Санкт-Петербургский государственный университет

Программная инженерия
Кафедра системного программирования

Катербарг Глеб Юрьевич

Библиотека алгоритмов деформации 3D модели мягкого тела для компьютерной системы планирования медицинских операций

Бакалаврская работа

Научный руководитель:
ст. преп. Немешев М. Х.

Рецензент:
инженер-программист, ООО "Системы компьютерного моделирования"
Монькин А. А.

Санкт-Петербург
2017

SAINT-PETERSBURG STATE UNIVERSITY

Software Engineering
Software Engineering Departmen

Gleb Katerbarg

Library of 3D soft body model deformation algorithms for surgery planning computer system

Graduation Thesis

Scientific supervisor:
Senior Lecturer Marat Nemeshev

Reviewer:
Software Engineer, Computer Simulation Systems Ltd
Alexander Monkin

Saint-Petersburg
2017

Оглавление

Введение

На сегодняшний день информационные технологии глубоко проникают в различные сферы жизни человека, и медицина, а в особенности пластическая хирургия, одна из первых идет в ногу со временем и начинает использовать предлагаемые технологические новшества.

Пластическая хирургия занимается устранением дефектов и деформаций органов, тканей, поверхностей человеческого тела. Операции этого раздела хирургии принято разделять на реконструктивные и эстетические. Реконструктивные пластические операции позволяют устранить деформации тканей и органов с восстановлением их функций методами пластической хирургии. Подобные операции проводятся у людей с какими-либо телесными повреждениями, полученными в результате травмы или болезни. Для улучшения же внешности пациента используются эстетические пластические операции, которые в настоящее время приобретают всё большую популярность.

Первостепенной целью пластической хирургии является восстановление функционального значения органа, но и проблема восприятия пациентом внешнего вида восстановленного органа остается немаловажной. Зачастую на консультации пациенту с трудом удается воспринимать предлагаемые варианты операций по фотоснимкам из архива врача, поэтому в настоящее время при планировании медицинских операций широко применяются компьютерные системы.

Компьютерное моделировние результатов пластических операций является немаловажным этапом на пути изменения внешности. Обычно пациенты затрудняются представить будущие изменения. В этом случае симуляция результатов операции дает возможность заранее увидеть планируемые перемены и, при необходимости, скорректировать их. Хирург же, в свою очередь, сможет дать ответы на большинство вопросов пациента и оправдать его ожидания. Для моделирования результатов пластической операции и их демонстрации пациенту производится сканирование, восстанавливается трехмерная модель интересующей области, и по полученной модели изучают различные ее свойства, произ-

водят замеры. К примеру, для коррекции фигуры удаляется лишняя жировая ткань и производятся различные подтяжки участков кожи. При планировании такой операции требуется определить объем жировой ткани, который необходимо удалить, установить места возможных подтяжек кожных покровов. Для моделирования результата необходимо имитировать физическое поведение тканей тела человека. Демонстрация возможных результатов этой операции пациенту может быть в дальнейшем использована для принятия решения о проведении операции.

В компьютерной системе планирования хирургических операций Phoenixcas 3D Viewer [11] встроена возможность восстановления поверхностной трехмерной модели пациента, визуализация, выполнение замеров, различные способы сравнения трехмерных моделей и другое. В данный программный продукт в целях расширения функциональности добавляются возможности симуляции результатов различных операций, и зачастую в реализации алгоритмов симуляции нужны вспомогательные деформаторы, реализацию которых целесообразно вынести в отдельную библиотеку для дальнейшего использования. О её разработке и пойдет речь в рамках данной работы.

1. Постановка задачи

Целью работы является разработка библиотеки алгоритмов деформации 3D модели мягкого тела в рамках компьютерной системы планирования хирургических операций Phoenixcas 3D Viewer.
Для этого необходимо выполнить следующие задачи:

1. Провести обзор методов деформации мягких тел.
2. Разработать и релаизовать архитектуру библиотеки.
3. Разработать алгоритмы деформации мягкого тела.
4. Провести апробацию библиотеки.

2. Обзор

2.1. Методы физического моделирования

Для симуляции результатов проведения операций на мягких тканях широко применяются методы физического моделирования. Для реализации поставленной задачи используются особенности моделирования тканей человека в биомеханике и модель деформации, основанная на механических законах физики. Существуют несколько основных физических моделей, подходящих для моделирования результатов операций.

2.1.1. Модель конечных элементов

При использовании данной модели физического моделирования объект разбивается на элементы, например, тетраэдры или кубы. На каждом элементе берется точка - либо одна из вершин тетраэдра или куба, либо его центр масс. Они будут нужны для описания конечных элементов в пространстве. Модель конечных элементов представлена на рисунке 1. Перемещение выделенных точек заранее неизвестно, и его нужно вычислить. Все остальные точки объекта выражаются через указанные. Для этого используются характеристики материала и форма конечного элемента. Вершинные точки соседних конечных элементов совпадают. В результате этого шага получается набор уравнений, по которым строится одно большое уравнение. Оно решается относительно перемещений рассматриваемых точек. Данная модель является очень точной, однако эта модель требует знания всех характеристик материала, необходимых для выведения сооотношений в конечном элементе. Подробнее данный метод описан в [5].

2.1.2. Модель масс-тензоров

Данная модель схожа с моделью конечных элементов при построении модели. Она отличается от нее на этапе решения механических задач. Данная модель описана также в [5]. Объект делится на конечные элементы, каждый из которых представляется в виде тензора. Верши-

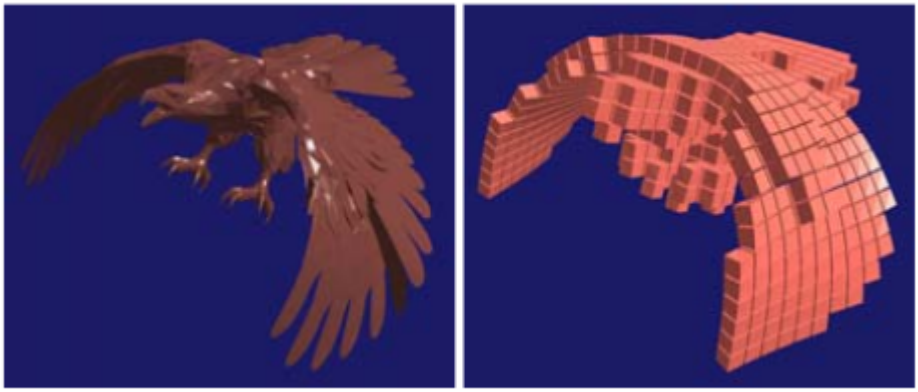

Рис. 1: Модель конечных элементов [10]

ны конечных элементов теперь имеют массы. Соответственно тензоры описывают физические величины, характеризующие способность твердого тела деформироваться. Моделирование происходит итеративно: выбирается величина временного промежутка и рассматривается состояние системы через эти промежутки. У модели есть небольшое преимущество в простоте реализации по сравнению с моделью конечных элементов, но она проигрывает в скорости вычислений.

2.1.3. Модель масс-с-пружинами

В рамках данной модели объект представляется в виде набора частиц, соединенных между собой пружинами. Частицы имеют свои массы, а сумма масс частиц определяет массу всего объекта. Пружины сохраняют расстояние между частицами, свойста объекта задаются расположением и жесткостью пружин. При изменении объема по этим пружинам передаются силы с целью восстановления объема объекта. Отсюда следует, что любая деформация объекта описывается воздействием внешних сил на частица. При этом пружины могут сохранять форму объекта, площадь поверхности, объем и другие свойства. Как и в предыдущем случае моделирование выполняется итеративно. Подробнее данная модель описана в [10], ее можно увидеть на рисунке 2. Основное преимущество модели масс-с-пружинами - это простота вычислений и скорость выполнения моделирования. Также при использо-

вании данной модели требуется знать только основные характеристики мягких тканей. Модель масс-с-пружинами дает приемлемую точность моделирования результатов операции для показа пациенту, хоть и является менее точной по сравнению с вышеописанными моделями.

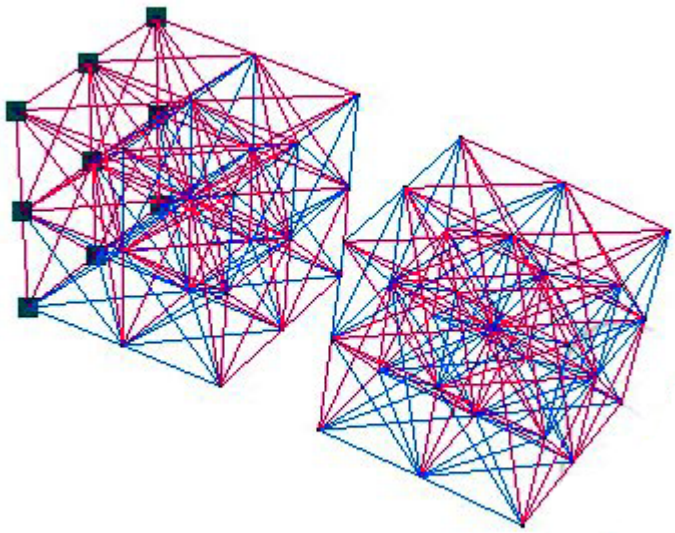

Рис. 2: Модель масс-с-пружинами [10]

2.1.4. Модель SPH

Объект представлен множеством частиц. Частица находятся в пространстве, имеют массу, плотность, другие свойства. Свойство объекта складывается из сумм свойств частиц. Изменение положение и формы объекта вычисляется через взаимодействие частиц друг с другом. Аналогично предыдущим случаям моделирование выполняется по итерациям. Данная модель очень хорошо подходит для моделирования свойств некоторой жидкости, газов. В остальных же случаях модель SPH проигрывает остальным рассматриваемым моделям по сложности работы и скорости вычисления. Модель SPH описана в [9] на примере применения данного метода для симуляции течения крови в сосуде.

2.2. Параметризация модели

Объект состоит из множества вершин. Для работы данного аналитического метода моделирования необходимо выделить некоторые особые точки, к примеру, точку по центру живота, груди, шеи. Расположение особых точек на модели человека представлено на рисунке 3. Далее по ним происходит поиск ключевых точек, т. е. точек, координаты которых находятся в определенных пропорциональных соотношениях с особыми. Ключевые точки сдвигаются и связываются кривыми Безье. После чего каркас заполняется интерполирующими кривыми. Из очевидных минусов данного метода можно выделить необходимость изначального выделения особых точек, по которым будет меняться вся модель в зависимости от конкретной реализации алгоритма. Подходы к параметризации трехмерной модели тела человека описаны в [7].

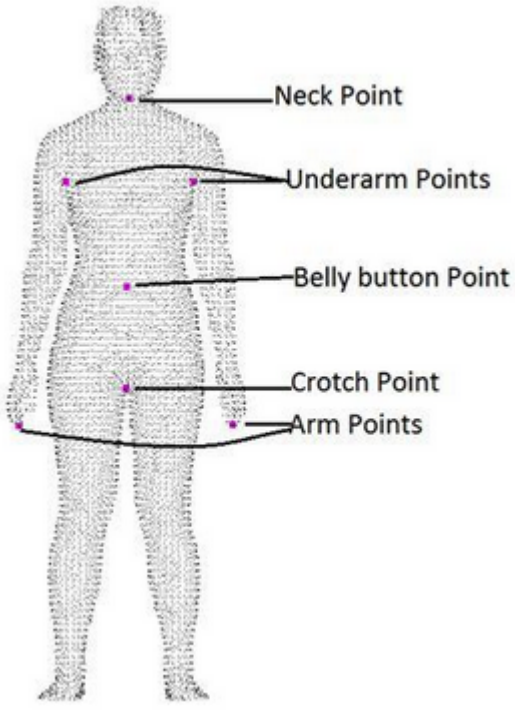

Рис. 3: Иллюстрация расположения особых точек на трехмерной модели человека для параметризации [7]

2.3. Методы оптимизации

2.3.1. Градиентный спуск

Пусть выделена область модели, вершины которой заданы координатами x в трехмерном пространстве. Задана некоторая функция f($\overline{x}$). Это может быть разность площадей, объемов, сумм длин ребер и так далее исходного значения модели и желаемого.

Задачей градиентного спуска является оптимизация функции в направлении скорейшего спуска, которое задается антиградиентом, то есть градиентом со знаком минус: $\overline{x}^{[j+1]} = \overline{x}^{[j]} - \lambda^{[j]} \nabla f(\overline{x}^{[j]})$. Где $\lambda^{[j]}$ может быть:

- константой,
- дробным шагом (длина шага делится на фиксированное число),
- наискорейшим спуском: $\lambda^{[j]} = argmin_{\lambda} f(\overline{x}^{[j+1]})$

Шаги алгоритма:

- Задается начальное значение и точность расчёта ε.
- Вычисляется $\overline{x}^{[j+1]} = \overline{x}^{[j]} - \lambda^{[j]} \nabla f(\overline{x}^{[j]})$, $\lambda^{[j]} = argmin_{\lambda} f(\overline{x}^{[j+1]})$
- Проверяется критерий останова: $|\overline{x}^{[j+1]} - \overline{x}^{[j]}| \leq \varepsilon$.

Данный метод решает задачу оптимизации итеративно и имеет свой критерий сходимости, который задается в зависимости от выбранной функции. Шаг градиентного спуска может быть фиксированным или изменяющимся, например, длина шага может делиться на какое-то фиксированное число в процессе работы самого метода. Если шаг выбран неверно, то градиентный спуск может не сойтись. Подробнее о данном алгоритме можно прочитать в [13]. На рисунке 4 представлена геометрическая интерпретация данного метода, где на каждом шаге происходит сдвиг по вектору антиградиента, уменьшенному в фиксированное число раз.

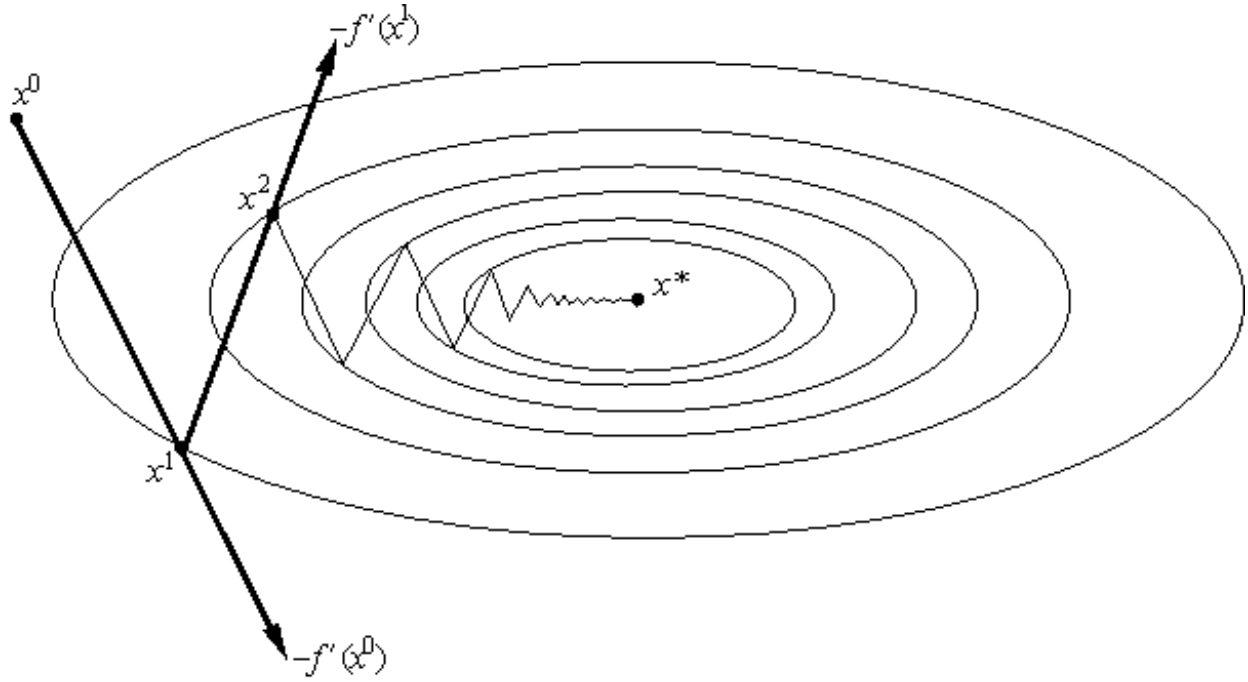

Рис. 4: Иллюстрация метода градиентного спуска [13]

2.3.2. Покоординатный спуск

Данный метод улучшает предыдущий метод за счёт того, что на очередной итерации спуск осуществляется постепенно вдоль каждой из координат, однако теперь необходимо вычислять новые λ n раз за один шаг, где n - размер вектора. Если шаг спуска выбрать неправильно, то данный метод может не сойтись и критерий останова не будет достигнут.

2.4. Инструменты

Существующие инструменты планирования хирургических операций в большинстве являются коммерческими продуктами. Среди них можно выделить Axis Three[3]. Данное программное средство работает с трехмерными моделями области лица и груди. В Canfield[4] представлены решения для задач анализа 3D скана и симуляции результатов различных операций на лице и теле. ANSYS[2] и ABAQUS[1] - коммерческие инструменты физического моделирования, симуляция которых основана на методе конечных элементов. Предоставляют возможность

обмена моделями и совместной работы над симуляцией. Эти продукты разработаны для проектирования различной техники, анализа прочности конструкций. Также их используют для симуляции челюстно-лицевых операций, моделируют инструменты хирургов.

Среди инструментов с открытым исходным кодом для моделирования поведения мягких тел в задачах медицины можно выделить SOFA[12]. Данный инструмент написан на языке C++ и поддерживает такие физические модели как: модель масс-с-пружинами, модель конечных элементов, SPH. FEBio[6], основан на методе конечных элементов и является свободным для некоммерческого использования.

3. Архитетура системы

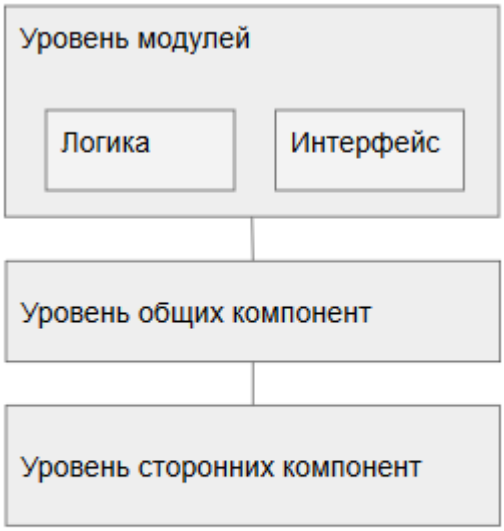

Рис. 5: Архитектура системы

Библиотека деформатором разрабатывается для системы планирования хирургических операций Phoenixcas 3D Viewer. На рисунке 5 представлена архитектура системы. Данный программный продукт разделен на три уровня: уровень модулей, уровень общих компонент и уровень сторонних компонент [14].

Модули отвечают за реализацию симуляции результатов операции с подключением ресурсов из других уровней, решают конкретные пользовательские задачи и не связаны между собой. Их также можно разделить на логику модуля и пользовательский интерфейс.

Уровень общих компонент нужен для подключения различных библиотек алгоритмов, структур данных, которые будут доступны для всех модулей.

Уровень сторонних компонент нужен для подключения различных сторонних библиотек, средств визуализации. Этот уровень закрыт от прямого доступа уровню модулей, работа с ним осуществляется на уровне общих компонентов.

Библиотека деформаторов реализована как новый класс статических функций и упакована в .lib файл, который подключается к модулям через уровень общих компонентов. Работа с алгоритмами библиотеки ведется через открытый интерфейс класса.

Система Phoenixcas 3D Viewer реализована на языка C++, поэтому алгоритмы библиотеки написаны также на языке C++. Используются некоторые структуры движка Ogre3d. Разработка велась в Visual Studio 2010.

4. Функциональность

Поскольку результатом работы алгоритмов является деформация существующей 3D модели, во входных параметрах у каждой функции присутствует трехмерная модель поверхности, которая представляется полигональной сеткой, записываемой как множество вершин и множество индексов треугольников, ссылающихся на вершины. Также необходимо передать в функцию коэффициент деформации k: $0 < k \leq 1$ (при k = 1 исходная поверхность не меняется). На данный момент в библиотеку включены такие алгоритмы деформации, реализаванные в рамках данной работы, как:

- posBasedDynGradientDescent(vertices, edges, k) - изменяет координаты точек из вектора vertices в соответствии с коэффициентом деформации k, методом градиентного спуска с ограничением на длину каждого ребра.

- sumEdgeGradientDescent(vertices, edges, k) - изменяет координаты точек из вектора vertices в соответствии с коэффициентом деформации k методом градиентного спуска с ограничением на сумму длин ребер

- subAreaGradientDescent(vertices, triangles, k) - изменяет координаты точек из вектора vertices в соответствии с коэффициентом деформации k методом покоординатного спуска с ограничением на разницу площадей.

5. Реализация

Поверхность трехмерной модели задана множеством вершин и множеством индексов треугольников, которые на эти вершины ссылаются. Однако в функциях с ограничением на длины ребер, например, также удобно заранее иметь информацию о ребрах модели, а не индексы треугольников. Для этого хранится структура, которая содержит 2 индекса вершин, которые это ребро связывает. Аналогичная структура была создана и для треугольников, вместо двух индексов теперь в ней хранится их тройка. Заполняются эти структуры при помощи средств Ogre3d. Вершины поверхности модели передаются по ссылке и изменяются в ходе работы алгоритма. Остальные входные параметры (вектора ребер и треугольников) в ходе работы программы не изменяются, ведь в структурах идет привязка именно к индексам вершин, а не к их координатам. Также стоит отметить, что в виду того, что границы области поверхности не должы быть сдвинуты, следует хранить еще один массив, заполненный 0 и 1. В нем единицами помечены вершины, которые можно сдвигать, а нулями - те самые вершины границ, координаты которых не должны быть изменены. Средством для визуализации является Ogre3d. Далее рассматривается реализация алгоритмов градиентного спуска с ограничениями.

5.1. Градиентный спуск с ограничением на длину каждого ребра

Суть метода состоит в том, чтобы итеративно применять поправку в соответствии с коэффициентом деформации k на длину каждого ребра поверхности трехмерной модели до возникновения критерия останова спуска. Далее подробнее рассматривается функция для минимизации спуском.

Рассмотрим применение ограничения на ребро, соединяющее вершины p_1 и p_2, рисунок 6. Для этого сначала для ребра подсчитывается константа d, которая не будет изменяется от итерации к итерации ал-

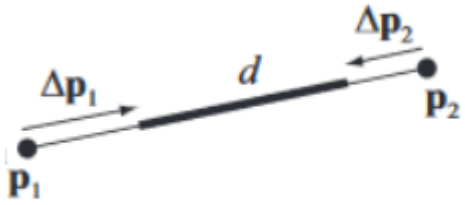

Рис. 6: Иллюстрация применения ограничения на ребро [8]

горитма. Она равна длине ребра, умноженной на коэффициент деформации:

$$d = |p_1 - p_2| * k \tag{1}$$

Далее производится нормализация вектора, направленного по ребру в сторону вершины p_2:

$$n = \frac{p_1 - p_2}{|p_1 - p_2|} \tag{2}$$

Теперь вычисление самого ограничения на ребро p_1p_2 на данной итерации алгоритма:

$$c = |p_1 - p_2| - d \tag{3}$$

И, наконец, подсчет сдвига для p_1 и p_2:

$$\begin{aligned} \Delta p_1 &= -\tfrac{1}{2} * n * c \\ \Delta p_2 &= +\tfrac{1}{2} * n * c \end{aligned} \tag{4}$$

Идеи данного метода описываются в [8]. В алгоритме из статьи также используются веса вершин, в данном случае они берутся равными единице. Отсюда появляется множитель $\frac{1}{2}$ в формуле градиента. Новые координаты вершин p_1 и p_2 будут получены путем прибавления соответствующего градиента к их старым координатам. Для одной итерации спуска подобные вычисления нужно провести для каждого ребра поверхности трехмерной модели.

Следует подсчитать ошибку, которая равна сумме длин сдвигов. Через какое-то число итераций ошибка будет или не уменьшаться, или достаточно мала, чтобы остальное считать погрешностью. Это условие является критерием останова. Также следует отметить, что константа

d для всех ребер поверхности модели считается 1 раз перед самим спуском и записывается в массив D, ограничения же пересчитываются на каждой итерации алгоритма.

5.2. Градиентный спуск с ограничением на сумму длин ребер

Целевой функцией в данном методе выбрана разница сумм длин ребер с требуемым значением. Основное отличие данного метода от предыдущего заключается в том, что ограничение применяется не к каждому ребру по отдельности, а сразу к группе смежных с текущей вершиной ребер, рисунок 7. Подробнее алгоритм приведен в псевдокоде 1. Фиксируется вершина p_0 в массиве вершин поверхности моде-

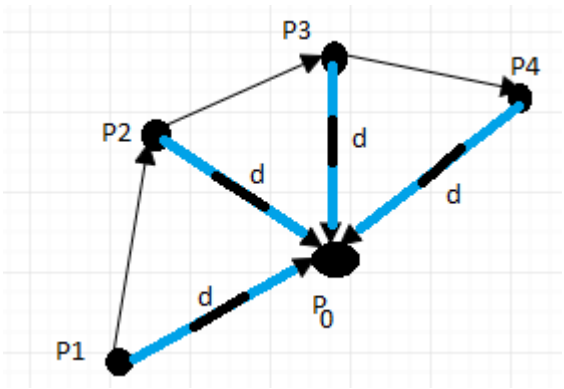

Рис. 7: Применение ограничения к ребрам, смежным с вершиной p_0

ли. Берется множество смежных с p_0 вершин, и для каждой из них подсчитывается n:

$$n = \sum_{i=1}^{edgesCount} \frac{p_0 - p_i}{|p_0 - p_i|} \tag{5}$$

Теперь константой d будет сумма длин всех ребер поверхности трехмерной модели:

$$d = \sum_{i=1}^{edgesCount} |p_0 - p_i| * k \tag{6}$$

```
Data: vertices, edges, k
Result: new vertices
ε = 1-e5;
sum = sum of edges lengths;
d = k * sum;
foreach v in vertices do
|  adjCount[v] = number of adjacent verticies;
end
while error > ε do
|  newSum = sum of new edges lengths;
|  c = newSum - d;
|  foreach (p_0 in vertices do
|  |  foreach p_1 in adjacent to v vertices do
|  |  |  vector = p_0 - p_1;
|  |  |  n[p_0] += vector / vector.length();
|  |  |  n[p_1] -= vector / vector.length();
|  |  end
|  |  if adjCount[p_0] != 0 then
|  |  |  grad = n[p_0] * C / adjCount[p_0];
|  |  |  shifts[p_0] -= grad;
|  |  |  error += grad;
|  |  end
|  end
end
apply shifts;
```

Algorithm 1: Псевдокод алгоритма

Соответственно ограничением станет разность сумм длин ребер новой модели на данной итерации и d:

$$c = \sum_{i=1}^{edgesCount} |p_0 - p_i| - d \tag{7}$$

В итоговой формуле для сдвигов появится деление не на двойку, как в предыдущей версии алгоритма, а на количество смежных с p_0 вершин. Согласно обобщенной формуле из [8] итоговый вид формулы для сдвигов будет таким:

$$\begin{aligned} \Delta p_i &= -\frac{1}{adjacentCount} * n * c \\ \Delta p_j &= +\frac{1}{adjacentCount} * n * c \end{aligned} \tag{8}$$

5.3. Покоординатный спуск с ограничением на разницу площадей

Data: vertices, edges, k
Result: new vertices
ε = 1-e8;
targetArea = k * originalArea;
repeat
  c = currentArea - targetArea;
  foreach *(p_0,p_1) in edges* **do**
    shifts[p_0] += (p_0 - p_1) * c;
    shifts[p_1] -= (p_0 - p_1) * c;
  end
  foreach *v in vertices* **do**
    error += $|shifts[v]|^2$;
  end
  apply shifts;
  newC = currentArea - targetArea;
  while *newC > c - error * ε* **do**
    apply shifts;
    newC = currentArea - targetArea;
  end
until *error <= ε*;

Algorithm 2: Псевдокод алгоритма

Метод покоординатного спуска с ограничением на разность площадей представлен в псевдокоде 2. В данном случае функцией для оптимизации в направлении скорейшего спуска является разница площадей исходной поверхности трехмерной модели и площади поверхности, домноженной на коэффициент $0 < k < 1$. В предыдущих двух случаях ошибкой являлась сумма длин сдвигов. Теперь она берется в квадрате. Ошибка будет уменьшаться или станет пренебрежимо малой, что станет критерием останова спуска. Идеи данного метода почерпнуты из [13]. Для каждой вершины подсчитывается градиент, после чего вершины сдвигаются до тех пор, пока новое значение разности площадей не будет примерно равняться старому за вычетом ошибки, умноженной на ε.

6. Апробация

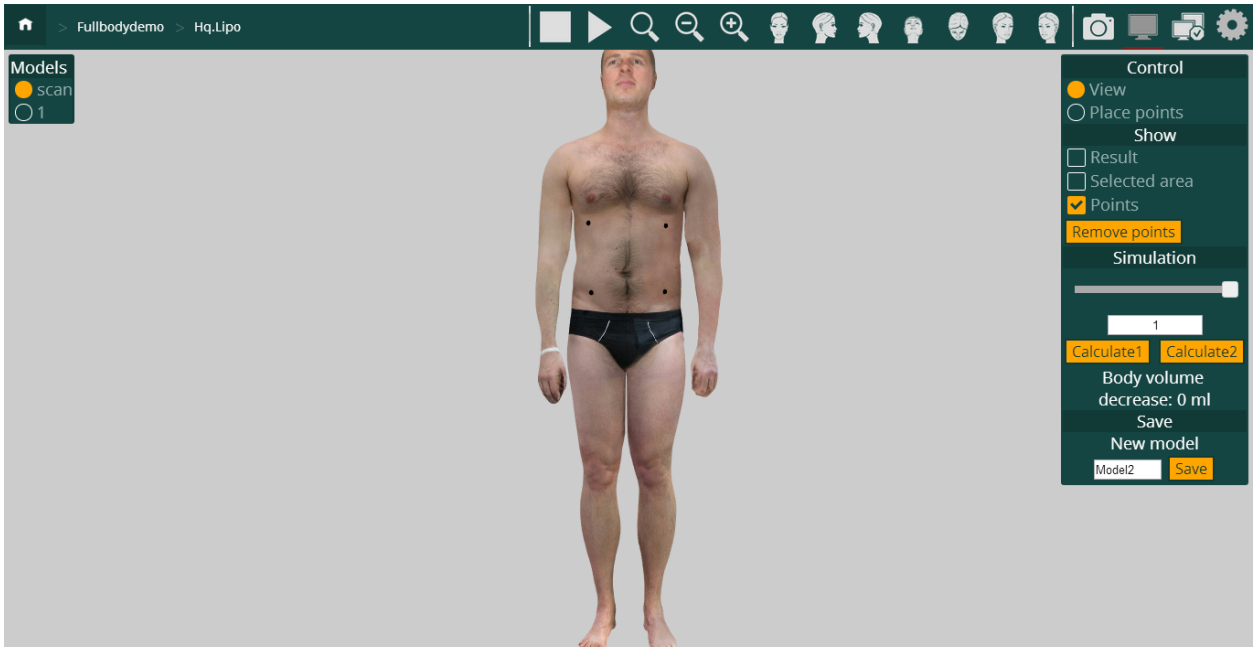

Рис. 8: Модуль

Изначально алгоритмы библиотеки разрабатывались и тестировались в отдельном репозитории деформаторов, в основном солюшене было много внутренних зависимостей. К тому же в репозитории деформаторов уже были реализованы некоторые удобные интерфейсы для организации тестирования и визуализации разрабатываемых алгоритмов.

Для апробации библиотеки было решено реализовать еще один модуль в системе Phoenixcas 3D Viewer, в котором предполагалось применить библиотечные алгоритмы к трехмерной модели поверхности тела человека. Интерфейс модуля представлен на рисунке 8.

В модуле сначала выбирается нужная трехмерная модель. Затем мышью понадобится отметить точки на выбранной модели (их должно быть не менее трех), сдвинуть ползунок, отвечающий за коэффициент деформации. Далее происходит обрезание участка модели в соответствии с отмеченными точками. После этого выполняется вызов деформирующей поверхность модели функции из библиотеки, происходят расчёты симуляции. В результате в коллекцию добавляется но-

вая модель, которая является результатом симуляции, она визуализируется. Библиотека показала свою работоспособность в разработанном модуле, результаты визуализации можно увидеть на рисунке 9.

Таким образом были получены наглядные результаты работы алгоритмов библиотеки на трехмерной модели человека. Алгоритмы внедрены в библиотеку деформаторов, а библиотека подключена к системе планирования хирургических операций Phoenixcas 3D Viewer. Реализован модуль симуляции операции с использованием этой библиотеки.

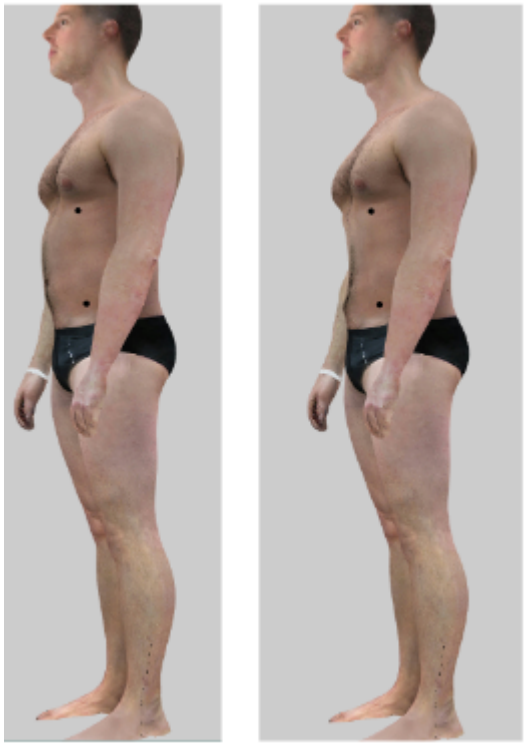

Рис. 9: Иллюстрация применения метода градиентного спуска с ограничениями из библиотеки

7. Результаты

Разработана библиотека алгоритмов деформации 3D модели мягкого тела в рамках компьютерной системы планирования хирургических операций Phoenixcas 3D Viewer.
В ходе работы были достигнуты следующие результаты.

1. Проведен обзор методов деформации мягких тел.
2. Разработана и реализована архитектура библиотеки.
3. Реализованы алгоритмы деформации мягкого тела.
4. Проведена апробация библиотеки.

Список литературы

[1] ABAQUS. — URL: ''https://www.3ds.com/ru/produkty-i-uslugi/simulia/produkty/abaqus/''.

[2] ANSYS. — URL: ''http://www.ansys.com/''.

[3] Axis Three. — URL: ''http://www.axisthree.com/''.

[4] Canfield. — URL: ''http://www.canfieldsci.com/''.

[5] Delingette H. Ayache N. Soft Tissue Modeling for Surgery Simulation. — 2004. — P. 29–40.

[6] FEbio. — URL: ''https://febio.org/''.

[7] M. Kasap. Parameterized human body model for real-time applications. — 2007.

[8] Müller M. Heidelberger B. Hennix M. Ratcliff J. Position Based Dynamics. — 2006.

[9] Müller M. Schirm S. Teschner M. Interactive blood simulation for virtual surgery based on smoothed particle hydrodynamics. — 2003.

[10] Nealen A. Müller M. Keiser R. Boxerman E. Carlson M. Physically Based Deformable Models in Computer Graphics. — 2005.

[11] Phoenixcas 3D. — URL: ''http://www.phoenixcas.com/''.

[12] SOFA. — URL: ''https://www.sofa-framework.org/''.

[13] Utexas.edu. Algorithms for Constrained Optimization. — URL: ''https://www.me.utexas.edu/~jensen/ORMM/supplements/units/nlp_methods/const_opt.pdf''.

[14] А. Монькин. Модуль расчета деформации трехмерной модели мягкого тела для компьютерной системы планирования хирургических операций. — 2016.