**МГТУ им. Н.Э. Баумана**

**КОМПЬЮТЕРНАЯ ГРАФИКА**

**Лабораторный практикум №8**

**по теме: «***Реализация алгоритма отсечения отрезка произвольным выпуклым отсекателем.*

*(Алгоритм Кируса-Бека)***»**

***Студент: Нгуен Фыок Санг***

***Группa: ИУ7И-46Б***

***Работу проверил:***

2020

**Цель работы: изучение и программная реализация алгоритма отсечения отрезка.**

1. Алгоритм Кируса-Бека:

* Входные данные:
  + Отсекатель – произвольного многоугольник
  + два конца отрезка: P1(X1, Y1), P2(X2, Y2)
* Выход:
  + R1, R2: R1R2 - это часть отрезки P1P2, расположенная полностью внутри откатателя

Параметрическое уравнение отрезка P1P2 имеет вид:

Предположим, что f - граничная точка выпуклой области С, а nв - внутренняя нормаль к одному из ограничивающих эту область ребер. Тогда для любой точки отрезка AB, то есть для любого значения параметра t:

* из условия nв [P(t) - f] < 0 следует, что вектор[P(t) - f] направлен из области C
* из условия nв [P(t) - f] = 0 следует, что вектор [P(t) - f] принадлежит плоскости, которая проходит через граничную точку k и перпендикулярна нормали nв
* из условия nв [P(t) - f] > 0 следует, что вектор [P(t) - f] направлен внутрь
* Из всех этих условий следует, что прямая пересекает замкнутую выпуклую область C равно в двух точках.
* Если точка f лежит на той граничной плоскости или на том ребре, для которых nв является внутренней нормалью , то точка на отрезке P(t) , которая удовлетворяет уравнению nв [ P(t) - f] = 0 , будет точкой пересечения этого отрезка с ребром окна отсечения.
* Уравнение для определения точки пересечения отрезка с ребром полигонального выпуклого окна отсечения:

nвi [P1 + (P2 – P1)t – fi] = 0 (2)

nвi [P1 – fi] + nвi [P2-P1] t = 0.

D = P2-P1

Wi = P1-fi

t ( nвi D ) + Wi nвi = 0

t = - ( Wi nвi) /( D nвi), D ≠ 0, i=1,2..m (3)

1. в случае вырождения отрезка в точку (D = 0), то есть когда P2=P1
   1. вне окна отсечения, когда числитель Wi nвi < 0
   2. на границе, когда числитель Wi nвi = 0
   3. внутри, когда числитель Wi nвi > 0
2. скалярное произведение D nвi = 0, т.е. когда отрезок параллелен ребру отсекателя

* Используя уравнение (3), можно определить значения параметра t, при которых происходят пересечения исследуемого отрезка с ребрами окна отсечения.
  + Если значения t ∉ [0;1], то их не рассматривают, поскольку они соответствуют точкам, лежащим вне исходного отрезка.
  + Найденное значение параметра t для очередной точки пересечения рассматривают в качестве
    - D nвi < 0, возможного верхнего предела tв,
    - D nвi > 0, относят к нижней границе видимости tн
  + Полученные решения следует разбить на две группы: верхнюю и нижнюю, в зависимости от близости найденной точки пересечения к началу или концу отрезка.
  + Чтобы отрезок был видимым относительно всего отсекателя, он должен быть видим относительно всех ребер отсекателя одновременно.
* Концам видимой части отрезка будут соответствовать два значения параметра t, максимальнoe значение из нижней группы и минимальнo из верхней группы
* Алгоритм Кируса - Бека

1. Ввод координат концевых точек отрезка P1(P1.x, P1.y) , P2(P2.x, P2.y)

2. Ввод числа сторон m выпуклого многоугольника - окна отсечения и координат его вершин (массив C).

3. Вычисление вектора ориентации отрезка D = S2 - S1.

4. Инициализация пределов значений параметра t при условии, что отрезок полностью видимый, то есть tн = 0, tв = 1.

5. Начало цикла по всем сторонам отсекающего окна (i = 1..m). Выполнение для каждой i-ой стороны окна следующих действий:

5.1. Вычисление вектора внутренней нормали к очередной i-ой стороне окна отсечения - nвi

5.2. Определение граничной точки fi каждой стороны отсекающего окна (точки, лежащей на i-ой стороне окна).

5.3. Вычисление вектора Wi = P1 - fi .

5.4 Вычисление скалярного произведения векторов Wск i= Wi nвi .

5.5 .Вычисление скалярного произведения векторов Dск i= Dnвi .

5.6. Проверка на равенство нулю скалярного произведения Dск (вырождение отрезка в точку или его параллельность стороне отсекателя). Если Dnвi =0 , тогда переход к пункту 5.9.

5.7. Вычисление параметра t: t = - ( Wскi / Dскi) (отрезок не вырожден в точку и не параллелен стороне отсекателя).

5.8. Определение верхнего и нижнего пределов параметра t :

5.8.1. Поиск нижней границы параметра t, если Dскi > 0 :

если t>1, то переход к п.7 (отрезок невидим, т.к. нижний предел параметра t превышает единицу и пересечение с отсекателем имеет место не для самого отрезка, а для его продолжения за вершину P2.;

если t ≤1, то tн = max ( t, tн ) (выбор максимального значения из текущего значения параметра t и ранее вычисленного значения нижней границы параметра).

5.8.2. Поиск верхней границы параметра t, если Dскi < 0 :

если t < 0, топереход к п.7 (отрезок невидим, т.к. верхний предел параметра t отрицателен и пересечение с отсекателем имеет место не для самого отрезка, а для его продолжения за вершину P1.;

если t ≥0, то tв = min ( t, tв ) (выбор минимального значения из текущего значения параметра t и ранее вычисленного значения верхней границы параметра)

5.9. Проверка видимости точки, в которую выродился отрезок, или проверка видимости произвольной точки отрезка в случае его параллельности стороне отсекателя:

если Wскi < 0, то отрезок (точка) невидим и переход к п. 7;

если Wскi >0, то отрезок (точка) видим относительно текущей стороне отсекателя и переход к п. 5.10.

5.10. Конец цикла по сторонам отсекателя.

6. Проверка фактической видимости отсеченного отрезка. Если tн > tв, то переход к пункту 7 , иначе визуализация отрезка в интервале от P(tн) до P(tв).

7. Конец алгоритма.

* Определение факта выпуклости многоугольника. Вычисление внутренних нормалей
* Если знаки всех векторных произведений смежных сторон многоугольника равны нулю, то многоугольник вырождается в отрезок. Если есть, как положительные так и отрицательные знаки, то многоугольник невыпуклый. Если все знаки векторных произведений смежных сторон неотрицательные, то отсекающий многоугольник выпуклый, а внутренние нормали ориентированы влево от его контура. Если же все знаки неположительны, то многоугольник также является выпуклым, а внутренние нормали ориентированы вправо от его контура
* Если вектор стороны многоугольника образован как разность векторов пары смежных его вершин Ai-1 и Ai и если скалярное произведение нормали n и вектора от Ai-1 до Ai+1 положительно, то n - внутренняя нормаль. В противном случае n - внешняя нормаль. Для получения внутренней нормали полученное значение n надо умножить на -1.

def clipping(win):

fl = isPolygonConvex(win.edges)

if fl == 0:

QMessageBox.warning(win, "Error", "Input Polygon is not convex")

return

for b in win.lines:

win.pen.setColor(blue)

cyrus\_beck\_lineClipping(b, win.edges, fl, win.scene, win.pen)

win.pen.setColor(red)

def cyrus\_beck\_lineClipping(line, edges, n, scene, pen):

# Инициализация пределов значений параметра t

tb = 0

te = 1

#Вычисление вектора ориентации отрезка D = P2 - P1.

D = QPointF()

D.setX(line[1][0] - line[0][0])

D.setY(line[1][1] - line[0][1])

# цикл по всем сторонам отсекающего окна

for i in range(len(edges)):

# Вычисление вектора внутренней нормали к очередной i-ой стороне

N = QPointF()

if i == len(edges) - 1:

N.setX(-n \* (edges[0].y() - edges[i].y()))

N.setY(n \* (edges[0].x() - edges[i].x()))

else:

N.setX(-n \* (edges[i + 1].y() - edges[i].y()))

N.setY(n \* (edges[i + 1].x() - edges[i].x()))

# Определение граничной точки fi

F = QPointF(edges[i].x(), edges[i].y())

# Вычисление вектора Wi = P1 - fi

W = QPointF()

W.setX(line[0][0] - F.x())

W.setY(line[0][1] - F.y())

# Вычисление скалярного произведения векторов Wскi= Wi nвi

Dsc = scalar\_mult(D, N)

# Вычисление скалярного произведения векторов Dск i= Dnвi .

Wsc = scalar\_mult(W, N)

# Проверка на равенство нулю скалярного произведения Dск

if Dsc == 0:

# отрезок (точка) невидим

if Wsc < 0:

return

else:

#Вычисление параметра t

t = - Wsc / Dsc

# Поиск нижней границы параметра t

if Dsc > 0:

# отрезок невидим

if t > 1:

return

else:

tb = max(tb, t)

# Поиск верхней границы параметра t

elif Dsc < 0:

# отрезок невидим

if t < 0:

return

else:

te = min(te, t)

# Конец цикла по сторонам отсекателя

# проверка фактической видимости отрезка

if tb <= te:

#визуализация отрезка в интервале от P(tн) до P(tв).

R1 = P\_t(line, te)

R2 = P\_t(line, tb)

scene.addLine(R1.x(), R1.y(), R2.x(), R2.y(), pen)







