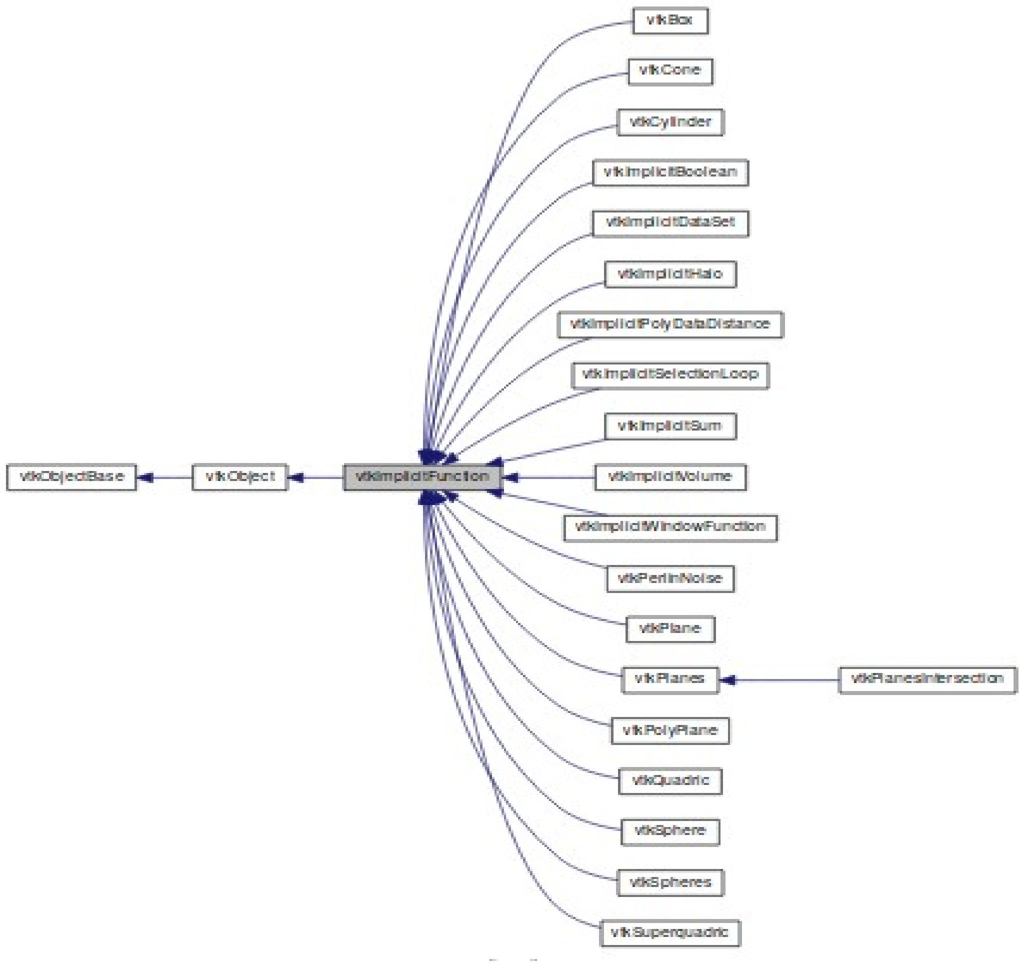
TP1

Représentation de formes 3D

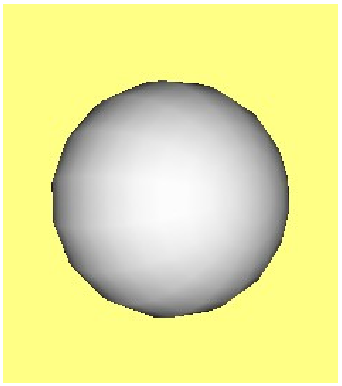
1. Enumérer et représenter toutes les classes vtk qui permettent la représentation d'objets synthétiques (Superquadriques, ...).



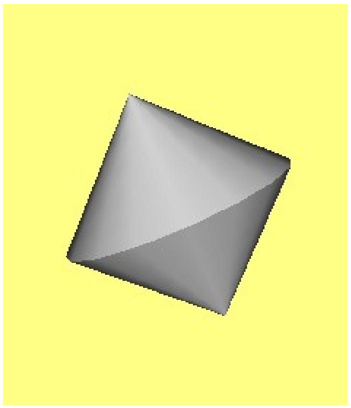
```
#include <vtkCubeSource.h>

#include <vtkSphereSource.h>

#include <vtkConeSource.h>

#include <vtkCylinderSource.h>
```

```
#include <vtkPolyData.h>

#include <vtkPolyDataMapper.h>

#include <vtkActor.h>

#include <vtkRenderWindow.h>

#include <vtkRenderer.h>
#include <vtkRenderWindowInteractor.h>

#include <vtkProperty.h>

#include <vtkSuperquadric.h>

#include <vtkSuperquadricSource.h>

int main()

{

vtkSuperquadricSource *sq = vtkSuperquadricSource::New();

sq->SetThetaRoundness(2);//epsilon 1

sq->SetPhiRoundness(1); //epsilon 2

//mapper

vtkPolyDataMapper *mapper = vtkPolyDataMapper::New();

mapper->SetInputConnection(sq->GetOutputPort());

//actor

vtkActor *actor = vtkActor::New();

actor->SetMapper(mapper);

//renderer

vtkRenderer *renderer = vtkRenderer::New();

renderer->AddActor(actor);

renderer->SetBackground(2, 1, 0.5);

//window

vtkRenderWindow *renderWindow = vtkRenderWindow::New();

renderWindow->SetWindowName("Cube");

renderWindow->AddRenderer(renderer);

//an interactor

vtkRenderWindowInteractor *iren = vtkRenderWindowInteractor::New();
iren->SetRenderWindow(renderWindow);

//start rendering

renderWindow->Render();

iren->Start();

return 0;

}
```

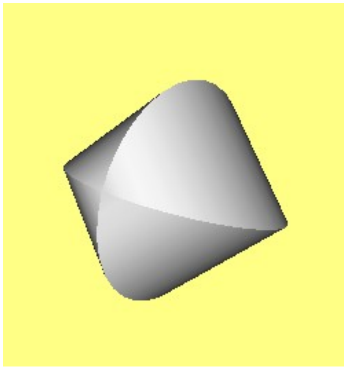
E1=1 ; E2=1



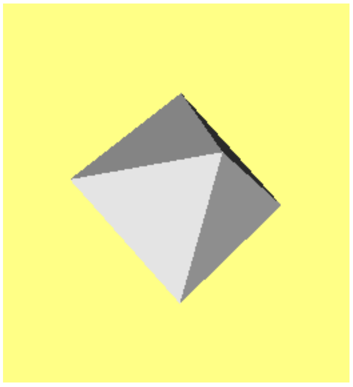
E1=1 ; E2=2 ;



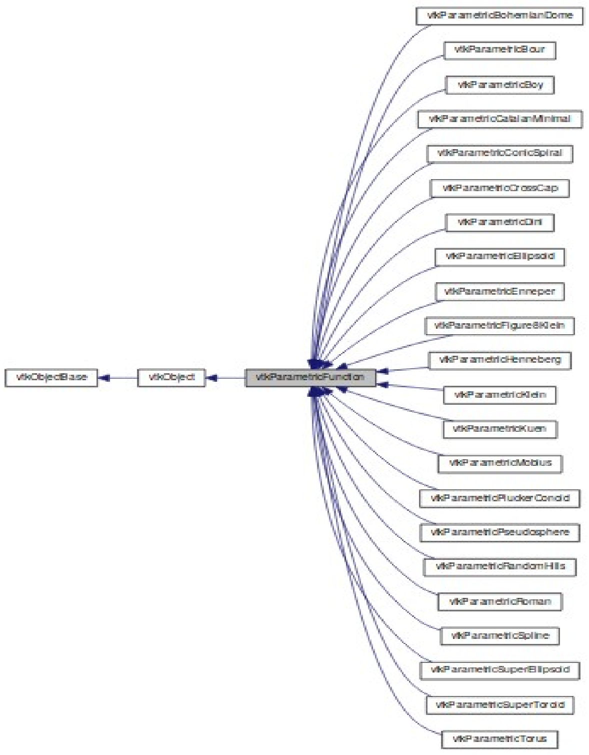
E1=2 ; E2=1



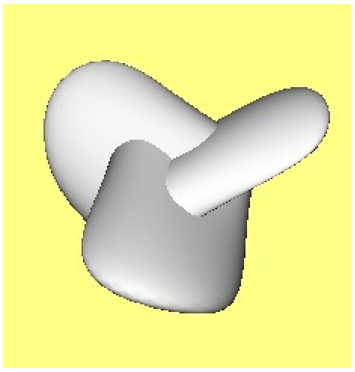
E2=2 ; E2=2



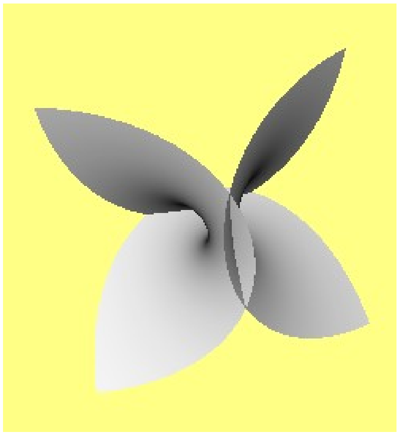
2. Étudier la représentation des surfaces paramétriques dans vtk. Traiter quelques exemples vus dans le cours.



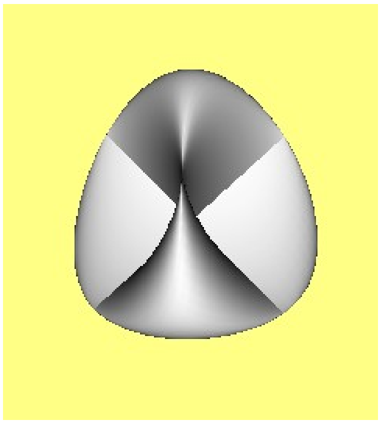
```
#include <vtkCubeSource.h>

#include <vtkSphereSource.h>

#include <vtkConeSource.h>

#include <vtkCylinderSource.h>

#include <vtkPolyData.h>

#include <vtkPolyDataMapper.h>

#include <vtkActor.h>

#include <vtkRenderWindow.h>

#include <vtkRenderer.h>

#include <vtkRenderWindowInteractor.h>

#include <vtkProperty.h>

#include <vtkSuperquadric.h>

#include <vtkSuperquadricSource.h>

#include <vtkPlatonicSolidSource.h>

#include <vtkEllipticalButtonSource.h>

#include <vtkParametricFunctionSource.h>

#include <vtkParametricConicSpiral.h>

#include <vtkParametricDini.h>

#include <vtkParametricEnneper.h>

#include <vtkParametricBoy.h>

#include <vtkParametricKlein.h>

#include <vtkParametricMobius.h>
#include <vtkParametricSpline.h>
#include <vtkParametricRoman.h>

int main()

{

vtkParametricConicSpiral *cs = vtkParametricConicSpiral::New();

vtkParametricDini *ds = vtkParametricDini::New();
vtkParametricKlein *hl=vtkParametricKlein::New();
vtkParametricMobius *ml=vtkParametricMobius::New();
vtkParametricRoman *sl=vtkParametricRoman::New();
vtkParametricEnneper *en = vtkParametricEnneper::New();
 vtkParametricBoy*dl=vtkParametricBoy::New();
vtkParametricFunctionSource *pf = vtkParametricFunctionSource::New();

pf->SetParametricFunction(en);

//mapper

vtkPolyDataMapper *mapper = vtkPolyDataMapper::New();

mapper->SetInputConnection(pf->GetOutputPort());

//actor

vtkActor *actor = vtkActor::New();

actor->SetMapper(mapper);

//renderer

vtkRenderer *renderer = vtkRenderer::New();

renderer->AddActor(actor);

renderer->SetBackground(2,1,0.5);

//window

vtkRenderWindow *renderWindow = vtkRenderWindow::New();

renderWindow->SetWindowName("Cube");

renderWindow->AddRenderer(renderer);

//an interactor

vtkRenderWindowInteractor *iren = vtkRenderWindowInteractor::New();

iren->SetRenderWindow(renderWindow);

//start rendering

renderWindow->Render();

iren->Start();

return 0;

}
```



boy



enneper



roman