの診断

#ifdef _DEBUG
void CMeshEditorDoc::AssertValid() const
{
    CDocument::AssertValid();
}

void CMeshEditorDoc::Dump(CDumpContext& dc) const
{
    CDocument::Dump(dc);
}
#endif //_DEBUG

/////////////////////////////////////////////////////////////////////////////
```

```
// CMeshEditorDoc コマンド

void CMeshEditorDoc::OnFileOpen()
{
    // TODO: この位置にコマンド ハンドラ用のコードを追加してください
    CFileDialog* dia = new CFileDialog(TRUE, NULL, NULL,
        OFN_HIDEREADONLY | OFN_OVERWRITEPROMPT, file_ex, NULL);

    if(dia->DoModal() == IDOK){
        if(mesh != NULL)
            delete mesh;
        mesh = new MeshData;

        CString file_name = dia->GetPathName();
        CString file_ext = dia->GetFileExt();
        FILE* file = fopen(file_name,"r");

        //Read mesh data from file
        //and create adjacent data for the mesh
        file_manager->setFile(file, file_name, file_ext);
        file_manager->open(mesh);

        fclose(file);

        smoother->setMeshData(mesh);
        feature_detector->setMeshData(mesh);
        //simplifier->setMeshData(mesh);
        //subdivder->setMeshData(mesh);

        //smoother->computeNRingMeanCurvature(3);

        mesh->computeFaceNormal();
        this->drawMesh(FLAT_SHADING);
        //this->drawMesh(COLOR);
    }
    delete dia;
}

void CMeshEditorDoc::drawMesh(int vis_mode)
{
    POSITION pos = GetFirstViewPosition();
    CMeshEditorView* pView = (CMeshEditorView*)GetNextView(pos);
    switch(vis_mode){
    case FLAT_SHADING:
        pView->flatShade(mesh);
        break;
    case SMOOTH_SHADING:
        pView->smoothShade(mesh);
        break;
    case WIRE_FRAME:
        pView->flatShade(mesh);
        break;
    case CURVATURE_COLOR:
        pView->flatShade(mesh);
        break;
    case PRINCIPAL_DIRECTION:
```

```
        pView->drawPrincipalDir(mesh, true, true, visual_dialog->m_transparent, visual_dialog->m_grey_scale);
        break;
    case RIDGE_PRINCIPAL:
        pView->drawRidgeVertexAsSegmet(mesh, visual_dialog->m_transparent, visual_dialog->m_grey_scale, true);
        break;
    case RIDGE_RIDGE:
        pView->drawRidgeVertexAsSegmet(mesh, visual_dialog->m_transparent, visual_dialog->m_grey_scale, false);
        break;
    case RIDGE_EDGE:
        pView->drawRidgeEdge(mesh, visual_dialog->m_transparent, visual_dialog->m_grey_scale);
        break;
    case RIDGE_FACE:
        //pView->drawRidgeFace(mesh, true, false);
        break;
    case COLOR:
        pView->drawColoredVertex(mesh, smoother->T);
    }
    pView->Invalidate(TRUE);
}

void CMeshEditorDoc::OnWire()
{
    // TODO: この位置にコマンド ハンドラ用のコードを追加してください

}

void CMeshEditorDoc::OnVisual()
{
    // TODO: この位置にコマンド ハンドラ用のコードを追加してください
    if(visual_dialog->DoModal() == IDOK){

    }
}

void CMeshEditorDoc::OnAdSmooth()
{
    // TODO: この位置にコマンド ハンドラ用のコードを追加してください
}

void CMeshEditorDoc::OnAngle()
{
    // TODO: この位置にコマンド ハンドラ用のコードを追加してください

}

void CMeshEditorDoc::OnEdge()
{
    // TODO: この位置にコマンド ハンドラ用のコードを追加してください
    if(mesh != NULL)
        drawMesh(RIDGE_PRINCIPAL);
}
```

```
void CMeshEditorDoc::OnEnhance()
{
    // TODO: この位置にコマンド ハンドラ用のコードを追加してください

}

void CMeshEditorDoc::OnFileSave()
{
    // TODO: この位置にコマンド ハンドラ用のコードを追加してください

}

void CMeshEditorDoc::OnFileSaveAs()
{
    // TODO: この位置にコマンド ハンドラ用のコードを追加してください

}

void CMeshEditorDoc::OnFlat()
{
    // TODO: この位置にコマンド ハンドラ用のコードを追加してください
    if(mesh != NULL){
        mesh->computeFaceNormal();
        this->drawMesh(FLAT_SHADING);
    }
}

void CMeshEditorDoc::OnRidge()
{
    // TODO: この位置にコマンド ハンドラ用のコードを追加してください
    if(mesh == NULL)
        return;

    if(ridge_dialog->DoModal() == IDOK){
        //pre-smoothing
        switch(ridge_dialog->m_pre_smoothing){
        case 0:
            break;
        case 1:
            break;
        case 2:
            break;
        }

        //detection of local curvature max/min along curvature line
        mesh->computeNormal();
        mesh->computePrincipal();
        feature_detector->ridge();

        //Thresholding
        switch(ridge_dialog->m_thresholding){
        case 0:
            feature_detector->constantThresholding(0);
            break;
        case 1:
            feature_detector->automaticThresholding(60,30);
```

```
            break;
        case 2:
            feature_detector->constantThresholding(ridge_dialog->m_constant);
            break;
        }

        //Visualization
        switch(ridge_dialog->m_data_type){
        case 0:
            if(ridge_dialog->m_ridge_dir)
                drawMesh(RIDGE_RIDGE);
            else
                drawMesh(RIDGE_PRINCIPAL);
            break;
        case 1:
            drawMesh(RIDGE_EDGE);
            break;
        case 2:
            if(ridge_dialog->m_extraction){
                //
                //
            }
            drawMesh(RIDGE_FACE);
            break;
        }
    }
}

void CMeshEditorDoc::OnSimplify()
{
    // TODO: この位置にコマンド ハンドラ用のコードを追加してください

}

void CMeshEditorDoc::OnSmooth()
{
    // TODO: この位置にコマンド ハンドラ用のコードを追加してください
    if(mesh == NULL)
        return;

    if(smooth_dialog->DoModal() == IDOK){
        CString method = smooth_dialog->m_method;
        int iteration = smooth_dialog->m_iteration;

        if(method == "Laplacian Flow")
            smoother->LaplacianFlow(smooth_dialog->m_iteration, smooth_dialog-
>m_step);
        else if(method == "Mean Curvature Flow (Desbrun)")
            //smoother->meanCurvatureFlow(smooth_dialog->m_iteration, smooth_d
ialog->m_step);
            smoother->meanCurvatureFlow12Ring(smooth_dialog->m_iteration, smoo
th_dialog->m_step, 0.2f);

        OnFlat();
    }
}
```

```
void CMeshEditorDoc::OnSubdivide()
{
	// TODO: この位置にコマンド ハンドラ用のコードを追加してください
}

void CMeshEditorDoc::OnSmoothShade()
{
	// TODO: この位置にコマンド ハンドラ用のコードを追加してください
	if(mesh != NULL){
		mesh->computeNormal();
		drawMesh(SMOOTH_SHADING);
	}
}

void CMeshEditorDoc::OnPrincipal()
{
	// TODO: この位置にコマンド ハンドラ用のコードを追加してください
	if(mesh != NULL){
		mesh->computePrincipal();
		drawMesh(PRINCIPAL_DIRECTION);
	}
}
```