Моделирование изгибов бумаги

Позднякова Алиса Б05-924

29 декабря 2022 г.

1 Описание проекта

Цель проекта: Написать библиотеку, с помощью которой возможно получить 3d mesh файл, содержащий представление изогнутой бумаги.

Получение представления:

Для каждого искривленного региона бумаги выполним следующие пункты, после чего объединим полученные 3d mesh в финальную:

(1) Построение геодезической кривой
Реализованный подход к моделированию изгиба бумаги основан на том, что развертывающаяся поверхность однозначно определяется геодезической кривой на ней.

Def. Геодезическая кривая - кривая на поверхности, такая, что на развертывающейся поверхности она представляет собой прямую линию.

Геодезическую кривую можно представить с помощью кривой Безье. В проекте использовались кривые Безье 3 степени.
Точки на кривой добавляются до тех пор, пока кривая не станет гладкой (все углы не меньше некоторого *epsilon*).

(2) Вычисление вершин и граней 3d mesh
По точкам геодезической кривой построим прямые. Назовем их правилами. По свойствам бумаги при изгибе сохраняются углы и расстояния. Значит по соответствующим им прямым на развертке можно вычислить длины сторон граней-четырехугольников.
Таким образом для каждой геодезической кривой получаем изогнутый регион. Ограничением в принимаемых данных является условие, что искривленные регионы не пересекаются (функциональность рассчета 3d mesh в точках сингулярности не реализована)

2 Зависимости и сборка проекта

Код: `https://github.com/alicepozd/crumpled_paper`
В этом же репозитории есть doxygen документация
Чтобы начать использовать модуль:

(1) Зависимости:
Проект написан с использованием библиотеки 3d визуализации libigl [4].

Для запуска проекта необходимо установить её, а так же ее зависимости (библиотека Eigen [5] для работы с векторами и матрицами).
Необходимо создать директорию crumpled_paper/lib и положить в нее библиотеки.

(2) Сборка проекта:
Соберите библиотеку papper_mesh:

```
doxygen
mkdir build
cd build
cmake ./path_to_project
make
```

3 API библиотеки:

(1) Структура Point:
Является представлением 3d точки.

Имеет поля:

```
double x;
double y;
double z;
```

И следующие функции:

```
Point(double x, double y, double z)
```

Конструктор точки по набору координат.

```
Point(const Point& another)
```

Конструктор точки по другой.

```
Point get_mid(const Point& another, double t = 0.5) const;
```

Функция, возвращающая точку, делящую отрезок в отношении t.

```
double get_dist(const Point& another) const;
```

Функция, находящая расстояние между точками.

```
double get_angle(const Point& first, const Point& second) const;
```

Функция находит угол, между лучами, проходящими через точки first и second.

(2) Структура Line:
Структура, реализующая прямую $ax + by + c = 0$.
Имеет поля:

```
double a;
double b;
double c;
```

И следующие функции:

```
Line(std::pair<double, double> p, std::pair<double, double> q)
```

Конструктор прямой по двум точкам.

```
Line(double a, double b, double c)
```

Конструктор прямой по ее коэффициентам.

```
std::pair<double, double> intersection(Line another) const
```

Возвращает точку пересечение прямых.

```
Line perpendicular(std::pair<double, double> p) const
```

Возвращает прямую, проходящую через точку p и являющуюся перпендикуляром.

```
std::pair<double, double> get_point_on_dist(const std::pair<double, double>& from,
                const std::pair<double, double>& to, double dist)
```

Параметры:
Point from - точка начала отсчета
Point to - точка к которой направлен вектор
double dist - расстояние

Находит точку на прямой, отстоящую от точки from на расстояние dist по направлению к точке to.

(3) Класс Geodesic_line:
Класс реализует геодезическую кривую, как кривую Безье.

Содержит следующие публичные функции:

```
Geodesic_line(Point begin, std::vector<Point>& main_tangent, Point end_side, double dist, double lenth, double theta = 175);
```

Point begin точка, в которой начинается кривая
std::vector<Point> main_tangent вектор из 2 точек, задающий касательные к бумаге в начале и конце геодезической кривой
std::vector<Point> end_side точка, задающая направление, в котором будет конец геодезической кривой
double dist расстояние на котором будет конец геодезической кривой
double lenth длина геодезической кривой
double theta угол, при достижении которого считаем бумагу гладкой

Рассчитывает точки геодезической кривой

```
std::vector<Point> get_geodesic_curve() const
```

Функция возвращает геодезическую кривую.

(4) Класс CurvedRegion:
Класс, реализующий представление региона бумаги в виде 3d mesh.

Содержит следующие публичные функции:

```
CurvedRegion(std::vector<Point>& start_region, std::vector<Point>& flat_geodesic,
        std::vector<Point>& flat_left_border, std::vector<Point>& flat_right_border,
                Point start, std::vector<Point>& main_tangent, Point end_side, double dist);
```

Параметры:
std::vector<Point> start_region вектор из 2 точек, край начала региона
std::vector<Point> flat_geodesic вектор из 2 точек, геодезическая кривая на развертке
std::vector<Point> flat_left_border вектор из 2 точек, левый край бумаги относительно геодезической кривой на развертке
std::vector<Point> flat_right_border вектор из 2 точек, правый край бумаги относительно геодезической кривой на развертке
Point start, точка начала геодезической кривой std::vector<Point> main_tangent вектор из 2 точек, задающий касательные к бумаге в начале и конце геодезической кривой
Point end_side точка, задающая направление, в котором будет конец геодезической кривой
double dist расстояние на котором будет конец геодезической кривой

Возвращает вектор из трех векторов: геодезическую кривую и соответствующие ее точкам точки на левой и правой границе бумаги

```
std::vector<std::vector<Point>> get_curved_region();
```

Возвращает вектор.

(4) Класс Paper:
Класс реализует алгоритм работы с 3d mesh представлением бумаги.
Содержит следующие публичные функции:

```
Paper(std::string& file_name);
```

Параметры:
std::string& file_name имя файла с заданными параметрами бумаги

```
void get_file(std::string& file_name);
```

Получает на вход file_name строку с названием, создает файл типа .obj с 3d mesh представлением бумаги.

4 Использование библиотеки:

Для создания 3d mesh файла необходимо выполнить следующие шаги.

(1) Создание файла, подаваемого на вход:

Формат файла:

Первая строка содержит ширину бумаги, деленную на 2 и высоту. Далее пустая строка.

В следующих строках задаются искривленные регионы. Параметры через разделитель |. Необходимые параметры:

Флаг: 1, если регион плоский, 0 иначе.
Точка левой границы на конце региона.
Точка геодезической кривой на конце региона.
Точка правой границы на конце региона.

В случае, если регион не плоский дополнительно:

Точка, задающая направление касательной в точке геодезической кривой (нормы 1).
Точка, задающая направление следующей точки на границе регионов (нормы 1).
Расстояние до этой точки.
Пример:

```
1000 7000

0 | -1000 2000 0 | 0 4000 0 | 1000 6000 0 | 0 1.41 -1.41 | 0 0 1 | 3000
0 | -1000 7000 0 | 0 7000 0 | 1000 7000 0 | 0 1.41 -1.41 | 0 0 1 | 2000
0 | -1000 8000 0 | 0 8000 0 | 1000 8000 0 | 0 1 0 | 0 0 1 | 900
0 | -1000 10000 0 | 0 9000 0 | 1000 8000 0 | 0 1 0 | 0 0 1 | 500
```

(2) Создание объекта класса Paper

(3) Создание 3d mesh:

С помощью функции get_file класса Paper.

(4) Интерпритация результатов:

Из Papper Mesh получаем obj файл с 3d mesh. Полученный файл можно открыть в стороннем редакторе. Например `https://3dviewer.net/`.

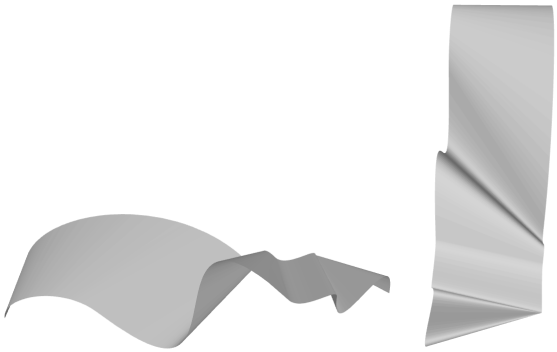
5 Примеры использования

В директории crumpled_papper/use_examples лежат файлы, с примерами работы алгоритма с разными параметрами. При выполнении сборки проекта в папке $your_path/source появится исполняемый файл. Запустив его из директории, в которой он находится, можно получить файлы со следующими примерами.

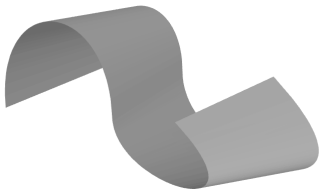
Первый пример использования:
different_parameter_example - демонстрирует использование.

Разные ракурсы:

Второй пример использования:
Разные ракурсы:



Третий пример использования:
Разные ракурсы:



6 Ссылки

[1] Camille Schreck. Interactive deformation of virtual paper. Modeling and Simulation. Université Grenoble Alpes, 2016

[2] D. Cohen-Or and P. Slavík. Geodesic-Controlled Developable Paper Bending. Surfaces for Modeling. EUROGRAPHICS 2007

[3] ZHANG Xing-wang, WANG Guo-jin. A new algorithm for designing developable Bézier surfaces. SCIENCE A 2006

[4] Библиотека libigl: `https://libigl.github.io/`

[5] Библиотека Eigen: `https://eigen.tuxfamily.org/index.php?title=Main_Page`