Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Московский государственный технический университет имени Н.Э. Баумана
(национальный исследовательский университет)»
(МГТУ им. Н.Э. Баумана)

ФАКУЛЬТЕТ «Информатика и системы управления»

КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»

РАСЧЕТНО-ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

К КУРСОВОМУ ПРОЕКТУ

НА ТЕМУ:

разработка ПО, которое:
1) реализует алгоритм деформации (сокращения и растяжения) человеческого бицепса с помощью геометрической модели на узлах, сохраняющей объем при деформации;
2) предоставляет возможности загрузки модели из конфигурационного файла, управления ее состоянием (сокращение и растяжение) и положением (вращение, перемещение и масштабирование)

Студент ИУ7-53Б _____ **П. Г. Пересторонин**
(Группа) (Подпись, дата) (И.О.Фамилия)

Руководитель курсового проекта _____ **А. А. Оленев**
(Подпись, дата) (И.О.Фамилия)

2020 г.

Оглавление

Введение **2**

1 Аналитическая часть **4**

1.1 Формальное описание геометрической модели мышцы 4

1.2 Расчёт формул деформации мышцы 5

1.3 Формальное описание геометрической модели каркаса мышцы 8

1.4 Анализ алгоритмов удаления невидимых линий и поверхностей 9

1.4.1 Алгоритм обратной трассировки лучей 10

1.4.2 Алгоритм, использующий Z-буфер 11

1.4.3 Алгоритм Робертса 12

1.5 Анализ методов закрашивания 13

1.5.1 Простая закраска 13

1.5.2 Закраска по Гуро 14

1.5.3 Закраска по Фонгу 15

2 Конструкторская часть **17**

3 Технологическая часть **18**

4 Исследовательская часть **19**

Заключение **20**

Литература **21**

Введение

Наиболее удобным представлением информации для восприятия человеком является графическое представление. С развитием вычислительной техники, немаловажной функцией стала обработка информации, связанной с изображениями. Данная обработка подразумевает преобразование информации о некоторой местности в графическое представление этой местности и называется задачей *синтеза изображения*.

В общем случае задача синтеза изображения трехмерных объектов представляет собой задачу имитации визуальной обстановки, т.е. искусственного построения изображений окружающей среды с такой степенью достоверности, которая достаточна для выработки и поддержания у пользователя программы навыков управления подвижными объектами. Таким образом, синтезируемое изображение должно быть динамическим и как можно более реалистичным.

В компьютерной графике решаются задачи разработки модели синтезируемой обстановки, задания исходных данных, связанных с определением положения наблюдателя и картинной плоскости, преобразования объектов сцены из одной системы координат в другую систему координат, отсечение объектов, вычисление перспективных проекций, удаление невидимых линий и поверхностей, создание реалистических изображений, растровой развертки простейших элементов изображения.

Цель данной работы - реализация ПО, которое предоставляет геометрическую модель человеческого бицепса на узлах, способную деформироваться (а именно сокращаться и растягиваться) с сохранением объема, а также предоставляет возможность деформации этой модели и применения афинного преобразования к ней.

Чтобы достигнуть поставленной цели, требуется решить следующие задачи:

- формально описать структуру геометрической модели мышцы;
- рассчитать формулы, позволяющие деформировать геометрическую модель и обеспечивающие сохранение его объема;
- формально описать структуру геометрической модели каркаса мыш-

цы;

- выбрать алгоритмы трехмёрной графики, позволяющие визуализировать модель;
- реализовать данные алгоритмы для визуализации описанных выше объектов.

1 Аналитическая часть

1.1 Формальное описание геометрической модели мышцы

Ввиду сложности прямого описания поверхности мышцы некоторым уравнением, модель мышцы в данной работе будет представляться с помощью группы усечённых конусов, полученной путем вращения группы отрезков, находящихся между узлами модели. Очевидно, что при такой модели радиусы основания конуса описываются значениями точек начала и конца отрезка, которые с данного момента будут отождествляться с *радиусом узла*.

Для более простой работы с моделью расстояние между узлами было выбрано постоянным. Данный факт не вносит никаких ограничений в представление модели, так как описывая некоторый узел пропорционально его соседям, можно получить представление, в котором визуально расстояние между этими двумя соседними узлами будет удвоено. Повторяя данные действия можно получить любое расстояние между двумя произвольными узлами.

Учитывая все, описанное выше, для исчерпывающего описания определенного состояния и формы мышцы требуется, во-первых, массив радиусов оснований конусов в узлах, а во-вторых, расстояние между узлами.

Однако для исчерпывающего описания свойств мышцы этого не достаточно. Данный набор не хранит информации о поведении узлов при деформациях, в связи с чем также для каждого узла требуется добавить коэффициент приращения, который будет означать относительный прирост данного узла.

- массив радиусов узлов;
- расстояние между узлами;
- массив коэффициентов прироста радиусов узлов.

1.2 Расчёт формул деформации мышцы

Как уже было сказано выше, общий объем модели составляется из объемов составляющих ее усечённых конусов, объем которых в свою очередь вычисляется как фигура вращения отрезка.

Отрезок в данном случае проще всего задавать в виде прямой, заданной уравнением $y = ax + b$, которая ограничена по x: $x \in [0; \Delta x]$, где Δx - расстояние между узлами. Рассматривать для i-ого и $(i+1)$-ого узлов отрезок $[0; \Delta x]$ гораздо проще, чем отрезок $[(i-1) \cdot \Delta x; i \cdot \Delta x]$, в связи с чем во всех последующих вычислениях подразумевается, что пара соседних узлов с индексами i и $(i+1)$ сдвигается на $(i-1) \cdot \Delta x$ влево (можно заметить, что во всех последующих расчетах нас будет интересовать исключительно величина Δx, поэтому все последующие расчёты справделивы и для исходного положения составных частей модели). Площадь усечённого конуса в таком случае можно рассчитывать по формуле (1.1):

$$\mathcal{V} = \pi \cdot \int_0^{\Delta x} f^2(x)dx, \qquad \text{где } f(x) = ax + b \tag{1.1}$$

Далее можно подставить значение функции и рассчитать интеграл (1.2):

$$\mathcal{V} = \pi \cdot \int_0^{\Delta x} \left(a^2x^2 + 2abx + b^2\right) dx = \pi \cdot \left(\frac{a^2x^3}{3} + abx^2 + b^2x\right)\Bigg|_0^{\Delta x} \tag{1.2}$$

Из чего следует (1.3):

$$\mathcal{V} = \pi \cdot \left(\frac{a^2\Delta x^3}{3} + ab\Delta x^2 + b^2\Delta x\right) \tag{1.3}$$

Коэффициенты a (угла наклона прямой) и b (точки пересечения прямой с Oy) можно найти исходя из двух имеющихся точек отрезка. Так, для отрезка между i-ым и $(i+1)$-ым узлами, получается (1.4) и (1.5):

$$a = \frac{y_{i+1} - y_i}{\Delta x} \tag{1.4}$$

$$b = y_i \tag{1.5}$$

Пусть n - кол-во узлов. Тогда $R = \{r_1, r_2, \dots r_n\}$ - массив радиусов узлов, $M = \{m_1, m_2, \dots, m_n\}$ - массив коэффициентов прироста радиусов узлов. Отсюда (учитывая формулы (1.3), (1.4) и (1.5)) получается, что общий объем, для удобства последующих вычислений поделённый на π, можно вычислить по формуле (1.6):

$$V = \frac{\mathcal{V}}{\pi} = \sum_{i=1}^{n-1} \left(\frac{(r_{i+1} - r_i)^2 \cdot \Delta x}{3} + 2r_i(r_{i+1} - r_i) \cdot \Delta x + r_i^2 \Delta x \right) \tag{1.6}$$

После выноса за знак суммы Δx получается итоговая формула вычисления объема (1.7):

$$V = \Delta x \cdot \sum_{i=1}^{n-1} \left(\frac{(r_{i+1} - r_i)^2}{3} + 2r_i(r_{i+1} - r_1) + r_i^2 \right) \tag{1.7}$$

Которую можно представить как функцию от расстояния между узлами Δx и массива радиусов узлов (1.8):

$$V(\Delta x, R) = \Delta x \cdot F(R), \tag{1.8}$$

где $F(R)$ - функция (1.9):

$$F(R) = \sum_{i=1}^{n-1} \left(\frac{(r_{i+1} - r_i)^2}{3} + 2r_i(r_{i+1} - r_1) + r_i^2 \right) \tag{1.9}$$

Длина модели L является суммой расстояний между узлами и для n узлов находится следующим образом (1.10):

$$L = (n - 1) \cdot \Delta x \tag{1.10}$$

Откуда можно вывести зависимость расстояния между узлами от длины (1.11):

$$\Delta x = \frac{L}{n - 1} \tag{1.11}$$

При деформации модели (сокращении или растяжении) объем должен

оставаться постоянным. Если посмотреть на выражение (1.8) становится очевидным, что для равенства $V = V'$, где V - объем до деформации, а V' - после, должно выполнятся равенство (1.12):

$$\Delta x \cdot F(R) = \Delta x' \cdot F(R') \tag{1.12}$$

где R' - массив радиусов узлов после деформации, $\Delta x'$, учитывая (1.11), находится из (1.13):

$$\Delta x' = \frac{L'}{n-1} = \frac{L + \delta x}{n-1} = \Delta x + \frac{\delta x}{n-1} \tag{1.13}$$

где δx - изменение длины, а $L' = L + \delta x$ - новое значение длины модели.

Отсюда получается, что $F(R')$ можно найти, как (1.14):

$$F(R') = \frac{F(R) \cdot \Delta x}{\Delta x'} = \frac{\Delta x}{\Delta x'} \cdot F(R) \tag{1.14}$$

Пусть δy - элементарное приращение радиуса. Тогда новое значение для радиуса каждого узла можно найти из соотношения:

$$r'_i = r_i + m_i \cdot \delta y \tag{1.15}$$

где m_i (как было описано выше) - коэффициент прироста радиуса.

Таким образом можно выразить через (1.15) радиусы массива R' и найти $F(R')$ как (1.16):

$$F(R') = \sum_{i=1}^{n-1} \Big(\frac{((r_{i+1} + m_{i+1} \cdot \delta y) - (r_i + m_i \cdot \delta y))^2}{3} + \\ +2(r_i + m_i \cdot \delta y) \cdot (r_{i+1} + m_{i+1} \cdot \delta y - r_i + mi \cdot \delta y) + (r_i + m_i \cdot \delta y)^2 \Big) \tag{1.16}$$

После приведения подобных слагаемых относительно δy получается (1.17):

$$F(R') = A\delta y^2 + B\delta y + C, \tag{1.17}$$

где A, B, C рассчитываются из (1.18), (1.19), (1.20) соответственно.

$$A = \sum_{i=1}^{n-1} \left(\frac{1}{3} m_{i+1}^2 - \frac{5}{3} m_{i+1} m_i + \frac{7}{3} m_i^2 \right) \tag{1.18}$$

$$B = \sum_{i=1}^{n-1} \left(m_{i+1} r_i + m_i r_{i+1} \right) \tag{1.19}$$

$$C = \sum_{i=1}^{n-1} \left(\frac{(r_{i+1} - r_i)^2}{3} + r_{n+1} r_n \right) \tag{1.20}$$

Квадратное уравнение (1.17) будет иметь решения (1.21):

$$\delta y_{1,2} = \frac{-B \pm \sqrt{B^2 - 4AC}}{2A} \tag{1.21}$$

Среди решений больший интерес представляет большее, потому что меньшее решение находит сохранение объема при отрицательных радиусах, что не является верных в рамках имеющейся задачи. Таким образом δy - больший корень среди корней из (1.21), который равен (1.22):

$$\delta y = \frac{-B + \sqrt{B^2 - 4AC}}{2A} \tag{1.22}$$

Таким образом при деформации модели на величину δx новые значения радиусов узлов будут иметь вид (1.15), где величина δy находится из выражения (1.22).

1.3 Формальное описание геометрической модели каркаса мышцы

В данной работе каркас будет отождествляться с 2 стержнями, соединенными вместе на одном из концов. Положение каркаса во многом зависит от положения и состояния модели мышцы, однако есть параметры, которые от нее не зависят, а именно - точки крепления. Для представления каркаса будет достаточно следующих параметров:

- расстояние от нескрепленного конца первого стержня до точки креп-

ления мышцы;

- расстояние от точки крепления мышцы к первому стержню до точки соединения стержней;
- расстояние от точки соединения стержней до точки крепления мышцы ко второму стержню;
- расстояние от точки крепления мышцы ко второму стержню до нескрепленного конца второго стержня;

Углы наклона стержней каркаса будут находится из теоремы косинусов, которая для треугольника с длинами сторон A, B, C и угла α, лежащего напротив стороны с длиной A, будет иметь вид (1.23):

$$A^2 = B^2 + C^2 - 2BC\cos(\alpha) \tag{1.23}$$

Откуда можно выразить угол α как (1.24):

$$\alpha = \arccos\left(\frac{B^2 + C^2 - A^2}{2BC}\right) \tag{1.24}$$

1.4 Анализ алгоритмов удаления невидимых линий и поверхностей

При выборе алгоритма удаления невидимых линий и поверхностей нужно учесть важную особенность поставленной задачи - работа программы будет выполняться в реальном режиме при взаимодействии с пользователем. Данный факт предъявляет к алгоритму в первую очередь требование быстрой скорости работы. Для выбора наиболее подходящего алгоритма следует рассмотреть уже имеющиеся алгоритмы удаления невидимых линий и поверхностей.

1.4.1 Алгоритм обратной трассировки лучей

Данный алгоритм позволяет получать очень реалистичные изображения, работает в пространстве изображения.

Идея состоит в следующем: для определения цвета пиксела экрана через него из точки наблюдения проводится луч, ищется его ближайшее пересечение со сценой и определяется освещенность точки пересечения. Эта освещенность складывается из отраженной и преломленной энергий, непосредственно полученных от источников света, а также отраженной и преломленной энергий, идущих от других объектов сцены. После определения освещенности этой точки учитывается ослабление света при прохождении через прозрачный материал и в результате получается цвет точки экрана.

Исходя из описания идеи алгоритма, можно сделать вывод о его плюсах и минусах.

Плюсы:

- качественное изображение, которое может быть построено с учётом явлений дисперсии лучей, преломления, а также внутреннего отражения;
- возможность использования в параллельных вычислительных системах.

Минусы:

- трудоёмкие вычисления;
- высокая сложность реализации версии, учитывающей все физические явления.

Вывод: так как в поставленной задаче в первую очередь стоит требование быстроты работы алгоритма, а также ввиду того, что модели, использующиеся в программе, не являются прозрачными и имеют преимущественно диффузное отражение, данный алгоритм не подходит в рамках данной работы.

1.4.2 Алгоритм, использующий Z-буфер

Данный алгоритм является одним из самых простых алгоритмов удаления невидимых линий и поверхностей, работает в простанстве изображения.

Идея состоит в следующем: имеется 2 буфера: буфер кадра, который используется для запоминания атрибутов (интенсивности) каждого пиксела в пространстве изображения, а также z-буфер - это отдельный буфер глубины, используемый для запоминания координаты z или глубины каждого видимого пиксела в пространстве изображения. В процессе работы глубина или значение z каждого нового пиксела, который нужно занести в буфер кадра, сравнивается с глубиной того пиксела, который уже занесен в z-буфер. Если это сравнение показывает, что новый пиксел расположен впереди пиксела, находящегося в буфере кадра, то новый пиксел заносится в этот буфер и, кроме того, производится корректировка z-буфера новым значением z. Если же сравнение дает противоположный результат, то никаких действий не производится. По сути, алгоритм является поиском по x и y наибольшего значения функции z (x, y).

Исходя из описания идеи алгоритма, можно сделать вывод о его плюсах и минусах. Плюсы:

- простота реализации;
- простота работа со сложными поверхностями;
- отсутствие требования сортировки объектов по глубине;
- высокая скорость работы.

Минусы:

- требование большого объема требуемой памяти (в рамках современных вычислительных систем несущественно);
- сложность работы с прозрачными и просвечивающими объектами.

Вывод: ввиду того, что алгоритм удовлетворяет главному требованию программы, а его минусы не входят в число требований к программе, дан-

ный алгоритм может быть выбран в качестве алгоритма удаления невидимых линий и поверхностей в данной работе.

1.4.3 Алгоритм Робертса

Данный алгоритм представляет собой первое известное решение задачи об удалении невидимых линий, работает в объектном пространстве.

Идея: алгоритм прежде всего удаляет из каждого тела те ребра или грани, которые экранируются самим телом. Затем каждое из видимых ребер каждого тела сравнивается с каждым из оставшихся тел для определения того, какая его часть или части, если таковые есть, экранируются этими телами.

Плюсы:

- простые, но при этом точные математические методы;
- реализации алгоритма, использующие предварительную приоритетную сортировку вдоль оси z и простые габаритные или минимаксные тесты, демонстрируют почти линейную зависимость от числа объектов.

Минусы:

- вычислительная трудоёмкость алгоритма теоретически растет, как квадрат числа объектов;
- большое количество этапов обработки и предварительных вычислений (из чего следует большое количество кода и высокая асимптотическая константа);
- отсутствие возможности работы с прозрачными и просвечивающими объектами.

Вывод: данный алгоритм сложен в реализации в виду большого количества требуемых оптимизаций, а так же будет работать медленнее с аппроксимированными фигурами вращения, нежели со стандартными многогранниками, что приведет к медленной работе и не будет удовлетворять главному требованию программы.

Вывод

Ввиду требования к быстрой работе алгоритма, а так же большого количества плоскостей, получающихся при аппроксимации фигур вращения и замедляющих алгоритмы, работающие в объектном пространстве, рациональным выбором будет выбор алгоритма z-буфера в качестве алгоритма удаления невидимых линий и поверхностей.

1.5 Анализ методов закрашивания

Методы закрашивания используются для затенения полигонов (или поверхностей, аппроксимированных полигонами) в условиях некоторой сцены, имеющей источники освещения. С учётом взаимного положения рассматриваемого полигона и источника света находится уровень освещенности по закону Ламберта (1.25):

$$I_\alpha = I_0 \cdot \cos(\alpha) \tag{1.25}$$

где I_α - уровень освёщенности в рассматриваемой точке, I_0 - максимальный уровень освёщенности, а α - угол между вектором нормали к плоскости и вектором, направленным от рассматриваемой точки к источнику освещения (в случае нормированных векторов может быть рассчитан как скалярное произведение данных векторов).

1.5.1 Простая закраска

Идея данного алгоритма очень проста: вся грань закрашивается одним уровнем интенсивности, который зависит высчитывается по закону Ламберта. При данной закраске все плоскости (в том числе и те, что аппроксимируют фигуры вращения), будут закрашены однотонно, что в случае с фигурами вращения будет давать ложные ребра.

Плюсы:

- быстрая работа;

- хорошо подходит для многогранников, обладающих преимущественно диффузным отражением.

- плохо подходит для фигур вращения: видны ребра.

Вывод: учитывая, что в рамках данной работы все модели - фигуры вращения, данных алгоритм не является подходящим.

1.5.2 Закраска по Гуро

Идея данного алгоритма заключается в билинейной интерполяции в каждой точки интенсивности освещения в вершинах.

Нормаль к вершине можно найти несколькими способами:

- интерполировать нормали прилегающих к вершине граней;

- использовать геометрические свойства фигуры (так, например, в случае со сферой ненормированный вектор нормали будет в точности соответствовать вектору от центра сферы до рассматриваемой точки).

После нахождения нормали ко всем вершинам находится интенсивность в каждой вершине по закону Ламберта (1.25). Затем алгоритм проходится сканирующими строками по рассматриваемому полигону для всех y : $y \in [y_{min}; y_{max}]$. Каждая сканирующая строка пересекает 2 ребра многоугольника, пусть для определённости это будут ребра через одноименные вершины: MN и KL. В точках пересечения высчитывается интенсивность путём интерполяции интенсивности в вершинах. Так, для точки пересечения с ребром MN интенсивность будет рассчитана как (1.26):

$$I_{MN} = \frac{l_1}{l_0} \cdot I_M + \frac{l_2}{l_0} \cdot I_N \tag{1.26}$$

где l_1 - расстояние от точки пересечения до вершины N, l_2 - расстояние от точки пересечения до вершины M, l_0 - длина ребра MN. Для точки пересечения сканирующей строки с ребром KL интенсивность высчитывается аналогично.

Далее, после нахождения точек пересечения, алгоритм двигается по Ox от левой точки пересечения X_{left} до правой точки пересечения X_{right} и в каждой точке $\mathcal{X}$ интенсивность рассчитывается как (1.27):

$$I_{\mathcal{X}} = \frac{\mathcal{X} - X_{left}}{X_{right} - X_{left}} \cdot I_{X_{right}} + \frac{X_{right} - \mathcal{X}}{X_{right} - X_{left}} \cdot I_{X_{left}} \tag{1.27}$$

Плюсы:

- хорошо подходит для фигур вращения, аппроксимированных полигонами, с диффузным отражением.

Минусы:

- при закраске многогранников ребра могут стать незаметными.

Вывод: с учетом наличия в данной работе исключительно фигур вращения данных алгоритм подходит гораздо больше предыдущего.

1.5.3 Закраска по Фонгу

Данный алгоритм работает похожим на алгоритм Гуро образом, однако ключевым отличием является то, что интерполируются не интенсивности в вершинах, а нормали. Таким образом, закон Ламберта в данном алгоритме применяется в каждой точке, а не только в вершинах, что делает этот алгоритм гораздо более трудоёмким, однако с его помощью можно гораздо лучше изображаются блики.

Плюсы:

- хорошо отображает блики, вследствие чего подходит для закраски фигур с зеркальным отражением.

Минусы:

- самый трудоёмкий алгоритм из рассмотренных.

Вывод

Ввиду требования к быстрой работе алгоритма, а так же большого наличия исключительно моделей с диффузным отражением, рациональным выбором будет выбор алгоритма закраски Гуро в качестве метода закрашивания.

Вывод

В данном разделе были формально описаны модели мышцы, каркаса и законы, по которым эти модели деформируются, были рассмотрены алгоритмы удаления невидимых линий и поверхностей, методы закрашиваания поверхностей. В качестве алгоритма удаления невидимых линий и поверхностей был выбран алгоритм z-буфера, в качестве метода закрашивания был выбран алгоритм закраски Гуро.

2 Конструкторская часть

Вывод

3 Технологическая часть

Вывод

4 Исследовательская часть

Вывод

Заключение

Литература