|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **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»

**Лабораторная работа № 7**

|  |  |
| --- | --- |
| **Тема** Реализация алгоритма отсечения отрезка произвольным выпуклым отсекателем (Алгоритм Кируса-Бека)  **Студент** Якуба Д. В.  **Группа** ИУ7-43  **Оценка (баллы) \_\_\_\_\_\_\_\_\_\_\_**  **Преподаватель** Куров А. В. |  |

Москва

2020 г.

Оглавление

[Цель работы 2](#_Toc41131398)

[Техническое задание 3](#_Toc41131399)

[Теоретическая часть 3](#_Toc41131400)

[Практическая часть 3](#_Toc41131401)

[Программная реализация алгоритма на ЯП Python 3](#_Toc41131402)

[Пользовательский интерфейс 3](#_Toc41131403)

[Демонстрация работы алгоритма 3](#_Toc41131404)

# Цель работы

Изучение и программная реализация алгоритма отсечения отрезка.

# Техническое задание

Алгоритм отсечения отрезка выпуклым отсекателем – алгоритм Кируса-Бека.

Необходимо обеспечить ввод отсекателя – произвольного многоугольника. Высветить его первым цветом. Также необходимо обеспечить ввод нескольких (до десяти) различных отрезков (высветить их вторым цветом). Отрезки могут иметь произвольное расположение – горизонтальные, вертикальные, имеющие произвольный наклон.

Предусмотреть ввод отрезков, параллельных границе отсекателя.

Ввод осуществлять с помощью мыши и нажатия других клавиш.

Выполнить отсечение отрезков, показав результат третьим цветом. Исходные отрезки не удалять.

# Теоретическая часть

## Отсечение отрезка произвольным выпуклым отсекателем

Поставленная задача отсечения в данном алгоритме, по сути, решается с использованием двух скалярных произведений, в которых задействован вектор нормали.

Легко заметить, что, в случае отсечения отрезка нерегулярным отсекателем, использование кодов концов отрезков (которые мы задействовали в рассмотрении предыдущего алгоритма отсечения отрезка регулярным отсекателем) невозможно. Данный факт вынуждает нас искать точки пересечения со сторонами отсекателя (или с их продолжениями). Количество точек пересечения будет равняться количеству сторон заданного многоугольного отсекателя, причём сам отрезок может пересекать многоугольник лишь в двух точках, которые нам и предстоит выбрать.

Таким образом, задача сводится к нахождению точек пересечения заданного отрезка с границами отсекателя и правильному определению из всех найденных точек пересечения вершины начала видимого отрезка и вершины конца видимого отрезка. В дополнении также потребуется определить полную невидимость отрезка. При этом стоит отметить, что определить полную видимость отрезка также не является некоторой тривиальной задачей, поэтому для тоже понадобится выполнить полный цикл нахождения точек пересечения. К моменту окончания работы алгоритма неизменным, что и будет означать полную видимость рассматриваемого отрезка.

Для идентификации полностью видимых или полностью невидимых отрезков удобно использовать параметрическую форму задания отрезка:

Рассмотрим следующий случай:



Рисунок , случай 1, полная видимость отрезка

В данном случае будем иметь:

– для нижней и левой границ.

– для верхней и правой границ.

Таким образом, для полностью видимого отрезка каждой точке пересечения соответствует недопустимое значение параметра.

Рассмотрим иной случай:



Рисунок , случай 2, полная невидимость отрезков

В данном случае точкам пересечения отрезков также соответствуют недопустимые значения параметра.

Получается, что простых способов определения полностью видимых или полностью невидимых отрезков предложено быть не может. То есть полностью видимые и полностью невидимые отрезки будут распознаваться уже по ходу работы алгоритма.

#### Видимость и невидимость отдельно взятой точки

В процессе отсечения, как правило, требуется определять видимость или невидимость отдельно взятой точки. Рассмотрим следующий случай:



Рисунок , рассмотрение видимости точки

Задача состоит в определении видимости точки *A* относительно каждой из сторон отсекателя.

Рассмотрим ребро :

Построим нормаль к рассматриваемой стороне из произвольной её точки:



В качестве второго вектора построим вектор, начинающийся в произвольной точке рассматриваемого ребра и заканчивающийся в рассматриваемой точке:

# Практическая часть

## Программная реализация алгоритма на ЯП Python

## Пользовательский интерфейс

## Демонстрация работы алгоритма