Изоповерхности – фигуры, описанные математическими функциями. В отличие от других фигур в PovRay, изоповерхности аппроксимируются во время рендеринга, что часто вызывает определенные неудобства в их использовании. Однако, такого рода поверхности имеют ряд полезных особенностей, в частности возможность моделировать реальные деформации, сдвиги в поверхностях и другое. С помощью изоповерхностей можно задать сложной формы объекты, что либо сложно, либо невозможно сделать с помощью методов CSG.

Описание изоповерхностей

Описание любой изоповерхности происходит в рамках следующего блока параметров:

```
isosurface {
  function { FUNCTION_ITEMS }
  [contained_by { SPHERE | BOX }]
  [threshold FLOAT_VALUE]
  [accuracy FLOAT_VALUE]
  [max_gradient FLOAT_VALUE]
  [evaluate P0, P1, P2]
  [open]
  [max_trace INTEGER] | [all_intersections]
  [OBJECT_MODIFIERS...]
}
```

со значениями по умолчанию:

```
contained_by : box{-1,1}
threshold    : 0.0
accuracy     : 0.001
max_gradient : 1.1
```

Детально рассмотрим назначение основных параметров.

`function { ... }` – описание математической функции, которая впоследствии и будет определять основные свойства изоповерхности.

Contained_by – объект-контейнер, который ограничивает область, в которой PovRay изображает описанную изоповерхность. Этим контейнером может быть либо параллелепипед, либо сфера. Задаются объекты–контейнеры с помощью стандартного синтаксиса:

```
contained_by { sphere { CENTER, RADIUS } }
contained_by { box { CORNER1, CORNER2 } }
```

Если контейнер не указывается, то принимается значение по умолчанию:
box {<-1,-1,-1>, <1,1,1>}

threshold – параметр, который определяет, насколько объемной будет изоповерхность. Поверхность рисуется тогда, когда её значение равно параметру threshold. По умолчанию используется threshold, равный 0.

accuracy

Для построения поверхностей в PovRay используется метод рекурсивного деления луча на отрезки, которое продолжается до тех пор, пока длина отрезка, в котором PovRay ищет точку пересечения луча и изоповерхности, не станет меньше значения параметра accuracy. По умолчанию, accuracy = 0,001. Меньшее значение позволяет строить более четкие поверхности, но при этом затрачивая больше ресурсов для рендеринга.

max_gradient

PovRay находит первую точку пересечения луча с изоповерхностью, описанной любой непрерывной функцией, если известно значение параметра max_gradient. Для получения корректного изображения, необходимо правильно задать величину параметра max_gradient . Если значение max_gradient будет указано слишком маленьким, то велика вероятность появления изъянов в изображении, например, в объекте могут получиться дырки или незаполненные полосы.

Для простых изоповерхностей, особенно тех, что имеют симметричную форму, очень просто рассчитать max_gradient из математических соображений.

Рассмотрим пример построения изоповерхности, задаваемой следующим образом:

```
isosurface {
  function { x*x + y*y + z*z - 1 }
       accuracy 0.0001
       contained_by{sphere{0,1.2}}
       pigment {rgb .9}
}
```

Поскольку параметр max_gradient не уточнён, принято значение по умолчанию max_gradient = 1.1, что значительно меньше реального, отсюда и результат изображения – часть сферы не прорисована, так как луч не проверил на наличие точек изоповерхности те области, которые выходят за пределы максимального угда наклона луча, принятого по умолчанию равным 1.1.

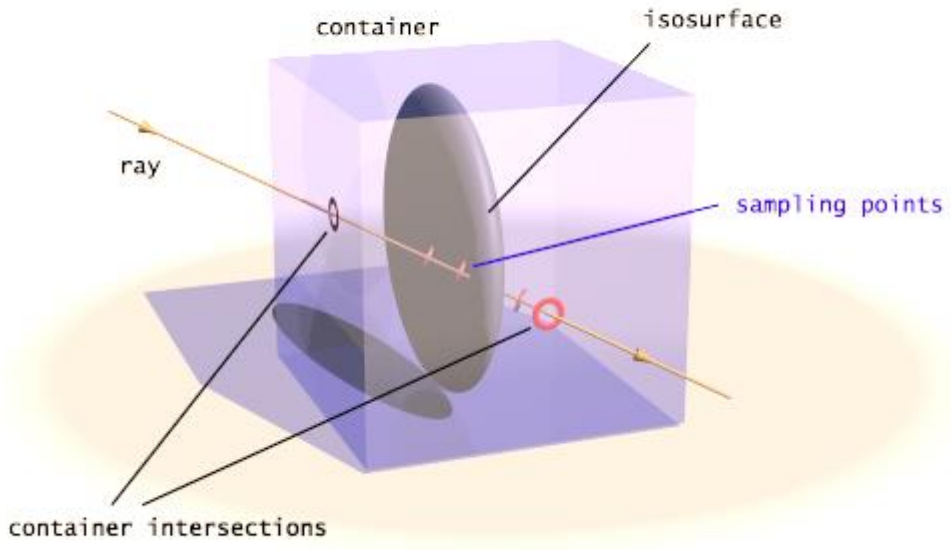
Для данной изоповерхности расчёт max_gradient можно произвести из следующих соображений:

сфера, задающая изоповерхность – объект, симметричный в любом направлении. Рассчитаем max_gradient в направлении вдоль оси x (значения y и z остаются неизменными). Тогда функция, задающая сферу будет иметь упрощенный вид: x*x – 1, её градиент gradient = 2*x. Функция ограничена сферой с центром в точке (0,0,0) и радиусом 1.2, следовательно градиент $\text{gradient} \subseteq [-2.4; 2.4]$, откуда находим max_gradient = 2.4.

Open

Наличие слова **open** в с писке параметров, задающих изоповерхность, говорит о том, что видимая часть объекта – контейнера должна быть удалена (см. пример в п. «примеры простейших изоповерхностей») .

Теперь, зная назначение параметров, задаваемых нашу изоповерхность, принцип её построения в системе PovRay становится совсем понятным:



как было сказано выше, при построении изоповерхности используется метод рекурсивного деления: отрезок луча прошедшего внутрь объекта-контейнера рекурсивно делится, при этом на границах этих отрезков производятся вычисления значений функции, задающей изоповерхность. Деление происходит до тех пор, когда при условии, что максимальный угол наклона луча не превышает max_gradient по значениям функции можно будет понять, что в исследуемых точках – концах отрезков нет пересечения луча и

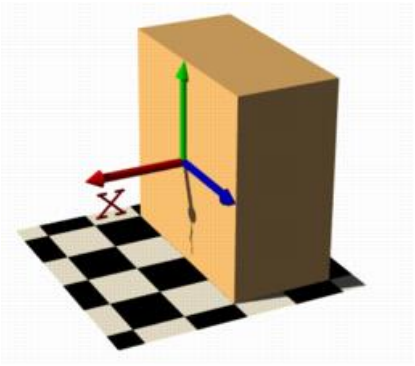
изоповерхности , либо когда длина отрезка луча достигнет заданного параметра точности accuracy.

Примеры простейших изоповерхностей

Для начала рассмотрим простую функцию f(x,y,z) = x. Значениями этой функции будет ось OX. Описание изоповерхности на встроенном языке PovRay будет выглядеть так:

```
isosurface {
   function { x }
   contained_by { box { -2, 2 } }
  }
```

Поверхность в результате так же проста:

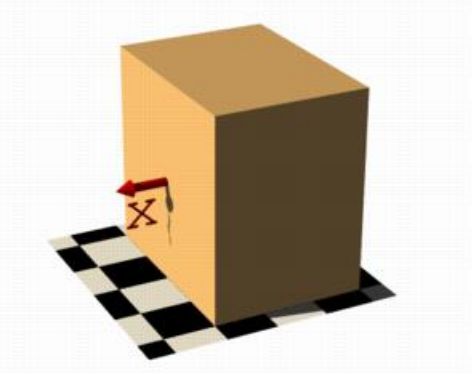


Причина, по которой мы получили коробку – использование параллелепипеда в качестве объекта – контейнера для изоповерхности, в списке параметров его нет, и, в данном примере по умолчанию используется box {<-1,-1,-1>, <1,1,1>}.

То есть фактически только одна сторона бокса получена описанной функцией, - плоскость, значение x в доль которой равно нулю, так как значение параметра threshold (граница, предел) по умолчанию установлено равным 0.

Рассмотрим вариант описанной изоповерхности, когда threshold = 1. Тогда

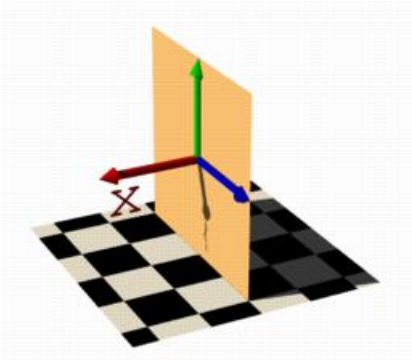
```
isosurface {
   function { x }
   contained_by { box { -2, 2 } }
   threshold 1
  }
```



Добавив слово open в определении изоповерхности, можно удалить видимую часть объекта- контейнера, тогда из определения

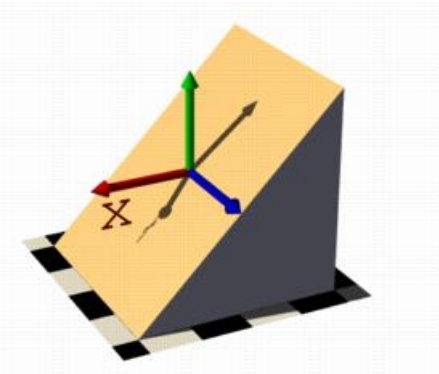
```
isosurface {
   function { x }
   contained_by { box { -2, 2 } }
   open
 }
```

получим плоскость, перпендикулярную оси OX, как показано на рисунке:

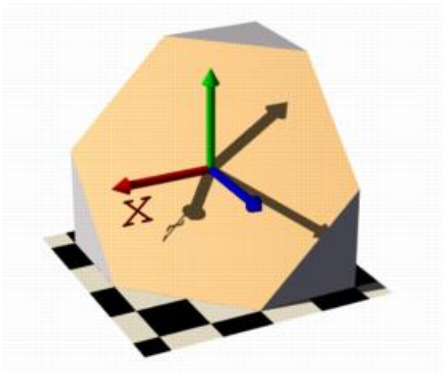


Далее рассмотрим изоповерхности, заданные более сложными функциями:

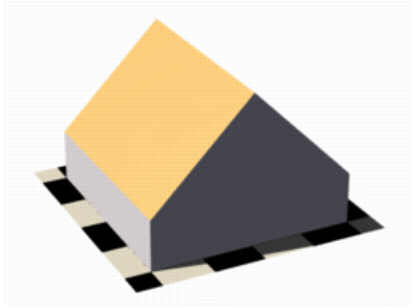
1. `function { x+y }`



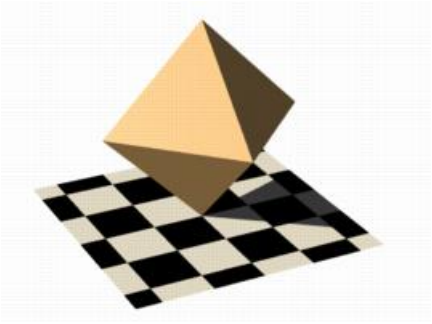
2. `function {x+y+z }`



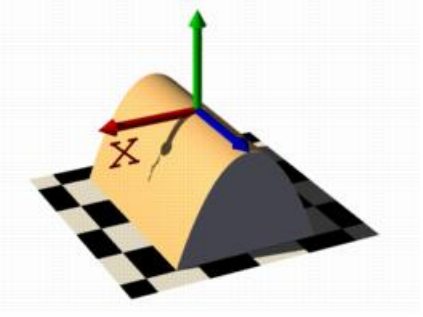
3. `function { abs(x)-1+y }`



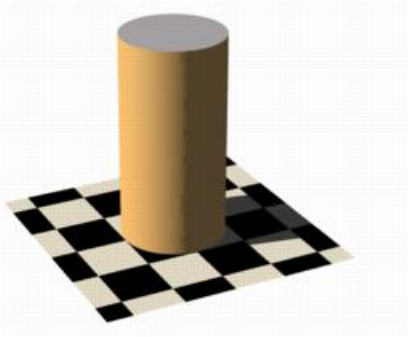
4. `function { abs(x)+abs(y)+abs(z)-2 }`



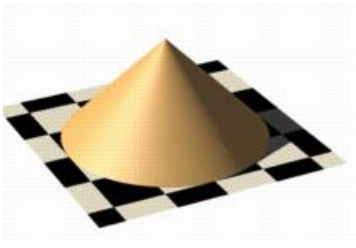
5. `function { pow(x,2) + y }`



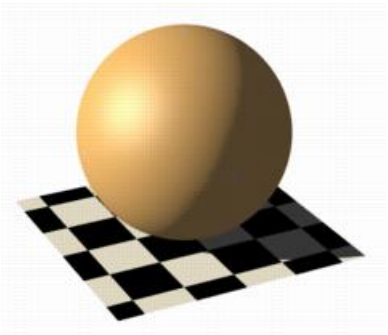
6. `function { sqrt(pow(x,2) + pow(z,2)) - 1 }`



7. `function { sqrt(pow(x,2) + pow(z,2)) + y }`



8. `function { sqrt(pow(x,2) + pow(y,2) + pow(z,2)) - 2 }`



Преобразования изоповерхностей

К изоповерхностям применимы все преобразования, доступные для любого другого объекта в PovRay. Если вы хотите изменить положение или размер изоповерхности внутри объекта – контейнера, то следует изменять значения параметров, от которых зависит функция, описывающая изоповерхность.

Следует так же отметить, что изменение значений параметров для достижения желаемого результата необходимо производить в направлении, обратном тому эффекту, который мы хотим получить. Например, чтобы перенести сферу Sphere(x,y,z) с центром в точке (0,0,0) на две единицы в положительном направлении вдоль оси y, преобразование координат должно выглядеть так: Sphere(x,y-2,z). Объясняется это просто: изначально координата центра сферы y = 0, наша цель – сделать её равной 2, то есть y = 2, отсюда получаем y-2 = 0.

Аналогичным образом делается масшабирование изоповерхности.

Итак, кратко об основных преобразованиях:

1. **Параллельный перенос** : $F(x,y,z) \to F(x - xtr, y - ytr, z - ztr)$, F – функция описывающая изоповерхность, xtr, ytr, ztr – величина переноса вдоль соответствующей оси.
2. **Масштабирование**: $F(x,y,z) \to F(\frac{x}{sc\_x}, \frac{y}{sc\_y}, \frac{z}{sc\_z})$, sc_x, sc_y, sc_z – величина масштабного преобразования изопоерхности вдоль соответствующей координаты. Для того, чтобы бесконечно масштабировать изоповерхность, необходимо заменить соответствующую координату на 0. В качестве примера можно привести преобразование сферы в цилиндр:

```
#include "functions.inc"
#include "colors.inc"

camera { location  <0, 0, -4> look_at <0, 0, 0>}
light_source {<-100,200,-100> colour rgb 1}
background  {Brown}

#declare R = 2;
#declare  S = function {x*x + y*y + z*z - 1}

isosurface {
  function { S(0,y,z) }
       max_gradient 3
       contained_by{sphere{0,R}}
       pigment {Yellow}
}
```

3. <u>Сдвиг относительно плоскости</u>

Осуществляется следующим преобразованием:

a) В плоскости XY: $F(x,y,z) \rightarrow F\left(x + y * \tan\left(radians(angle)\right), y, z\right)$;

b) В плоскости YZ: $F(x,y,z) \rightarrow F\left(x, y + z * \tan\left(radians(angle)\right), z\right)$;

c) В плоскости XZ: $F(x,y,z) \rightarrow F\left(x + z * \tan\left(radians(angle)\right), y, z\right)$.

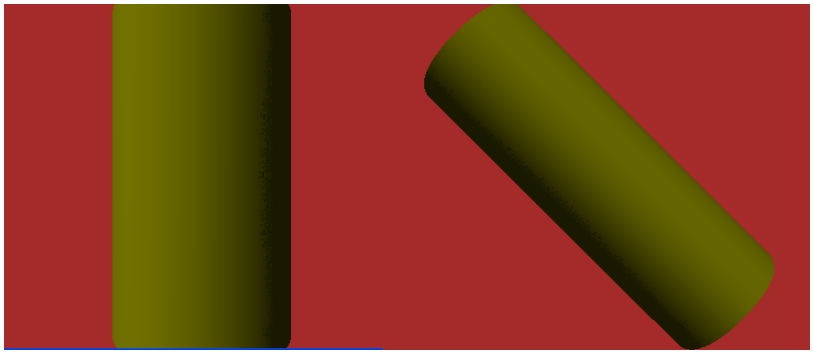
Пример: сдвиг цилиндра, направленного вдоль оси Y, на 45 градусов в плоскости XY

```
#include "functions.inc"
#include "colors.inc"

camera { location  <0, 0, -4> look_at <0, 0, 0>}
light_source {<-100,200,-100> colour rgb 1}
background  { Brown}

#declare R = 2;
#declare  S = function {x*x + z*z  - 1}

isosurface {
  function { S(x+y*tan(radians(45)),y,z) }
       max_gradient 11
       contained_by{sphere{0,R}}
       pigment {Yellow}
}
```



4. **Поворот** осуществляется следующим образом:

   a) Вокруг оси **X**: $F(x,y,z) \to F(x,y',z')$, где
   $y' = z * \sin(\text{radians}(\text{angle})) + y * \cos(\text{radians}(\text{angle}))$,
   $z' = z * \cos(\text{radians}(\text{angle})) - y * \sin(\text{radians}(\text{angle}))$.

   b) Вокруг оси **Y**: $F(x,y,z) \to F(x',y,z')$, где
   $x' = x * \cos(\text{radians}(\text{angle})) - z * \sin(\text{radians}(\text{angle}))$,
   $z' = x * \sin(\text{radians}(\text{angle})) + z * \cos(\text{radians}(\text{angle}))$.

   c) Вокруг оси **Z**: $F(x,y,z) \to F(x',y',z)$, где
   $x' = x * \cos(\text{radians}(\text{angle})) + y * \sin(\text{radians}(\text{angle}))$,
   $z' = -x * \sin(\text{radians}(\text{angle})) + y * \text{scos}(\text{radians}(\text{angle}))$.

5. **Зеркальное отображение**

   1) Относительно плоскости XY: $F(x,y,z) \to F(y,-x,z)$;
   2) Относительно плоскости YZ: $F(x,y,z) \to F(x,z,-y)$;
   3) Относительно плоскости XZ: $F(x,y,z) \to F(-z,y,x)$.

6. **Кручение** изоповерхности на n оборотов вокруг оси X достигается следующим преобразованием: $F(x,y,z) \to F(x,y',z')$, где
   $y' = z * \sin(x * 2 * \text{pi} * n) + y * \cos(x * 2 * \text{pi} * n)$,
   $z' = z * \cos(x * 2 * \text{pi} * n) - y * \sin(x * 2 * \text{pi} * n)$.
   Аналогично для других осей (см. преобразования для поворота).

Операции над изоповерхностями

Для работы с изоповерхностями доступны аналоги всех тех операций над объектами, которые определены в CSG:

1. **Объединения** нескольких изоповерхностей можно добиться при использовании функции min (S1,S2,..,Sn), где S1,..Sn – функции, описывающие изоповерхности.

   Пример:

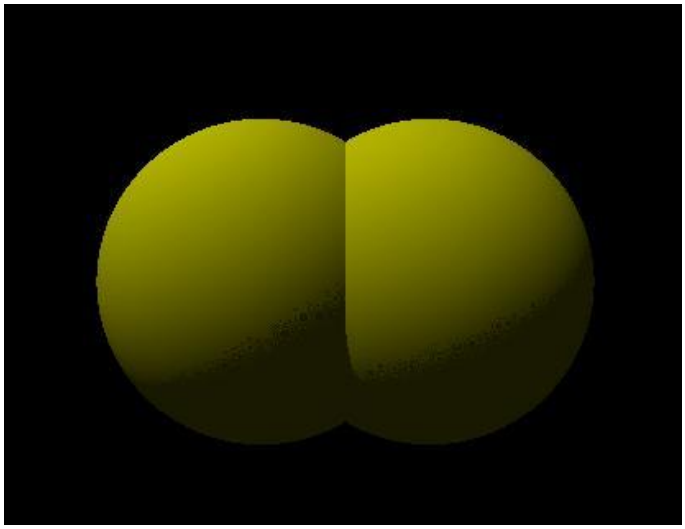
```
 #include "colors.inc"

camera { location  <0, 0, -4> look_at <0, 0, 0>}
light_source {<-100,200,-100> colour rgb 1}

#declare R = 2;
#declare  S = function {x*x + y*y + z*z - 1}

isosurface {
  function { min(S(x+0.5,y,z), S(x-0.5,y,z)) }
       max_gradient 2
       contained_by{sphere{0,R}}
       pigment {Yellow}
}
```

   В результате получаем объединение двух сфер, параллельно перенесённых вдоль оси x на -+ 0.5 соответственно.
   Данная операция может выполняться в отношение двух и более изоповерхностей.



2. Аналогичным образом определяется операция **пересечения** нескольких изоповерхностей. При этом используется функция max (S1,S2,..,Sn), где S1,..Sn – функции, описывающие изоповерхности.
   Рассмотрим действие функции max (S1,S2), где S1, S2 – функции, задающие цилиндр и параллелепипед соответственно.

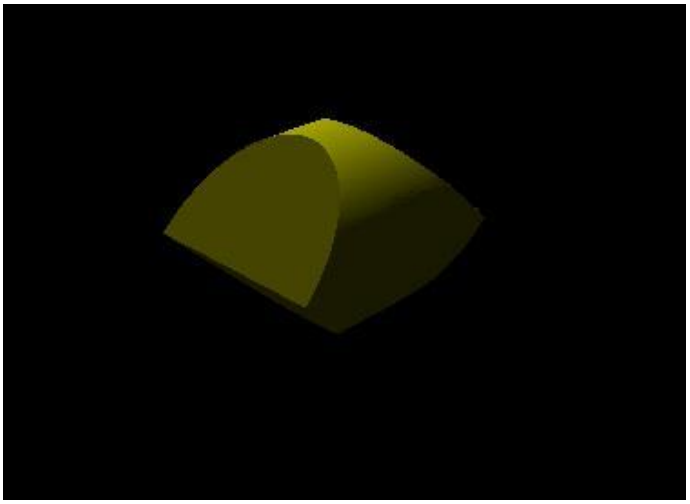
```
#include "colors.inc"
#include "functions.inc"

camera {
      location <5, 3.5, 5>
      look_at  <3, 2,  3>

      angle 50
      }
light_source {<-100,200,-100> colour rgb 1}

#declare S1 = function { sqrt(pow(y,2) + pow(z,2)) - 0.8 }
#declare S2 = function { abs(x)+abs(y)-1 }

isosurface {
 function { max(S1(x,y,z), S2(x,y,z)) }
      max_gradient 2
      pigment {Yellow}
}
```



3. **<u>Суммирование или вычитание</u>** изоповерхностей друг из друга можно задать с помощь обычного сложения или вычитания функций, задающих эти изоповерхности.
   Рассмотрим примеры:
   a) Сумирование двух изоповерхностей:

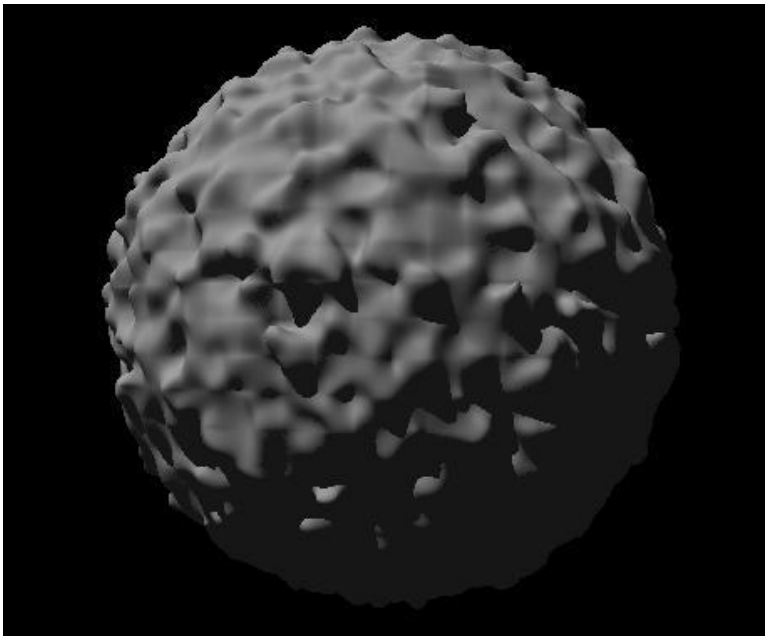
```
#include "functions.inc"

   camera { location  <0, 0, -4> look_at <0, 0, 0> angle 40}
   light_source {<-100,200,-100> colour rgb 1}

   #declare R = 1.1;
   #declare  S = function {x*x + y*y + z*z - 1}

    isosurface {
  function { S(x,y,z) +
  f_noise3d(x*10, y*10, z*10)*0.3 }
       max_gradient 7
```

```
        contained_by{sphere{0,R}}
        pigment {rgb .9}
}
```



Как видно из рисунка, в результате сложения двух функций получаем что-то подобное «смеси» этих функций. В данном примере мы получили фигуру сферической формы, но с шероховатостью на поверхности, что объясняет наложение шероховатости функцией f_noise3d(x*10, y*10, z*10)*0.3 на сферу S = function {x*x + y*y + z*z - 1}.

b) Разность двух изоповерхностей:

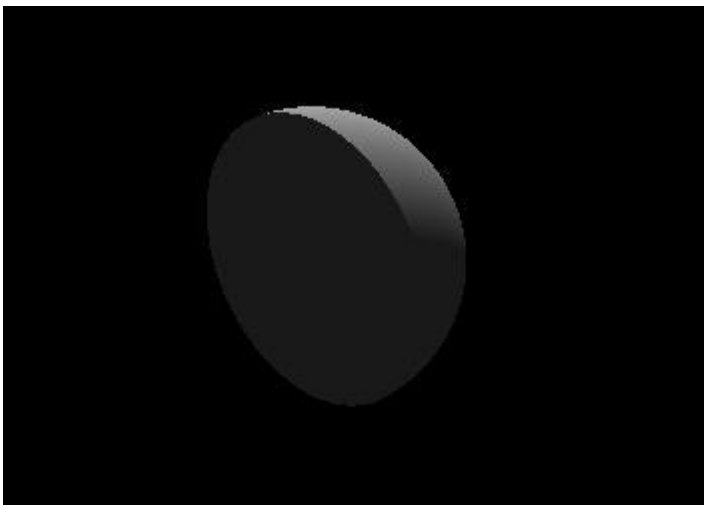
```
#include "functions.inc"
#include "colors.inc"

camera {
        location <5, 3.5, 5>
        look_at  <3, 2,  3>
          angle 50
        }
light_source {<-100,200,-100> colour rgb 1}

#declare R = 1.1;
#declare  S = function {x*x + y*y + z*z - 1}

isosurface {
  function { S(x,y,z) - S(x-0.1, y,z)}

        max_gradient 2
        contained_by{sphere{0,R}}
        pigment {White}
}
```



Разность двух сфер

4. Blob–подобные комбинации

С помощью умножения функций, задающих изоповерхности, можно добиться эффекта, подобного тому, что достигается при работе с blob – объектами – к примеру, плавного перехода одной поверхности в другую.

Рассмотрим пример, в котором параллелепипед плавно переходит в цилиндр:

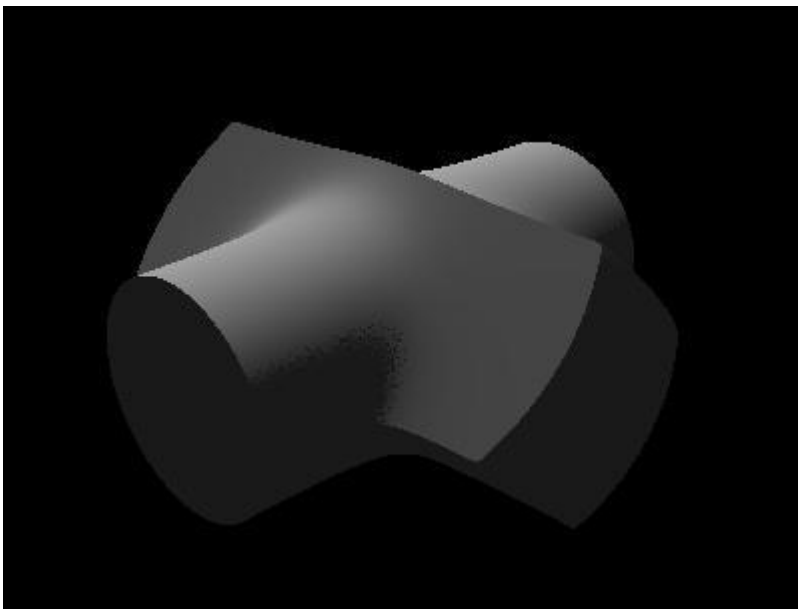
```
#include "functions.inc"
#include "colors.inc"

camera {
      location <5, 3.5, 5>
      look_at  <3, 2,  3>
      angle 50
      }
light_source {<-100,200,-100> colour rgb 1}

#declare S1 = function { sqrt(pow(y,2) + pow(z,2)) - 0.8 } // цилиндр
#declare S2 = function { abs(x)+abs(y)-1 } // перевернутый параллелепипед


isosurface {
 function { S1(x,y,z)*S2(x, y,z)-0.03}

      max_gradient 6
      contained_by{sphere{0,2}}
      pigment {White}
}
```



Данный прием удобнее тем, что он применим при работе с любыми поверхностями, а не только со сферами и цилиндрами, как при работе с blob-объектами.

3.7.12 shapes.inc, shapes_old.inc, shapes2.inc, shapesq.inc

Файлы с таким расширением содержат предопределённые формы и макросы, их генерирующие.

"shapes.inc" включает в себя "shapes_old.inc" и содержит множество макросов для работы с объектами , а тка же для создания специальных объектов - таких, как наклонный текст, шарообразные фигуры и прочее.

Ряд объектов в рамках "shapes_old.inc" не очень полезны при работе с новейшими версиями POV-Ray, они сохранены для обеспечения совместимости для сцен, написанных в старых версиях, в которых отсутствовали такие объкты, как понусы, диски, плоскости и другие.

Файл "shapes2.inc" содержит несколько полезных форм, в том числе правильные многогранники, а "shapesq.inc" содержит определения нескольких прямоугольникной и кубической формы объектов.

Было бы хорошо, если бы объекты из "shapesq.inc" проще генерировались, были более гибкими по своим свойствам , и их рендеринг был так же быстр, как в случае работы с изоповерхностя. Но несмотря на эти недостатки, они полезны для обеспечения совместимости со старыми версиями, а так же могут быть неограниченными, в отличие от изоповерхностей.

3.7.12.1 shapes.inc

Макросы Isect(Pt, Dir, Obj, OPt) и IsectN(Pt, Dir, Obj, OPt, ONorm) взаимодействуют с функцией trace(). Первый возвращает точку пересечения, второй – нормаль к поверхности, с которой пересекается луч.

Информацию о точке пересечения и нормали к поверхности макросы передают через свои параметра, а так же правду или ложь в зависимости от того, есть ли пересечение или нет: если пересечение найдено, макросы возвращают правду и устанавливают на месте параметра Opt точку пересечения, а на месте ONorm – нормаль. В противном случае параметры Opt ONorm не изменяются и макросы возвращают ложь.

Описание параметров:

- Pt – точка, из которой исходит луч.
- Dir – направление луча.
- Obj – объект, наличие пересеченией с которым определяют макросы.
- Opt – переменная, в которую макрос заносит значение точки пересечения.
- ONorm – переменная, в качестве значения которой устанавливается нормаль к поверхности в точке пересечения с лучом.

Extents(Obj, Min, Max) - макрос, при вызове которого, в качестве параметров Min и Max устанавливаются значения максимального и минимального размера объекта (то есть для получения крайних углов ограничивающего объект бокса).

- Obj = объект, границы которого вы устанавливаете.
- Min = объявленная переменная, значение которой макрос установит в качестве нижней границы объекта.
- Max = объявленная переменная, значение которой макрос установит в качестве нижней границы объекта.

Center_Object(Object, Axis) – макрос, который выполняет центрирование объекта.

Параметры:

- Object = центрируемый объект.
- Axis = См. Center_Trans() в документации для transforms.inc .

Align_Object(Object, Axis, Pt) – объявление для применения к объекту макроса Align_Trans().

Описание параметров:

- Object – объект для выравнивания.
- Axis – смотрите Align_Trans() в документации по transforms.inc

- Point – точка, относительно которой выравнивается ограничивающий объект бокс.

Bevelled_Text(Font, String, Cuts, BevelAng, BevelDepth, Depth, Offset, UseMerge) – макрос, используемый для того, чтобы добиться скошенности противоположных граней текстовых объектов. Эффект достигается построением пересечения множества копий текстового объекта, каждая из которых сдвинута в каком-либо направлении. Результат, конечно, не превосходный, но может быть полезным для некоторых целей.

Предупреждение: объект, для которого используют макрос Bevelled_Text визуализируется гораздо медленнее, чем если этот макрос применять к объекту.

Описание параметров:

- Font – параметр, который определяет применимый к объекту шрифт.
- String – текстовая строка, на основе которой генерируется объект.
- Cuts – параметр, определяющий количество пересечений для обеспечения скошенности текста. Большее значение параметра даёт более гладкий объект, но при этом используется больше ресурсов компьютора и процесс визуализации в итоге медленней.
- BevelAng – угол скоса граней текстового объекта.
- BevelDepth – толщина скошенной части оьъекта.
- Depth – общая толщина текстового оьъекта, который в итоге получается.
- Offset – парамерт, определяющий величину отступа текстового объекта. Координата по z не учитывается, так как для осуществления сдвига противоположные грани объекта должны быть компланарными.
- UseMerge – переключатель между merge(1) и union (0).

  Text_Space(Font, String, Size, Spacing) вычисляет ширину текстовой строки, включая пробелы, возвращает ширину всех n символов.

  Параметры:

- Font – параметр, задающий шрифт;

- String – текстовая строка, из которой сгенерирован объект;
- Size – параметр для масштабирования объекта;
- Spacing - количество пробелов для добавления между символами.

Text_Width(Font, String, Size, Spacing)вычисляет длину текстовой строки и возвращает длину первых n-1 символов + длину глифа последнего. Text_Width вычисляет «физическую» длину текста и, если вы используете только один символ, - «физическую» ширину глифа.

Парамтеры:

- Font – параметр, задающий шрифт;
- String – текстовая строка, из которой сгенерирован объект;
- Size – параметр для масштабирования объекта;
- Spacing - количество пробелов для добавления между символами.

Text_Space(Font, String, Size, Spacing) вычисляет ширину текстовой строки, включая пробелы, возвращает ширину всех n символов.

Параметры:

- Font – параметр, задающий шрифт;
- String – текстовая строка, из которой сгенерирован объект;
- Size – параметр для масштабирования объекта;
- Spacing - количество пробелов для добавления между символами.

Константы Align_Left, Align_Right, Align_Centerиспользуются при работе с макросом Circle_Text().

Circle_Text(Font, String, Size, Spacing, Depth, Radius, Inverted, Justification, Angle)создаёт текстовый объект, верхняя или нижняя грань которого полностью или частично выровнены по кругу. Этот макрос следует использовать внутри блока, описывающего объект object{...}.

Параметры:

- Font – параметр, задающий шрифт;
- String – текстовая строка, из которой сгенерирован объект;
- Size – параметр для масштабирования объекта;
- Spacing - количество пробелов для добавления между символами;
- Depth – толщина текстового объекта;
- Radius – радиус круга, по которому выровнены знаки;
- Inverted – если параметр ненулевой, то верхняя часть символа будет напрвлена к центру круга, в противном случае – нижняя ;
- Justification – параметр, который принимает значения
- Align_Left, Align_Right, или Align_Center – выравнивание по левому, правому краю или по центру.
- Angle – точка на круга, с которой начинается визуализация , увеличивается в направлении против часовой стрелки.

Макрос Wedge(Angle)создает бесконечную клинообраазную форму – пересечение двух плоскостей. Имеет основное применение в CSG, например, для того, чтобы получить специфической формы дугу в торе. При построении такого рода объектов одно ребро располагается вдоль оси y, другая сторона совмещается с плоскостью zy, остальные вращаются вокруг оси y по часовой стрелке.

Параметры:

- Angle – угол в градусах между сторонами клинообразной формы.

Spheroid(Center, Radius) – макрос, создающий неровно масштабируванную сферу.

Параметры:

- Center – центр сферойда;
- Radius – вектор, задающий радиусы сферойда.

Макрос Supertorus(MajorRadius, MinorRadius, MajorControl, MinorControl, Accuracy, MaxGradient)предназначен для построения тора. Параметр MaxGradient следует установить таким, чтобы он соответствовал параметрам, котороые вы выбираете, больший по площади тор должен иметь большой MaxGradient.

Параметры:

- majorRadius, MinorRadius – внешний и внутренниё радиусы тора;
- MagorControl, MinorControl – параметры ,отвечающие за округлённость тора. Используйте значения в пределах отрезка [0;1];
- Accuracy – параметр точности;
- MaxGradient – параметр, определяющий максимальный градиент луча.

Supercone(EndA, A, B, EndB, C, D) – макрос, создающий объект, подобный конусу, но у которого конечный точки – эллипсы. Объект является пересечением поверхности четвёртого порядка и цилиндра.

Параметры:

- EndA – центр конца A;
- A,B – радиусы эллипса с центром в точке A;
- EndB – центр конца B;
- C, D – радиусы эллипса с центром в точке B.

Макрос Connect_Spheres(PtA, RadiusA, PtB, RadiusB)моделирует конус, который получается в результате плавного слияния двух сфер.

Параметры:

- PtA – центр сферы A;
- RadiusA – радиус сферы A;
- PtB – центр сферы B;
- RadiusB – радиус сферы B.

Wire_Box_Union(PtA, PtB, Radius),
Wire_Box_Merge(PtA, PtB, Radius),
Wire_Box(PtA, PtB, Radius, UseMerge) – макросы, создающие каркасные боксы из цилиндров и сфер. Результирующий объект будет полностью заполнять объект бокс с теми же угловыми точками.

Параметры:

- PtA – нижний левый угол бокса;

- PtB – верхний правый угол бокса;
- Radius – радиус цилиндра или сферы, из которой формируется бокс;
- UseMerge – параметр, принимающий два значения – 0,1, которые говорят использовать или нет соединение объектов.

Round_Box_Union(PtA, PtB, EdgeRadius),
Round_Box_Merge(PtA, PtB, EdgeRadius),
Round_Box(PtA, PtB, EdgeRadius, UseMerge) – макрос, который из боксов, цилиндров или сфер создает бокс с округлёнными углами. Результирующий объект будет полностью заполнять объект бокс с теми же угловыми точками.

Параметры:

- PtA – нижний левый угол бокса;
- PtB – верхний правый угол бокса;
- EdgeRadius – радиус рёбер бокса;
- UseMerge – параметр, принимающий два значения – 0,1, которые говорят использовать или нет соединение объектов.

Round_Cylinder_Union(PtA, PtB, Radius, EdgeRadius),
Round_Cylinder_Merge(PtA, PtB, Radius, EdgeRadius),
Round_Cylinder(PtA, PtB, Radius, EdgeRadius, UseMerge) – макрос, создающий из цилиндров и торов цилиндр с округлёнными гранями.

Результирующий объект будет полностью заполнять объект цилиндр с теми же угловыми точками и радиусом.

Параметры:

- PtA, PtB – концевые точки цилиндра;
- Radius – радиус цилиндра;
- EdgeRadius – радиус граней цилиндра;
- UseMerge - параметр, принимающий два значения – 0,1, которые говорят использовать или нет соединение объектов.

Макрос Round_Cone_Union(PtA, RadiusA, PtB, RadiusB, EdgeRadius),
Round_Cone_Merge(PtA, RadiusA, PtB, RadiusB, EdgeRadius),

Round_Cone(PtA, RadiusA, PtB, RadiusB, EdgeRadius, UseMerge) из конусов и торов создает цилиндр с округлёнными гранями.

Результирующий объект будет полностью заполнять объект конус с теми же угловыми точками и радиусом.

Параметры:

- PtA, PtB – концевые точки цилиндра;
- Radius – радиус цилиндра;
- EdgeRadius – радиус граней цилиндра;
- UseMerge - параметр, принимающий два значения – 0,1, которые говорят использовать или нет соединение объектов.

Макрос Round_Cone2_Union(PtA, RadiusA, PtB, RadiusB),
Round_Cone2_Merge(PtA, RadiusA, PtB, RadiusB),
Round_Cone2(PtA, RadiusA, PtB, RadiusB, UseMerge)создаёт из конуса и двух сфер конус с округлёнными гранями.

Результирующий объект будет полностью заполнять объект конус с теми же угловыми точками и радиусами.

Параметры:

- PtA, PtB – центры сферических крышек;
- Radius – радиусы сферический крышек;
- UseMerge - параметр, принимающий два значения – 0,1, которые говорят использовать или нет соединение объектов.

Round_Cone3_Union(PtA, RadiusA, PtB, RadiusB),
Round_Cone3_Merge(PtA, RadiusA, PtB, RadiusB)
Round_Cone3(PtA, RadiusA, PtB, RadiusB, UseMerge). Подобно Round_Cone2()макрос из конуса и двух сфер создаёт конус с округлёнными гранями, и результирующий объект будет полностью заполнять объект конус с теми же угловыми точками и радиусами. Разница в том, что это макрос берет концевые точки части конуса и передвигает сферы так, что быть совмещённой с поверхностью, вместо помещения сфер в концевые точки.

Параметры:

- PtA, PtB – центры сферических крышек;
- Radius – радиусы сферический крышек;
- UseMerge - параметр, принимающий два значения – 0,1, которые говорят использовать или нет соединение объектов.

Макросы Quad(A, B, C, D) и Smooth_Quad(A, NA, B, NB, C, NC, D, ND)создают 4-х – сторонние полигональные объекты, используя пары треугольников.

Параметры:

- A,B,C,D – вершины четырёхгранника;
- NA, NB, NC, ND – нормали к вершинам четырёхгранника.

3.7.12.1.1 The HF Macros (Высокочастотные макросы)

Библиотека shapes.inc содержит несколько высокочастотных макросов. Которые генерируют сетки в различных объектах. Общие особенности для высокочастотных макросов:

- Все высокочастотные макросы не используют напрямую изображения в качестве вводимых параметров, они вычисляют определённую пользователем функцию. Макросы деформируют поверхность, построенную на вычисленных значениях функции.
- Макросы либо записывают в файл, либо содают объект напрямую. Если вы хотите организовать запись в файл,укажите имя файла в списке параметров, если же Вы хотите выводить изображение напрямую, в параметре для имени файла укажите пустую строку – « ».

  Способ использования различных высокочастотных макросов описан ниже.

HF_Square (Function, UseUVheight, UseUVtexture, Res, Smooth, FileName, MnExt, MxExt) – генерирует сетвку в форме прямоугольной карты высот, подобной встроенному примитиву height_field.

Параметры:

- Function - функция для использования с целью деформровать построенную область;
- UseUVheight – логическая величина, говорящая макросу, использовать или нет UV – преобразование;
- UseUVtexture - логическая величина, говорящая макросу, использовать или нет текстурированное UV – преобразование;
- Res – двухмерный вектор, задающий разрешение геререируемой сетки;
- Smooth - логическая величина, говорящая макросу, использовать или нет сглаживание сгенерированной сетки;

- Filename – название файла, в который будет выводиться результат;
- MnExt – передний нижний левый угол бокса, содержащего обработанную область;
- MxExt – задний правый верхний угол бокса, содержащего обработанную область.

Макрос HF_Sphere(Function, UseUVheight, UseUVtexture, Res, Smooth, FileName, Center, Radius, Depth)генерирует сетку в форме сферической карты высот. Когда использовано UV – отображение (преобразование плоского объекта в трёхмерный при переходе к u-v координатам), прямоугольник натягивается на сферу начиная с положительного направления оси x против часовой стрелки вокруг оси y.

Параметры:

- Function - функция для использования с целью деформровать карту высот;
- UseUVheight – логическая величина, говорящая макросу, использовать или нет UV – преобразование карты высот;
- UseUVtexture - логическая величина, говорящая макросу, использовать или нет текстурированное UV – преобразование карты высот;
- Res – двумерный вектор, задающий разрешение герерируемой сетки;
- Smooth - логическая величина, говорящая макросу, использовать или нет сглаживание сгенерированной сетки;
- Filename – название файла, в который будет выводиться результат;
- Center – центр объёмной области перед сдвигом; сдвиг, вероятнее всего, сделает объект сдвинутым по отношению к центру;
- Radius – начальный радиус сферы перед сдвигом;
- Depth – глубина карты высот.

HF_Cylinder(Function, UseUVheight, UseUVtexture, Res, Smooth, FileName, EndA, EndB, Radius,Depth)- генерирует сетку в форме бесконечной цилиндрической карты высот. Когда используется UV-ghtj,преобразование, прямоугольник натягивается на цилиндр.

Параметры:

- Function - функция для использования с целью деформровать построенную область;
- UseUVheight – логическая величина, говорящая макросу, использовать или нет UV – преобразование карты высот;

- UseUVtexture - логическая величина, говорящая макросу, использовать или нет текстурированное UV – преобразование карты высот;
- Res – двухмерный вектор, задающий разрешение геререируемой сетки;
- Smooth - логическая величина, говорящая макросу, использовать или нет сглаживание сгенерированной сетки;
- Filename – название файла, в который будет выводиться результат;
- EndA, EndB – крайние точки цилиндра;
- Radius - радиус цилиндра до сдвига;
- Depth - глубина карты высот.

HF_Torus (Function, UseUVheight, UseUVtexture, Res, Smooth, FileName, Major, Minor, Depth)- макрос, генерирующий сетку в форме торообразной карты высот. Когда использовано UV-преобразование,прямоугольник натягивается на тор.

Параметры:

- Function - функция для использования с целью деформровать построенную область;
- UseUVheight – логическая величина, говорящая макросу, использовать или нет UV – преобразование карты высот;
- UseUVtexture - логическая величина, говорящая макросу, использовать или нет текстурированное UV – преобразование карты высот;
- Res – двухмерный вектор, задающий разрешение геререируемой сетки;
- Smooth - логическая величина, говорящая макросу, использовать или нет сглаживание сгенерированной сетки;
- Filename – название файла, в который будет выводиться результат;
- Major – внешний радиус тора;
- Minor – внутренний радиус тора.

3.7.12.2 shapes_old.inc

Ellipsoid, Sphere

Сфера с единичным радиусом в начале координат.

Cylinder_X, Cylinder_Y, Cylinder_Z

Цилиндры, бесконечные вдоль соответствующих осей координат.

QCone_X, QCone_Y, QCone_Z

Бесконечные вдоль соответствующих осей координат конусы.

Cone_X, Cone_Y, Cone_Z

Закрытые в основании конусы: единичного радиуса при X/Y/Z равном -1 и нулевого радиуса при X/Y/Z равном 0.

Plane_YZ, Plane_XZ, Plane_XY

Бесконечные плоскости, проходящие через начало координат.

Paraboloid_X, Paraboloid_Y, Paraboloid_Z

Параболоиды, заданные уравнением: y^2 + z^2 - x = 0/ x^2 + z^2 - y = 0/ y^2 + x^2 - x = 0

Hyperboloid, Hyperboloid_Y

Гиперболоид, задающийся уравнением: y - x^2 + z^2 = 0

UnitBox, Cube

Куб с ребров длиной 2, центрированный по началу координат.

Disk_X, Disk_Y, Disk_Z

Замкнутые цилиндры с радиусом основания, равным 1, и высотой, равной 2.

3.7.12.3 shapes2.inc

Tetrahedron

Тетраэдр -- 4-гранный правильный многогранник.

Octahedron

Октаэдр -- 8-гранный правильный многогранник.

Dodecahedron

Додекаэдр – 12-гранный правильный многогранник.

Icosahedron

Икосаэдр – 20-гранный правильный многогранник.

Rhomboid

Трёхмерный четырехгранный ромб, основанный на сдвинутом боксе.

Hexagon

Правильный шестигранный многогранник с осью вдоль x.

HalfCone_Y

Конечный конус, направленный вдоль оси y.

Pyramid

Четырёхгранная пирамида (объединение треугольников, не может быть использована в конструктивной геометрии).

Pyramid2

Четырёхгранная пирамида – пересечение плоскостей. Может быть использована в конструктивной геометрии.

Square_X, Square_Y, Square_Z

Конечные плоскости, перпендикулярные соответствующим осям, площадью 2 единицы.

3.7.12.4 shapesq.inc

Bicorn

Эта кривая выглядит, как верхняя часть параболоида, ограниченная снизу другим параболоидом. Задаётся уравнением:

y^2 - (x^2 + z^2) y^2 - (x^2 + z^2 + 2 y - 1)^2 =

Cubic_Cylinder

Кривая, задаваемая уравнением y^2 + z^2 = 0.5 (x^3 + x^2) .

Cubic_Saddle_1

Седло, задающееся кубическим уравнением

z = x^3 - y^3 .

Devils_Curve

Вариант кривой в трехмерном пространстве. Эта фигура имею подобные гиперболоиду верхнюю и нижнюю части, средняя часть объекта сжата и содержит два похожих на каплю отверстия.

Уравнение, задающее объект:

x^4 + 2 x^2 z^2 - 0.36 x^2 - y^4 + 0.25 y^2 + z^4 = 0

Folium

Это лист, повернутый вокруг оси x. Задаётся уравнением:

2 x^2 - 3 x y^2 - 3 x z^2 + y^2 + z^2 = 0

Glob_5

Фигура, имеющая форму капли, задающаяся уравнением:

y^2 + z^2 = 0.5 x^5 + 0.5 x^4

Twin_Glob

Модификация лемнискаты – две доли, похожие на слезинки.

Helix, Helix_1

Уравнение, аппроксимирующее спираль:

z = arctan(y/x).

Спираль может быть аппроксимирована алгебраическим уравнением в результате следующих шагов:

tan(z) = y/x => sin(z)/cos(z) = y/x =>
(1) x sin(z) - y cos(z) = 0. Используем разложение синуса и косинуса в ряд Тейлора около нуля

sin(z) = z - z^3/3! + z^5/5! - ...
cos(z) = 1 +z^2/2! - z^6/6! - ...

Опустив старшие степени, записываем выражение (1) в виде:

x (z - z^3/6) - y (1 + z^2/2) = 0 или

(2) -1/6 x z^3 + x z + 1/2 y z^2 - y = 0

Спираль (2) закручена на 90 градусов в пределах интервала 0 <= z <= sqrt(2)/2 и в интервале 0 <= z <= sqrt(2) = 1.4042, если к спирали применено масштабирующее преобразование <2 2 2>.

Hyperbolic_Torus

Внешний радиус гиперболического тора равен sqrt(40), внутренний sqrt(12). Объект генерируется путём изгиба в кольцо ветвей гиперболы. Задаётся уравнением:

x^4 + 2 x^2 y^2 - 2 x^2 z^2 - 104 x^2 + y^4 - 2 y^2 z^2 + 56 y^2 + z^4 + 104 z^2 + 784 = 0

Lemniscate

Лемниската Бернулли выглядит как две капил, соединенные своими концами. Фигура получается вращением Лемнискаты вокруг оси X. Задаётся уравнением:

x^4 - x^2 + y^2 + z^2 = 0

Quartic_Loop_1

Фигура с шероховатой поверхностью с одной стороны и чем-то похожим на параболоид

(но заполненный изнутри).

Задается уравнением:

(x^2 + y^2 + a c x)^2 - (x^2 + y^2)(c - a x)^2

-99*x^4+40*x^3-98*x^2*y^2-98*x^2*z^2+99*x^2+40*x*y^2

+40*x*z^2+y^4+2*y^2*z^2-y^2+z^4-z^2

Monkey_Saddle

Эта поверхности состоит из трех частей, загнутых вверх и трех вниз.

Уравнение:

z = c (x^3 - 3 x y^2)

Величина параметра c задаёт вертикальный масштаб поверхности: чем меньше c, тем более плоской в районе начала координат будет поверхность.

Parabolic_Torus_40_12

Параболический тор с внешним радиусом, равым sqrt(40), внутренним sqrt(12). Объект генерируется путём изгиба в кольцо ветвей параболы. Задаётся уравнением:

x^4 + 2 x^2 y^2 - 2 x^2 z - 104 x^2 + y^4 - 2 y^2 z + 56 y^2 + z^2 + 104 z + 784 = 0

Quartic_Paraboloid

Quartic parabola - полином четвёртой степени, изогнутый вокруг оси z. Задаётся уравнением:

0.1 x^4 - x^2 - y^2 - z^2 + 0.9 = 0.

Torus_40_12

Тор с внешним радиусом, равным sqrt(40), и внутренним, равным sqrt(12).

Sinsurf

Очень грубая аппроксимация синусоидальной поверхности z = sin(2 pi x y).

Чтобы точность аппроксимации составляла, к примеру, 7 знаков после запятой на расстоянии в единицу от начала координат, требуется полином степени 60 и выше, который содержит около 200 тысяч коэффициентов.

Для лучшего результата используйте масштаб

<1 1 0.2>.

3.7.7 functions.inc

Этот включаемый файл содержит интерфейсы ко внутренним функциям, а также к нескольким предопределённым. Идентификаторы, используемые для их вызова, могут не быть одинаковыми в различных версиях POV-Ray, для этого создана эта библиотека.

Количество требуемых параметров и то, за что они отвечают, также дано во включаемом файле, эта глава даёт подробную информацию.

Для того, чтобы посмотреть начальные значения параметров, посмотрите демонстрационный файл "i_internal.pov".

Используемый синтаксис:

```
#include "functions.inc"
isosurface {
  function { f_torus_gumdrop(x,y,z, P0) }
  ...
}

pigment {
  function { f_cross_ellipsoids(x,y,z, P0, P1, P2, P3) }
  COLOR_MAP ...
)
```

В некоторых описанных функциях есть специальные параметры, они будут описаны в следующей секции.

3.7.7.1 Общие параметры

Тип поперечного сечения:

В спиралях и спиралевидных функциях девятый параметр – параметр, задающий тип поперечного сечения. Например:

0 :

Квадрат

0.0 to 1.0 :

Закруглённый квадрат

1 :

Круг

1.0 to 2.0 :

Закруглённые ромбы

2 :

Ромб

2.0 to 3.0 :

Частично вогнутый ромб

3 :

Вогнутый ромб

3.7.7.1.1 Field strength (Интенсивность Поля)

Численное значение в точке пространства вычисляемое функцией умножается на Интенсивность Поля. Набор точек, значение функции в которых равно нулю, остается без изменений для каждого (любого) положительного значения данного параметра, т.е. если вы используете функцию с пороговым значением = 0, выходная поверхность останется прежней.

В некоторых случаях, Интенсивность Поля вносит заметный эффект в скорость и точность рендеринга. В общем случае увеличение интенсивности ведёт к ускорению рендеринга, но если значение слишком велико, это может привести к излому или полному исчезновению поверхности.

Установка интенсивности поля в отрицательное значение приводит к инверсии поверхности, явно, как и установка функции отрицательного знака.

3.7.7.1.2 Field Limit (Граница Поля)

Не привносит никаких изменений на выходную поверхность, если вы используете пороговое значение внутри границ поля (или полностью уничтожит поверхность при пороговом значении выше границ). Тем не менее, это может существенно повлиять на время рендеринга.

Если вы используете функцию для пигментации, тогда все точки, достаточно далёкие от поверхности, будут одного цвета, того цвета, который соответствует величине границы поля.

3.7.7.1.3 SOR Switch (Переключатель ПВ (ППВ))

Если данное значение больше нуля, тогда кривая представляется в виде поверхности вращения (ПВ).

В случае отрицательного или нулевого значения кривая вытягивается линейно в направлении оси Z.

3.7.7.1.4 SOR offset (Смещение ПВ)

В случае включённого ППВ кривая смещается на данную дистанцию в направлении оси X перед формообразованием.

3.7.7.1.5 SOR Angle (Угол ПВ)

В случае включённого ППВ кривая поворачивается вокруг оси Z перед формообразованием.

3.7.7.1.6 Invert isosurface (Инверсия Изоповерхности)

Иногда, когда вы рендерите поверхность, вы можете обнаружить, что на выходе получается только фигура контейнера (оболочки). Это может быть вызвано тем, что некоторые встроенные функции определены наизнанку.

Можно инвертировать всю Изоповерхность при помощи отрицания всей функции:

-(function) - threshold

3.7.7.2 Встроенные функции

Ниже приводится список всех встроенных функций в порядке их появления в файле “Functions.h”

f_algbr_cyl1(x,y,z, P0, P1, P2, P3, P4). Алгебраический цилиндр, это то, что вы получите, если возьмёте двухмерную кривую и проецируете её в 3D. Двухмерная кривая просто выдавливается вдоль третьей оси, в данном случае это ось Z.
В случае включённого SOR Switch , цилиндр не выдавливается вдоль оси Z, а получается вращением вокруг оси Y.

- P0 : Field Strength
- P1 : Field Limit
- P2 : SOR Switch
- P3 : SOR Offset
- P4 : SOR Angle

f_algbr_cyl2(x,y,z, P0, P1, P2, P3, P4). Алгебраический цилиндр, это то, что вы получите, если возьмёте двухмерную кривую и проецируете её в 3D. Двухмерная кривая просто выдавливается вдоль третьей оси, в данном случае это ось Z.
В случае включённого SOR Switch, область пересечения кривой будет образующей при вращении вокруг оси Y вместо выдавливания по оси Z.

- P0 : Field Strength (Необходима отрицательная величина или отрицательная функция.)
- P1 : Field Limit
- P2 : SOR Switch
- P3 : SOR Offset
- P4 : SOR Angle

f_algbr_cyl3(x,y,z, P0, P1, P2, P3, P4). Алгебраический цилиндр, это то, что вы получите, если возьмёте двухмерную кривую и проецируете её в 3D. Двухмерная кривая просто выдавливается вдоль третьей оси, в данном случае это ось Z.
В случае включённого SOR Switch, область пересечения кривой будет образующей при вращении вокруг оси Y вместо выдавливания по оси Z.

- P0 : Field Strength (Необходима отрицательная величина или отрицательная функция.)
- P1 : Field Limit
- P2 : SOR Switch
- P3 : SOR Offset
- P4 : SOR Angle

f_algbr_cyl4(x,y,z, P0, P1, P2, P3, P4). Алгебраический цилиндр, это то, что вы получите, если возьмёте двухмерную кривую и проецируете её в 3D. Двухмерная кривая просто выдавливается вдоль третьей оси, в данном случае это ось Z.
В случае включённого SOR Switch, область пересечения кривой будет образующей при вращении вокруг оси Y вместо выдавливания по оси Z.

- P0 : Field Strength (Необходима отрицательная величина или отрицательная функция.)
- P1 : Field Limit
- P2 : SOR Switch
- P3 : SOR Offset
- P4 : SOR Angle

f_bicorn(x,y,z, P0, P1). Поверхность является поверхностью вращения.

- P0 : Field Strength (Необходима отрицательная величина или отрицательная функция.)
- P1 : Scale. Матмодель данной поверхности предполагает, что форма поверхности будет различной для различных значений данного параметра. Установка значения 3 даёт в результате поверхность с радиусом в 1 единицу.

f_bifolia(x,y,z, P0, P1). Бифолия (Двулистник) выглядит как параболоид связанный у основания с другим параболоидом.

- P0 : Field Strength (Необходима отрицательная величина или отрицательная функция.)
- P1 : Scale. Поверхность всегда имеет один и тот же вид. Изменение данного параметра имеет схожий эффект с добавлением модификатора Scale. Установка значения 1 даёт в результате поверхность с радиусом в 1 единицу.

f_blob(x,y,z, P0, P1, P2, P3, P4). Эта функция генерирует пузырь схожий с «CSG blob» с двумя сферическими компонентами. Данная функция работает только с отрицательными значениями “threshold”.

- P0 : Расстояние между двумя компонентами
- P1 : Натяжение первой компоненты
- P2 : Инверсия радиуса пузыря для первой компоненты
- P3 : Натяжение второй компоненты
- P4 : Инверсия радиуса пузыря для второй компоненты

f_blob2(x,y,z, P0, P1, P2, P3). Эта функция генерирует пузырь схожий с «CSG blob» с двумя сферическими компонентами.

- P0 : Separation. Одна компонента находится по центру, в то время как другая смещена по оси X.
- P1 : Inverse size. Увеличьте для уменьшения размеров поверхности.
- P2 : Натяжение.
- P3 : Инверсивный параметр к модификатору “threshold”

f_boy_surface(x,y,z, P0, P1). Для данной поверхности желательно выставить малое значение “Field Strength”,в противном случае поверхность может изломаться или вовсе исчезнуть. Это имеет сторонний эффект замедлять скорость рендеринга.

- P0 : Field Strength (Необходима отрицательная величина или отрицательная функция.)
- P1 : Scale. Поверхность всегда имеет один и тот же вид. Изменение данного параметра имеет схожий эффект с добавлением модификатора Scale.

f_comma(x,y,z, P0). Поверхность имеет вид пунктуационного знака «запятая» .

- P0 : Scale

f_cross_ellipsoids(x,y,z, P0, P1, P2, P3). Данная поверхность имеет вид объединения 3х эллипсоид ориентированных вдоль каждой оси.

- P0 : Eccentricity. В случае значения меньшего единицы эллипсы выглядят сплющенными, в случае значения более 1 растянутыми, в случае со значением равным нулю – эллипсоиды принимают форму сферы.
- P1 : Inverse size. Увеличьте для уменьшения размеров поверхности.
- P2 : Diameter. Увеличьте для увеличения размеров эллипсоид.
- P3 : Threshold. Инверсивный параметр к параметру “threshold”

f_crossed_trough(x,y,z, P0)

- P0 : Field Strength (Необходима отрицательная величина или отрицательная функция.)

f_cubic_saddle(x,y,z, P0). Для данной поверхности желательно выставить малое значение “Field Strength”,в противном случае поверхность может изломаться или вовсе исчезнуть.

- P0 : Field Strength (Необходима отрицательная величина или отрицательная функция.)

f_cushion(x,y,z, P0)

- P0 : Field Strength (Необходима отрицательная величина или отрицательная функция.)

f_devils_curve(x,y,z, P0)

- P0 : Field Strength (Необходима отрицательная величина или отрицательная функция.)

f_devils_curve_2d(x,y,z, P0, P1, P2, P3, P4, P5). Кривая f_devils_curve_2d может быть повёрнута вокруг оси X для образования поверхности вращения, или может быть вытянута вдоль оси Z (выключив SOR switch)

- P0 : Field Strength (Необходима отрицательная величина или отрицательная функция.)
- P1 : X factor
- P2 : Y factor
- P3 : SOR Switch
- P4 : SOR Offset
- P5 : SOR Angle

f_dupin_cyclid(x,y,z, P0, P1, P2, P3, P4, P5)

- P0 : Field Strength (Необходима отрицательная величина или отрицательная функция.)
- P1 : Большой радиус

- P2 : Малый радиус
- P3 : смещение по оси X
- P4 : смещение по оси Y
- P5 : Радиус инверсии

f_ellipsoid(x,y,z, P0, P1, P2). f_ellipsoid Генерирует сферы и эллипсоиды. Необходимо значение "threshold 1".
Изменение параметров масштаба ведёт к линейному аффинному изменению фигуры.

- P0 : Инверсивный масштаб по оси X
- P1 : Инверсивный масштаб по оси Y
- P2 : Инверсивный масштаб по оси Z

f_enneper(x,y,z, P0)

- P0 : Field Strength (Необходима отрицательная величина или отрицательная функция.)

f_flange_cover(x,y,z, P0, P1, P2, P3)

- P0 : Spikiness. Уменьшение значения ведёт к увеличению шипов. Установка 1 выдаст сферу
- P1 : Inverse size. Увеличьте для уменьшения размеров поверхности.
- P2 : Flange. Увеличение ведёт к увеличению разброса шипов. Установка 1 – без разброса
- P3 : Threshold. Инверсивный параметр к модификатору “threshold”

f_folium_surface(x,y,z, P0, P1, P2). Парабола соединённая с плоскостью.

- P0 : Field Strength (Необходима отрицательная величина или отрицательная функция.)
- P1 : Показатель ширины – чем больше значение, тем уже место соединения с плоскостью.
- P2 : Дивергенция – чем больше значение, тем шире сама парабола.

f_folium_surface_2d(x,y,z, P0, P1, P2, P3, P4, P5). Кривая f_folium_surface_2d может быть повёрнута вокруг оси X для образования поверхности f_folium_surface, или может быть вытянута вдоль оси Z (выключив SOR switch)

- P0 : Field Strength (Необходима отрицательная величина или отрицательная функция.)
- P1 : Показатель ширины - Аналогично для трёхмерной поверхности вращения вокруг оси Y
- P2 : Дивергенция – Аналогично для трёхмерной поверхности вращения вокруг оси Y
- P3 : SOR Switch
- P4 : SOR Offset
- P5 : SOR Angle

f_glob(x,y,z, P0).Частично уходит в бесконечность, если это не запрещено параметром “contained_by”

- P0 : Field Strength (Необходима отрицательная величина или отрицательная функция.)

f_heart(x,y,z, P0)

- P0 : Field Strength (Необходима отрицательная величина или отрицательная функция.)

f_helical_torus(x,y,z, P0, P1, P2, P3, P4, P5, P6, P7, P8, P9). На некоторых наборах параметров поверхность выглядит как тор со спиральной намоткой вокруг. Опционально намотка прорезается вокруг внешнего пространства.

- P0 : Больший радиус
- P1 : Число петель намотки
- P2 : Закрученость намотки. В случае нулевого значения каждая петля отделена от другой. В случае с единицей петли закручиваются к следующим. В случае со значением 2 закручивается в петлю через одну, и т.д.
- P3 : Толщина накрутки (?)
- P4 : Прямой параметр Threshold.
- P5 : Инверсия малого радиуса
- P6 : Толщина накрутки (?)
- P7 : Период разрезов.
- P8 : Амплитуда разрезов
- P9 : Фаза разрезов. При нуле разрезы симметричны.

f_helix1(x,y,z, P0, P1, P2, P3, P4, P5, P6)

- P0 : Число витков спирали.
- P1 : Период, - число оборотов за единицу длины
- P2 : Малый радиус
- P3 : Большой радиус
- P4 : Параметр формы. Если больше 1 тогда труба - огибающая становится толще в направлении оси Y
- P5 : Cross section type
- P6 : Cross section rotation angle (degrees)

f_helix2(x,y,z, P0, P1, P2, P3, P4, P5, P6). Необходима отрицательная функция.

- P0 : Не используется
- P1 : Период, - число оборотов за единицу длины
- P2 : Малый радиус
- P3 : Большой радиус
- P4 : Не используется
- P5 : Cross section type
- P6 : Cross section rotation angle (degrees)

f_hex_x(x,y,z, P0). Создаёт сетку из шестигранных цилиндров вытянутых вдоль оси z. Толщина контролирутся параметром “threshold”. Когда оно равно 0.8660254 или cos 30 Стороны соединятся, т.к. это растояние между центрами. Установка отрицания на функцию приведёт к созданию поверхности в виде пчелиных сот. Также эта функция полезна при использовании в качестве пигментации.

- P0 : Нет эффекта (хотя по синтаксису положен хотя бы один параметр)

f_hex_y(x,y,z, P0). Данная поверхность образует решётку из параллелепипедов вытянутых вдоль оси Z. Толщина контролирутся параметром “threshold”.Все параллелепипеды повёрнуты на 60 градусов и расстояние между их центрами составляет 0.8660254 или cos 30. Также эта функция полезна при использовании в качестве пигментации.

- P0 : Нет эффекта (хотя по синтаксису положен хотя бы один параметр)

f_hetero_mf(x,y,z, P0, P1, P2, P3, P4, P5). f_hetero_mf (x,0,z) создаёт мультифрактальное поле высот.
Мультифрактальность относится к характеристике фрактальной размерности, которая разнится от высоты. Строится по сумме шумов нескольких частот, параметр hetero_mf определяет количество и вид частот которые участвуют в суммировании. Преимущество в использовании вместо height_field {} из изображения является то, что область определения функции hetero_mf достаточно широка в плоскостях Z и X. Другие полезные функции f_ridged_mf и f_ridge.

- P0 : H - является отрицательным показателем базовых шумовых частот используемых для построения функции (Каждая из частот f определяется как f-H). In landscapes, and many natural forms, the amplitude of high frequency contributions are usually less than the lower frequencies.
  В случае значения равным единицы фрактальность достаточно гладка.
  В случае нулевого значения функция определена как «белый шум».
- P1 : Lacunarity – мультипликатор используемый при переходе от одной «октавы» фрактала к другой. Предполагается устанавливать значения более 1.0
- P2 : Octaves – число различных частот добавляемых к фракталу. Каждая следующая октава умножается на параметр Lacunarity. Соответственно при большом количестве октавы могут формировать очень высокие частоты.
- P3 : Базовая высота, используемая для гетерогенного масштабирования.
- P4 : T - Гетерогенно масштабирует фрактал. T=0 получается прямая зависимость 1/f (не гетерогенное масштабирование). T=1 подавляет высокие частоты на низких высотах.
- P5 : Тип генератора noise3d. Допустимые значения: 0, 1, 2 и 3.

f_hunt_surface(x,y,z, P0)

- P0 : Field Strength (Необходима отрицательная величина или отрицательная функция.)

f_hyperbolic_torus(x,y,z, P0, P1, P2)

- P0 : Field Strength (Необходима отрицательная величина или отрицательная функция.)
- P1 : Больший радиус: зазор между центрами огибающей трубы и ближайшей точкой.
- P2 : Малый радиус: толщина огибающей трубы.

f_isect_ellipsoids(x,y,z, P0, P1, P2, P3). 'isect ellipsoids' – поверхность похожая на пересечение эллипсов ориентированных вдоль каждой оси.

- P0 : Eccentricity. В случае значения меньшего единицы эллипсы выглядят сплющенными, в случае значения более 1 растянутыми, в случае со значением равным нулю – эллипсоиды принимают форму сферы.
- P1 : Inverse size. Увеличьте для уменьшения размеров поверхности.
- P2 : Diameter. Увеличьте для увеличения размеров эллипсоид.
- P3 : Threshold. Инверсивный параметр к параметру “threshold”

f_kampyle_of_eudoxus(x,y,z, P0, P1, P2). 'kampyle of eudoxus' – две бесконечные плоскости с впадиной посередине.

- P0 : Field Strength (Необходима отрицательная величина или отрицательная функция.)
- P1 : Dimple: В случае нулевого значения выпуклости встречаются в цетре. ненулевое значение даёт в результате меньшую выпуклость.
- P2 : Closeness: Чем больше значение, тем ближе плоскости.

f_kampyle_of_eudoxus_2d(x,y,z, P0, P1, P2, P3, P4, P5)Двумерная кривая которая образует вращением предыдущую поверхность.

- P0 : Field Strength (Необходима отрицательная величина или отрицательная функция.)
- P1 : Dimple: В случае нулевого значения выпуклости встречаются в цетре. ненулевое значение даёт в результате меньшую выпуклость.
- P2 : Closeness: Чем больше значение, тем ближе плоскости.
- P3 : SOR Switch
- P4 : SOR Offset
- P5 : SOR Angle

f_klein_bottle(x,y,z, P0)

- P0 : Field Strength (Необходима отрицательная величина или отрицательная функция.)

f_kummer_surface_v1(x,y,z, P0). Набор светящихся стержней.

- P0 : Field Strength (Необходима отрицательная величина или отрицательная функция.)

f_kummer_surface_v2(x,y,z, P0, P1, P2, P3). Другая вариация функции kummer_surface. выглядит как светящиеся стержни при отрицательных значениях параметров. В случае с положительными значениями выглядит как суперэллипсоида.

- P0 : Field Strength (Необходима отрицательная величина или отрицательная функция.)
- P1 : Rod width (negative): При больших отрицательных значениях увеличивает диаметр стержней.
- P2 : Divergence (negative): Установка значения -1 приводит стержни к виду цилиндров. Большие отрицательные приводят стержни к утолщению. Малые отрицательные значения приводят к сужению стержней.
- P3 : Влияет на длину половины стержней. Изменение знака смещает влияние на другую половину стержня. Значение 0 не влияет на результат.

f_lemniscate_of_gerono(x,y,z, P0). Лемниската Герона. Имеет вид песочных часов.

- P0 : Field Strength (Необходима отрицательная величина или отрицательная функция.)

f_lemniscate_of_gerono_2d(x,y,z, P0, P1, P2, P3, P4, P5). Двумерная версия лемнискаты. Предназначена для образованя тел вращения или растяжения по осям.

- P0 : Field Strength (Необходима отрицательная величина или отрицательная функция.)
- P1 : Size: увеличение значения ведёт к увеличению кривой и уменьшению округлости.
- P2 : Width: увеличение значения ведёт к утолщению кривой
- P3 : SOR Switch
- P4 : SOR Offset
- P5 : SOR Angle

f_mesh1(x,y,z, P0, P1, P2, P3, P4) Общая толщина потоков контролируется параметром threshold изоповерхности. В случае нулевого значения данного параметра потоки имеют нулевую толщину и, следовательно, невидимы. Параметры P2 и P4 контролируют относительную толщину потоков.

- P0 : Расстояние между соседним потоком в направлении оси x.
- P1 : Расстояние между соседним потоком в направлении оси z.
- P2 : Относительная толщина в направлениях осей x и z
- P3 : Амплитуда волнового эффекта. В случае нулевого значения получается плоская сетка.
- P4 : Относительная толщина в направление оси y

f_mitre(x,y,z, P0). Выглядит как эллипс сжатый на концах плоскогубцами.

- P0 : Field Strength (Необходима отрицательная величина или отрицательная функция.)

f_nodal_cubic(x,y,z, P0). Нодальный Куб – вытяжение кривой Stophid2D вдоль оси X с последующим наклоном.

- P0 : Field Strength (Необходима отрицательная величина или отрицательная функция.)

f_noise3d(x,y,z)

f_noise_generator(x,y,z, P0)

- P0 : Номер генератора шума.

f_odd(x,y,z, P0)

- P0 : Field Strength (Необходима отрицательная величина или отрицательная функция.)

f_ovals_of_cassini(x,y,z, P0, P1, P2, P3). Обобщение тора.

- P0 : Field Strength (Необходима отрицательная величина или отрицательная функция.)
- P1 : Больший радиус – аналогично тору
- P2 : Наполнение. 0 – Тор. При больших значениях дыра в центре затягивается. При сверхвысоких результатом является эллипсоид с выпуклостями.
- P3 : Толщине. Линейно влияет на толщину тора.

f_paraboloid(x,y,z, P0). Поверхность вращения с образующей параболой.

- P0 : Field Strength (Необходима отрицательная величина или отрицательная функция.)

f_parabolic_torus(x,y,z, P0, P1, P2)

- P0 : Field Strength (Необходима отрицательная величина или отрицательная функция.)
- P1 : Больший радиус
- P2 : Малый радиус

f_ph(x,y,z) = atan2(sqrt(x*x + z*z), y) В случае одиночного использования функция "PH" строит поверхность всех точек на особой широты, в т.ч. конус..В случае значения threshold равным нулю даёт в результате конус нулевой толщины. См также f_th и f_r

f_pillow(x,y,z, P0)

- P0 : Field Strength

f_piriform(x,y,z, P0). Выглядит как половина леминискаты.

- P0 : Field Strength

f_piriform_2d(x,y,z, P0, P1, P2, P3, P4, P5, P6). Плоская версия предыдущей поверхности.

- P0 : Field Strength(Необходима отрицательная величина или отрицательная функция.)
- P1 : Size factor 1: Увеличение даёт большую кривую
- P2 : Size factor 2: малые отрицательные значения также увеличивают в размерах но делают более тонкими.
- P3 : Fatness: Утолщение кривой.
- P4 : SOR Switch
- P5 : SOR Offset
- P6 : SOR Angle

f_poly4(x,y,z, P0, P1, P2, P3, P4). Предназначена для получения поверхности вращения полинома степени до 4 включительно. Общий вид "x = A + By + Cy2 + Dy3 + Ey4". Вращение происходит вокруг оси Y.

- P0 : Коэффициент A
- P1 : Коэффициент B
- P2 : Коэффициент C
- P3 : Коэффициент D
- P4 : Коэффициент E

f_polytubes(x,y,z, P0, P1, P2, P3, P4, P5). Состоит из определённого числа труб. Каждая труба следует двумерной кривой определяемой полиномом до 4го порядка включительно. Общий вид полинома "x = A + By + Cy2 + Dy3 + Ey4". Выстраиваются вдоль оси Y. Функция требует положительного значения параметра threshold (толщина труб).

- P0 : Число труб
- P1 : Коэффициент A
- P2 : Коэффициент B
- P3 : Коэффициент C
- P4 : Коэффициент D
- P5 : Коэффициент E

f_quantum(x,y,z, P0). Схоже с областью электрона на d орбите.

- P0 : Не используется, но необходимо

f_quartic_paraboloid(x,y,z, P0). Параболоида, но с менее гладкой формой.

- P0 : Field Strength (Необходима отрицательная величина или отрицательная функция.)

f_quartic_saddle(x,y,z, P0). Седло, но с менее гладкой формой.

- P0 : Field Strength (Необходима отрицательная величина или отрицательная функция.)

f_quartic_cylinder(x,y,z, P0, P1, P2).

'Quartic cylinder' выглядит как цилиндр, в который помещено яйцо.

- P0 : Field Strength (требуется отрицательная интенсивность поля)
- P1 : Диаметр «яйца».
- P2 : Параметр, определяющий ширину цилиндра и вертикальный размер «яйца».

f_r(x,y,z) = sqrt(x*x + y*y + z*z)
Макрос, создающий поверхность, состоящую из точек, отстоящих на определённом расстоянии (величина threshold) от начала координат, то есть сферу. Смотрите также f_ph и f_th

f_ridge(x,y,z, P0, P1, P2, P3, P4, P5) – функция, модифицирующая другие поверхности подобно полю высот или пигментной финкции. Смотрите также f_hetero_mf и f_ridged_mf.

Список параметров:

- P0 : Лямбда
- P1 : Октавы
- P2 :Омега
- P3 : Сдвиг

- P4 : Выступ (гребень)
- P5 : Тип, генерирующий noise3d. 0, 1, 2 и 3 - допустимые значения.

f_ridged_mf(x,y,z, P0, P1, P2, P3, P4, P5) – «Заострённый мультифрактал», может использоваться для создания мультифрактальных полей высот и моделей. Название «Мультифрактал» происходит от свойства объекта,который обладает дробной размерностью, изменяющейся с высотой.

Поверхность получается в результате суммирования «шумов» большого количества частот.

Параметры функции f_ridged_mf задают, как много и какие частоты суммировать, а так же какой вклад различные частоты должны внести в результирующую сумму.

Преимущество f_ridged_mf перед height_field{}в том,что область, определяемая ridged_mf может быть произвольно увеличена вдоль осей x и y без потери разрешения. См. также функции

f_hetero_mf и f_ridge.

Список параметров:

- P0 : H –отрицательная степень базовой частоты шума, использующаяся при построении функции f_ridged_. Когда H равен единице, фрактализация относительно плавная. Когда H приближается к 0, высокие частоты становятся близки к низким.
- P1 : параметр для перехода от одной «октавы» к другой в процессе фрактализации.
- Этот параметр влияет на величину разрыва между частотами. (Рекомендуется использовать значение, большее 1.0).
- P2 : Октава – число различных частот, добавленных к фракталу. Каждая октавная чистота равна предыдущей, умноженной на “Лакунарность” – параметр P1. Таким образов, используюя большое количество октав, можно очень быстро получить высокие частоты.
- P3: Смещение применяется к фракталам, чья размерность меняется от высоты к высоте.

  Высокие частоты на низких высотах более амортизированы, чем на больших высотак, то есть более низкие уровни плавне и более гладкие, чем высокие.
- P5: Тип, генерирующий noise3d. 0, 1, 2 и 3 – допустимые значения.

f_rounded_box(x,y,z, P0, P1, P2, P3) – бокс с закругленными угламы. Определяется как куб <-1, -1, -1> to <1, 1, 1>. Изменяя величину параметров, задающих куб, размер можно регулировать, при этом не оказывая влияния на радиус кривизны углов.

Описание параметров:

- P0 : Радиус кривизны углов. Нулевой дает прямоугольные углы, 0.1, к примеру – углы с закруглением, соответствующем сфере sphere {0, 0.1}.
- P1 : Масштаб x .
- P2 : масштаб y.
- P3 : Масштаб z .

f_sphere(x,y,z, P0)- сфера.

- P0: радиус сферы.

f_spikes_2d(x,y,z, P0, P1, P2, P3) =2-D function : f = f(x, z) - y

- P0 : высота центрального острого выступа
- P1 : Частота выступов вдоль оси x
- P2 : Частота выступов вдоль оси y
- P3 : Величина, на которую уменьшается выступ по мере удаления от центра

f_spiral(x,y,z, P0, P1, P2, P3, P4, P5)

- P0 : Расстояние между витками спирали
- P1 : Толщина
- P2 : Внешний диаметр спирали. Поверхность выглядит так, как если бы она бала заключена в сфере этого диаметра
- P3 : Не используется
- P4 : не используется

f_steiners_roman(x,y,z, P0). Объект "Steiners Roman" есть композиция четырёх идентичных треугольных блока, которые в совокупности составляют округлённый тетраэдр. Вдоль осей X, Y и Z в местах, где соединяются блоки, есть сгибы.

P0 : Поле напряжённости.

f_strophoid(x,y,z, P0, P1, P2, P3). "Strophoid" подобен бесконечной плоскости с колбой, выступающей из неё.

- P0 : Поле напряжённости (требуется отрицательное поле или отрицательная функция)
- P1 : размер колбы. Чем больше значение парамтера, тем больше размер колбы. Отрицательная величина параметра сделает колбу с нижней стороны плоскости.
- P2 : Острота. Когда параметр равен нулю, колбы выглядит как сфера, лишь касающаяся плоскости.
- P3 : Плоскостность. Чем больше значение параметра, тем больше толщина колбы.

f_strophoid_2d(x,y,z, P0, P1, P2, P3, P4, P5, P6). Плоская кривая «strophoid» может быть вытянута вдоль оси Z или повернута вокруг любой из осей с помощью SOR-параметров.

- P0 : Поле напряжённости (требуется отрицательное поле или отрицательная функция)
- P1 : размер колбы. Чем больше значение парамтера, тем больше размер колбы. Отрицательная величина параметра сделает колбу с нижней стороны плоскости.

- P2 : Острота. Когда параметр равен нулю, колбы выглядит как сфера, лишь касающаяся плоскости.
- P3 : Плоскостность. Чем больше значение параметра, тем больше толщина колбы.

- P4 : SOR -перенос
- P5 : SOR -сдвиг
- P6 : SOR -угол

f_superellipsoid(x,y,z, P0, P1). Для построения этого объекта («Суперэллипсоид») необходимо отрицательное поле напряжённости или отрицательная функция.

- P0 : величина горизонтальной полуоси
- P1 : величина вертикальной полуоси

f_th(x,y,z) = atan2(x, z)
f_th() –функция, полезная только в сочетании с другими.
В любой точке создаёт величину, равную углу тета в радианах. Угол тета подобен координате долготы для Земли. Угол остаётся неизменным, если вы двигаетесь в вертикальном направлении, и меняется в противном случае.Смотрите так же f_ph и f_r .

f_torus(x,y,z, P0, P1)

- P0 : Внешний радиус
- P1 : Внутренний радиус

f_torus2(x,y,z, P0, P1, P2)

- P0 : Поле напряжений (Требуется либо отрицательное поле, либо отрицательная функция)
- P1 : Внешний радиус
- P2 : Внутренний радиус

f_umbrella(x,y,z, P0)

P0 : Поле напряжений (Требуется либо отрицательное поле, либо отрицательная функция)

3.7.7.3 Предопределённые функции

eval_pigment(Pigm, Vect)-макрос, вычисляющий цвет пигмента в любой точке. Некоторые пигменты требуют больше информации, чем просто координаты точки и не работают с этим макросом.

Параметры:

- Vect = точка, в которой нужно вычислить пигмент.
- **Pigm =** пигмент для вычисления.

f_snoise3d(x, y, z). То же самое, что и f_noise3d(), но возвращает значения в пределах [-1, 1].

f_sine_wave(val, amplitude, frequency). Превращает идущую под уклон форму волны в синусоидальную.

3.7.7.3.1 Шаблонные функции

Предопределённые шаблонные функции полезны для построения специальных шаблонов.

Некоторые из шаблонных функций не используются вообще, так как их задание требует специальных параметров, которые должны быть заданы в определении функции, или они требуют информацию, недоступную для них. По этой причине, Вы, возможно, захотите определить собственную модификацию этих функций.

Все шаблонные функции принимают три параметра – координаты XYZ точек для определения значения функции в них.

Примеры:

f_agate(x, y, z)

f_boxed(x, y, z)

f_bozo(x, y, z)

f_brick(x, y, z)

f_bumps(x, y, z)

f_checker(x, y, z)

f_crackle(x, y, z)

f_cylindrical(x, y, z)

f_dents(x, y, z)

f_gradientX(x, y, z)

f_gradientY(x, y, z)

f_gradientZ(x, y, z)

f_granite(x, y, z)

f_hexagon(x, y, z)

f_leopard(x, y, z)

f_mandel(x, y, z)

f_marble(x, y, z)

f_onion(x, y, z)

f_planar(x, y, z)

f_radial(x, y, z)

f_ripples(x, y, z)

f_spherical(x, y, z)

f_spiral1(x, y, z)

f_spiral2(x, y, z)

f_spotted(x, y, z)

f_waves(x, y, z)

f_wood(x, y, z)

f_wrinkles(x, y, z)