МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное
образовательное учреждение высшего образования
«Юго-Западный государственный университет»
(ЮЗГУ)

Кафедра программной инженерии

УТВЕРЖДАЮ
Проректор по учебной работе
_____О. Г. Локтионова
«10» 02 2022 г.

АЛГОРИТМЫ РАСТЕРИЗАЦИИ ОКРУЖНОСТЕЙ

Методические указания по выполнению лабораторной работы
по дисциплине «Компьютерная графика»
для студентов всех форм обучения направления подготовки
09.03.04 «Программная инженерия»

Курск 2022

УДК 004.92
Составитель Е.А. Петрик

Рецензент
Кандидат технических наук, доцент Т.И.Лапина

Алгоритмы растеризации окружностей: методические указания по выполнению лабораторной работы / Юго-Зап. гос. ун-т; сост.: Е. А. Петрик. Курск, 2022. 11 с.: ил.4. Библиогр.: с.11.

Содержат краткие теоретические сведения об алгоритмах растеризации окружностей, а также приведены примеры и задания для лабораторной работы.

Методические указания соответствуют требованиям программы по направлению подготовки бакалавров: 09.03.04 «Программная инженерия»

Предназначены для студентов всех форм обучения направления подготовки бакалавров 09.03.04 «Программная инженерия»

Текст печатается в авторской редакции

Подписано в печать Формат 60х84 1/16.
Усл. печ. л. Уч. – изд. л. .Тираж экз. Заказ . Бесплатно.
Юго - Западный государственный университет.
305040, г. Курск, ул. 50 лет Октября, 94.

Цель работы

Изучение алгоритмов растеризации окружностей, создание программы для визуализации работы алгоритмов.

Основные понятия

Растровая развертка окружности. Простые алгоритмы растровой развертки окружности

Существует несколько простых, но неэффективных способов преобразования окружностей в растровую форму.

Рассмотрим окружность с центром в начале координат, для которой

$x^2 + y^2 = R^2$.

Решая это уравнение относительно y, получим

$y = \pm\sqrt{R^2 - x^2}$.

Чтобы изобразить четвертую часть окружности, можно увеличивать x с единичным шагом от 0 до R и на каждом шаге вычислять y и делать округление полученного значения (остальные четверти изображаются симметрично). Этот метод работоспособен, но не эффективен, поскольку в него входят операции умножения и извлечения квадратного корня. Более того, при значениях x, близких к R, в окружности появляются заметные незаполненные промежутки, так как при этих значениях x тангенс угла наклона касательной к окружности стремится к бесконечности (рис. 1).

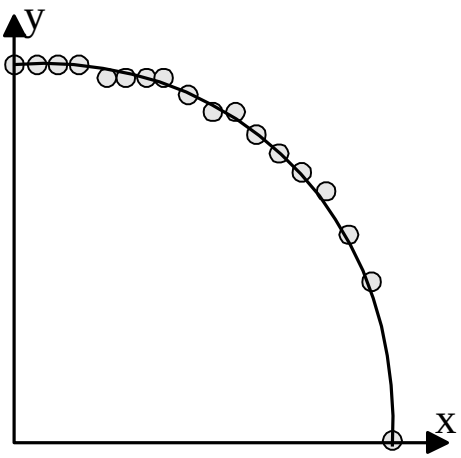

Рисунок 1 – Четвертая часть окружности, построенная с использованием единичных шагов по x и вычисленная с последующим округлением значений y

Еще один простой подход к построению окружности также заключается в использовании ее уравнения

$$(x - x_0)^2 + (y - y_0)^2 = R^2$$

или

$$x = x_0 + R\cos..., \quad y = y_0 + R\sin...$$

где x_0, y_0 - координаты центра,

R - радиус окружности,

τ - параметр.

При этом, учитывая тот факт, что шаг изменения аргумента должен составлять величину $\tau=1/R$, потребуется рассчитать для каждого значения параметра τ значения координат соответствующих точек окружности и соединить их затем отрезками прямых. Такой путь также требует довольно большого количества вычислений.

Другим, более простым и эффективным, путем является использование алгоритма Брезенхема для генерации окружности. Чтобы построить полную окружность, достаточно сгенерировать ее одну восьмую часть. Остальные части получаются затем путем симметричного отражения относительно определенной прямой. Так, отражая одну восьмую часть, построенную в первом октанте для углов в диапазоне 0-45, относительно прямой с уравнением y=x, получим одну четвертую часть, лежащую в первом квадранте. Отразив эту четверть относительно прямой x=0, получим одну вторую часть, лежащую выше оси абсцисс, наконец, отразив эту полуокружность относительно прямой y=0, получим полную окружность.

Рассмотрим построение части окружности, лежащей в первом квадранте. Предположим, что центр окружности лежит в начале координат, ее радиус равен R, ось абсцисс направлена вправо, ось ординат - вверх. Пусть алгоритм начинает работу в точке с координатами x=0, y=R, тогда он заканчивает работу в точке x=R, y=0. При таком направлении построения окружности y как функция аргумента x будет являться монотонно убывающей ($y=\sqrt{R^2 - x^2}$).

Для любого пикселя при выбранном направлении генерации окружности существует только три варианта выбора следующего пикселя, наилучшим образом аппроксимирующего окружность:

- горизонтальный пиксель вправо,

- диагональный вниз и вправо,
- вертикальный вниз.

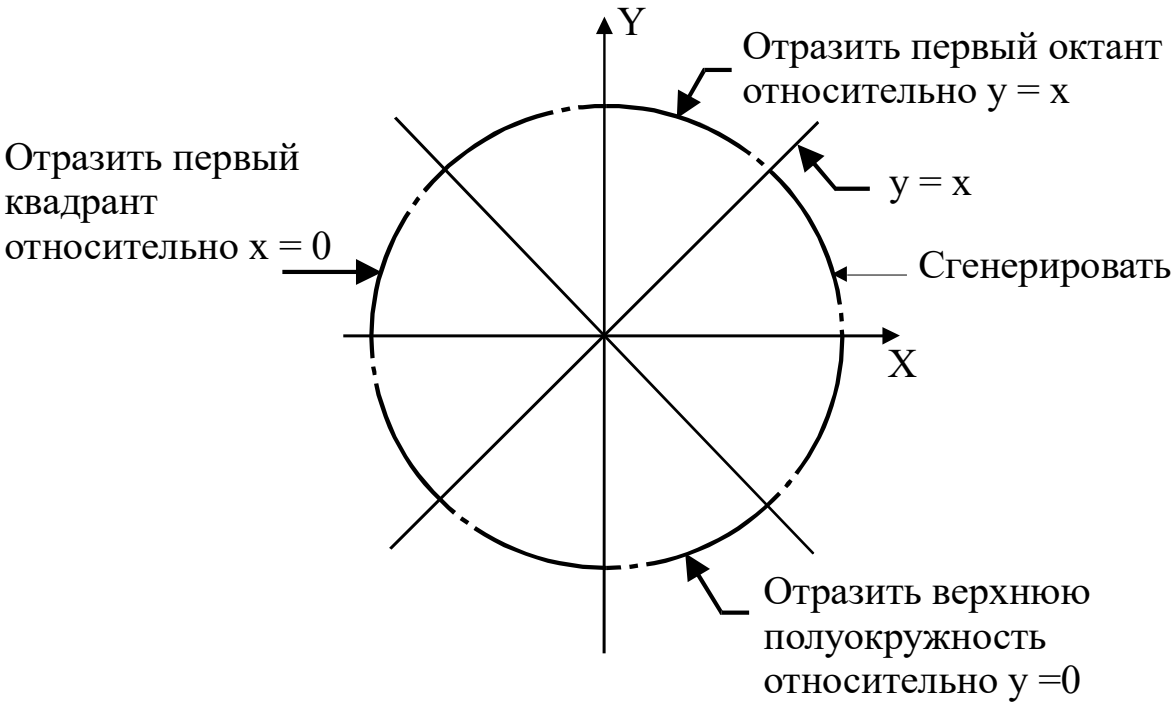

Рисунок 2 – Генерация полной окружности из дуги в первом октанте

В качестве критерия выбора очередного пикселя используется минимум модуля разности квадратов расстояний от центра окружности до пикселя и до идеальной окружности. Эти величины имеют следующие значения:

$m_h=|(x_i+1)^2 + y_i^2- R^2|$

$m_d=|(x_i+1)^2 + (y_i-1)^2 - R^2|$

$m_v =|x_i^2+ (y_i-1)^2 - R^2|$,

где m_h, m_d, m_v - искомые величины расстояний для горизонтального, диагонального и вертикального направлений соответственно,

x_i, y_i - координаты текущего пикселя.

Вычисления можно упростить, если заметить, что в окрестности текущей точки возможны пять вариантов прохождения окружности (рис. 3.3):

1) между горизонтальным (x_i+1, y_i) и диагональным (x_i+1, y_i-1) пикселами;

2) между горизонтальным (x_i+1, y_i) и вторым диагональным (x_i+1, y_i+1) пикселами;

3) между вертикальным (x_i, y_i-1) и диагональным (x_i+1, y_i-1);

4) между вертикальным (x_i, y_i-1) и третьим диагональным (x_i-1, y_i-1) пикселями;

5) прохождение точно через диагональный (x_i+1, y_i-1) пиксель.

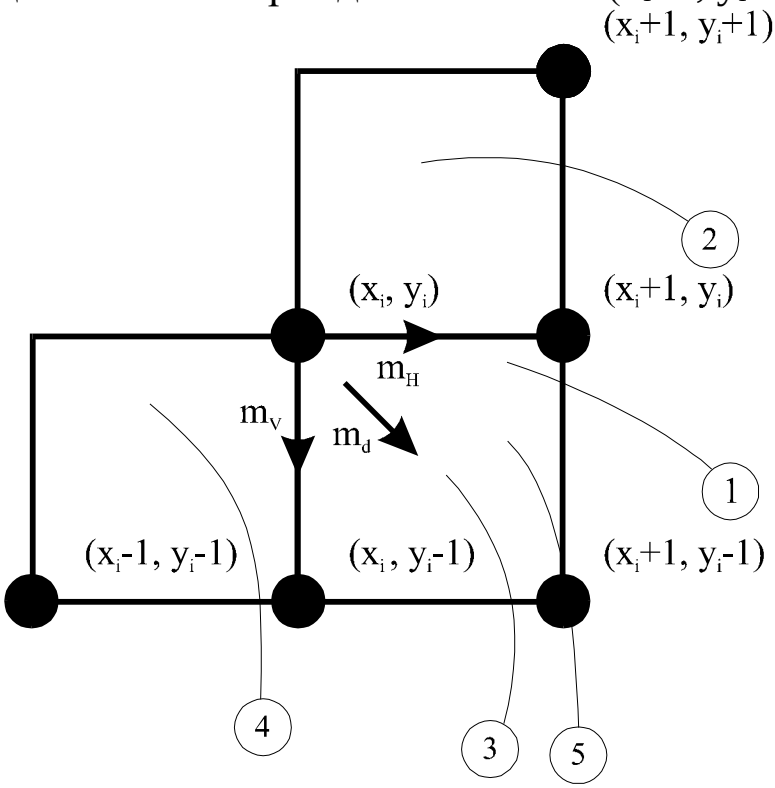

Рисунок 3 – Пересечение окружности и сетки растра

По аналогии с ошибкой в алгоритме построения отрезков выберем и здесь некоторую величину, анализ знака которой позволит выбрать нужный пиксель.

Такой величиной Δ является разность квадратов расстояний от центра окружности до диагонального (x_i+1,y_i-1) пикселя и до идеальной окружности:

$\Delta_\iota=(\xi_\iota+1)^2 + (\psi_\iota-1)^2 - P^2$

Отрицательное значение $\Delta_i<0$ может быть только в том случае, когда диагональный пиксель (x_i+1, y_i-1) лежит внутри окружности, то есть это случаи 1 и 2 и следует выбирать горизонтальный (x_i+1, y_i) или диагональный (x_i+1,y_i-1) пиксель.

Рассмотрим случай 1 и вычислим разность Δ_1 расстояний, то есть

$\Delta_1= \mu_\eta - \mu_\delta=|(\xi_\iota+1)^2 + \psi_\iota^2 - P^2| - |(\xi_\iota+1)^2 + (\psi_\iota-1)^2 - P^2|$

При $m_h<m_d$, $\Delta_1<0$ и в этом случае расстояние до горизонтального пикселя меньше, чем до диагонального, поэтому выбирается горизонтальный пиксель. При $m_h>m_d$, $\Delta_1>0$ и в этом

случае расстояние до диагонального пикселя меньше, чем до горизонтального, поэтому выбирается диагональный пиксель.

При $m_h=m_d$, $\Delta_1=0$ и можно выбирать либо горизонтальный, либо диагональный пиксель. Для определенности выберем горизонтальный пиксель.

Для сокращения вычислений упростим выражение. При этом следует учесть, что для случая 1 $m_h>0$, а $m_d<0$, так как горизонтальный пиксель лежит вне окружности, а диагональный внутри окружности.

Таким образом, можно записать:

$\Delta_1 =(x_i+1)^2+y_i^2-R^2+(x_i+1)^2+(y_i-1)^2-R^2$

Дополнение до полного квадрата члена y_i^2 с помощью добавления и вычитания $-2y_i+1$ дает:

$\Delta_1 =2[(x_i+1)^2+(y_i-1)^2-R^2]+2y_i-1.$

В квадратных скобках стоит по определению Δ_i и его подстановка существенно упрощает выражение:

$\Delta_1 =2(\Delta_\iota+\psi_\iota)-1.$

Во втором случае выбирается горизонтальный пиксель, второй диагональный пиксель (x_i+1, y_i+1) выбран быть не может в силу монотонного убывания функции. Для этого случая $mh<0$ и $md<0$, поэтому

$\Delta_1=P^2 - (\xi_\iota+1)^2 - \psi_\iota^2 + (\xi_\iota+1)^2 + (\psi_\iota-1)^2 - P^2 = 1 -2\psi_\iota$

Для первого квадранта $1<y_i<R$, поэтому $\Delta_1=1-2y_i<0$. Таким образом, и в случае 2, если $\Delta_1<0$, то выбирается горизонтальный пиксель.

Рассмотрим случай $\Delta_i>0$. Это будет справедливо только в том случае, если диагональный пиксель лежит вне окружности, то есть это случаи 3 и 4 и надо выбирать либо диагональный (x_i+1, y_i-1), либо вертикальный (x_i, y_i-1) пиксель.

Рассмотрим случай 3 и вычислим разность расстояний:

$\Delta_2= m_d - m_v =|(x_i+1)^2 + (y_i-1)^2 - R^2| - |x_i^2+ (y_i-1)^2 - R^2|$

При $\Delta_2<0$ и расстояние до диагонального пикселя меньше, чем до вертикального, поэтому выбирается диагональный пиксель. При $m_d> m_v$, $\Delta_2>0$ - расстояние до вертикального пикселя меньше, чем до диагонального, поэтому выбирается вертикальный пиксель. При $m_d = m_v$, $\Delta_2=0$ и можно выбирать либо диагональный, либо вертикальный пиксель. Для определенности выберем диагональный пиксель.

Для упрощения вычисления выражения учтем, что для случая 3 $m_d>0$, а $m_v<0$, так как диагональный пиксель лежит вне окружности, а вертикальный внутри окружности.

$\Delta_2=(\xi_\iota+1)^2+(\psi_\iota-1)^2-P^2+\xi_\iota^2+(\psi_\iota-1)^2-P^2==2[(\xi_\iota+1)^2+(\psi_\iota-1)^2-P^2]-2\xi_\iota-1=2(\Delta_\iota-\xi_\iota)-1$

В четвертом случае выбирается вертикальный пиксель, третий диагональный пиксель (x_i-1, y_i-1) выбран быть не может в силу монотонного убывания функции. Для этого случая $m_d>0$ и $m_v>0$ и

$\Delta_2=(x_i+1)^2 + (y_i-1)^2 - R^2 - x_i^2 - (y_i-1)^2 + R^2=2x_i+1$

Для первого квадранта $0<x_i<R$, поэтому $\Delta_2=2x_i+1>0$. Таким образом, и в случае 4 если $\Delta_2>0$, то выбирается вертикальный пиксель.

Остается проверить случай 5, для которого $\Delta_i=0$, то есть диагональный пиксель лежит точно на окружности.

Вычислим значение $\Delta_1=m_h-m_d$, учитывая, что $m_h>0$, а $m_d=0$:

$\Delta_1=(\xi_\iota+1)^2 + \psi_\iota^2 - P^2 > 0$,

следовательно, выбирается диагональный пиксель.

Вычислим значение $\Delta_2=m_d-m_v$, учитывая, что $m_d=0$, $m_v<0$:

$\Delta_2=\xi_\iota^2 - (\psi_\iota-1)^2 - P^2 < 0$,

таким образом, выполняется условие выбора правильного диагонального шага к (x_i+1, y_i-1).

Таким образом, при $\Delta_i=0$ выполняются те же условия для Δ_1 и Δ_2, как и в ранее рассмотренных случаях.

Подведем итог полученных результатов:

$\Delta_i<0$

$\Delta_1=0$ выбираем пиксель (x_i+1, y_i) m_h

$\Delta_1>0$ выбираем пиксель (x_i+1, y_i-1) m_d

$\Delta_i>0$

$\Delta_2=0$ выбираем пиксель (x_i+1, y_i-1) m_d

$\Delta_2>0$ выбираем пиксель (x_i, y_i-1) m_v

$\Delta_i=0$ выбираем пиксель (x_i+1, y_i-1) m_d

Теперь остается записать формулы для вычисления координат очередного пикселя и критерия для каждого из трех возможных направлений движения. Обозначим новое положение пикселя как (i+1) в индексе x,y координат и критерия Δ.

При переходе к горизонтальному пикселу:

$x_{i+1}=x_i+1$

$y_{i+1}=y_i$

$\Delta_{i+1}=(x_{i+1}+1)^2+(y_{i+1}-1)^2-R^2 \quad =(x_{i+1})^2+2x_{i+1}+1+(y_i-1)^2-R^2$
$=(x_i+1)^2+(y_i-1)^2-R^2+2x_{i+1}+1$

$=\Delta_i+2x_{i+1}+1$

При переходе к диагональному пикселю:

$x_{i+1}=x_i+1$

$y_{i+1}=y_i-1$

$\Delta_{i+1}=\Delta_i+2\xi_{i+1}-2\psi_{i+1}+2$

При переходе к вертикальному пикселю:

$x_{i+1}=x_i$

$y_{i+1}=y_i-1$

$\Delta_{i+1}=\Delta_i-2\psi_{i+1}+1$

Начальное значение для x=0, y=R будет равно:

$\Delta=(\xi_i+1)^2+(\psi_i-1)^2-P^2$, $\xi_i=0$, $\psi_i=1$, тогда:

$\Delta=1+(P-1)^2-P^2=2(1-P)$.

Алгоритм Брезенхема для генерации окружности в первом квадранте может быть записан в следующем виде:

1. Ввод исходных данных R (радиус окружности) и при необходимости Xc,Yc (координаты центра окружности).

2. Задание начальных значений текущих координат пикселя x:=0, y:=R, параметра Δ:=2(1-R), установка конечного значения ординаты пикселя y_k:=0.

3. Высвечивание текущего пикселя (x,y).

4. Проверка окончания работы: если $y<y_k$, то переход к п.11.

5. Анализ значения параметра Δ:

если $\Delta<0$, то переход к п.6;

если $\Delta=0$, то переход к п.9;

если $\Delta>0$, то переход к п.7.

6. Вычисление параметра $\Delta_1:=2\Delta+2\psi-1$ и анализ полученного значения:

если $\Delta_1=0$, то переход к п.8;

иначе - переход к п.9.

7. Вычисление параметра $\Delta 2:=2\Delta-2\xi-1$ и анализ полученного значения:

если $\Delta_2=0$, то переход к п.9;

иначе - переход к п.10.

8. Вычисление новых значений x и Δ (горизонтальный шаг):

$\xi:=\xi+1$; $\Delta:=\Delta+2\xi+1$. Переход к п.3.

9. Вычисление новых значений x,y и Δ (диагональный шаг):

ξ:=ξ+1; ψ:=ψ−1; Δ:=Δ+2(ξ−ψ+1). Переход к п.3.

10. Вычисление новых значений y и (вертикальный шаг):

ψ:=ψ−1; Δ:=Δ−2ψ+1. Переход к п.3.

11. Конец.

Примечание. Данный алгоритм необходимо дополнить с тем, чтобы учитывались вводимые координаты центра окружности и разные разрешающие способности экрана, а также обеспечивалось бы рисование всей окружности.

Учет ненулевых координат центра окружности означает фактически выполнение операции переноса, поэтому координаты точек окружности (X1,Y1) будут вычисляться по формулам: X1=Xc+X; Y1=Yc+Y. Неодинаковые разрешающие способности можно учесть, скорректировав ординату точки окружности.

```
// R - радиус, X1, Y1 - координаты центра
  int x := 0
  int y := R
  int delta := 1 - 2 * R
  int error := 0
  while (y >= x)
    drawpixel(X1 + x, Y1 + y)
    drawpixel(X1 + x, Y1 - y)
    drawpixel(X1 - x, Y1 + y)
    drawpixel(X1 - x, Y1 - y)
    drawpixel(X1 + y, Y1 + x)
    drawpixel(X1 + y, Y1 - x)
    drawpixel(X1 - y, Y1 + x)
    drawpixel(X1 - y, Y1 - x)
    error = 2 * (delta + y) - 1
    if ((delta < 0) && (error <= 0))
      delta += 2 * ++x + 1
      continue
    if ((delta > 0) && (error > 0))
      delta -= 2 * --y + 1
      continue
    delta += 2 * (++x - --y)
```

Рисунок 4 – Пример программы

Задание

Написать программу (на языке высокого уровня), реализующую алгоритмы растеризации окружности с их последующей отрисовкой. Координаты центра и радиуса должны задаваться пользователем.

Содержание отчета

Отчет должен содержать:

1. Титульный лист.
2. Задание.
3. Блок-схемы алгоритмов.
4. Листинг программы.
5. Пример работы программы.
6. Ответы на контрольные вопросы.

Контрольные вопросы

1. Что такое пиксель?
2. Что такое растровое изображение?
3. Для чего нужны алгоритмы растеризации окружностей?
4. Какие ещё алгоритмы растеризации окружностей существуют?

Список используемой литературы

1. Аммерал, Л. Принципы программирования в машинной графике / Л. Аммерал; пер. с англ. – Москва : Сол Систем, 1992. – 224 с. – ISBN 5-85316-001-X. – Текст : непосредственный.
2. Роджерс, Д. Алгоритмические основы машинной графики / Д. Роджерс; пер. с англ. – Москва : Мир, 1989. – 512 с. – ISBN 5-03-000476-9. – Текст : непосредственный.
3. Роджерс, Д. Математические основы машинной графики / Д. Роджерс, Дж. Адамс; пер. с англ. – Москва : Мир, 2001. – 604 с. – ISBN 5-03-002143-4. – Текст : непосредственный.

4. Шикин, Е. В. Начала компьютерной графики / Е. В. Шикин, А. В. Боресков, А. А. Зайцев. – Москва : ДИАЛОГ-МИФИ, 1993. – 138 с. – ISBN 5-86404-035-5. – Текст : непосредственный.