Название доклада:

Algorithms of Color Interpolation in Visualization of Results of Numerical Simulation

D. V. Mogilenskikh, I. V. Pavlov
Russian Federal Nuclear Center – the All-Russian Scientific and Research Institute of Technical Physics named after academician E. I. Zababakhin
(RFNC-VNIITF)

Алгоритмы цветовой интерполяции при визуализации результатов численного моделирования

Д. В. Могиленских, И. В. Павлов
Российский Федеральный Ядерный Центр - Всероссийский научно-исследовательский институт технической физики им. акад. Е. И. Забабахина
(РФЯЦ-ВНИИТФ)

Авторы:

Могиленских Дмитрий Владимирович
Начальник лаборатории научной визуализации
Российский Федеральный Ядерный Центр-ВНИИТФ им. акад. Е. И. Забабахина.
Палов Игорь Васильевич
Инженер-программист
Российский Федеральный Ядерный Центр-ВНИИТФ им. акад. Е. И. Забабахина.

Dmitry V. Mogilenskikh
Head of visualization laboratory
Russian Federal Nuclear Center - Institute of Technical Physics
Igor V. Pavlov
Engineer-programmer
Russian Federal Nuclear Center - Institute of Technical Physics

Категория доклада:

Обычный доклад с параллельным представлением на компьютере.

Секция: Визуализация

Информация для контактов:

Могиленских Дмитрий Владимирович
E-mail: d.v.mogilenskikh@vniitf.ru
456770, Снежинск, Челябинской обл.
А.Я 245
Факс: (35172) 3-29-19
Тел.: (35172) 2-22-22 доп. 5-61-56

Павлов Игорь Васильевич
E-mail: i.v.pavlov@vniitf.ru
456770, Снежинск, Челябинской обл.
А.Я 245
Факс: (35172) 3-29-19
Тел.: (35172) 2-22-22 доп. 5-61-56

Dmitry V. Mogilenskikh
Head of visualization laboratory
456770, Snezhinsk, Chelyabinsk Region, Russia
P.O. Box 245
E-mail: d.v.mogilenskikh@vniitf.ru
FAX: (35172) 3-29-19
TEL.: (35172) 2-22-22 (+) 5-61-56
Igor V. Pavlov
Engineer-programmer
456770, Snezhinsk, Chelyabinsk Region, Russia
P.O. Box 245
E-mail: i.v.pavlov@vniitf.ru
FAX: (35172) 3-29-19
TEL.: (35172) 2-22-22 (+) 5-61-56

Тезисы:

Techniques and algorithms of smooth color changing generation inside a polygon are typically applied in visualization of 3D objects for generating the photorealistic effect of the objects' smooth surface with lightening considered. Techniques of Gourot and Phong belong to such techniques. The similar algorithms of color interpolation are hardly applied in scientific visualization applications for presenting the physical content of mathematical models, and in scientific 2D-visualization, only partly. Processes of interpolation or smooth color changing inside polygons in visualization of scientific and experimental data are called "graphical replenishment" (GR) of the initial data for better visualization. The strict mathematical term "interpolation" corresponds to the work sense more precisely, but it is fundamental and strict.

The basic aim: the application of GR algorithms in visualization of results of numerical simulation of physical processes with difference and finite-element techniques for enhancing information density of graphical interpretation of these processes.

The problem on replenishment of discretely specified functions of two variables is well studied and is described in the scientific literature. In this paper the emphasis shifts towards creating the adequate and fast algorithms of color interpretation of the function change in the intermediate cell points in the visualization.

The process of analyzing and comparing the results together with the discrete color cell image requires to obtain the picture of smooth changing of a characteristics inside the cells as well. In this way of visualization of a model, of physical content, we can distinguish the general trends of characteristics changes, vortexes, turbulence, fine state transformations, specific objects, shock waves, discontinuity lines, interfaces, isolines and stream-lines.

GR is a good interpretation instrument with its joint application together with other visualization means, for example, constructing of isolines and stream-lines, grid application and others.

The material presentation is accompanied by additional illustrated sources.

The approaches and algorithms stated within this study were investigated and implemented within the frameworks of RFNC-VNIITF works on creating the system of scientific visualization of the results of numerical simulation and VIZI2D experiments.

Алгоритмы цветовой интерполяции при визуализации результатов численного моделирования

Д. В. Могиленских, И. В. Павлов
Российский Федеральный Ядерный Центр - Всероссийский научно-исследовательский институт технической физики им. акад. Е. И. Забабахина
(РФЯЦ-ВНИИТФ)

Введение

Методы и алгоритмы генерации плавного изменения цвета внутри многоугольника (полигона) применяются обычно при визуализации геометрии трехмерных объектов для формирования фотореалистичного эффекта гладкой поверхности объектов с учетом освещенности. К таким методам относятся методы Гуро, Фонга [1-4] и алгоритмы, предложенные в работе [5]. Подобные алгоритмы интерполяции цветов (освещенности) внутри многоугольников почти не применяются в приложениях научной визуализации для представления физического содержания математических моделей. В частности в научной 2D-визуализации, хотя такой способ визуализации является очень информативным для анализа моделей. В данной работе представлены алгоритмы и подходы, которые отчасти можно считать модификациями известных алгоритмов, а также некоторые иные алгоритмы интерполяции. Процесс интерполяции или плавного изменения цветов внутри полигонов при визуализации научных и экспериментальных данных назовем как “графическое восполнение” (ГВ) [6] исходных данных, для их адекватной визуализации. Строгий математический термин “интерполяция” более точно отвечает смыслу работы, но он является фундаментальным и строгим, в смысле оценки погрешности [3, 6-8]. В данной работе применяется специальное название “графическое восполнение”, чтобы отразить прикладную область и подчеркнуть, что не только точные оценки важны при визуализации, но и быстрое приближенное получения адекватного результата, который сопоставим по точности с классическими методами интерполяции, а также визуальное восприятие.

Основная цель: применение алгоритмов ГВ при визуализации результатов численного моделирования физических процессов разностными и конечно-элементными методами для повышения информативности графической интерпретации этих процессов.

Основным объектом разностных и конечно-элементных методов является сетка с вектором физических характеристик, заданных на ней. Составными элементами сетки являются ячейки и узлы. Иначе говоря, результатом численного моделирования данными методами являются сеточные скалярные и векторные поля, заданные на множестве ячеек или в узлах сетки. Представленные ниже алгоритмы ГВ можно успешно применять как для двумерных, так и для трехмерных сеток, а также для сеток разной структуры. В контексте данной работы геометрический объект многоугольник заменяется объектом ячейка сетки.

1. Постановка задачи

1.1 Предварительные сведения

Задача восполнения дискретно заданных функций двух переменных хорошо изучена и описана в научной литературе, как по численным методам, так и в литературе по машинной графике[3, 6-8]. Однако, в этих работах акцент делается на трехмерное представление графиков функций или наиболее точное нахождение промежуточных значений функций. В данной работе акцент смещается в сторону создания адекватных и быстрых алгоритмов цветной интерпретации изменения функции в промежуточных точках ячейки при ее визуализации.

Применение классических сплайнов, Безье кривых, рядов Фурье, полиномов Чебышева или метода наименьших квадратов и т. д. возможно, но они требуют больших вычислительных затрат. Для цветового представления результатов численного моделирования эти затраты не оправданы, не смотря на то, что точность приближения хорошо оценивается. Размерность современных решаемых задач в смысле размерности сеток, например, в механике сплошной среды, огромны, что приводит к необходимости решать очень большие системы уравнений для применения классических подходов.

Более оправдано применение локальных сплайнов или B-сплайнов, однако, здесь тоже есть необходимость решать большое количество небольших систем уравнений. В данной работе этот подход не рассматривается, но при необходимости его легко может реализовать заинтересованный исследователь.

Описание алгоритмов будем проводить на примере моделей, которые заданы двумерными регулярными разностными сетками. Данное предположение не ограничивает общность алгоритмов для других структур данных. Регулярные разностные сетки - наиболее часто встречающийся метод дискретизации сплошной среды, и на примере таких сеток удобнее вести изложение материала.

Замечание 1: Краткое описание численных моделей, которые представлены в виде иллюстраций будет представлено ниже в разделе “Оценки и выводы”.

Замечание 2: Часть иллюстративного материала вынесено в приложение для более удобного сравнительного анализа представленных алгоритмов. Нумерация в приложении будет отдельная и выделена звездочкой после номера рисунка. Например, Рисунок 1*.

Геометрию исходных данных, т. е. регулярную двумерную сетку, можно представить двумерными массивами X[M, N], Y[M, N], где M, N - размерность сетки. Будем предполагать, что в каждом узле заданы некоторые физические характеристики. Обозначим одну из физических характеристик через U[M, N]. Для всех алгоритмов ГВ необходима предобработка массива U. Она состоит в нахождении минимума и максимума величины U по всей сетке. Пусть U_{max} и U_{min} - соответственно максимум и минимум значений U по всей сетке.

1.2 Цель ГВ

Существует необходимость в процессе анализа и сравнения результатов, на ряду, с дискретным цветным ячеечным изображением (Рис. 1) получать картину плавного изменения характеристики и внутри ячеек. (Рис.2) При этом способе визуализации физического содержания модели выделяются общие тенденции изменения характеристик, вихри, турбулентность, тонкие переходы состояний, особые объекты, например, ударные волны, линии разрыва, контактные границы, изолинии и линии тока. ГВ является хорошим инструментом интерпретации при его совместном применении с другими средствами визуализации, например,

построение изолиний и линий тока, наложение сетки и т. д. Рис.3

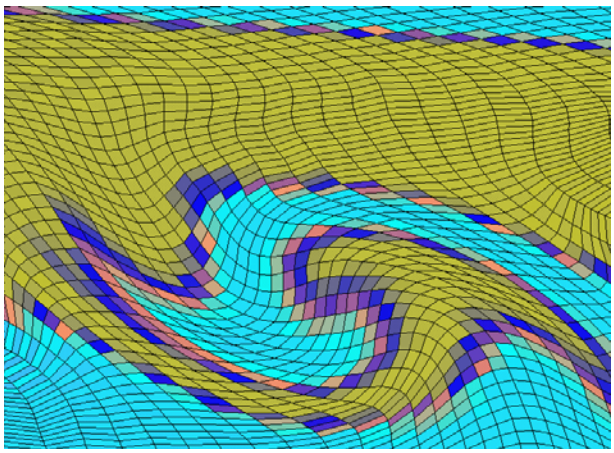

Рис. 1

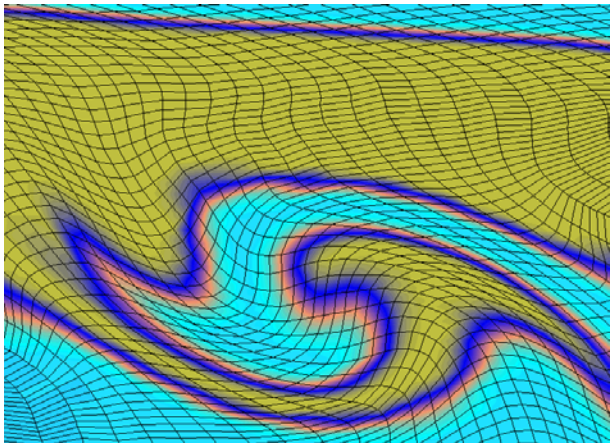

Рис. 2

На рисунках 1 и 2 представлен один и тот же фрагмент численной модели с разностной сеткой. На рисунке 1 ячеечный способ закраски, т. е. ячейки закрашиваются одним цветом без восполнения внутри. На рисунке 2 применен алгоритм 4 ГВ.

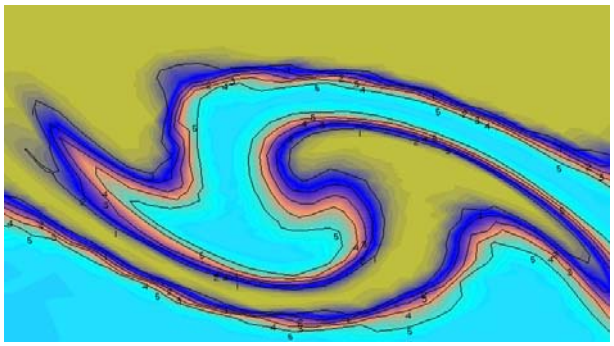

Рис. 3. Этот рисунок аналогичен рисунку 2, только без изображения сетки, но с визуализацией изолиний. Хорошо видно согласование формы изолиний и алгоритма 4 ГВ.

Задача заключается в том, чтобы получить хорошие, адекватные алгоритмы ГВ для цветового представления изменения физических величин внутри ячеек. Можно предложить несколько способов того, как оценить правильность восполнения или сравнить два алгоритма восполнения:

- **Обоснование** самого алгоритма, т. е. насколько полно учитывается исходная информация и с какой точностью интерполируется внутри ячейки.
- Визуальная оценка корректности изображения, применяя **сравнение с закраской по ячейкам**.
- **Наложение** на изображение с восполненными цветами сетки **других объектов**, которые стоятся на базе тех же данных, например, изолинии, линии тока и т. д. Рис. 3.
- Применение численных методов сравнения, вычисление **отклонения от "эталона"**, например, производя операцию вычитания одного изображения из другого и анализируя разность изображений по некоторому критерию или вычисление **меры информативности**[9].

Описание методов оценки качества ГВ выходит за рамки данной работы.

2. Варианты цветового решения задачи восполнения

2.1 Палитра

Наиболее распространенный метод цветовой интерпретации физического содержания моделей – это линейный закон соответствия (ЛЗС) между физической характеристикой в ячейках или узлах сетки и цветовым пространством. Применяются и другие нелинейные законы соответствия [10]. В данной работе для простоты изложения будем использовать ЛЗС.

Одно из центральных понятий в этой теме - понятие палитры цветов.

Определение 1: Палитра цветов - упорядоченный набор цветов из пространства RGB, который представляется одномерным массивом, где каждому цвету поставлен в соответствие целочисленный индекс, т. е. его порядковый номер в палитре.

Пусть дана палитра из (NC+1)-цветов. За каждым цветом из палитры закреплен некоторый порядковый номер. Из этих цветов можно составить различные палитры, меняя количество цветов и их порядок в одномерном массиве. Для изложения будем использовать палитру из (MC+1)-цветов, а индексы этих цветов будут представлены в некотором порядке в одномерном массиве PAL[MC], где $MC \in [0, NC]$. Данное представление называется индексным представлением цветов, т. е. каждому индексу соответствует цвет из цветового пространства RGB. Производные красный, зеленый, синий цвета называются аддитивной системой смешивания основных цветов RGB и это не единственный базис для представления цветового пространства. Применяется, например, субтрактивная система смешивания цветов – голубой, пурпурный и желтый цвета (CMY). Как правило, палитра создается плавными переходами от одного цвета к другому и каждый пользователь для своих целей может подобрать нужную палитру. Рис.4.

Согласно ЛЗС цвет для ячеек сетки вычисляется так [10]:

Диапазон изменение величины U: $IN_u = |U_{max}, U_{min}|$.

Диапазон дискретного изменение цветов в палитре: $IN_c = | 0, MC |$.

Вычислим коэффициент пропорциональности k длины отрезка IN_p к длине IN_c:

$$k = (U_{max} - U_{min}) / MC .$$

k - длина отрезка деления IN_p с таким свойством, что ячейки со значениями U из одного отрезка деления будут окрашены одним цветом. Возьмем текущую ячейку сетки со значением величины $U_{ij} = U_{ij} [i, j]$, где $i \in [0, M]$, $j \in [0, N]$ и вычислим для нее индекс цвета из палитры:

$$I_{i,j} = (int)((U_{i,j} / k) + 0.5), \quad \mathbf{(1)}$$

где (int) – выделение целой части.

В результате для каждой ячейки или узла сетки получаем определенный цвет из палитры, и каждая ячейка закрашивается этим цветом. В итоге получается дискретное ячеечное изображение сеточной модели в виде

многоугольников, которое отражает распределение физической характеристики U в ячейках сетки. Рис. 1.

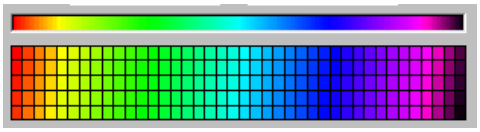

Рис. 4. Два варианте представления палитры.

Необходимо дать следующие определения и объяснения.

Определение 2: Палитра PAL называется **Линейной**, если в пространстве RGB все множество цветов палитры {PAL[0],.., PAL[MC]} принадлежит отрезку одной прямой и упорядочено относительно начала данного отрезка. Иначе палитра PAL называется **Нелинейной**, т. е. множество цветов {PAL[0],.., PAL[MC]} образует криволинейную траекторию в пространстве RGB. Рис.5.

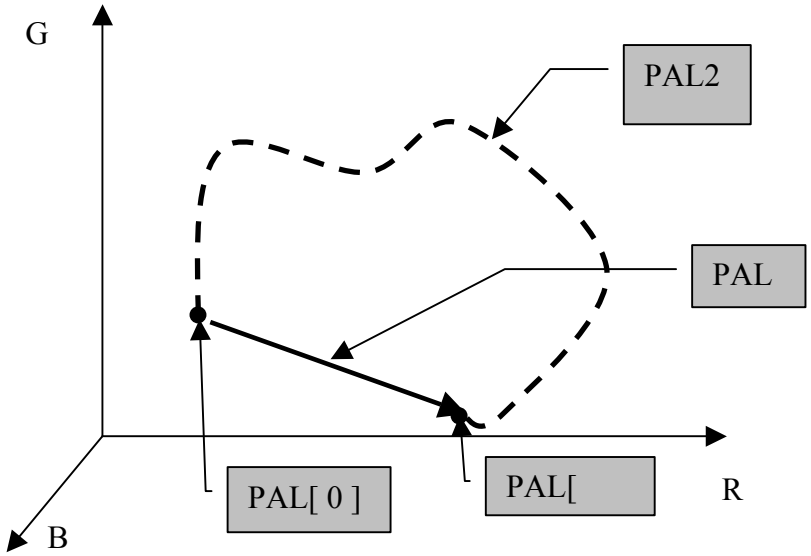

Рис. 5. Представлено цветовое пространство RGB. Схематично изображены примеры линейной палитры PAL1 и нелинейной палитры PAL2. У обоих палитр совпадают начальный и конечный цвета, т. е. PAL[0] и PAL[MC].

2.2 Восполнение в палитре.

Суть данного способа цветовой визуализации восполнения заключается в использовании только цветов из заданной палитры, за каждым цветом которой закреплен диапазон изменения величины U. В этом случае интерполяция происходит на уровне номеров для цветов в палитре. Этот способ удобно тем, что при любом законе соответствия цвета значению физической характеристики, ГВ дает гарантию получения цвета строго из этой палитры. Рис. 1*.

2.3 Восполнение в цветовом пространстве RGB.

Есть другой метод интерполяции между цветами палитры. Можно применить напрямую пространства RGB, т. е. использовать для восполнения цвета не из палитры.

Замечание 3: Если палитра линейная и достаточно представительна по количеству цветов, то результат будет совпадать с восполнением в палитре.

Рассмотрим случай нелинейной палитры. Казалось бы, что надо ожидать более плавные переходы цветов на изображении. Однако, результат практической реализации оказался отрицательным. Получается плохое согласование и не адекватность соответствия цветов и физического содержания. На рисунках 1* и 2* представлены результаты работы алгоритма 4. Для рисунка 1* применялось восполнение по палитре, а для рисунка 2* по пространству RGB. На рисунке 3* представлен вариант восполнения тех же данных при помощи OpenGL. Рисунок 4* – палитра при реализации этих примеров.

Реализация ГВ в RGB:

Шаг 1. По значению U в вершине ячейки вычислялся номер цвета в палитре.

Шаг 2. Полученный цвет разлагался на компоненты RGB.

Шаг 3. Восполнение осуществлялось отдельно по компонентам. Обычно применяются следующие интервалы изменения компонент:

- От 0 до 1 при вещественном представлении (OpenGL);
- От 0 до 255 при целочисленном представлении.

При этом подходе с RGB появляются цвета не из палитры. Этот факт схематично можно проиллюстрировать на нелинейной палитре на рисунке 5. PAL[0] –первой цвет палитры, PAL[MC] – последний цвет. Кривая PAL2 представляет траектории палитры в пространстве RGB, т. е. проходит через промежуточные точки палитры между цветами PAL[0] и PAL[MC]. Отрезок PAL1 представляет кратчайшее расстояние между цветами PAL[0] и PAL[MC] в RGB и проходит через другие цвета пространства RGB.

Недостатки данного подхода:

1. Нарушается корректность представления данной информации в связи с появлением цветов не из палитры. В итоге изображение становится менее информативным.
2. Появляются “сорные” эффекты наложения цветов, псевдопрозрачность, ореолы. Рис. 2*.
3. Большее количество вычислений, т. к. интерполировать приходиться три компоненты, а не одну как при восполнении в палитре.

На рисунках 5* и 6* представлена другая численная модель и алгоритм восполнения 4. Рисунок 5* - восполнение в палитре. Рисунок 6* - восполнение в RGB.

3. Применение для восполнения OpenGL.

Существует возможность реализации ГВ, используя функции стандартной графической библиотеки OpenGL.

Определение 3: Графическая библиотека OpenGL является графическим стандартом, который разработан и утвержден в 1992 году. Он признан ведущими разработчиками программного обеспечения и производителями графических ускорителей.

На рисунках 8*, 9*, 10* приводятся примеры ГВ. Очевидно, что результат не удовлетворительный. Несмотря на быструю скорость реализации восполнения в OpenGL, такой результат для научной визуализации строгих математических данных является, мягко говоря, не очень приемлемым. Алгоритм восполнения в OpenGL не очень понятен.

Рассмотрим три способа реализации восполнения для четырехугольных ячеек.

1. Задать режим рисования всей четырехугольной ячейки, задавая цвет для каждой вершины. Рис. 8*.

   В терминах OpenGL, например:

```
glBegin( GL_QUADS );
   glColor3f( red[c0], green[c0], blue[c0] );
   glVertex3f( x0, y0, z0 );
   glColor3f( red[c1], green[c1], blue[c1] );
   glVertex3f( x1, y1, z1 );
   glColor3f( red[c2], green[c2], blue[c2] );
   glVertex3f( x2, y2, z2 );
   glColor3f( red[c3], green[c3], blue[c3] );
   glVertex3f( x3, y3, z3 );
glEnd();
```

   Заметим, координаты Z являются значениями величины U. c0, c1, c2, c3 – соответствующие номера цветов в палитре, которые отвечают согласно закону соответствия (ЗС) некоторым значениям величины U. red[], green[], blue[] – соответствующие компоненты разложения цвета в RGB. Согласно рисунку 7:

   x0 = x(0, 0), x1 = x(0, 1), x2 = x(1, 1), x3 = x(1, 0),……,z3(1, 0).

2. Вариант деления ячейки диагоналями на два треугольника. Есть два способа разбиения. Первый: Δ(x0, x1, x2) и Δ(x0, x2, x3). Второй: Δ(x0, x1, x3) и Δ(x1, x2, x3). Практическая реализация показала, что первый вариант абсолютно идентичен первому способу: glBegin(GL_QUADS). OpenGL работает именно так. Второй способ дал немного другой результат, но он принципиально по качеству не отличается от первого способа. Рис. 9*.

   Второй вариант в терминах OpenGL, например:

```
glBegin( GL_TRIANGLES );
   glColor3f( red[c0], green[c0], blue[c0] );
   glVertex3f( x0, y0, z0 );
   glColor3f( red[c1], green[c1], blue[c1] );
   glVertex3f( x1, y1, z1 );
   glColor3f( red[c3], green[c3], blue[c3] );
   glVertex3f( x3, y3, z3 );

   glColor3f( red[c1], green[c1], blue[c1] );
   glVertex3f( x1, y1, z1 );
   glColor3f( red[c2], green[c2], blue[c2] );
   glVertex3f( x2, y2, z2 );
   glColor3f( red[c3], green[c3], blue[c3] );
   glVertex3f( x3, y3, z3 );
glEnd();
```

   Результат данной реализации отрицательный.

3. Вариант деления ячейки на четыре треугольника. Рис. 10*. Можно найти геометрический центр масс ячейки (x_v, y_v, z_v) и соединить все вершины с ним. Например так:

```
x_v = ( x0+x1+x2+x3)/4.0;
y_v = ( y0+y1+y2+y3)/4.0;
z_v = ( z0+z1+z2+z3)/4.0;
```

   Результат немного лучше, чем на рисунках 8*, 9*, но не значительно. Реализация аналогична второму способу.

Можно представить U как поверхность в R^3 и применить освещение. Однако, этот способ беден в цветовом плане и не удобен для постановки строго соответствия между цветом и значением U.

На рисунке 5* представлен результат работы алгоритма 4, который изложен ниже для той же модели. Результат значительно лучше, не смотря на необходимость временных затрат, такое изображение значительно точнее и корректней интерпретирует в цвете численную модель.

Практика показала, что OpenGL дает неплохие результаты на сетках с плавным изменением величины U, или же на крупных сетках, или при большом увеличении. Рис. 6.

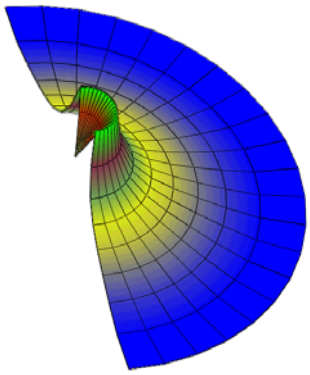

Рис. 6. Пример хорошего результата восполнения с OpenGL на крупной сетке с плавным изменением величины.

4. Алгоритмы генерации растровой развертки ячейки.

4.1 Общая часть

Областью графического восполнения является растровый экран, который состоит из двумерного множества дискретных точек – пикселов. Для всех графических объектов, которые изображаются на экране, необходим процесс генерации растровой развертки или разложение в растр, т. е. все объекты аппроксимируются множеством пикселов. Обычно методы генерации растровой развертки скрыты от пользователя за графическими функциями, но есть задачи, для которых этот процесс надо сделать явно. Ниже мы рассмотрим методы разложения в растр счетной ячейки сетки для алгоритмов ГВ.

Описание алгоритмов будем рассматривать на одной текущей ячейки сетки. Для других ячеек алгоритмы работают аналогично. Рассмотрим текущую ячейку сетки с индексами узлов в сетке: (i, j), (i + 1, j), (i + 1, j + 1), (i, j + 1).

Не ограничивая общность, допускать, что i = 0 и j = 0, т. е. получим номера для узлов ячейки: (0, 0), (1, 0), (1, 1), (0, 1). Рис. 7. В узлах ячейки задана некоторая физическая характеристика U.

Соответственно обозначим: U_{00}, U_{10}, U_{11}, U_{01}.
Обозначим координаты узлов в мировой системе координат и в экранной системе координат соответственно: (x1, y1), (x2, y2), (x3, y3), (x4, y4) и (xv1, yv1), (xv2, yv2), (xv3, yv3), (xv4, yv4).

Опишем по шагам общую схему генерации растровой развертки ячейки.
Шаг 1. Определение габаритов прямоугольника, который описывает ячейку в экранной системе координат. Рис. 7. Соответственно обозначим габариты: X_{min}, X_{max}, Y_{min}, Y_{max}.
Шаг 2. Локализация всех пикселов, которые составляют ячейку.
Шаг 3. Восполнение значений физической характеристики U во внутренних пикселах ячейки методами ГВ, исходя из значений U в узлах ячейки.
Шаг 4. Определение номера цвета из палитры или цвет из пространства RGB по любому закону соответствия между цветом и характеристикой U.
Процесс генерации растровой развертки тесно связан с алгоритмами ГВ.

Далее для каждой внутренней точки или внутреннего пиксела строки, линейной интерполяцией, можно вычислить значение величины U, или номер цвета I из палитры, или компоненты RGB. В зависимости от этого можно разными способами восполнять характеристику U внутри ячейки. Сами алгоритмы графического восполнения описаны ниже в разделе 5.

Применение алгоритма Брезенхейма дает очень хорошее скоростные показатели при генерации растровой развертки.

4.3 Применение аналитических алгоритмов сканирования по строкам или столбцам

Подобные алгоритмы сканирования при 3D-визуализации описаны в работах [2, 4]. На рисунке 7 схематично показан пример направления сканирования по столбцу I или по строке J. Сканирование можно проводить как по строкам, так и по столбцам.

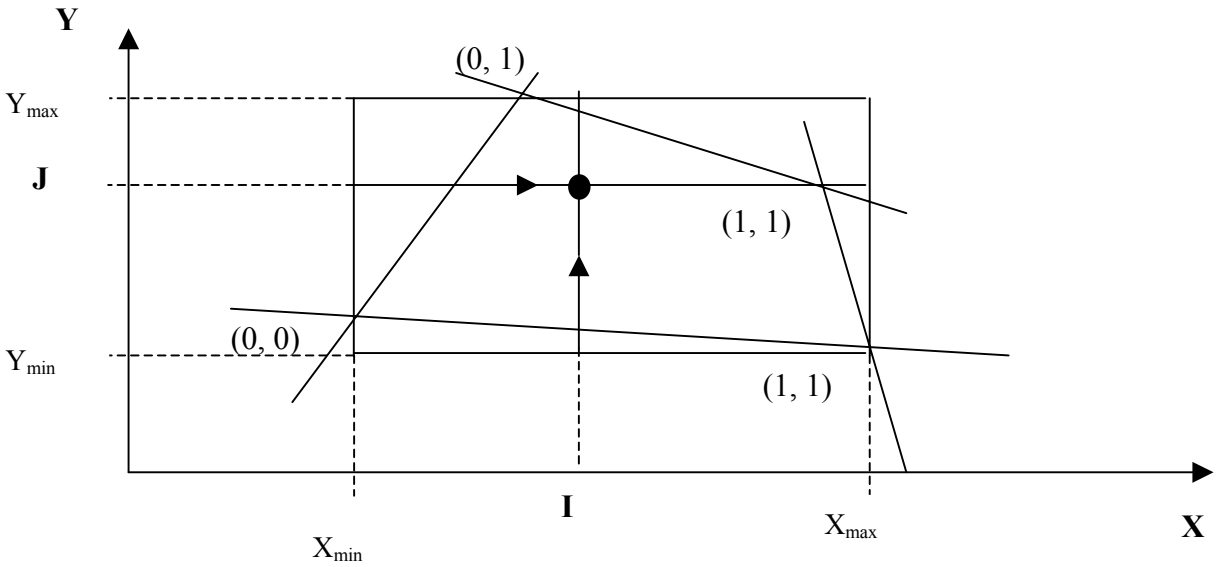

Рис. 7. Изображение текущей ячейки сетки, ограничивающего прямоугольника ячейки и схематичных направлений сканирования по строкам и столбцам.

4.2 Алгоритм 1. Применение Алгоритма Брезенхейма для растровой развертки отрезка прямой

Для получения растровой развертки ячейки можно применить классический целочисленный алгоритм растровой развертки отрезка прямой Брезенхейма [2]. Опишем его применение по шагам.

1. Вычисление растровой развертки ребер ячейки в целочисленных экранных координатах по алгоритму Брезенхейма.
2. Точки (пикселы) растровой развертки ребер сортируются, например, по вертикали, т. е. по координате Y.
3. Предварительно выбросить из списка точек растровой развертки ребер соседние точки с одинаковыми координатами Y и разными координатами X, оставляя самую левую и правую точку развертки.
4. Поиск пар, которые определяют начало и конец внутреннего сканирующего отрезка для ячейки.
5. В итоге получили растровую упорядоченную развертку границы ячейки.

Алгоритм 2. Локальное сканирование.

Один из самых очевидных способов определения растровой развертки или набора пикселов, которые определяют границы и внутренность ячейки – это по строчное сканирование габаритного прямоугольника. Описание по шагам:

1. Поиск точек пересечения текущей строки сканирования с ребрами ячейки в вещественном виде, применяя аналитические уравнения строки сканирования и прямой, которой принадлежит ребро.
2. Определение самой левой и самой правой точки пересечения сканирующей строки с ребрами ячейки.
3. Вычисление экранных координат точек пересечения, используя преобразования к экранным координатам.
4. В итоге получим растровую упорядоченную развертку границы ячейки.

Сейчас можно выполнить сканирование по пикселам в горизонтальном направлении.

Алгоритм 3. Глобальное сканирование.

Возможно, производить сканирование аналогичное трехмерному сканированию при 3D-визуализации, т. е. сканируется сразу вся строка экрана. Далее аналитически находятся пересечения с границами ячеек. Затем локализуются внутренние точки ячеек.

Результат. Растровую развертку текущей ячейки можно генерировать любым из Алгоритмов 1-3 с разной степенью эффективности и с разной точностью.
Обобщение для описания алгоритмов ГВ из раздела 5: В результате работы алгоритмов 1-3 локализуется внутренняя точка (пиксел) А (Рис.8), которая принадлежит текущей ячейке с экранными координатами (Xa, Ya). Рис. 8. Описание алгоритмов ГВ ниже будем проводить, относительно точки А. Для других пикселов внутри текущей ячейки алгоритмы ГВ работают аналогично.

Замечание 4:
Вычисление факта принадлежности пиксела текущей ячейке, рекомендуется осуществлять в мировой системе координат в вещественном представлении, для сохранения точности вычислений.

4.4 Особенности применения алгоритмов растровой развертки

4.4.1 Преимущества алгоритма Брезенхейма

Практическая реализация разных алгоритмов растровой развертки ячеек показала, что в среднем на любых типах сеток, особенно при увеличении их размерности применение алгоритма Брезенхейма для ребер дает значительный выигрыш в скорости визуализации. Такой подход к генерации растровой развертки показал очень хорошие результаты при визуализации криволинейных сеток.

4.4.2 Прямоугольные сетки

Для моделей, которые описаны прямоугольными регулярными или адаптивными сетками лучше применять глобальное сканирование по строке. Заранее нужно определить описывающий глобальный прямоугольник. Сканирование нужно проводить по-пиксельное, т. к. при прямоугольных ячейках просто и быстро определить принадлежность точки ячейке.

4.4.3 Эйлеровы сетки

Особенность Эйлеровых сеток, заключается в том, что их геометрия не изменяется при расчетах, а изменяются ее физическое содержание. Поэтому растровая развертка ячеек не меняется с течением времени счета задачи и ее можно для модели вычислить один раз. Зачастую, при эйлеровом подходе к моделированию, сетки являются прямоугольными, поэтому можно применять глобальное сканирование. Этот подход полезно и рационально применять для динамической визуализации одного расчета, т. е. при визуализации последовательных по времени состояний системы.

4.4.4 Лагранжевы сетки

Особенность данного подхода к численному моделированию заключается в том, что геометрия сетки изменяется постоянно при счете шага по времени, поэтому нужно на каждом шаге счета генерировать растровую развертку ячеек. По этой причине рекомендуется применять **Алгоритм 3**, на каждом временном шаге для увеличения скорости генерации развертки.

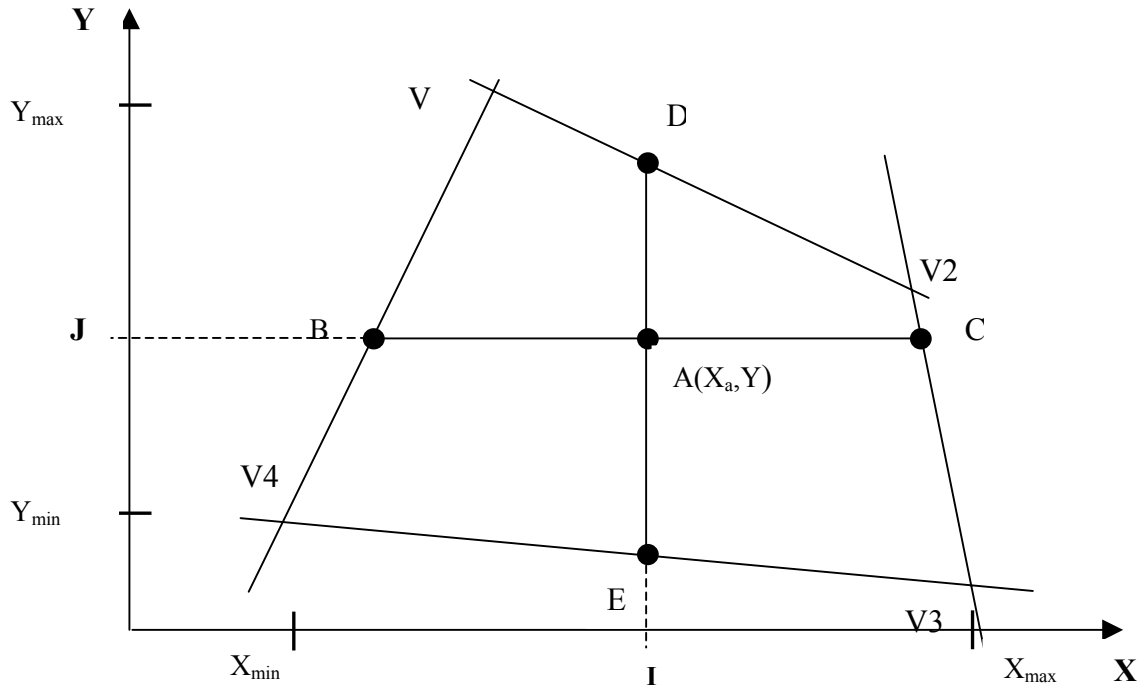

Рис. 8. Изображение текущей ячейки с локализованной внутренней точкой А и со сканирующими отрезками прямых |BC| и |DE|.

5. Алгоритмы “Графического восполнения”

5.1 Алгоритм 4. Билинейная интерполяция

Алгоритмы восполнения физических величин и цвета внутри ячеек, не ограничивая общности, будем излагать в предположении, что внутренние пиксели уже локализованы с помощью алгоритмов 1, 2 или 3.

Алгоритмы 4 и 5 можно считать двумерными модификациями методов Гуро и Фонга применительно к дискретной скалярной функции двух переменных, заданной на сетке для двумерной научной визуализации.

Шаг первой интерполяции: Для текущей сканирующей строки J найдем точки пересечения с ребрами ячейки. Обозначим их: B, C. Рис. 8. Теперь в эти точки необходимо интерполировать значения характеристики U из вершин. Воспользуемся линейной интерполяцией.

$U_B = (1 – t) * U_{v1} + t * U_{v4}$;

$U_c = (1 - t^*) * U_{v2} + t^* * U_{v3}$, где U_B, U_c, U_{v1}, U_{v2}, U_{v3}, U_{v4} значения характеристики U в соответствующих точках и параметры t и t*: $0 \le t \le 1$, $0 \le t^* \le 1$; $t = |V1\ B|/|V1\ V4|$, $t^* = |V2\ C| / |V2\ V3|$.

Шаг второй интерполяции: Сейчас необходимо интерполировать значение U в пиксель или точку A из точек B, C. Аналогично шагу выше:

$U_A = (1 – p) * U_b + p * U_c$, где $0 \le p \le 1$ и $p = |A\ B| / |B\ C|$. (*)

Шаг вычисления цвета:

Цвет можно вычислить по любому закону соответствия между физической характеристикой и палитрой цветов, например, по формуле **(1)** для ЛЗС или по формулам нелинейного соответствия из работы [10].

Примером работы данного алгоритма является рисунок 5*. На рисунках 11* и 12* выделен фрагмент модели, где представлено различие в восполнении при сканировании по столбцам рис. 11* и по строкам рис. 12*.

Однако, обычно изображения общей модели мало отличаются от типа сканирования.

Замечание 5: Шаг вычисления цвета можно сделать гораздо раньше, например, сразу для узлов ячейки вычислить цвет, и далее вычисления вести относительно индексов цвета из палитры, т.е. применять целочисленные вычисления. Например, при генерации растровой развертки ребер ячейки алгоритмом Брезенхейма для каждой точки развертки вычислять уже не величину U, а номер цвета в палитре. Такой подход применяется в методе Гуро, когда интенсивность освещения вычисляется для узлов многоугольника и далее интерполируется уже интенсивность. Однако, в целях сохранения максимально возможной точности, рекомендуется переходить к цвету как можно позже.

Замечание 6: Ранний переход к целочисленным вычислениям приводит к увеличению скорости визуализации.

На рисунке 9а представлен эффект раннего перехода к целочисленным вычислениям для характеристики U, и как следствие потеря качества интерполяции. На рисунке 9б представлен тот же пример, но интерполяция U осуществлялась в вещественном виде, а переход к индексу цвета был сделан только один раз при округлении результирующего значения физической величины U во внутреннем пикселе. Гладкость линий перехода от цвета к цвету показывает точность восполнения.

5.2 Алгоритм 5. Двойная билинейная интерполяция

Этот алгоритм является модификацией алгоритма 4 и является более времяемким. Идея заключается в том, чтобы одновременно применять интерполяцию по строке и столбцу. Допустим, получили восполненное значение U_A по алгоритму 4. Далее проводим аналогичную интерполяцию по столбцу I. Рис. 8.

Шаг первой интерполяции по столбцу: Для текущего сканирующего столбца I найдем точки пересечения с ребрами ячейки. Обозначим их: D, E. Рис.8. Теперь в эти точки необходимо интерполировать значения характеристики U из вершин. Воспользуемся линейной интерполяцией.

$U_D = (1 – t) * U_{v1} + t * U_{v2}$;

$U_E = (1 - t^*) * U_{v3} + t^* * U_{v4}$, где U_D, U_E, U_{v1}, U_{v2}, U_{v3}, U_{v4} значения характеристики U в соответствующих точках и параметры t и t*: $0 \le t \le 1$, $0 \le t^* \le 1$; $t = |V1\ D| / |V1\ V2|$, $t^* = |V3\ E|/|V3\ V4|$.

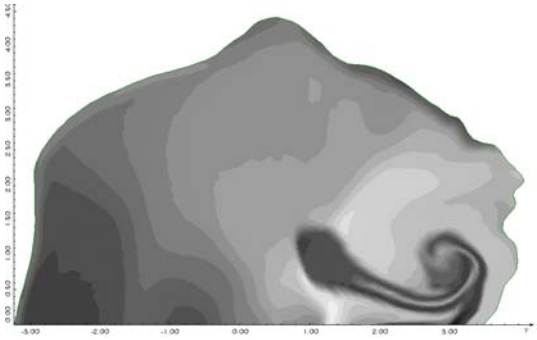

Рис. 9а

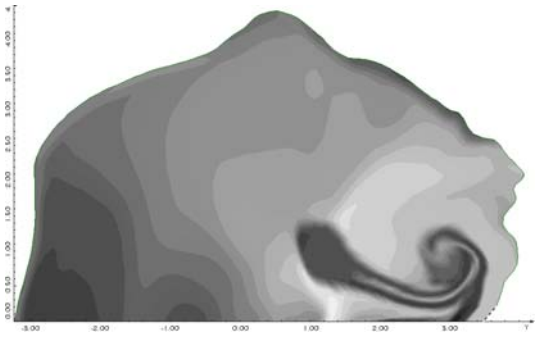

Рис. 9б

Сравнение результатов восполнения раннего перехода к целочисленным вычислениям с вычислением значений U в вещественном представлении.

Шаг второй интерполяции по столбцу: Сейчас необходимо интерполировать значение U в пиксел или точку A из точек D, E. Аналогично шагу выше:

$U_A^* = (1 – p) * U_D + p * U_E$, где $0 \le p \le 1$ и $p = |A\ D| / |D\ E|$.

Шаг вычисления результирующего значения U в точке A: Предложим, например, использовать операцию среднего арифметического:

$U_A = (U_A + U_A^*) / 2$.

Возможны и другие методы комбинации значений.

Шаг вычисления цвета: Аналогичен Алгоритму 4.

Применение двойной интерполяции очень редко оправдано из-за больших затрат времени при незначительном улучшении качества результата. В большинстве случаев визуальная разница не заметна с результатом работы алгоритма 4. Результат работы данного алгоритма приводится на рисунке 7*. Очевидно, что заметной визуальной разницы с рисунком 5* почти нет, однако, вычислительные усилия больше. Рисунок 13* представляет тот же фрагмент модели, что и на рисунках 11*, 12*.

5.3 Алгоритм 6. Восполнение по высоте

Предложим еще один алгоритм ГВ. Допустим, что внутренние пикселы локализуются с помощью алгоритмов 1, 2

или 3. Главная идея заключается в том, чтобы представить величину U как функцию двух переменных U = U(X, Y). Рис. 10. Значит, ячейка будет представлять собой некоторую часть поверхности в R^3. В общем случае узлы, например, четырехугольной ячейки не принадлежат одной плоскости.

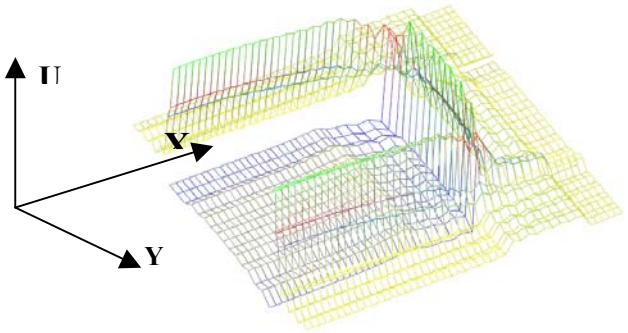

Рис. 10. Трехмерное представление дискретной функции двух переменных.

Можно предложить несколько способов реализации алгоритма 6.

1 Способ. Сделаем предположение, что узлы принадлежат одной плоскости. Составим уравнение этой плоскости по любым трем узлам из четырех, например, по узлам с индексами {(0, 0) (0, 1) (1, 1)}. Далее, можно привести его к общему уравнению плоскости: A*X+B*Y+C*Z+D = 0, где Z является величиной U. Определим коэффициенты уравнения согласно координатам:

A = ((y2-y1) * (z3-z1) - (y3-y1) * (z2-z1));
B = -((x2-x1) * (z3-z1) - (x3-x1) * (z2-z1));
C = ((x2-x1) * (y3-y1) - (x3-x1) * (y2-y1));
D = -A*X - B*Y - C*Z.

Когда пиксел локализован в ячейке (точка A на рисунке 8), то координаты X и Y известны, остается найти координату Z или в нашем случае величину U_A для этого пиксела: U_A = A*X+B*Y+D/-C.

Затем, по закону соответствия определяется соответствующий номер цвета в палитре. Этот способ дает хорошие результаты, когда существует плоскость, которой принадлежат все ее узлы. Результат по качеству и скорости сопоставим с алгоритмом 4. Недостатки алгоритма 6 в случае, если не существует плоскости, которой принадлежат все узлы ячейки:

1. Нет хорошо согласованного перехода цветов на границах между ячейками. Рисунок 17* представляет алгоритм 4, а рисунок 18* алгоритм 6.
2. Изображение получается не согласованным, скачкообразным. Рис. 14*. Модель та же, что и на рисунках 11*, 12*, 13*.

2 Способ. Этот способ и способ 3 будем рассматривать в предположении, что ячейки необязательно принадлежат одной плоскости. Разбить ячейку на два треугольника. Далее, найти общие уравнения обоих плоскостей соответствующих данным треугольникам, аналогично 1 способу. В результате получим два значения величины U для текущего внутреннего пиксела: U_A^1 и U_A^2. Предложим вариант определения результирующего значения U_A : $(U_A^1 + U_A^2)/2$. Результат получился немного лучше, чем для первого способа. Рис. 19*.

3 Способ. Аналогично второму способу разбить ячейку на два треугольника. Разница заключается в том, для каждого внутреннего пиксела нужно определить в какой именно треугольник он попал. Затем для поиска значения U_A применить либо одно уравнение плоскости, либо другое. Результат работы алгоритма восполнения является таким же хорошим, как и восполнение по алгоритму 4. По времени он работает чуть медленнее, чем алгоритм 4 на 10-15%. Но это результат нашей реализации, на наш взгляд это можно соотношение свести к нулю. Этот способ дает точное согласование переходов цветов на границах ячеек (Рис. 15*), в этом компоненте он чуть лучше, чем алгоритм 4. Эта разница выделена на рисунках 11а и 11б. Однако, в некоторых случаях этот алгоритм дает общую картину хуже. Рис. 15*. Рисунок 11а представляет результат работы алгоритма 4, рисунок 11б результат этого алгоритма. Надо обратить внимание на кривизну линии перехода от цвета к цвету, эти линии можно считать изолиниями. Разбиение, например четырехугольной ячейки, на два треугольника не однозначно, возможны два варианта разбиения диагоналями, но принципиальной разницы результатов в общем случае нет.

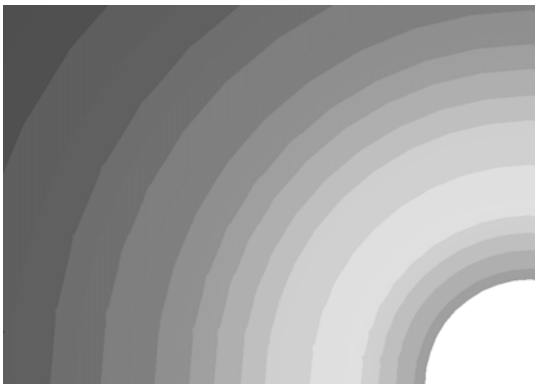

Рис. 11а

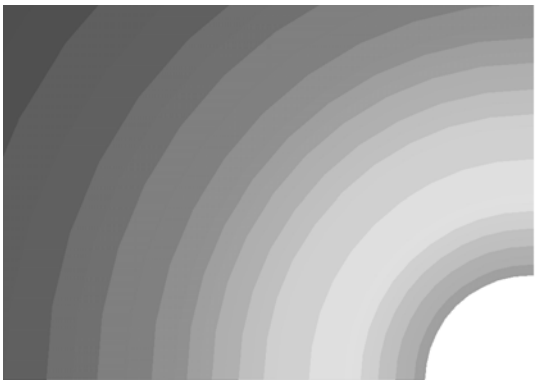

Рис. 11б

Представлено сравнение алгоритмов ГВ 4 и 6 по гладкости перехода от цвета к цвету. Выделен фрагмент, на который надо обратить внимание.

4 Способ. Можно предложить разбить ячейку на четыре треугольника, вычисляя геометрический центр ячейки, т. е. получим четыре уравнения плоскостей. Далее треугольники формируются соединением вершин с геометрическим центром. Практическая реализация показала, что этот способ имеет хуже, и качественные показатели, и особенно временные, чем алгоритм 4. На рисунках 20*, 16* показан качественный пример работы этого алгоритма на задачах, что и на рисунках 11* - 15*, 17* - 19*. Временные показатели становятся плохими при увеличении размерности сетки, это очевидно предсказывается. Однако, этот способ по корректности и качеству лучше предыдущих способов алгоритма 6.

6 Особенности ГВ для физических характеристик заданных в ячейках

Изложенные выше алгоритмы ГВ рассматривались в предположении, что физические характеристики заданы в узлах сетки. В разных методах численного моделирования применяются разные способы привязки характеристик к ячейкам.

1. Самый очевидный способ - применить представленные алгоритмы, после интерполяции значения U из ячеек в узлы. Например, с узлом можно ассоциировать среднее арифметическое значение U из смежных ячеек для этого узла.
2. Предложим алгоритм с дополнительной сеткой, узлы которой расположены в геометрическом центре масс ячеек. Размерность дополнительной сетки на единицу больше, чем размерность основной. Далее применяются алгоритмы для узловых характеристик U.

7 Ресурсы и преимущества алгоритмов ГВ 1-3

1. Результаты растровой развертки ребер ячеек можно использовать для **смежных ячеек**.
2. Изображение можно формировать в **оперативной памяти** в виде некоторого буфера, например, образ в формате **BitMap**. Затем выводить этот образ целиком на экран, например, функциями **OpenGL**.
3. Есть хорошая возможность применить алгоритмы 4-6 один раз и записать результат опять же, как образ в формате BitMap. Далее если необходимо **увеличение части модели**, то уже можно не применять эти алгоритмы, а применить более быстрые алгоритмы или стандартные возможности графических библиотек для увеличения размера образа BitMap.
4. Существует отличная возможность применения алгоритмов 4-6 для **трехмерных сеточных поверхностей** с одноразовой плоской растровой разверткой, используя, например, функции **OpenGL** для работы с **текстурами**.
5. Алгоритмы являются очень удобными для **аппаратной реализации**.

8 Оценки и выводы

Сделаем некоторые выводы и рекомендации.

1. В общем, **самые хорошие результаты** показал **алгоритм 4**. Он является самым быстрым из алгоритмов, которые работаю без применения OpenGL. По качеству и адекватности изображения показал лучшие результаты.
2. Для генерации растровой развертки более предпочтительно и эффективно применять **алгоритм 1**.
3. Применение **OpenGL** оправдано для режима быстрого **предпросмотра** изображения.
4. Возможно применение **OpenGL** для работы с BitMap образами, при **увеличении** или с применением **текстур**, которые заранее подготовлены по алгоритмам 4-6.
5. Графическое восполнение рекомендуется делать в рамках **цветового диапазона палитры**, а не в пространстве RGB.
6. Рекомендуется поздний переход к целочисленным номерам цветов в палитре, т. е. для **устранения лестничного эффекта** при переходе от цвета к цвету нужно вести вычисления в **вещественном представлении**, несмотря на некоторые временные затраты.

Главное, что алгоритмы 4-6 для восполнения одной ячейки **не требуют предварительных вычислений и информации о других ячейках** модели, в **отличие** от применения для восполнения сплайнов или интерполяционных многочленов.

Надо особо отметить, что применение для восполнения **сплайнов или интерполяционных многочленов** для **конечно-элементных сеток** вообще является **сложной и времяемкой задачей**, поэтому **алгоритмы 4-6** очень **актуальны** для ГВ таких сеточных полей.

Описание данных, которые использовались для иллюстраций работы алгоритмов:

1. Рисунки 1 – 3 и 5^* - 10^* представляют фрагмент - развития Реллей-Тейлоровской неустойчивости в жидкости. Лагранжевая методика [14], метод концентраций. Скалярным полем является энергия.
2. Рисунки 1^* – 3^* и 17^* - 20^* представляют тестовые данные, заданные вручную на маленькой сетке. Соответственно на рисунках 1^* – 3^* размер сетки 2x3 ячейки, а на рисунках 17^* – 20^* 2x2 ячейки.
3. На рисунках 9а и 9б представлена отдельная область, в которой происходит сильное перемешивание веществ, расчет проводился по лагранжевой методике [15]. На рисунках 11^* - 16^* представлен выделенный фрагмент того же расчета. Скалярным полем является плотность.
4. На рисунках 6, 11а и 11б представлена простая область модели с начальным распределением плотности по сетке.
5. На рисунке 10 представлен расчет гипотетической аварии реактора с распределением давления по высоте. [14]

Мы благодарим сотрудников математического отделения РФЯЦ-ВНИИТФ за их внимательное прочтение этой работы и за их важные советы и замечания по содержанию работы, а также за помощь в реализации иллюстративного материала.

Изложенные в этой работе подходы и алгоритмы были исследованы и реализованы, в рамках работ по созданию системы научной визуализации результатов численного моделирования и экспериментов VIZI2D. [11-13].

9 Литература

1. Труды международных конференций по компьютерной графике “Графикон’95, 96, 97, 98, 99,2000”.
2. Д.Роджерс. Алгоритмические основы машинной графики. - М. , Мир, 1989
3. Е.В. Шикин, А. В. Боресков. Компьютерная графика. Динамика, реалистические изображения. Диалог-МИФИ. Москва. 1997.
4. В. П. Иванов, А. С. Батраков. Трехмерная компьютерная графика. Радио и связь. Москва. 1995.
5. Ф.А.Плетнев. Быстрые методы закраски в реалистической графике. Вопросы атомной науки и техники (ВАНТ). Серия математическое моделирование физических процессов. 1997 г. Вып.4. Стр. 39-50.
6. Ю. М. Баяковский, В. А. Галактионов, Т. Н. Михайлова. “Графор. Графическое расширение фортрана”. Москва. “Наука”. 1985.
7. Бахвалов Н.С. Численные методы. Москва. Наука. 1973 г.
8. В. В. Кобков, Ю. И. Шокин. “Сплайн-функции в численном анализе”. Уч. пособие. Новосибирск. 1983.
9. Ш.-К. Чэн Принципы проектирования систем визуальной информации. М., Мир, 1994.
10. Д. В. Могиленских. Нелинейная цветовая интерпретации физических процессов. Труды международной конференции по компьютерной графике “Графикон’2000”.
11. Д. В. Могиленских, И. В. Павлов, В. В. Федоров и др. Принципы построения и функциональное содержание системы визуализации для анализа скалярных и векторных полей заданных на двумерных регулярных сетках. Препринт ВНИИТФ №172.
12. Могиленских Д.В., Федоров В. В., Павлов И. В. Системы научной визуализации “VIZI” для графического представления результатов математического моделирования. 3-ий сибирский конгресс по прикладной

индустриальной математике (ИНПРИМ-98). Новосибирск 22-27 июня 1998 год. Изд. Института математики СО РАН.
13. Могиленских Д.В., Федоров В. В., Павлов И. В. Визуализация двумерных результатов численного моделирования физических процессов. Система визуализации “VIZI” в ОС Windows. V Забабахинские научные чтения. Международная конференция 21-25 сентября 1998 год. Снежинск.
14. Гаджиев А. Д., Кузмин С. Ю., Лебедев С. Н., Писарев В. Н. Неявный конечно-разностный метод “Ромб” для численного решения двумерных уравнений газовой динамики на произвольных лагранжево-эйлеровых сетках. Препринт РФЯЦ-ВНИИТФ № 94, 1997.
15. Гаджиева В. В., Горин Н. В., Ким А. В., Кузмин С. Ю., Лебедев С. Н., Лебедева Т. В., Орлов Г. В., Писарев В. Н., Птицына Н. В. и др. Пакет программ SINARA для математического моделирования динамики аварийных процессов в реакторах на быстрых нейтронах. Вопросы Атомной Науки и Техники. Серия: Математическое моделирование физических процессов. 2000, Выпуск 3. МинАтом.