Научная визуализация с помощью VTK - 2

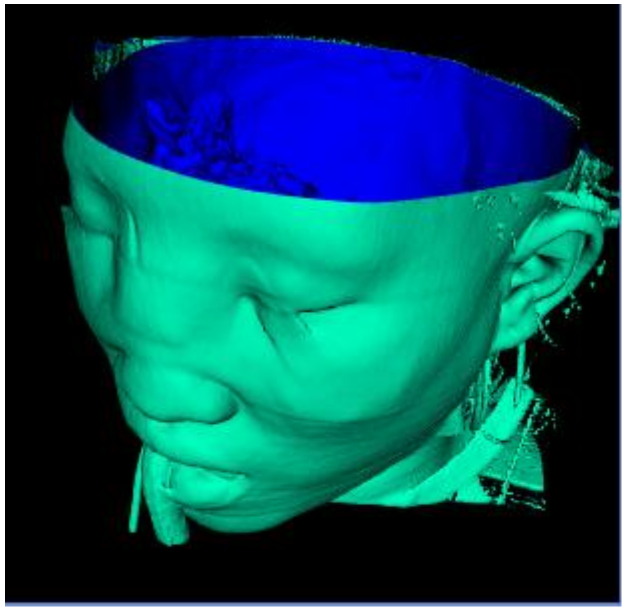
Васильев Евгений

ИИТММ ННГУ

Содержание

Изоповерхности

```
vtkSmartPointer<vtkContourFilter> contours =
vtkSmartPointer<vtkContourFilter>::New();
contours->SetInputConnection(reader->GetOutputPort());
contours->GenerateValues(4, 200.0, 400.0);

vtkSmartPointer<vtkPolyDataMapper> mapper =
vtkSmartPointer<vtkPolyDataMapper>::New();
mapper->SetInputConnection(contours->GetOutputPort());
mapper->SetScalarRange(100.0, 250.0);
```

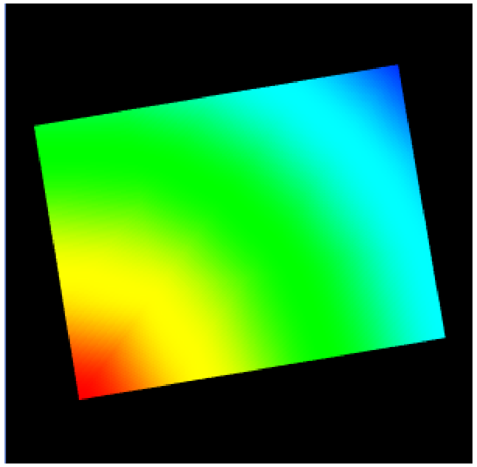


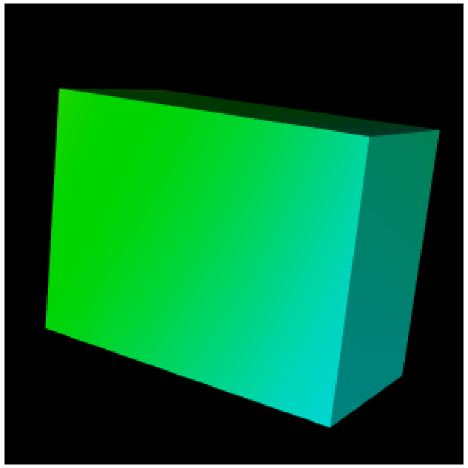
Source: example_contourfilter.cpp

Фильтр Гаусса

```
vtkSmartPointer<vtkImageGaussianSmooth> gauss =
    vtkSmartPointer<vtkImageGaussianSmooth>::New();
gauss->SetInputConnection(reader->GetOutputPort());
gauss->SetRadiusFactor(0.5);
```

Source: [example_gaussfilter.cpp]

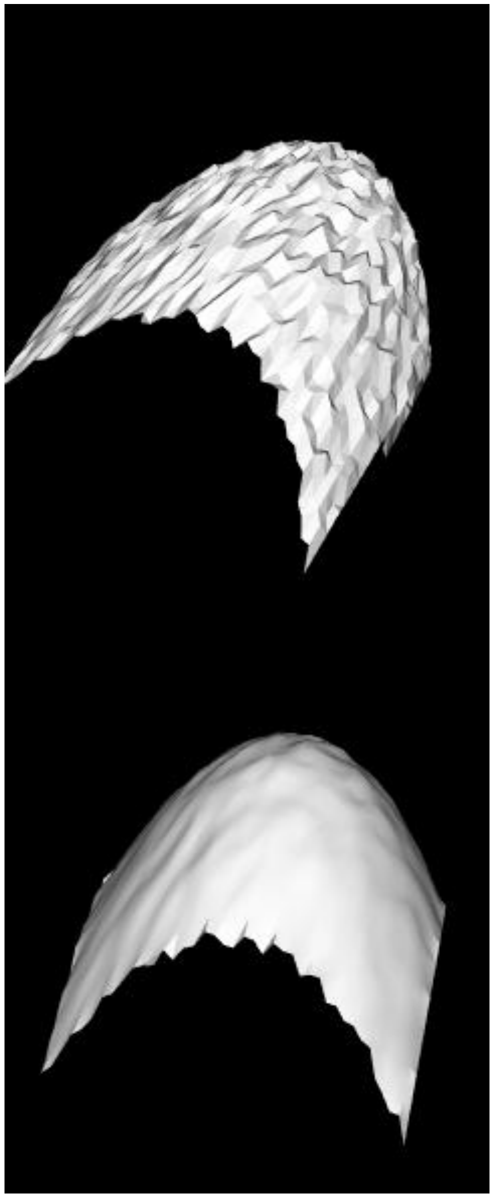




Сглаживание полигонов

```
vtkSmartPointer<vtkDelaunay2D> delaunay =
    vtkSmartPointer<vtkDelaunay2D>::New();
delaunay->SetInputData(inputPolyData);
delaunay->Update();

vtkSmartPointer<vtkSmoothPolyDataFilter> smoothFilter =
    vtkSmartPointer<vtkSmoothPolyDataFilter>::New();
smoothFilter->SetInputConnection(delaunay->GetOutputPort());
smoothFilter->SetNumberOfIterations(2);
smoothFilter->SetRelaxationFactor(0.5);
smoothFilter->FeatureEdgeSmoothingOff();
smoothFilter->BoundarySmoothingOn();
smoothFilter->Update();
```

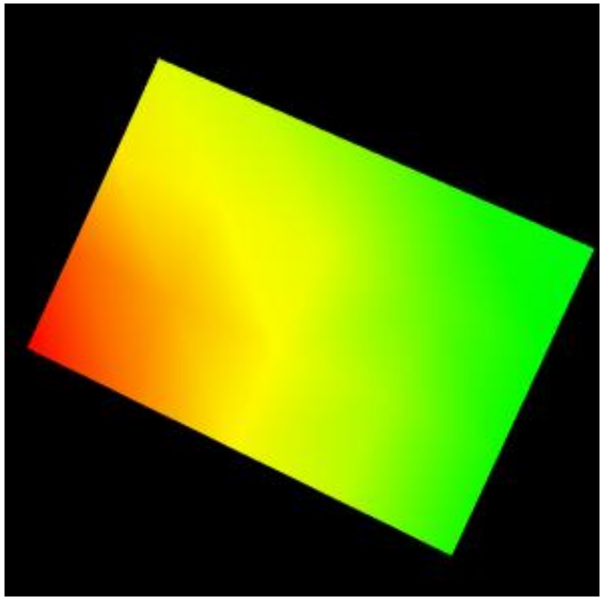
Source: example_smoothpolyfilter.cpp



Срез данных плоскостью

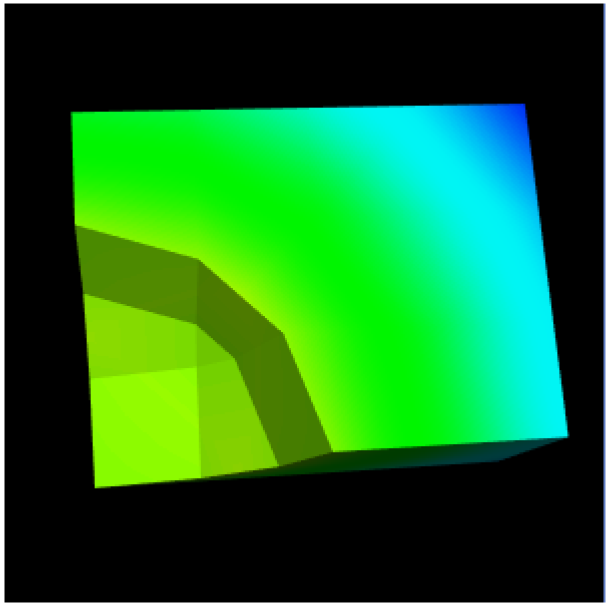
```
vtkSmartPointer<vtkPlane> plane =
    vtkSmartPointer<vtkPlane>::New();
plane->SetOrigin(1.0, 1.5, 2.0);
plane->SetNormal(0.9, 0.0, -0.4);

vtkSmartPointer<vtkCutter> cutter =
    vtkSmartPointer<vtkCutter>::New();
cutter->SetInputConnection(reader->GetOutputPort());
cutter->SetCutFunction(plane);
```

Source: [example_cutterfilter.cpp]



Отсечение данных сферой

```
vtkSmartPointer<vtkSphere> sphere =
    vtkSmartPointer<vtkSphere>::New();
sphere->SetCenter(0.0, 0.0, 0.0);
sphere->SetRadius(2.0);

vtkSmartPointer<vtkClipDataSet> clip =
    vtkSmartPointer<vtkClipDataSet>::New();
clip->SetInputConnection(reader->GetOutputPort());
clip->SetClipFunction(sphere);
clip->InsideOutOff();
clip->Update();
```

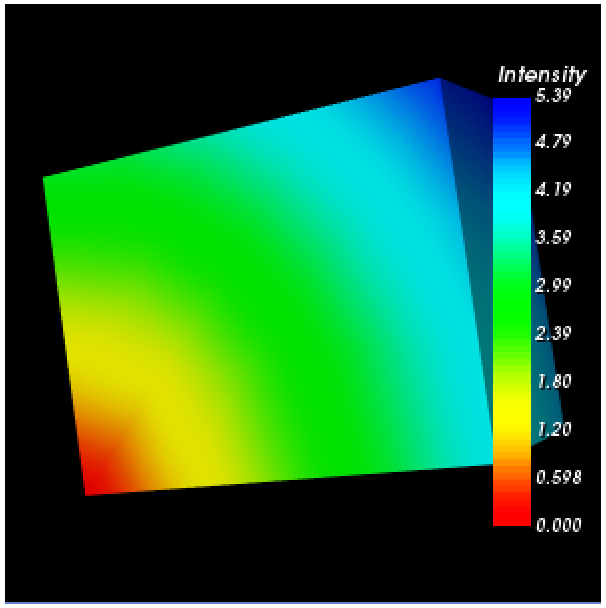


Source: example_clipdata.cpp

График цвета

```
vtkSmartPointer<vtkScalarBarActor> scalarBar =
    vtkSmartPointer<vtkScalarBarActor>::New();
scalarBar->SetLookupTable(mapper->GetLookupTable());
scalarBar->SetTitle("Intensity");
scalarBar->SetNumberOfLabels(10);

vtkSmartPointer<vtkRenderer> renderer =
    vtkSmartPointer<vtkRenderer>::New();
renderer->AddActor(actor);
renderer->AddActor2D(scalarBar);
```

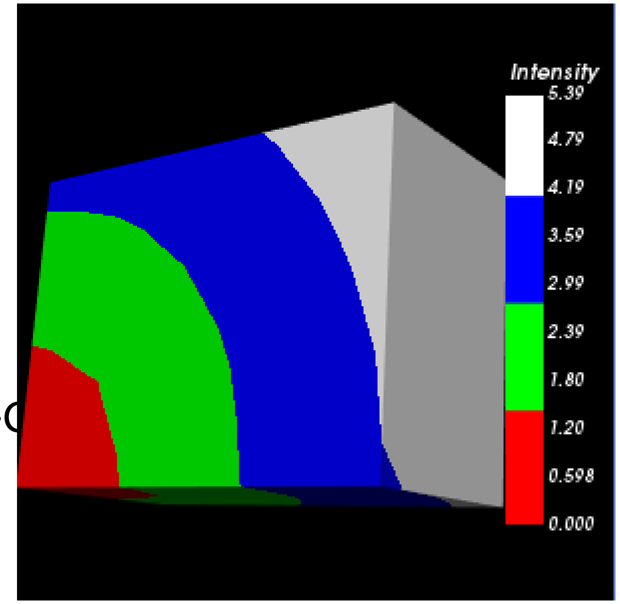
Source: example_scalarbar.cpp



Настройка палитры

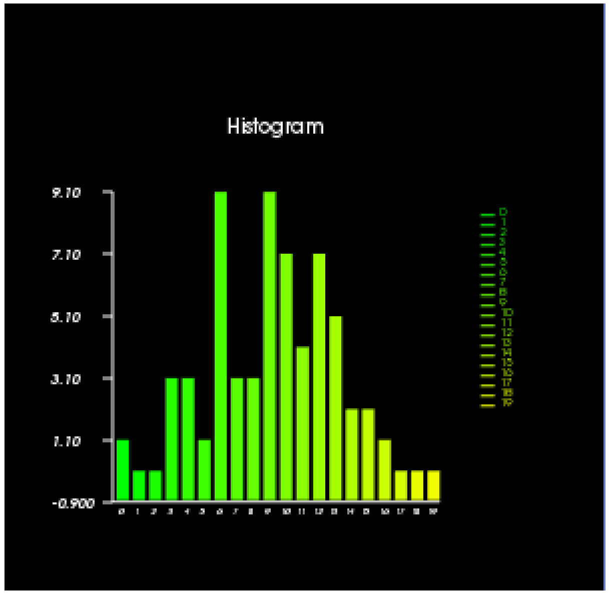
```
vtkSmartPointer<vtkLookupTable> lookuptable =
    vtkSmartPointer<vtkLookupTable>::New();
lookuptable->SetNumberOfTableValues(4);
lookuptable->SetRange(reader->GetOutputAsDataSet()->GetScalarRange());
lookuptable->Build();
lookuptable->SetTableValue(0, 1.0, 0.0, 0.0, 1.0);
lookuptable->SetTableValue(1, 0.0, 1.0, 0.0, 1.0);
lookuptable->SetTableValue(2, 0.0, 0.0, 1.0, 1.0);
lookuptable->SetTableValue(3, 1.0, 1.0, 1.0, 1.0);

mapper->SetScalarRange(reader->GetOutputAsDataSet()->G
mapper->SetLookupTable(lookuptable);
mapper->SetScalarRange(lookuptable->GetRange());
mapper->InterpolateScalarsBeforeMappingOn();
```

Source: [example_lookuptable.cpp]



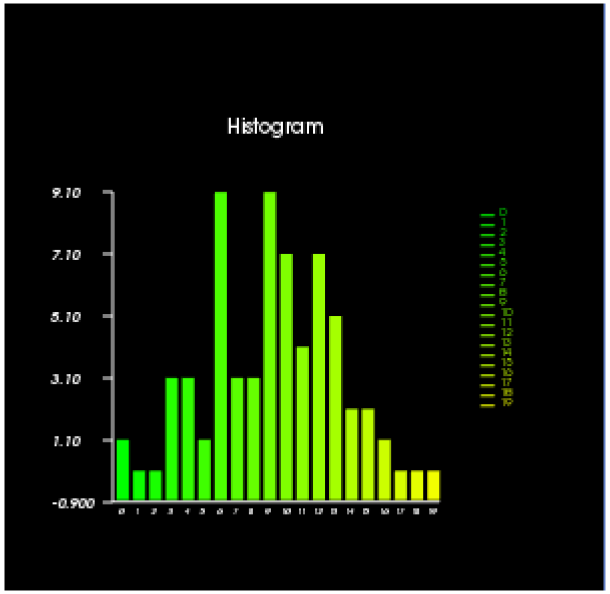
Гистограмма - 1

```
double spacing = 0.33;
int numberOfTuples = 20;

vtkSmartPointer<vtkImageAccumulate> histogram =
    vtkSmartPointer<vtkImageAccumulate>::New();
histogram->SetInputConnection(reader->GetOutputPort());
histogram->SetComponentOrigin(0, 0, 0);
histogram->SetComponentSpacing(spacing, 0, 0);
histogram->Update();
```



Гистограмма - 2

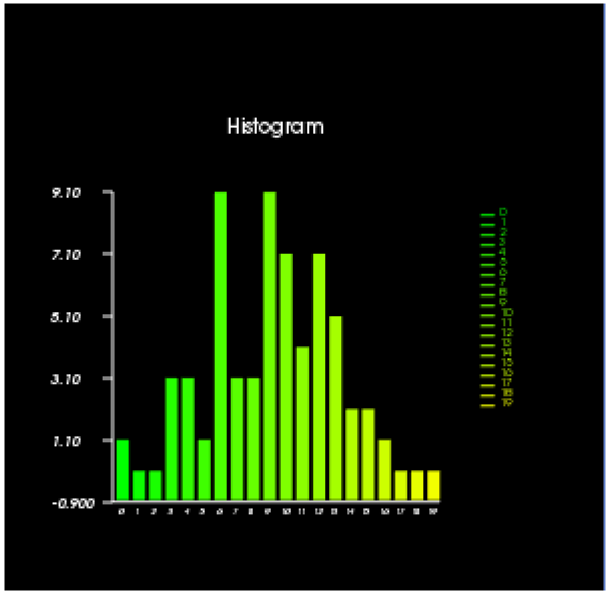
```
vtkSmartPointer<vtkIntArray> frequencies =
    vtkSmartPointer<vtkIntArray>::New();
frequencies->SetNumberOfComponents(1);
frequencies->SetNumberOfTuples(numberOfTuples);
vtkIdType* output = static_cast<vtkIdType*>
    (histogram->GetOutput()->GetScalarPointer());
for (int j = 0; j < numberOfTuples; ++j)
    frequencies->SetTuple1(j, *output++);

vtkSmartPointer<vtkDataObject> dataObject =
vtkSmartPointer<vtkDataObject>::New();
dataObject->GetFieldData()->AddArray(frequencies);
```



Гистограмма - 3

```
vtkSmartPointer<vtkBarChartActor> barChart =
    vtkSmartPointer<vtkBarChartActor>::New();

barChart->SetInput(dataObject);
barChart->SetTitle("Histogram");
barChart->GetLegendActor()->SetNumberOfEntries(
    frequencies ->GetNumberOfTuples());
for (int i = 0; i < numberOfTuples; ++i)
    barChart->SetBarColor
            (i, i/(double)numberOfTuples, 1.0, 0.0);
```

Source: example_histogram.cpp

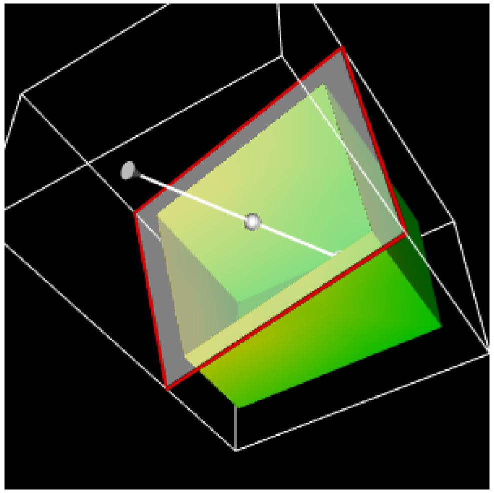


Интерактивный срез

- Создаем собственный интерактор, чтобы управлять плоскостью среза;
- Создаем PlaneWidget – интерактивную плоскость;
- Обрезаем наши данные с помощью этой плоскости;

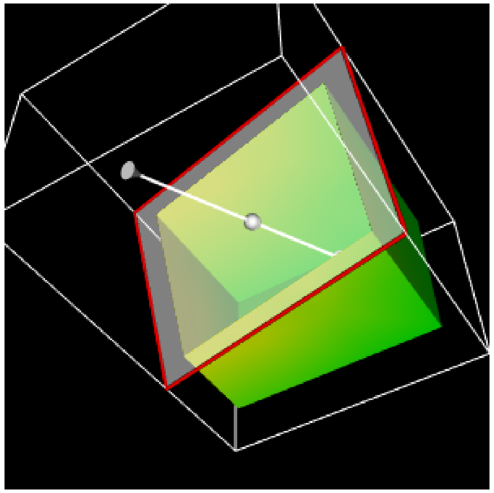
Интерактивный срез - 1

```
class PlaneMoveCallback : public vtkCommand  {
public:
      static PlaneMoveCallback *New()
      {    return new PlaneMoveCallback;    }
      virtual void Execute(vtkObject *caller, unsigned long, void*) {
         vtkImplicitPlaneWidget2 *planeWidget =
         reinterpret_cast<vtkImplicitPlaneWidget2*>(caller);
         vtkImplicitPlaneRepresentation *rep =
         reinterpret_cast<vtkImplicitPlaneRepresentation*>(planeWidget->GetRepresentation());
         rep->GetPlane(this->Plane);
      }
      PlaneMoveCallback() :Plane(0), Actor(0) {}
      vtkPlane *Plane;
      vtkActor *Actor;
};
```



Интерактивный срез - 2

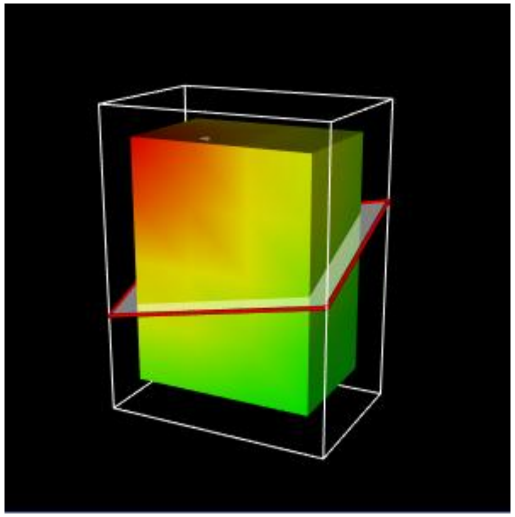
```
vtkSmartPointer<PlaneMoveCallback> myCallback =
    vtkSmartPointer<PlaneMoveCallback>::New();
myCallback->Plane = plane;
myCallback->Actor = actor;
vtkSmartPointer<vtkImplicitPlaneRepresentation> rep =
    vtkSmartPointer<vtkImplicitPlaneRepresentation>::New();
rep->SetPlaceFactor(1.25);
rep->PlaceWidget(actor->GetBounds());
rep->SetNormal(plane->GetNormal());
rep->SetOrigin(plane->GetOrigin());

vtkSmartPointer<vtkImplicitPlaneWidget2> planeWidget =
    vtkSmartPointer<vtkImplicitPlaneWidget2>::New();
planeWidget->SetInteractor(interactor);
planeWidget->SetRepresentation(rep);
planeWidget->AddObserver(vtkCommand::InteractionEvent, myCallback);
planeWidget->On();
```

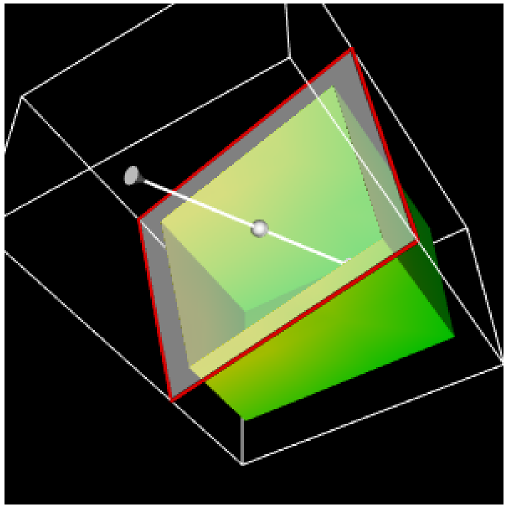


Интерактивный срез - 3

```
vtkSmartPointer<vtkDataSetMapper> mapper =
vtkSmartPointer<vtkDataSetMapper>::New();
    mapper->SetInputConnection(
                    reader->GetOutputPort());
mapper->AddClippingPlane(plane);
```

Source: [example_planecutwidget.cpp]





14.03.2018

Домашнее задание

- Найти и скачать трехмерный датасет;
- Сконвертировать его для VTK;
- Визуализировать;
- Применить любой фильтр;
- Добавить любой виджет;