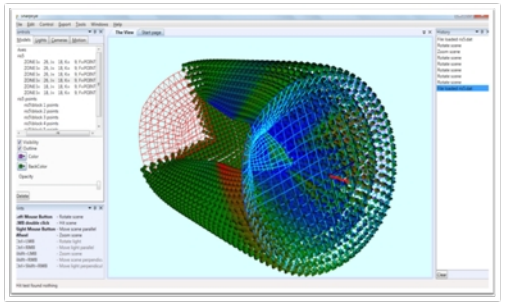
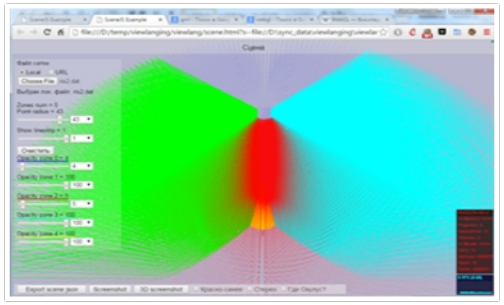
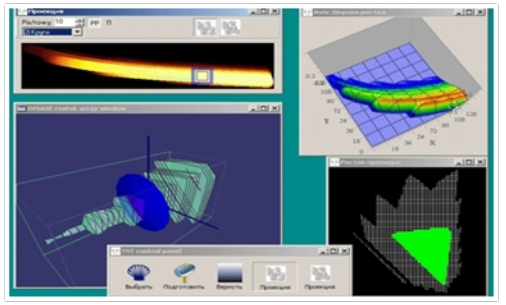
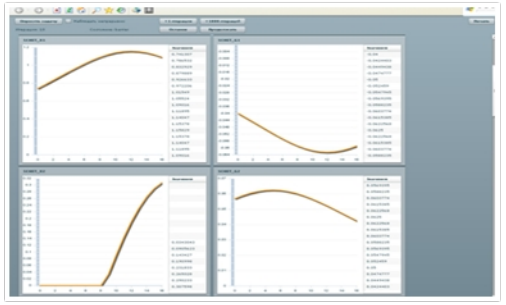
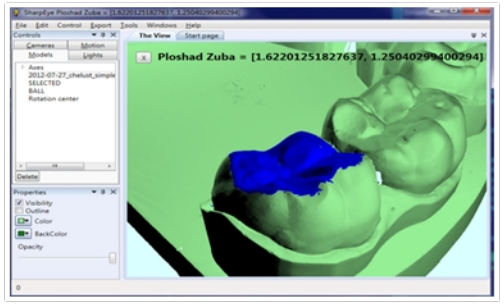
Развитие подходов к разработке специализированных систем компьютерной визуализации

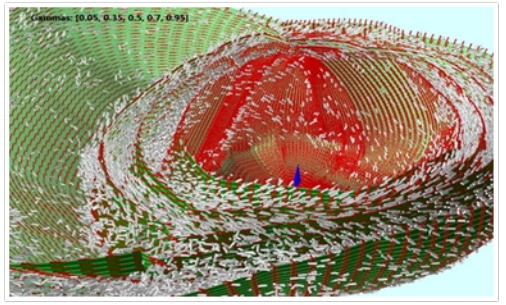
В.Л. Авербух, М.О. Бахтерев, П.А. Васёв,
Д.В. Манаков, И.С. Стародубцев

ИММ УрО РАН, г. Екатеринбург
http://cv.imm.uran.ru

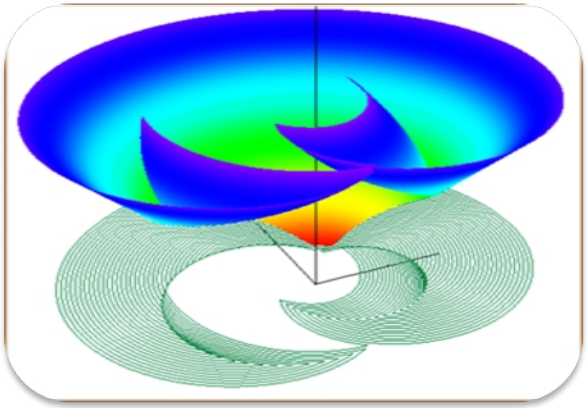
Сектор компьютерной визуализации ИММ УрО РАН

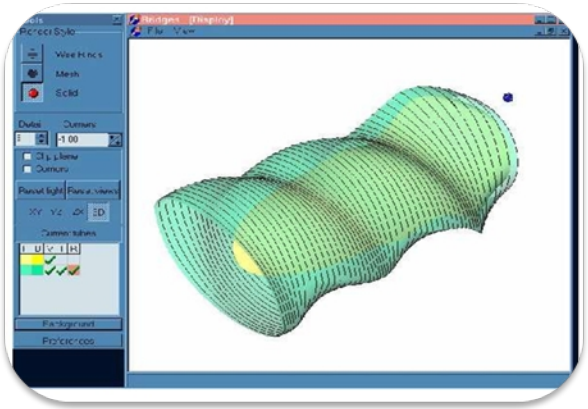
- Разработка специализированных систем визуализации

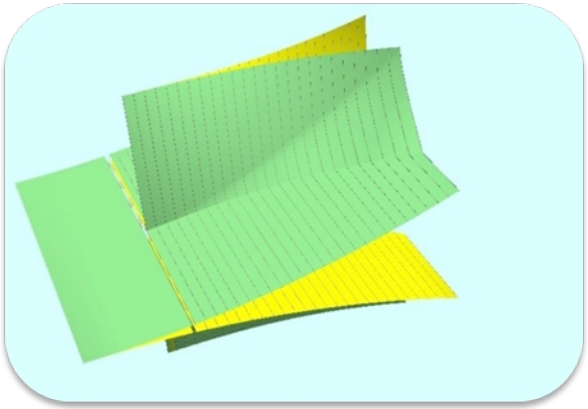






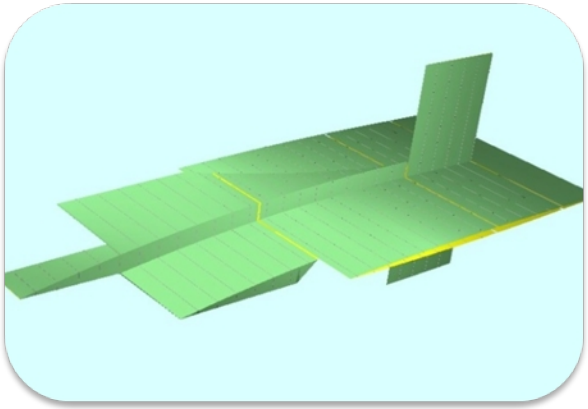


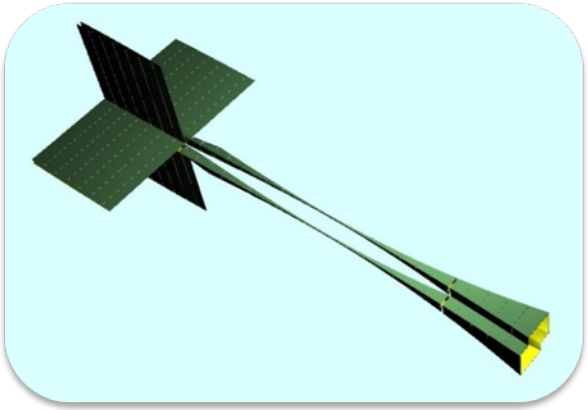
Визуализация максимальных стабильных мостов в линейных дифференциальных играх с двумерной эквивалентной фазовой переменной

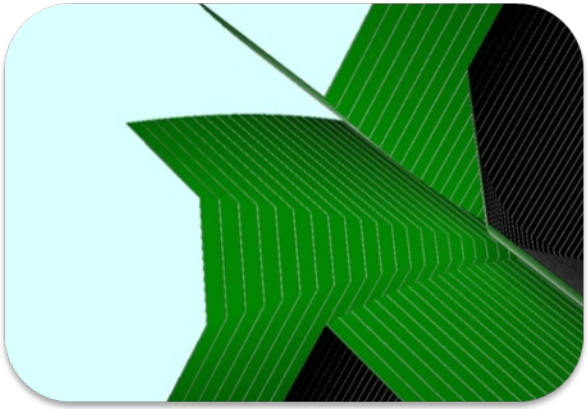






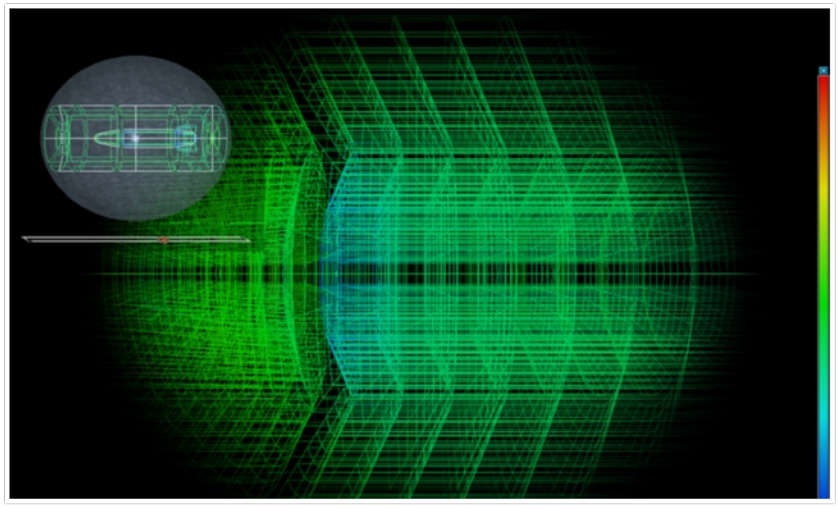


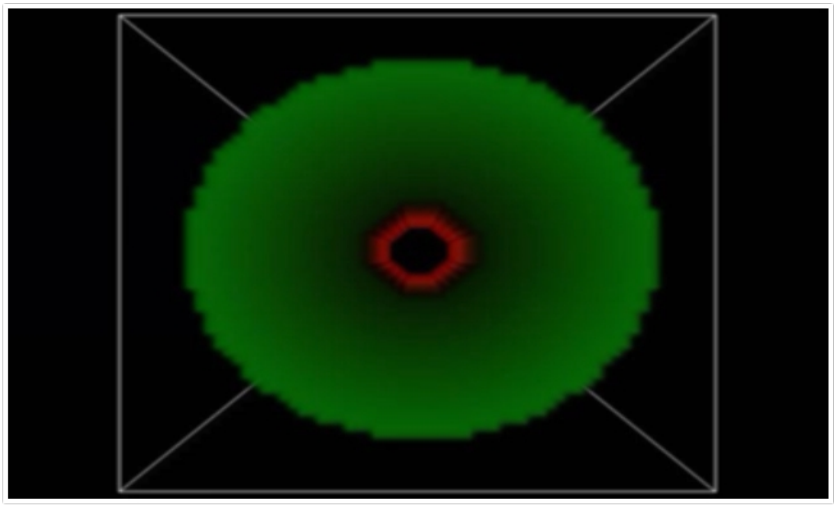


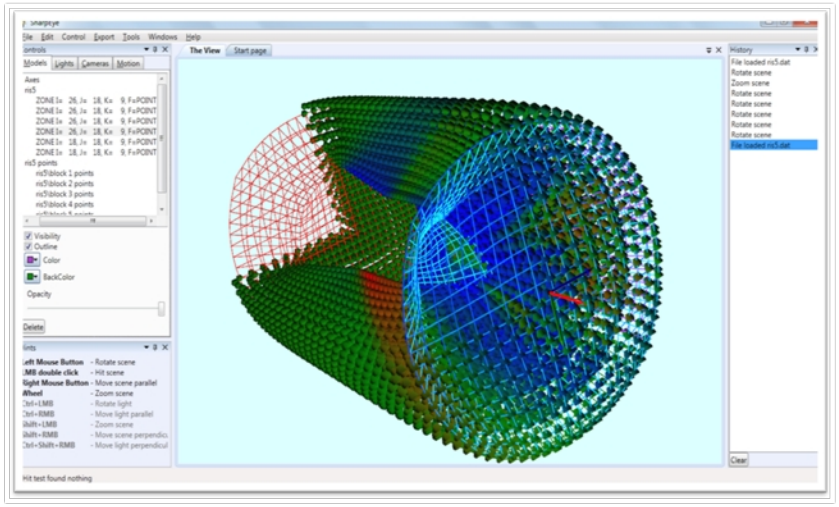


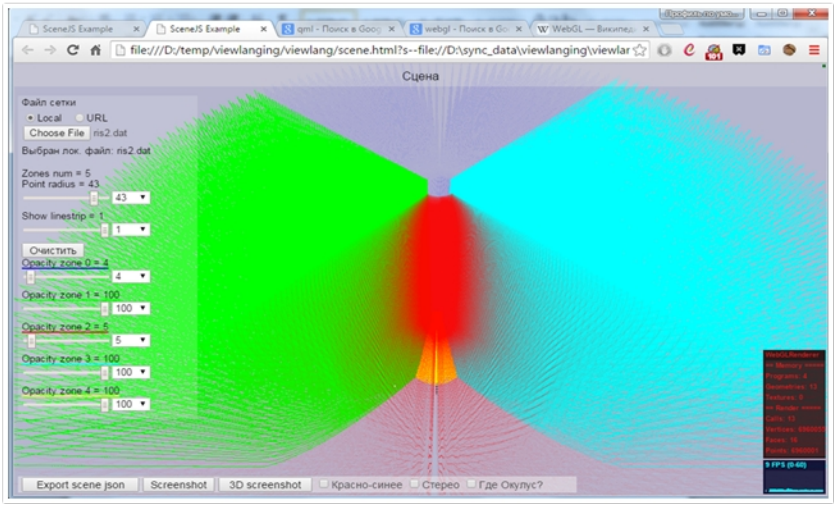
- Ganebny S.A., Kumkov S.S., Le Menec S., Patsko V.S., Model problem in a line with two pursuers and one evader // Dynamic Games and Applications, No.2, 2012, pp. 228–257.

Визуализация сеточных данных

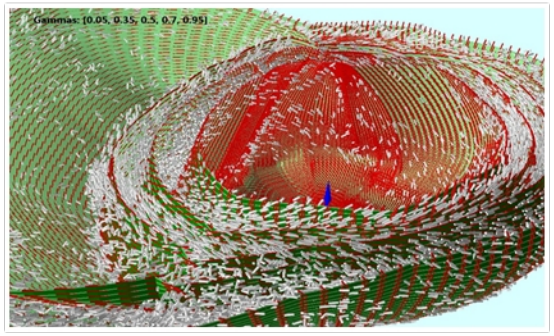


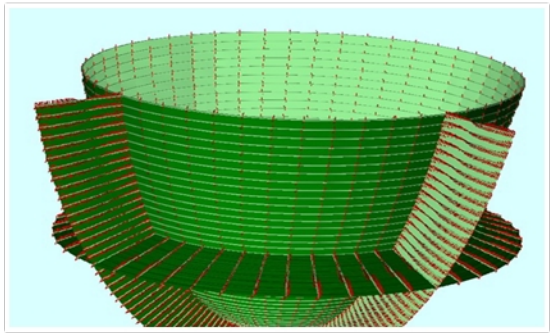


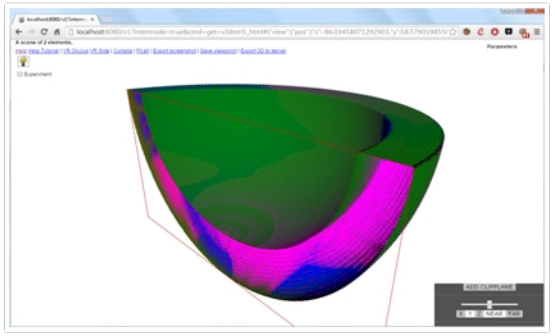


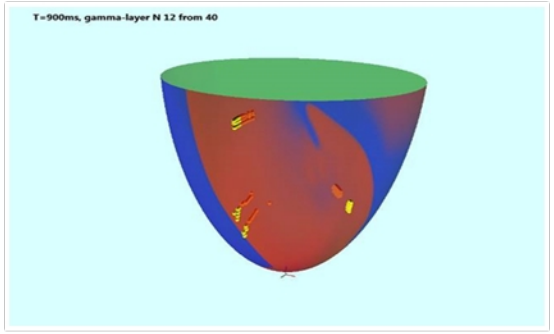


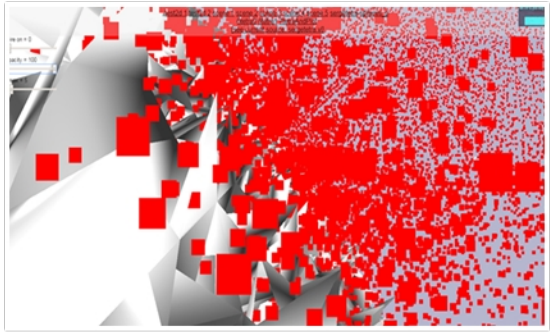
Задачи визуализации при моделировании сердца

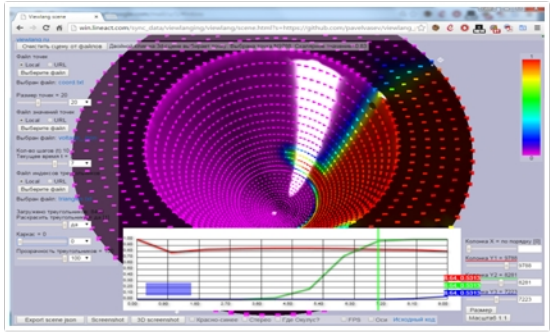






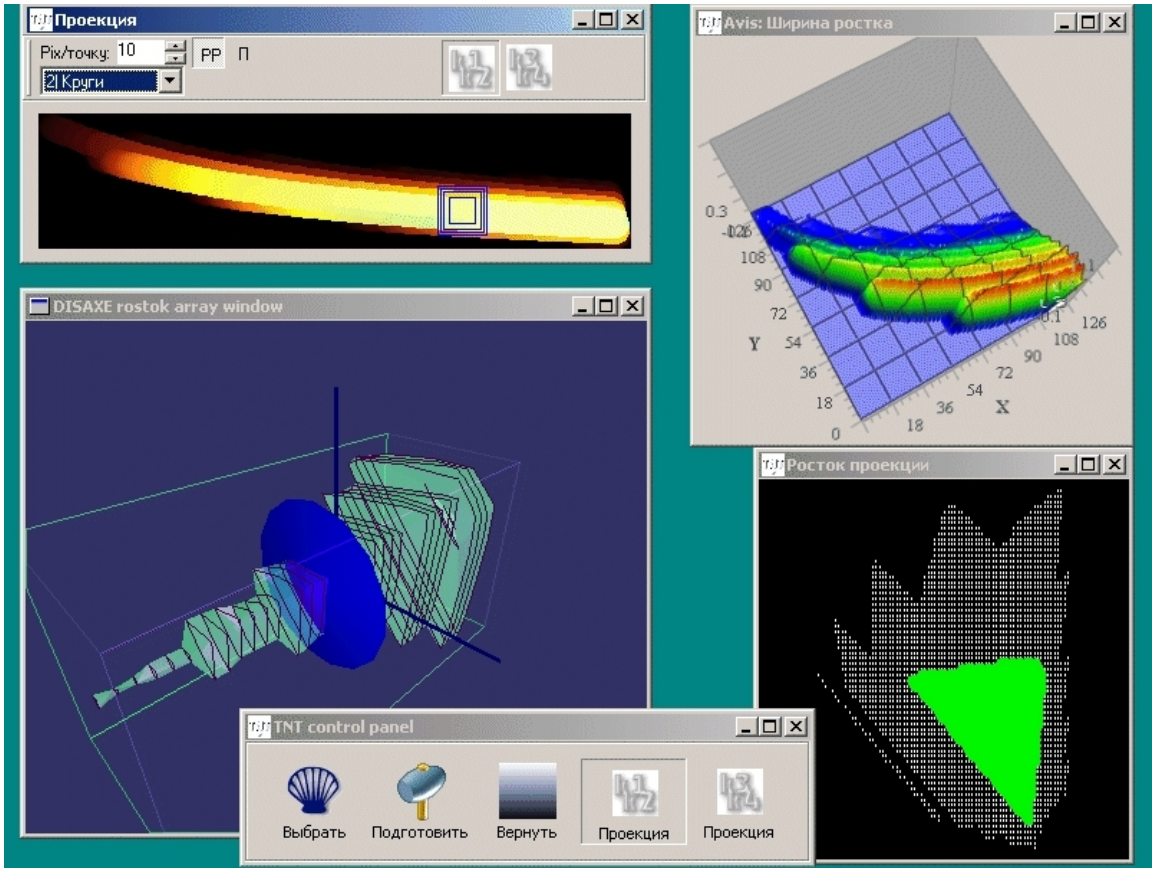


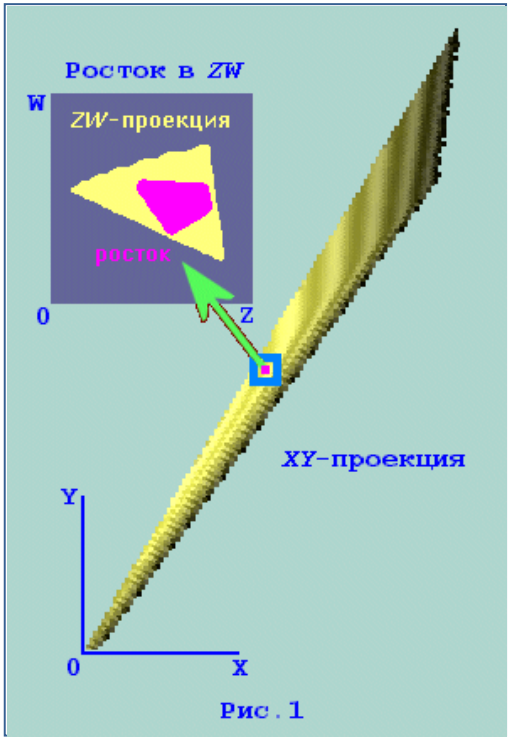




Sergei Pravdin, Hans Dierckx, Vladimir S. Markhasin, and Alexander V. Panfilov, “Drift of Scroll Wave Filaments in an Anisotropic Model of the Left Ventricle of the Human Heart”, BioMed Research International, Article ID 389830, in press.

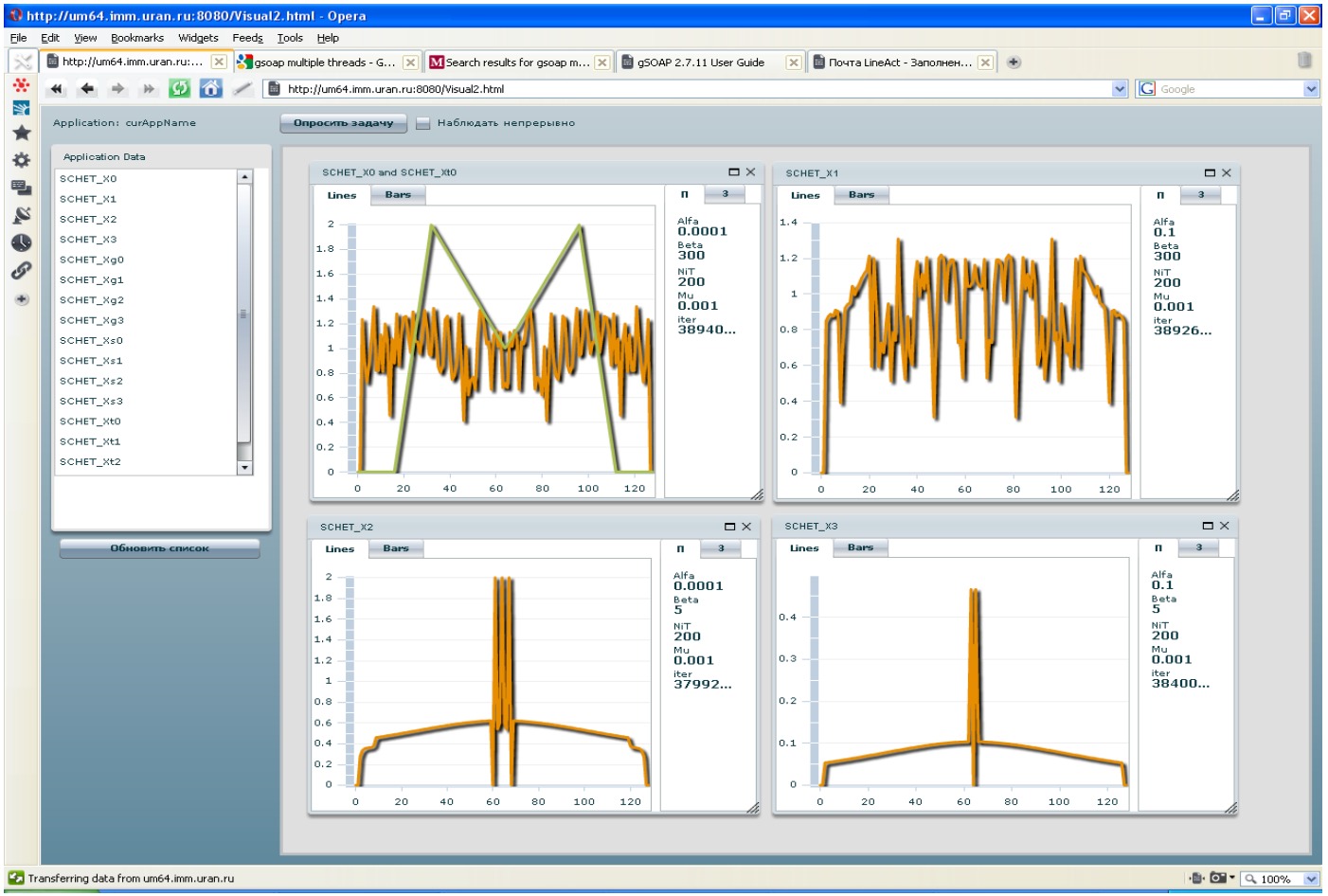
Визуализация 4-мерных множеств, характеризующих химическую реакцию динамического кинетического расщепления





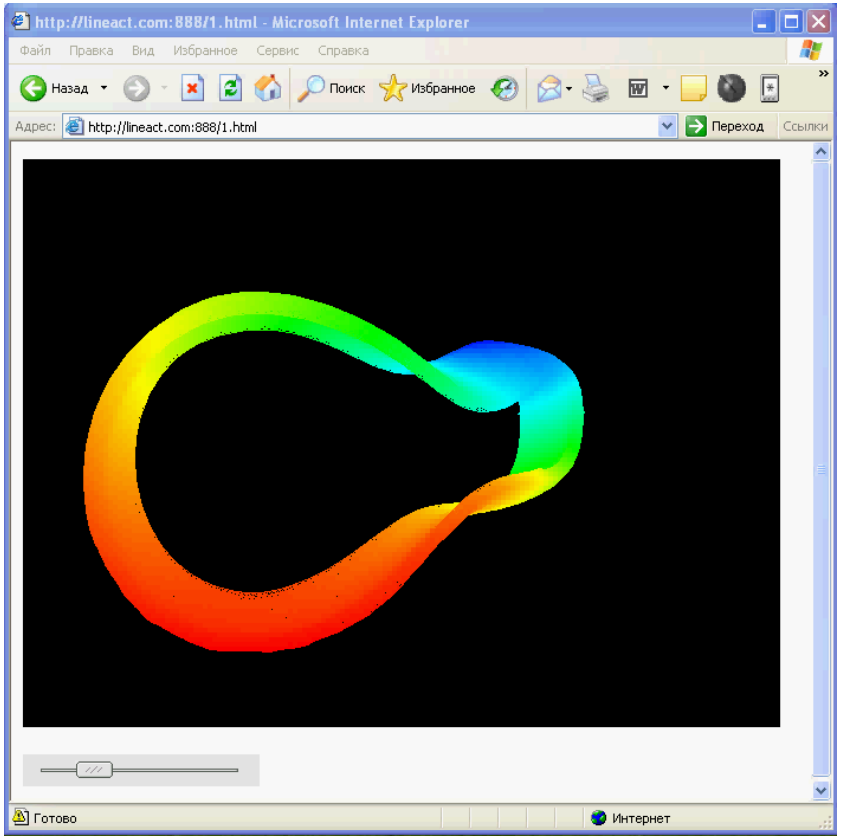
Иванов А.Г., Краснов В.П., Кумков С.И. Информационные множества констант скоростей химической реакции // Высокопроизводительные вычисления и их приложения: Труды Всероссийской научной конференции (30 октября - 2 ноября 2000 г., г. Черноголовка). - М.: Изд-во МГУ, 2000. С. 247-250.

Онлайн-визуализация



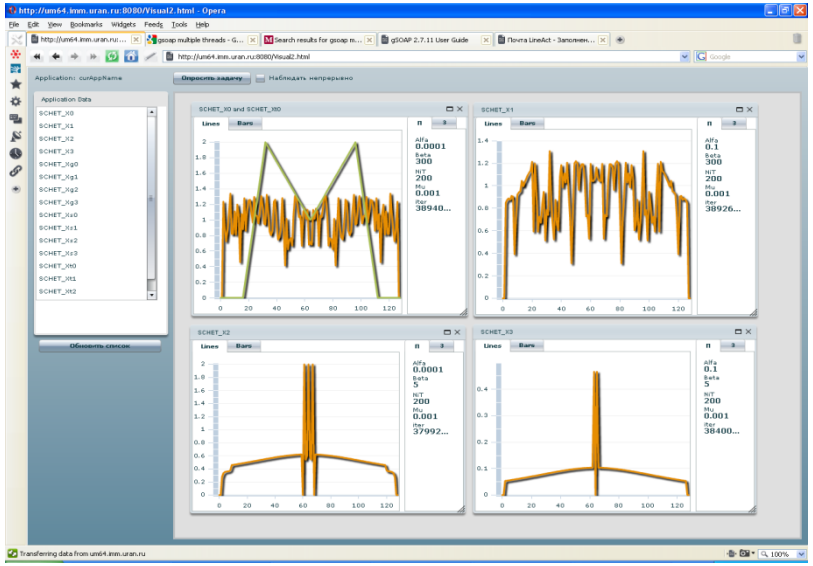
Serezgnikova T., Sharf S., Vasev P., Parallel computing on "Uran" cluster: development of algorithms and software for restoring blurred images // Proceedings of Second International Conference "Cluster Computing" CC 2013 (Ukraine, Lviv, June 3-5, 2013), pp. 198-203.

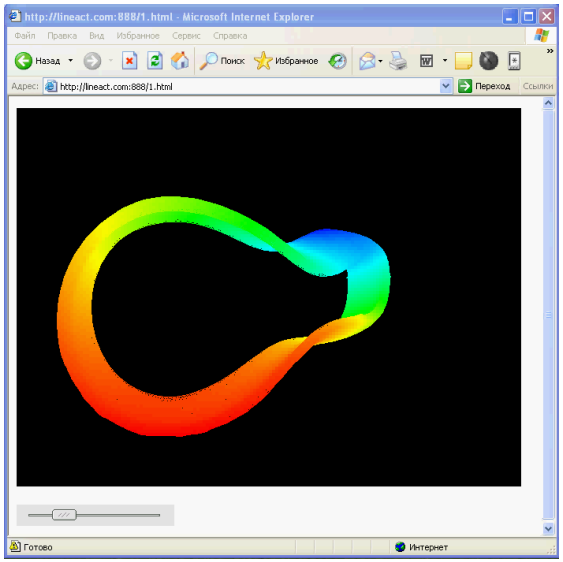
Удалённая визуализация



Бахтерев М.О., Васёв П.А., Казанцев А.Ю., Манаков Д.В. , Система удалённой визуализации для инженерных и суперкомпьютерных вычислений // Вестник ЮУрГУ. Серия "Математическое моделирование и программирование". 2009. № 17 (150). Вып. 3. С. 4-11.

Онлайн и удалённая визуализация





Serezgnikova T., Sharf S., Vasev P., Parallel computing on "Uran" cluster: development of algorithms and software for restoring blurred images // Proceedings of Second International Conference "Cluster Computing" CC 2013 (Ukraine, Lviv, June 3-5, 2013), pp. 198-203.

Бахтерев М.О., Васёв П.А., Казанцев А.Ю., Манаков Д.В. , Система удалённой визуализации для инженерных и суперкомпьютерных вычислений // Вестник ЮУрГУ. Серия "Математическое моделирование и программирование". 2009. № 17 (150). Вып. 3. С. 4-11.

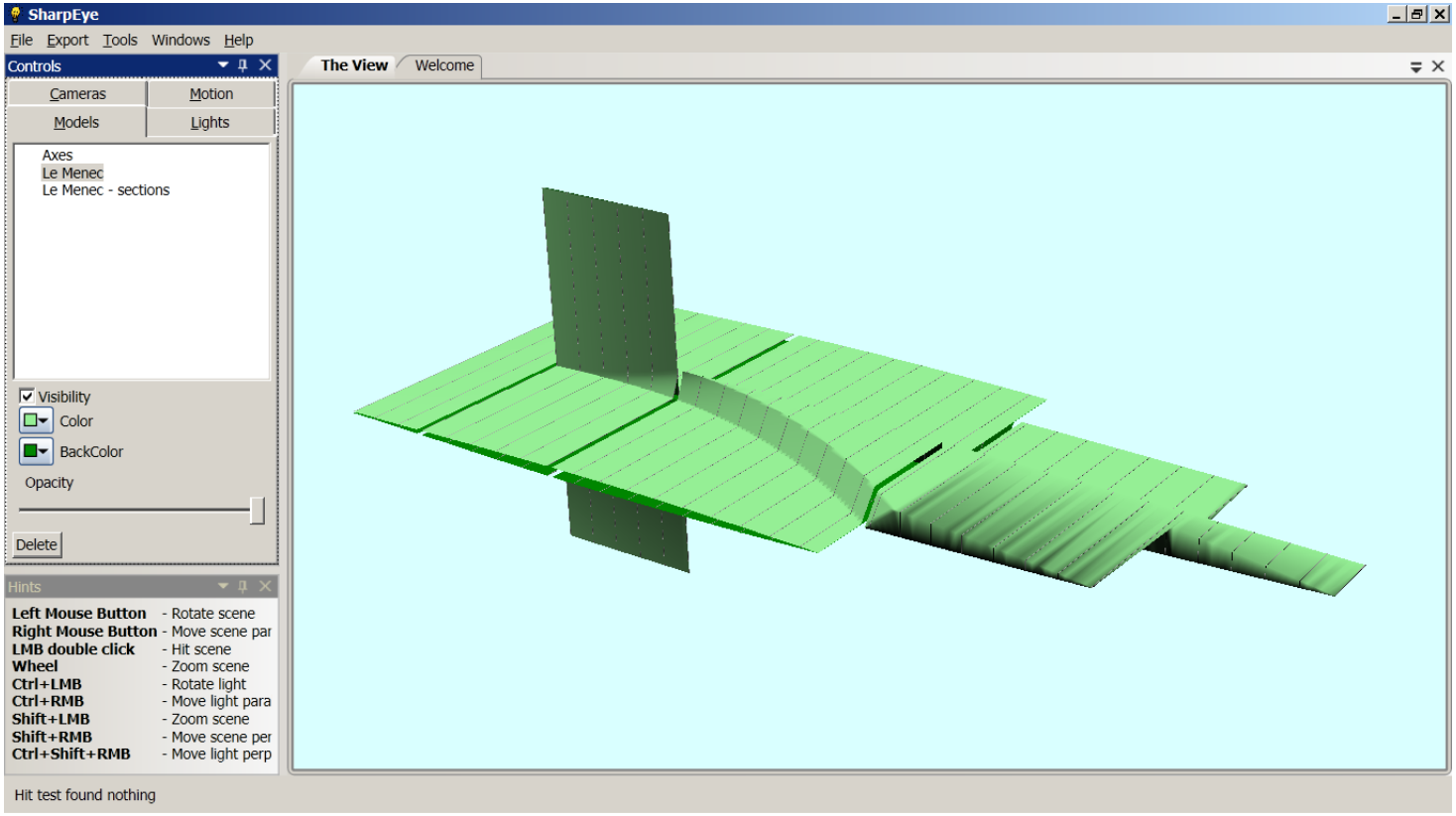
- Разработка специализированных систем визуализации

=>

- Создание теории компьютерной визуализации
- Поиск средств автоматизации разработки спец. систем
- Новые и дополнительные направления

Конструктор систем визуализации

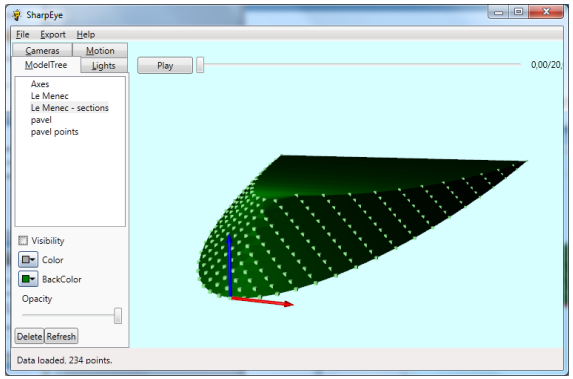
Гипотеза: существенная часть программ трехмерной визуализации содержит повторяющиеся элементы, которые можно выделить в универсальный слой.



SharpEye

Функции конструктора

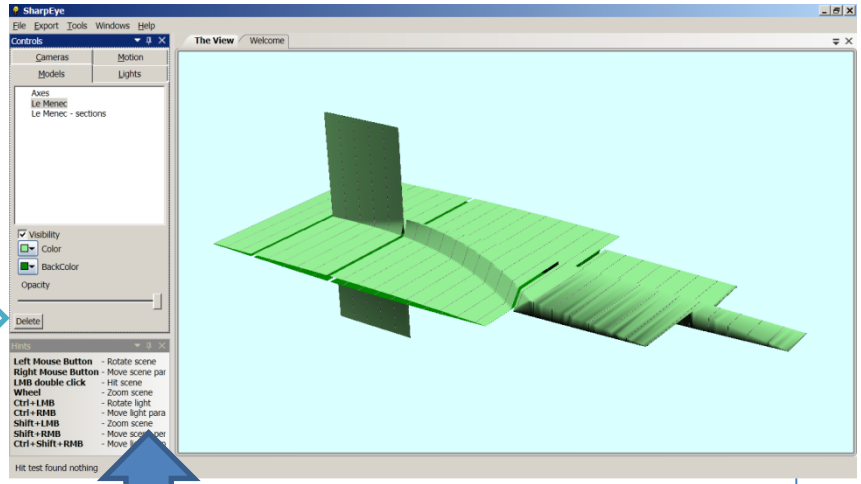
- Отображение 3D-объектов
  - Загрузка-удаление объектов, их хранение в нескольких иерархиях, операции над группами.
  - Операции с атрибутами объектов: изменение цвета, прозрачность, видимость-невидимость.
  - Геометрические операции со сценой (повороты, масштабирование, перемещение).
  - Операции со светом (перемещение источника, добавление, удаление, изменение яркости, включение-выключение) и камерами.
- Загрузка файлов с помощью модулей
- Управление сценой через API
- Экспорт изображений



Структура и потоки

1. Пользователь выбирает файл данных
2. Происходит голосование модулей
3. Модуль-победитель загружает файл и формирует сцену через API

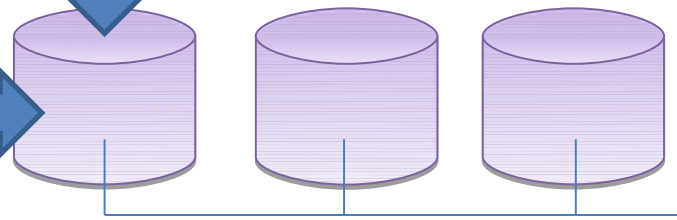
Основная программа конструктора



Файл данных

```
*** data.txt
0 0 0
1 1 1
-1 1 1
2 5 3
-4 2 1
2 4 1
7 5 12
4 4 3
14 15 2
```

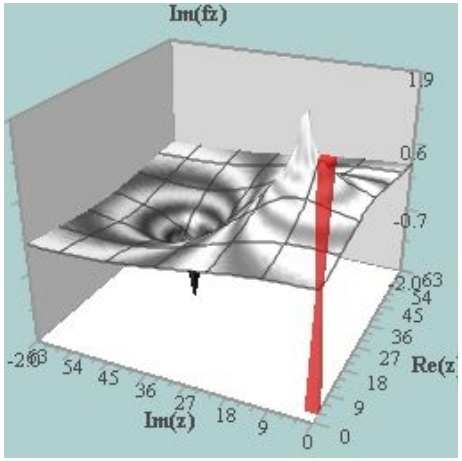
Модули визуализации



Типовой сценарий разработки

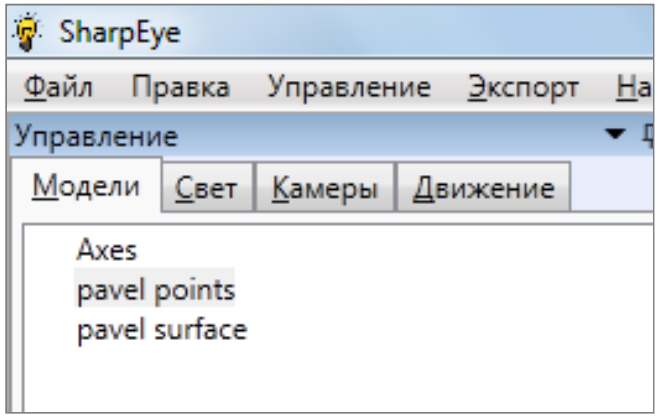
- Реализовать функцию преобразования данных в визуальные сущности. C# (dll), Ruby (скрипт).

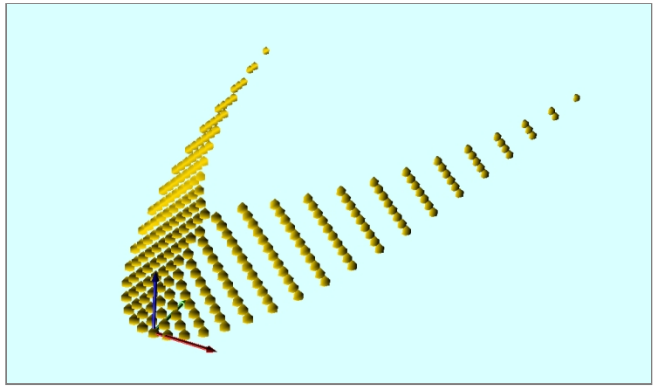
- При желании, добавить элементы управления.

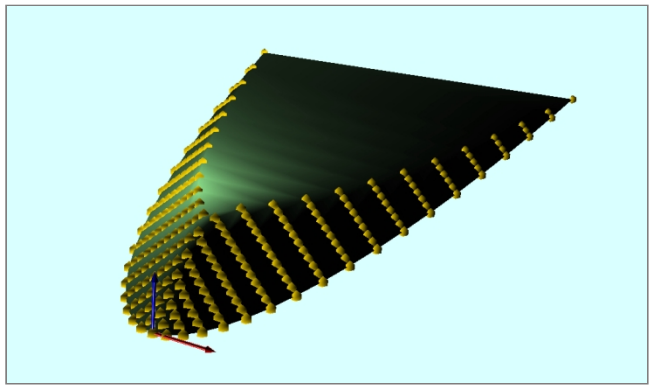


Задача. Дан набор точек {xyz} в файле D3.txt. Необходимо отобразить эти точки. А также построить по этим точкам триангуляцию функции z(x,y) и нарисовать и её.

```
def feel(p)     # Определяем, наш ли это файл?
  p.first_lines =~ /X\s+Y\s+Z/ ? 2.5 : 0.0
end

def load(p)    # Загрузка файла
  m = add_model( "#{p.name} points" )

  file = File.new( p.path , "r")
  file.gets

  vertices = List.of(Triangulator::Geometry::Point).new

  while line = file.gets
     items = line.strip.split(/\s+/) [0..2]
     x,y,z = items.map{ |i| i.to_f }

     m.add_sphere( x, y, z, 0.10, 3, 2 )
     vertices << Triangulator::Geometry::Point.new(x,y,z)
  end

  tris = Triangulator::Delauney::Triangulate(vertices)

  m2 = add_model( "#{p.name} surface" )
  vertices.each{ |v| m2.add_node( v.X, v.Y, v.Z ) }
  tris.each{ |t| m2.add_triangle( t.p2, t.p1, t.p3 ) }

end
```





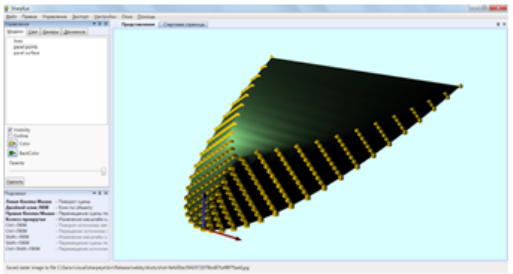


Декларативный подход к описанию сцен научной визуализации

Гипотеза: декларативный подход позволит создавать специализированные системы быстрее.

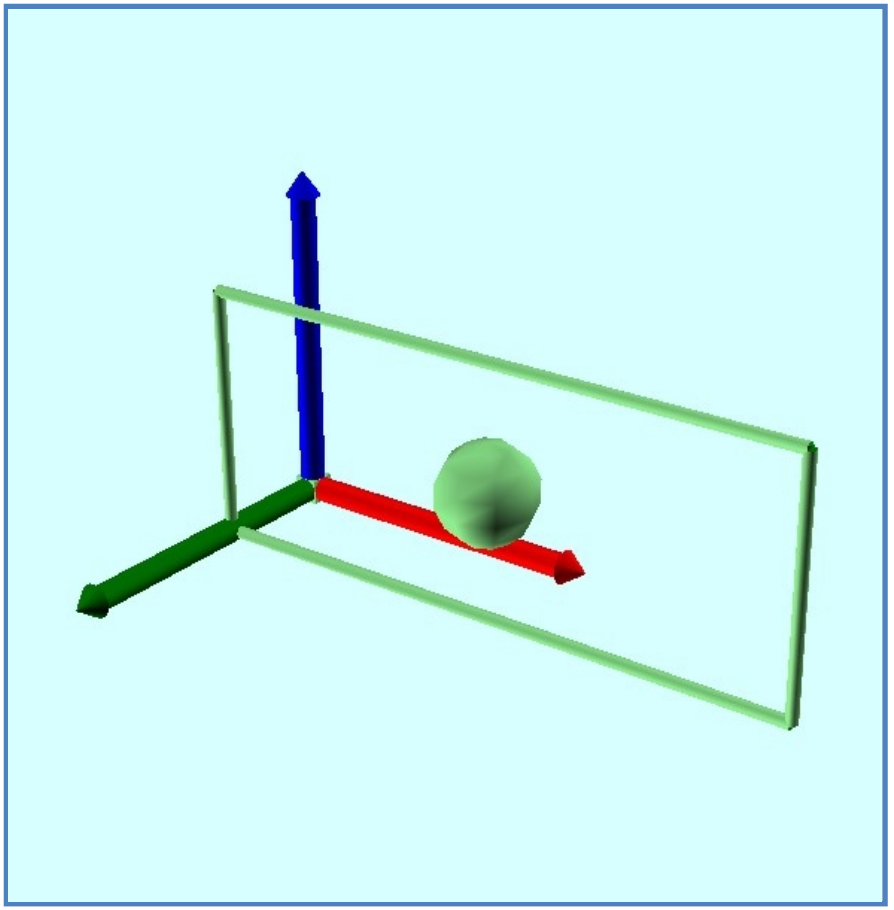
```
class LebedevSurface
        def feel(p)     # Определяем, наш ли это файл?
          p.first_lines =~ /X\s+Y\s+Z/ ? 2.5 : 0.0
        end

        def load(p)    # Загрузка файла
          m = add_model( "#{p.name} points" )

          file = File.new( p.path , "r")
          file.gets

          vertices = List.of(Triangulator::Geometry::Point).new

          while line = file.gets
            items = line.strip.split(/\s+/) [0..2]
            x,y,z  = items.map{ |i| i.to_f }

            m.add_sphere( x, y, z, 0.10, 3, 2 )
            vertices << Triangulator::Geometry::Point.new(x,y,z)
          end

          tris = Triangulator::Delauney::Triangulate(vertices)

          m2 = add_model( "#{p.name} surface" )
          vertices.each{ |v| m2.add_node( v.X, v.Y, v.Z ) }
          tris.each{ |t| m2.add_triangle( t.p2, t.p1, t.p3 ) }
        end
end
```

```
set scene.d3.spheres          = array * 3 <<file D3.txt
set scene.d3.spheres.radius = 0.03

set scene.d3.trimesh          = &scene.d3.spheres.triangulate
set scene.d3.trimesh.nodes = &scene.d3.spheres
set scene.d3.trimesh.color   = green
```

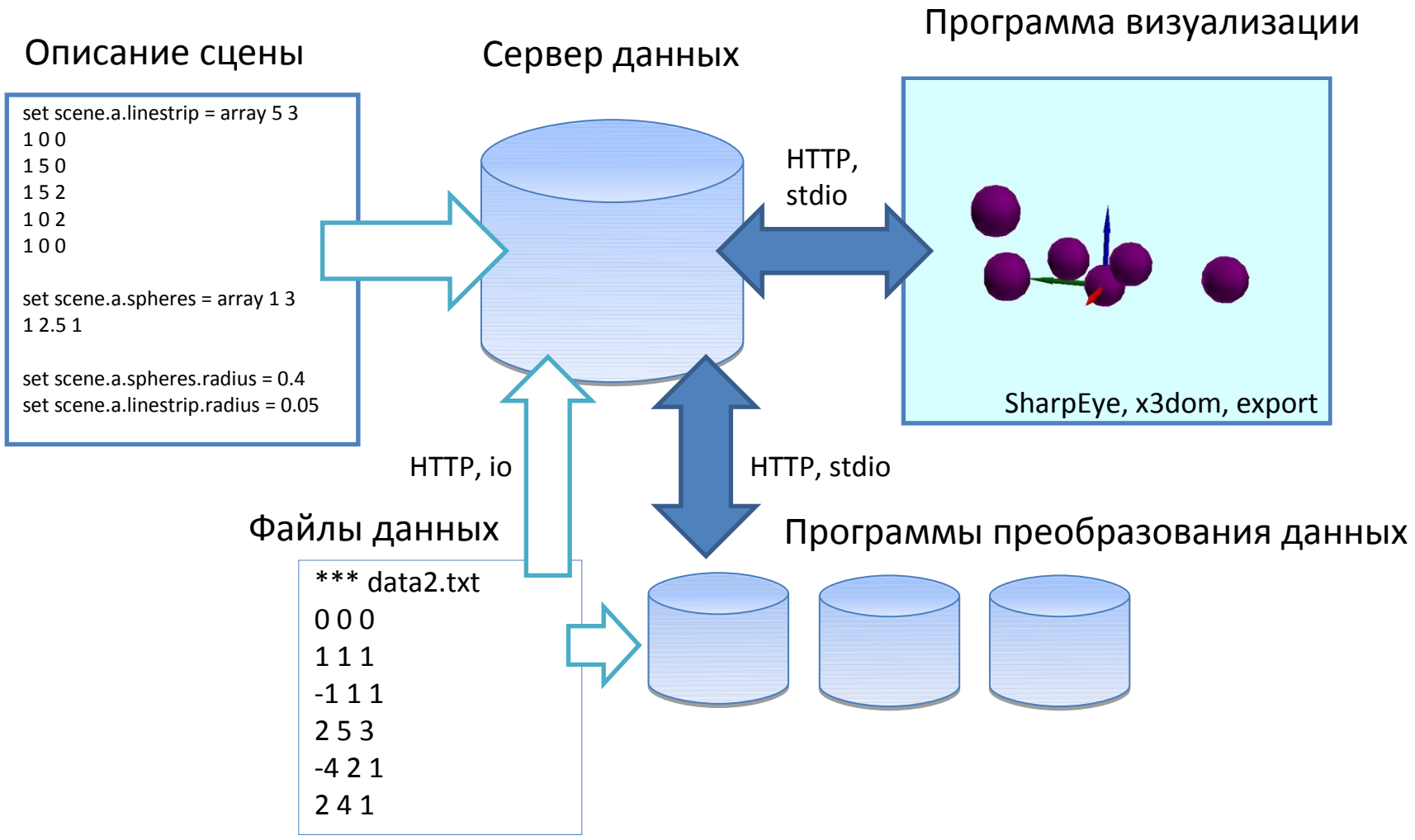


Задача: нарисовать прямоугольник и сферу.

```
set scene.a.linestrip = array 5 3
1 0 0
1 5 0
1 5 2
1 0 2
1 0 0

set scene.a.spheres = array 1 3
1 2.5 1

set scene.a.spheres.radius = 0.4
set scene.a.linestrip.radius = 0.05
```



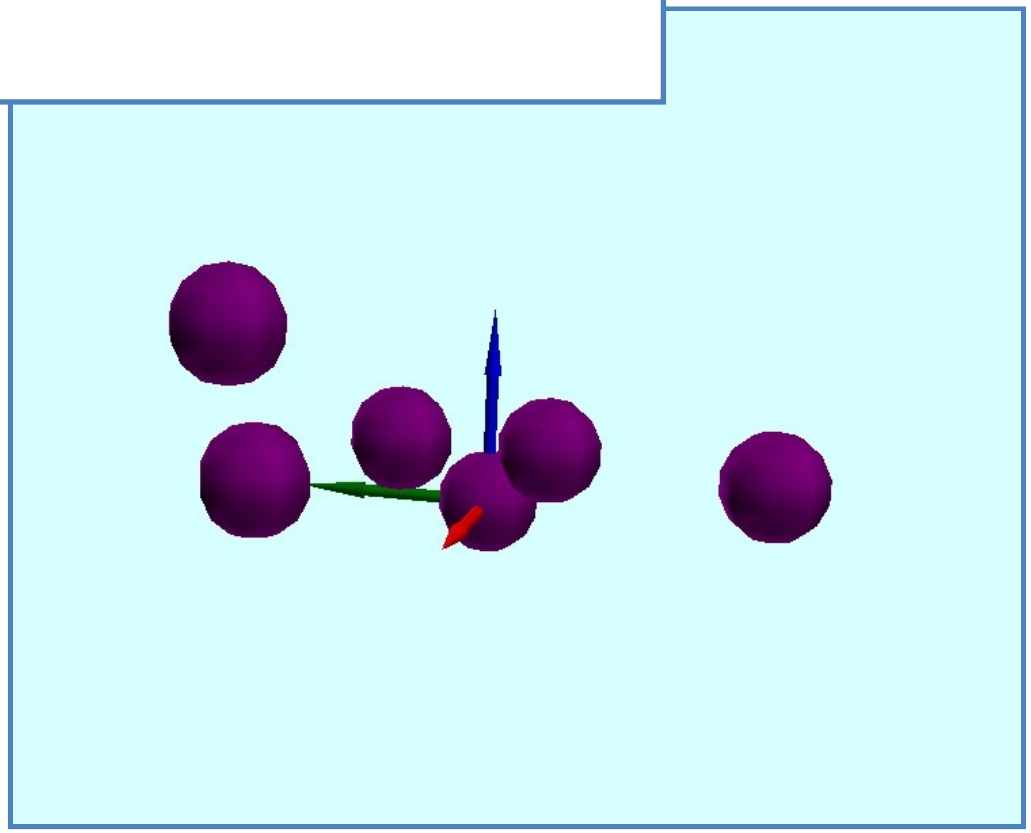
Структура и потоки



Задача: нарисовать сферы с координатам из файла data2.txt.

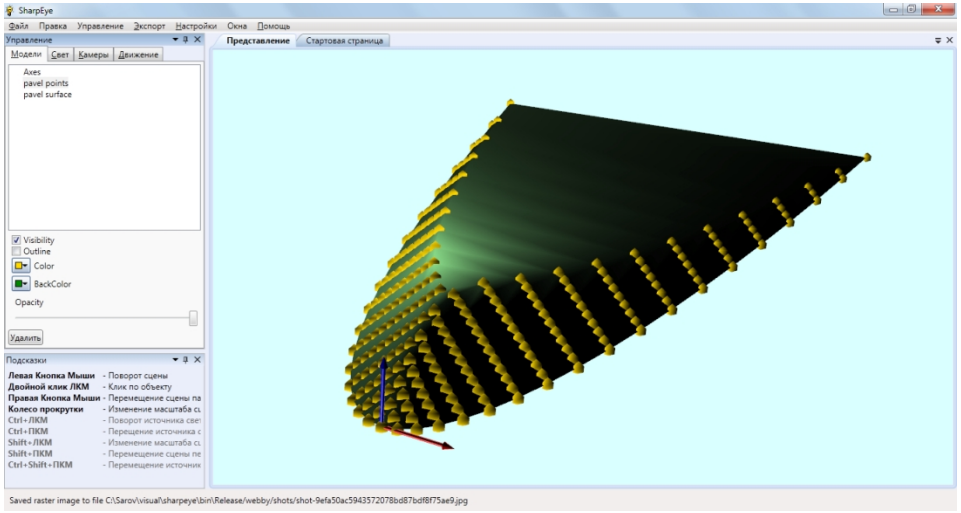
```
set scene.a.spheres.color = purple
set scene.a.spheres.radius = 0.7
set scene.a.spheres = array 6 3 <<file data2.txt
```

```
*** data2.txt
0 0 0
1 1 1
-1 1 1
2 5 3
-4 2 1
2 4 1
```



Задача. Дан набор точек {xyz} в файле D3.txt. Необходимо отобразить эти точки. А также построить по этим точкам триангуляцию функции z(x,y) и нарисовать и её.

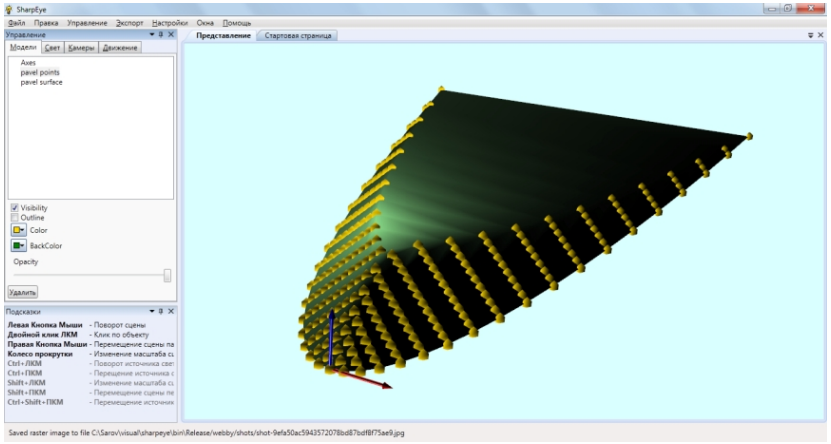
```
set scene.d3.spheres          = array * 3 <<file D3.txt
set scene.d3.spheres.radius = 0.03

set triangulate[input]          = array * 3 <<exec tri.exe <- {{get input}}

set scene.d3.trimesh          = &triangulate[input=scene.d3.spheres]
set scene.d3.trimesh.nodes  = &scene.d3.spheres
set scene.d3.trimesh.color    = green
```



Сравнение подходов

```
class LebedevSurface
        def feel(p)     # Определяем, наш ли это файл?
          p.first_lines =~ /X\s+Y\s+Z/ ? 2.5 : 0.0
        end

        def load(p)    # Загрузка файла
          m = add_model( "#{p.name} points" )

          file = File.new( p.path , "r")
          file.gets

          vertices = List.of(Triangulator::Geometry::Point).new

          while line = file.gets
             items = line.strip.split(/\s+/) [0..2]
             x,y,z = items.map{ |i| i.to_f }

             m.add_sphere( x, y, z, 0.10, 3, 2 )
             vertices << Triangulator::Geometry::Point.new(x,y,z)
          end

          tris = Triangulator::Delauney::Triangulate(vertices)

          m2 = add_model( "#{p.name} surface" )
          vertices.each{ |v| m2.add_node( v.X, v.Y, v.Z ) }
          tris.each{ |t| m2.add_triangle( t.p2, t.p1, t.p3 ) }
        end
end
```

```
set scene.d3.spheres          = array * 3 <<file D3.txt
set scene.d3.spheres.radius = 0.03

set scene.d3.trimesh          = &scene.d3.spheres.triangulate
set scene.d3.trimesh.nodes = &scene.d3.spheres
set scene.d3.trimesh.color   = green
```



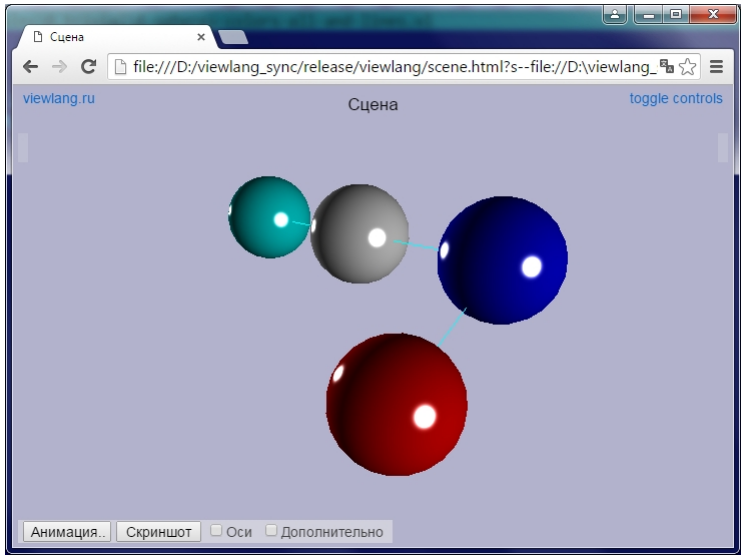
Веб-подход на основе QML

Гипотеза: язык QML интересен для описания сцен визуализации, так как он:

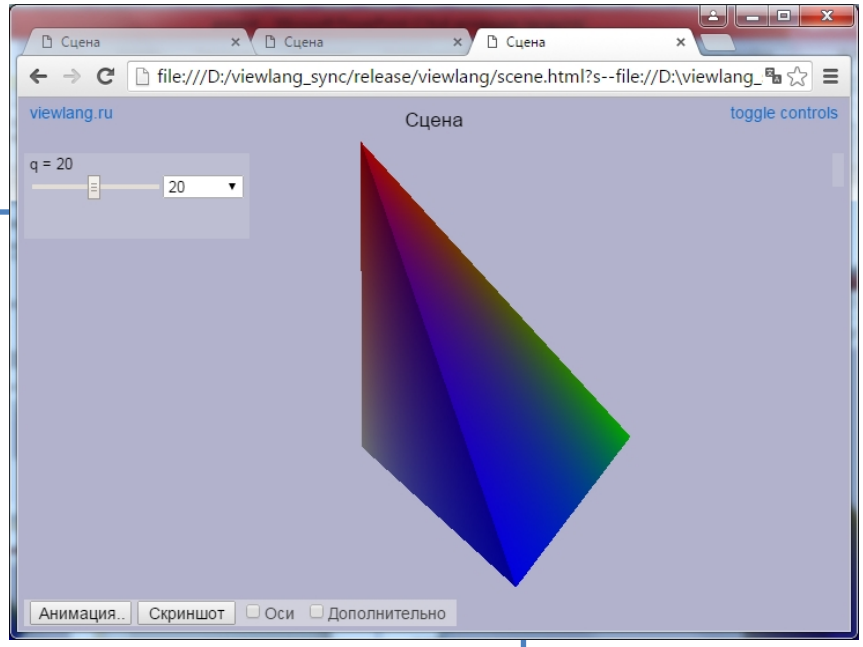
1. Декларативный.
2. Реактивный.
3. Может исполняться в Веб.

Пример сцены на QML

```
Scene {
  Spheres {
    id: sp
    positions: [0,0,0, 1,1,1, 2,2,2, 1,2,3 ]
    colors: [0,1,1, 1,1,1, 0,0,1, 1,0,0]
    radius: 0.5
  }
  Linestrip {
    positions: sp.positions
    color: [0,1,1]
  }
}
```



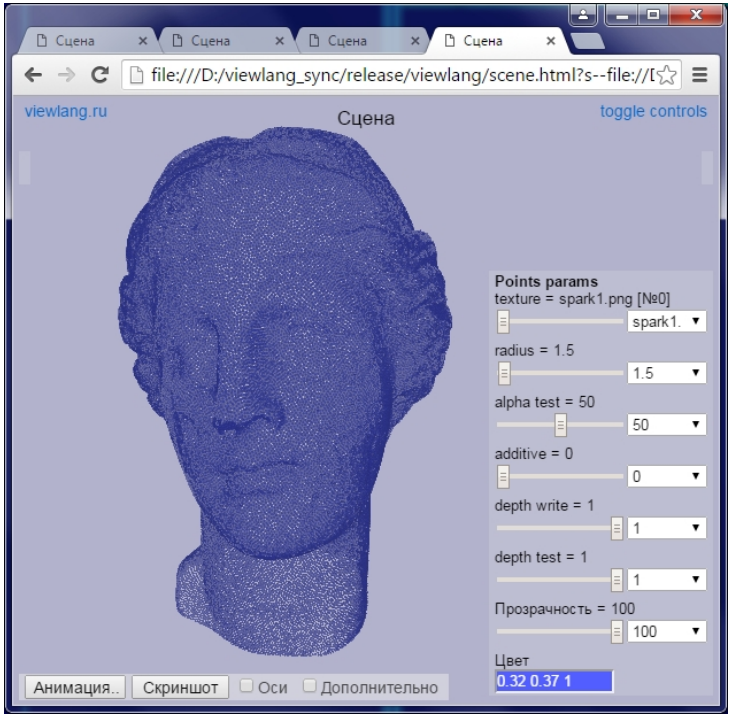
Реактивность




```
Scene {
  Param {
    id: qParam
    title: "q"; min: 4; max: 37
  }

  Tetras {
    positions: [ 0,0,0, 1,0,0,  0,1,0, 0,0,qParam.value/15]
    colors: [1,1,1,  0,1,0,  1,0,0, 0,0,qParam.value/15 ]
  }
}
```

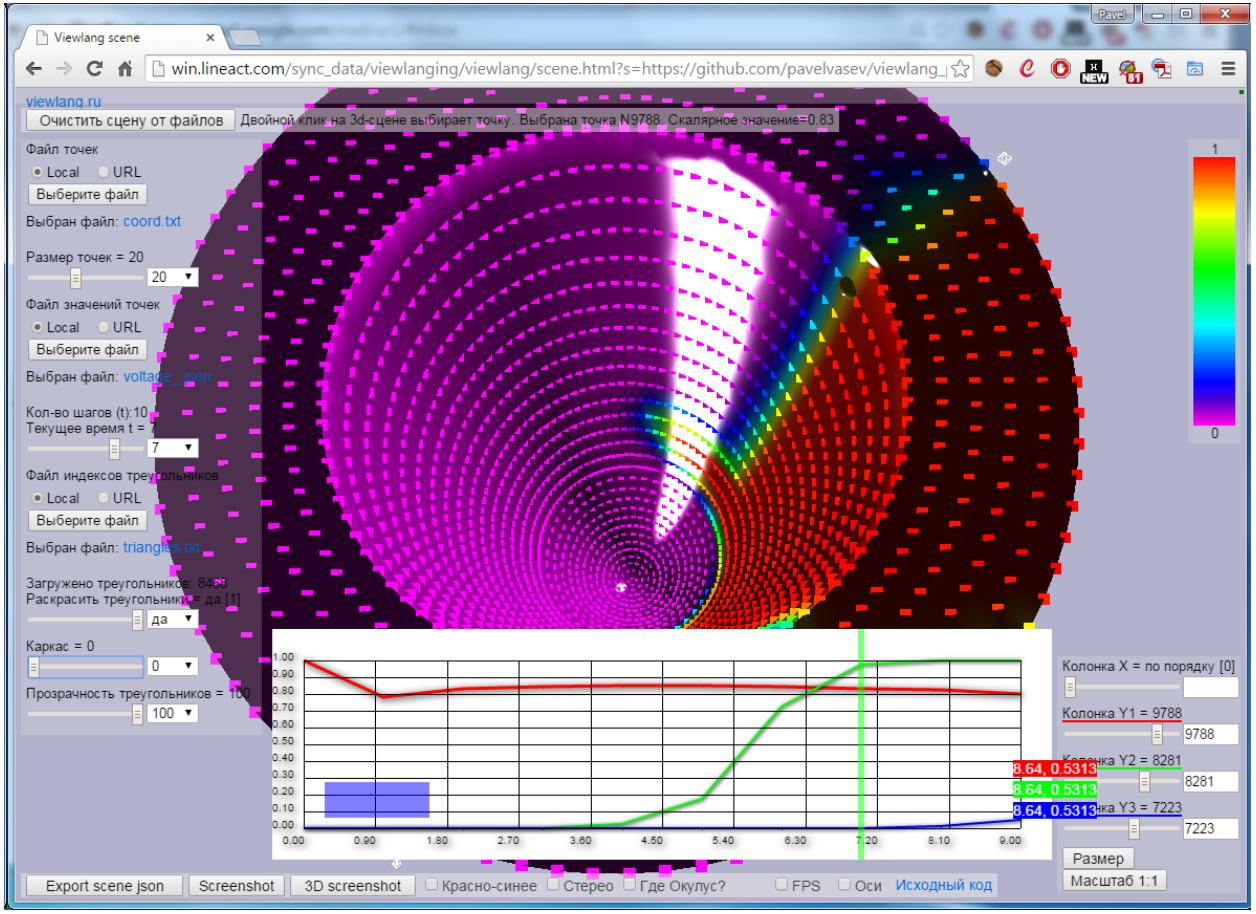
Чтение данных

```
Scene {
  CsvLoader {
    file: "http://server.com/data/s2.txt"
    Flatten {
      id: arr
    }
  }
  Points {
    positions: arr.output
  }
}
```



Визуализация фазовых переменных во времени по расчетам электрофизиологической модели левого желудочка сердца

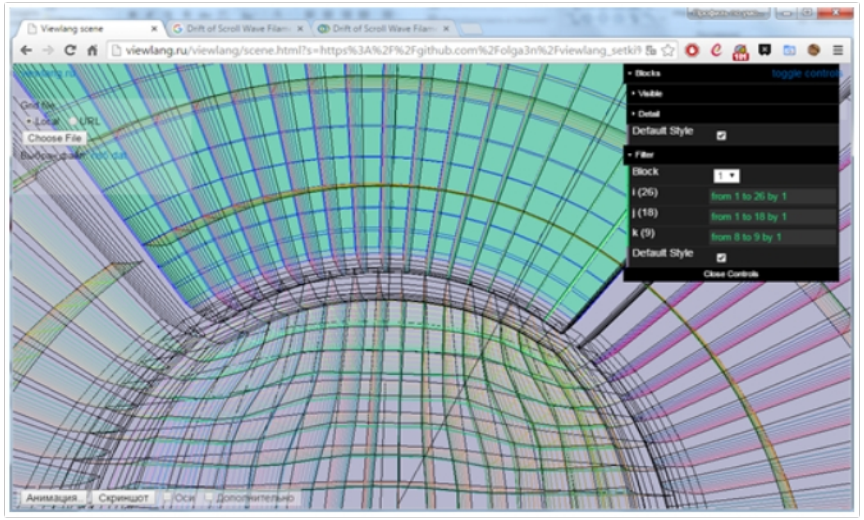
При расчётах электрофизиологии каждый узел сетки характеризуется набором значений нескольких (от 2 до ~50) фазовых переменных. Необходима визуализация сердца с раскраской узлов по значениям одной из переменных; срезы плоскостями (чтобы увидеть, что происходит внутри толщи стенки); графики одной фазовой переменной в разных узлах для оценки (не)однородности миокарда и разных переменных в одном узле для локального изучения электрических процессов в клетках.

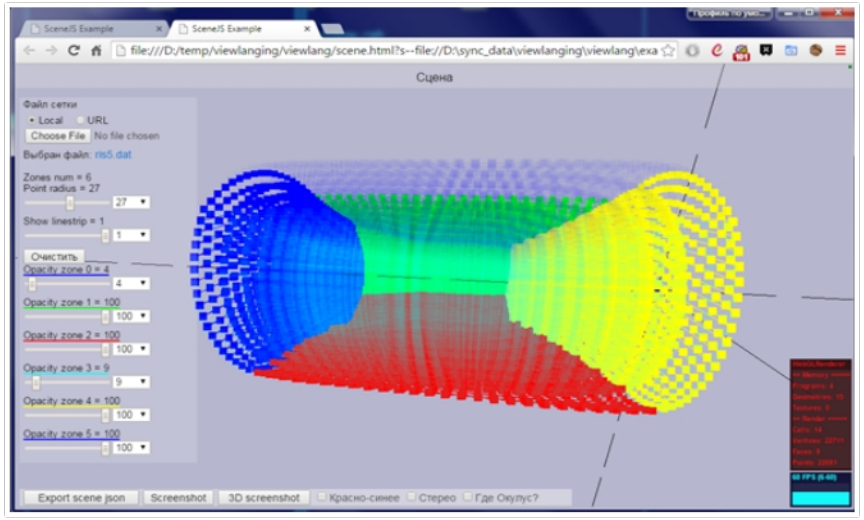


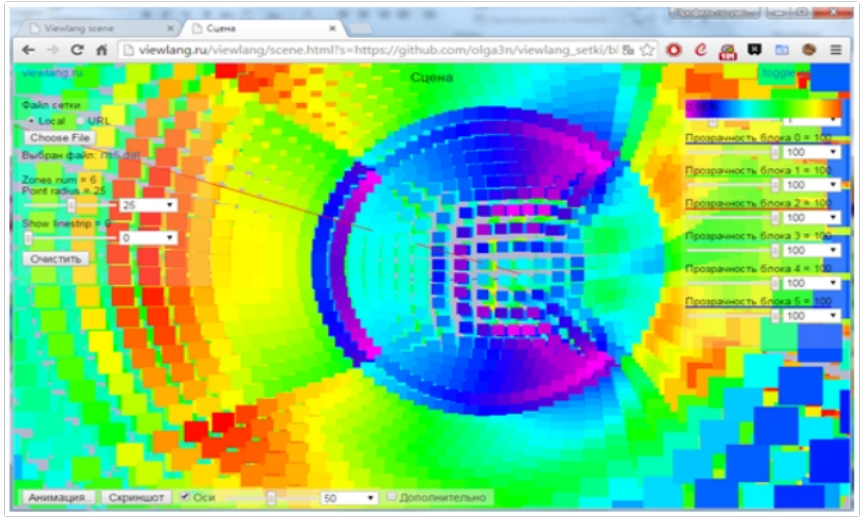
http://win.lineact.com/sync_data/viewlanging/viewlang/scene.html?s=https://github.com/pavelvasev/viewlang_pravdin_heart/blob/master/heart-may-15/se_all.vl

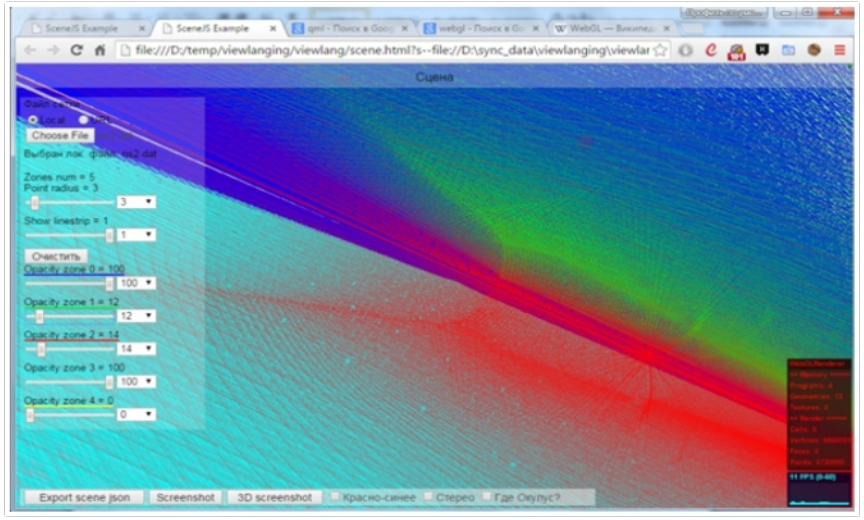
Sergei Pravdin, Hans Dierckx, Vladimir S. Markhasin, and Alexander V. Panfilov, “Drift of Scroll Wave Filaments in an Anisotropic Model of the Left Ventricle of the Human Heart,” BioMed Research International, Article ID 389830, in press.

Элементы визуализации сеток









http://viewlang.ru

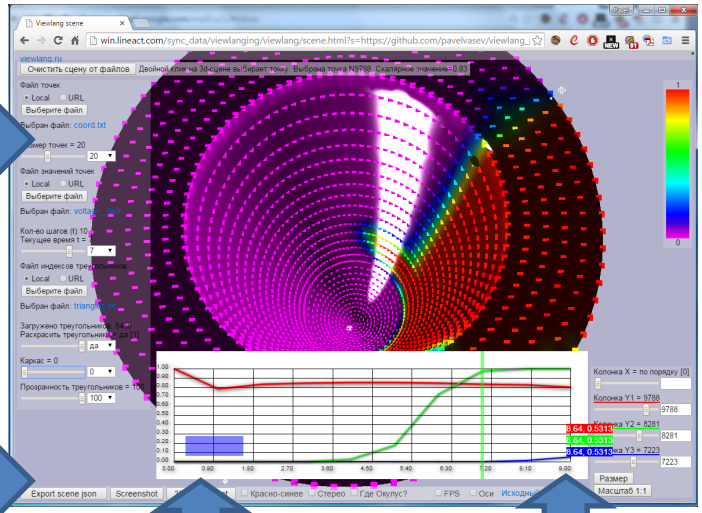
Структура и потоки

Описание сцены на QML

```
Scene {
  Points {
  }
  Lines {
  }
  Triangles {
  }
}
```

HTTP

Браузер пользователя =
= программа визуализации



Файлы данных

```
*** data2.txt
0 0 0
1 1 1
-1 1 1
2 5 3
-4 2 1
2 4 1
```

HTTP

HTTP

HTTP

Вычислитель как
поставщик данных

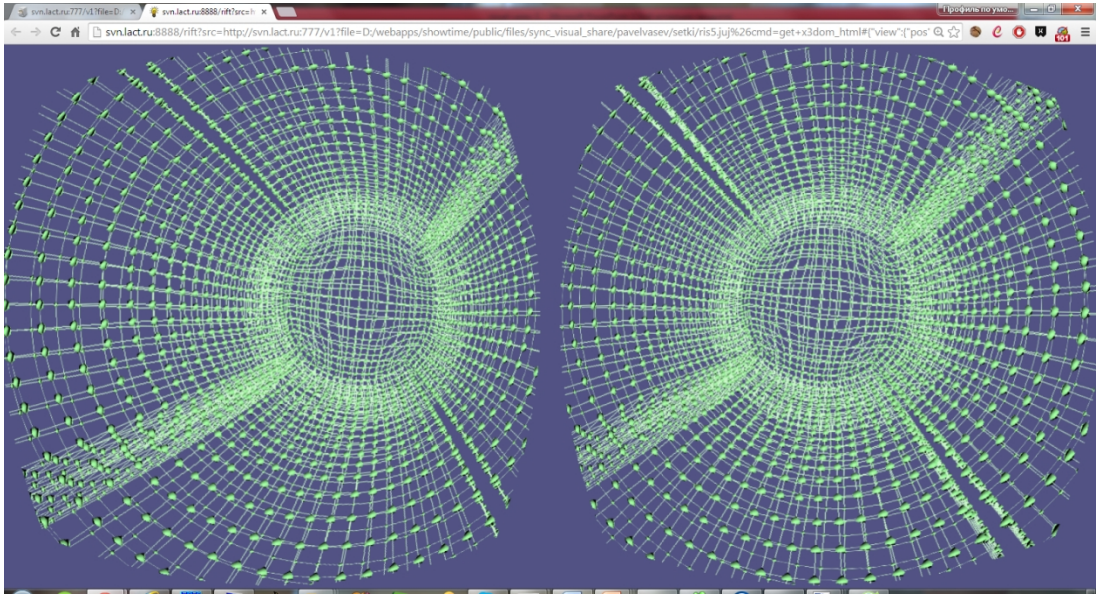
```
*** data3.txt
0 0 0
1 1 1
-1 1 1
2 5 3
-4 2 1
2 4 1
```

Модули обработки
и визуализации

```
Scene {
  Scene {
    Scene {
      Points {
      }
      Lines {
      }
      Triangles {
      }
    }
  }
}
```

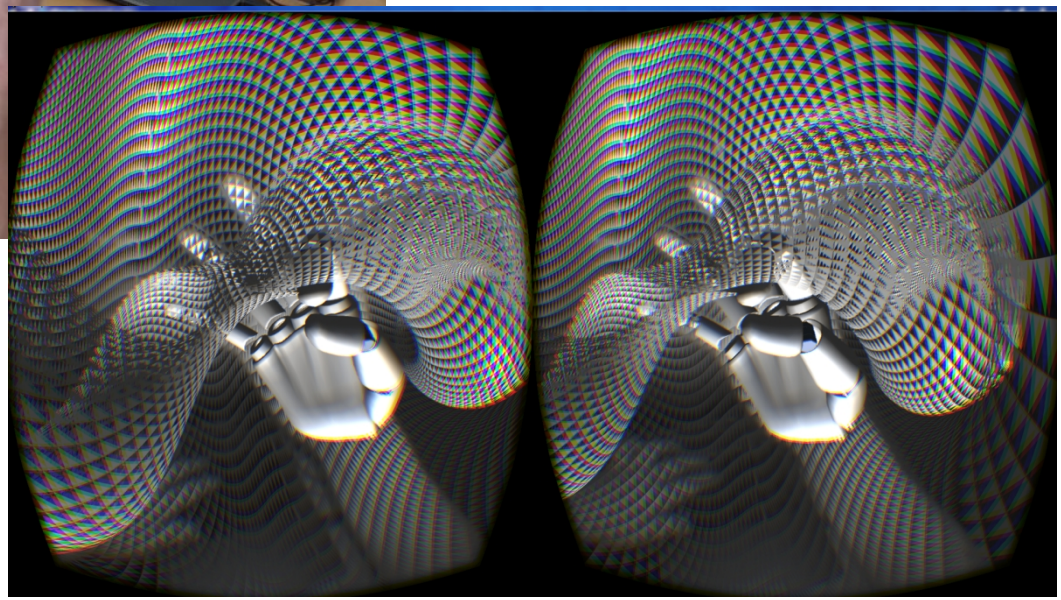
Виртуальная реальность

- Side by side (sony hmz-t1, etc)
- Анаглифические изображения
- nVidia 3dvision
- Oculus Rift
- OpenGL | DirectX | WebGL | Unity
- TV?



Человеко-машинное взаимодействие

- Leap motion
- Физическая визуализация

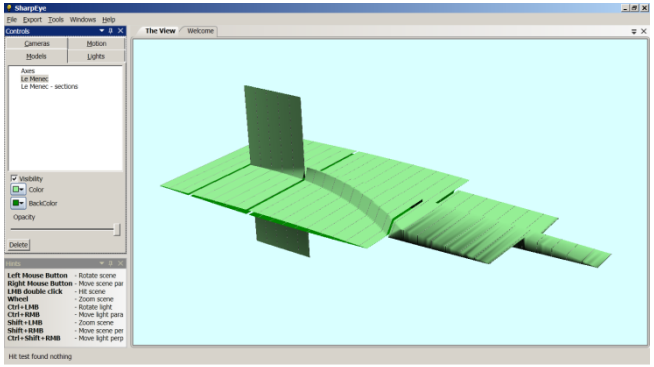


Развитие подходов к разработке специализированных систем компьютерной визуализации

В.Л. Авербух, М.О. Бахтерев, П.А. Васёв,
Д.В. Манаков, И.С. Стародубцев

ИММ УрО РАН, г. Екатеринбург
http://cv.imm.uran.ru

Конструктор систем визуализации



Гипотеза: существенная часть программ трехмерной визуализации содержит повторяющиеся элементы, которые можно выделить в универсальный слой.