Компьютерная графика

Домашнее задание 1: график функции на плоскости

2021

Задание

- Функция на плоскости, зависящая от времени $f(x, y, t)$

Задание

- Функция на плоскости, зависящая от времени $f(x, y, t)$
- Фиксированный прямоугольник $[x_0, x_1] \times [y_0, y_1]$, разбитый на сетку из $W \times H$ прямоугольных ячеек

Задание

- Функция на плоскости, зависящая от времени $f(x, y, t)$
- Фиксированный прямоугольник $[x_0, x_1] \times [y_0, y_1]$, разбитый на сетку из $W \times H$ прямоугольных ячеек
- Нужно нарисовать:
  - ‘График’ функции цветом, используя вершины сетки как основу

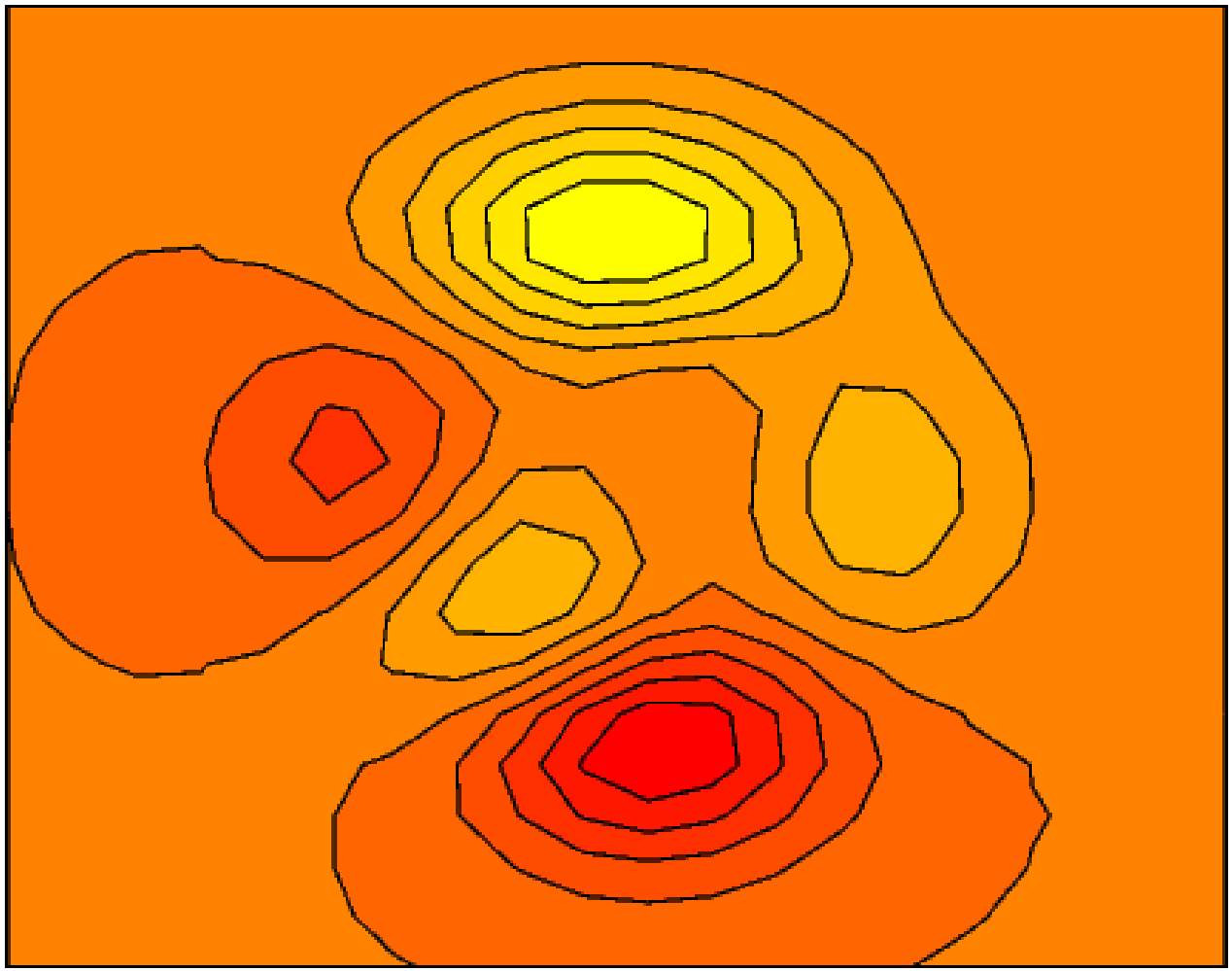
Задание

- Функция на плоскости, зависящая от времени $f(x, y, t)$
- Фиксированный прямоугольник $[x_0, x_1] \times [y_0, y_1]$, разбитый на сетку из $W \times H$ прямоугольных ячеек
- Нужно нарисовать:
  - 'График' функции цветом, используя вершины сетки как основу
  - Изолинии – линии $f(x, y, t) = const$ поверх графика

Задание

- Функция на плоскости, зависящая от времени $f(x, y, t)$
- Фиксированный прямоугольник $[x_0, x_1] \times [y_0, y_1]$, разбитый на сетку из $W \times H$ прямоугольных ячеек
- Нужно нарисовать:
  - 'График' функции цветом, используя вершины сетки как основу
  - Изолинии – линии $f(x, y, t) = const$ поверх графика
- График и изолинии вычисляются заново на каждый кадр

Пример



Функция

Любая интересная меняющаяся во времени функция на плоскости

- Metaballs: $f(x, y, t) = \sum c_i \exp\left(-\frac{(x-x_i)^2+(y-y_i)^2}{r_i^2}\right)$, где (x_i, y_i) – координаты движущейся по какому-то закону точки

Функция

Любая интересная меняющаяся во времени функция на плоскости

- Metaballs: $f(x, y, t) = \sum c_i \exp\left(-\frac{(x-x_i)^2+(y-y_i)^2}{r_i^2}\right)$, где (x_i, y_i) – координаты движущейся по какому-то закону точки
- Шум Перлина: строится на основе сетки двумерных единичных векторов, которые можно крутить в зависимости от времени (эта сетка никак не связана с сеткой использующейся для рендеринга)

Функция

Любая интересная меняющаяся во времени функция на плоскости

- Metaballs: $f(x, y, t) = \sum c_i \exp\left(-\frac{(x-x_i)^2+(y-y_i)^2}{r_i^2}\right)$, где (x_i, y_i) – координаты движущейся по какому-то закону точки
- Шум Перлина: строится на основе сетки двумерных единичных векторов, которые можно крутить в зависимости от времени (эта сетка никак не связана с сеткой использующейся для рендеринга)
- Комбинация синусов/косинусов с разными амплитудами и фазами

Функция

Любая интересная меняющаяся во времени функция на плоскости

- Metaballs: $f(x, y, t) = \sum c_i \exp\left(-\frac{(x-x_i)^2+(y-y_i)^2}{r_i^2}\right)$, где (x_i, y_i) – координаты движущейся по какому-то закону точки
- Шум Перлина: строится на основе сетки двумерных единичных векторов, которые можно крутить в зависимости от времени (эта сетка никак не связана с сеткой использующейся для рендеринга)
- Комбинация синусов/косинусов с разными амплитудами и фазами
- etc.

График

- Вершина сетки – (x_i, y_j) + цвет
- Раскрасить в зависимости от значения функции

График

- Вершина сетки – (x_i, y_j) + цвет
- Раскрасить в зависимости от значения функции
- Прямоугольники сетки придётся разбить на пары треугольников

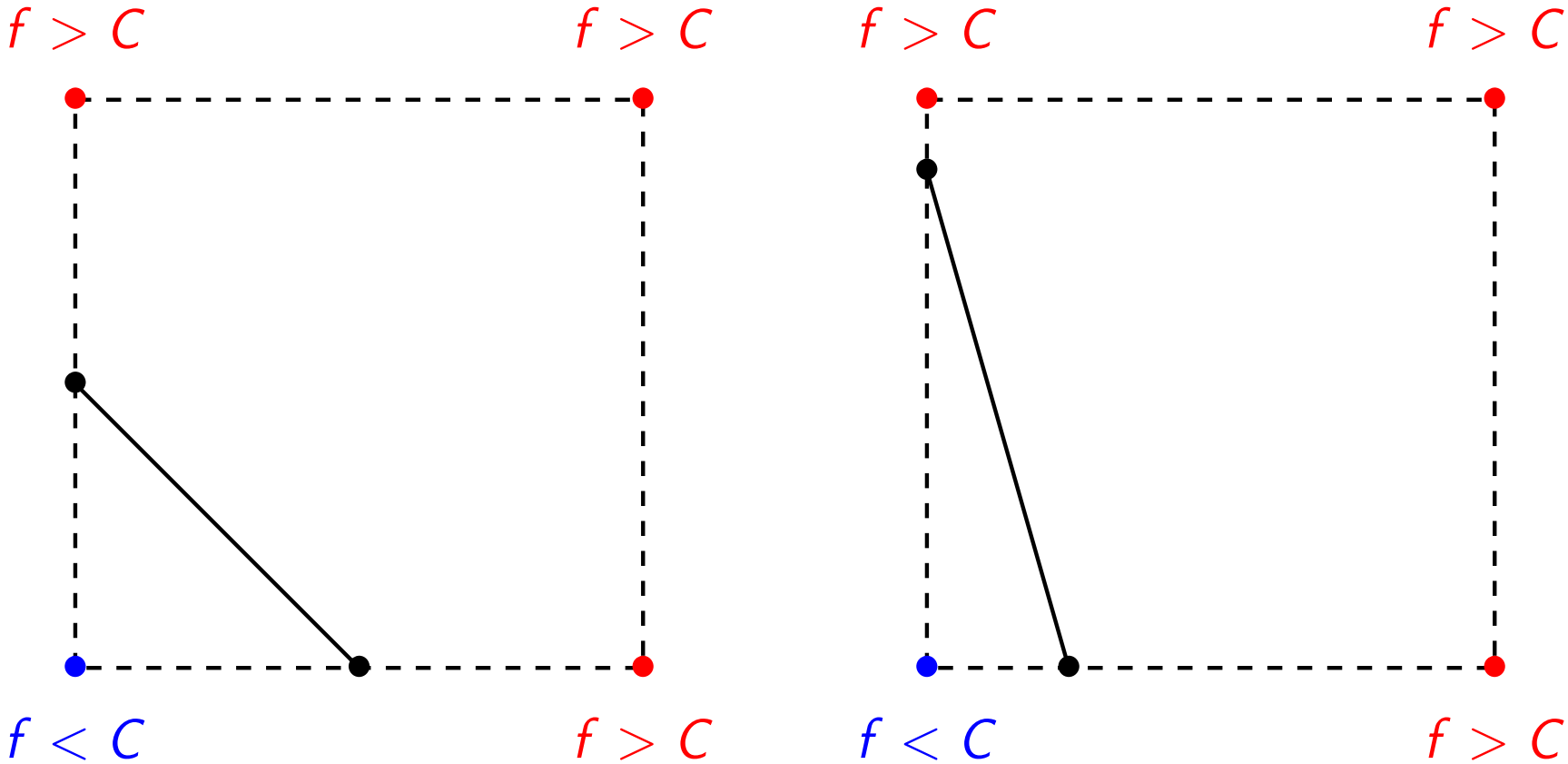
Изолинии

- Линии $z = f(x, y, t) = C_i$ для некоторых C_i, разного цвета
- Набор значений $C_1, C_2, C_3, \ldots, C_k$ выбрать на основе вашей функции

Изолинии

- Линии $z = f(x, y, t) = C_i$ для некоторых C_i, разного цвета
- Набор значений $C_1, C_2, C_3, \ldots, C_k$ выбрать на основе вашей функции
- Строить алгоритмом marching squares с линейной интерполяцией (или аналогичным алгоритмом на треугольниках) на основе той же сетки, что и график

Marching squares



- Есть вариант алгоритма, соединящий центры рёбер
- Есть вариант алгоритма, линейно интерполирующий значение функции вдоль ребра чтобы найти точку $f = C$ – нужно использовать его

Marching triangles

- Есть вариант алгоритма (иногда называется marching triangles), использующий треугольники и избавляющийся от неоднозначностей алгоритма на квадратах, – можно использовать его

Marching triangles

- Есть вариант алгоритма (иногда называется marching triangles), использующий треугольники и избавляющийся от неоднозначностей алгоритма на квадратах, – можно использовать его
- Ещё marching triangles называют трёхмерный алгоритм построения модели по облаку точек – это **не оно**

Немного деталей

- Часть данных вершин будут обновляться каждый кадр – цвета точек графика, координаты изолиний
- Часть данных постоянна – XY-координаты вершин сетки – их имеет смысл хранить в отдельном VBO
- Как для графика, так и для изолиний имеет смысл использовать индексированный рендеринг
- Можно использовать простые `GL_LINES` и `GL_TRIANGLES`, а можно `GL_LINE_STRIP`/`GL_LINE_LOOP` и `GL_TRIANGLE_STRIP` вместе с `primitive restart`
- Чтобы избавиться от дублирования вершин в изолиниях, придётся для каждого ребра исходной сетки запомнить индекс вершины изолинии, лежащей на этом ребре: можно использовать, например, `std::unordered_map` или просто `std::vector`, придумав какую-то нумерацию для рёбер

Баллы

- 3 балла: рисуется динамический график функции (т.е. цвета меняются во времени)
- 3 балла: рисуются динамические изолинии
- 2 балла: все данные в VBO обновляются только при их изменении, статические данные хранятся в отдельных VBO
- 2 балла: используется индексированный рендеринг для графика и его вершины не дублируются
- 2 балла: используется индексированный рендеринг для изолиний и их вершины не дублируются
- 1 балл: можно динамически менять количество изолиний
- 1 балл: можно динамически менять детализацию сетки
- 1 балл: корректно обрабатывается aspect ratio (т.е. соотношение ширины к высоте видимого графика не меняется при изменении размеров экрана)

Всего: 15 баллов

Защита заданий на практике 26 сентября

Ссылки

- jamie-wong.com/2014/08/19/metaballs-and-marching-squares
- jacobzelko.com/marching-squares
- ckcollab.com/2020/11/08/Marching-Squares-Algorithm.html