بيانيات الحاسوب

Computer Graphics

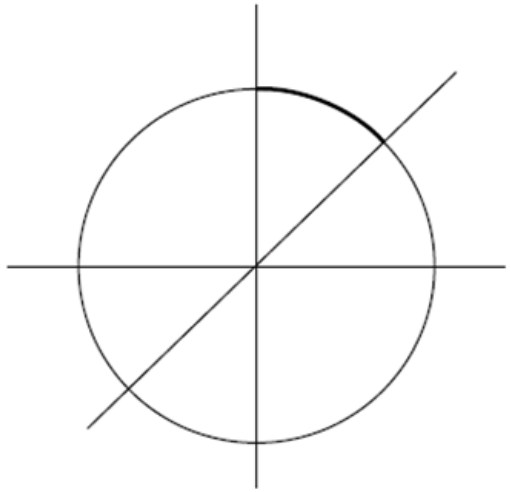
Dr.-Eng. Samer Sulaiman

2022-2023

مفردات المنهاج

- **أساسيات بيانيات الحاسوب**
  - مقدمة: مفاهيم أساسية
  - التحويلات ثنائية البعد 2D
  - التحويلات ثلاثية البعد 3D
- **خوارزميات بيانيات الحاسوب**
  - الخوارزميات الهندسية
  - الخوارزميات النقطية الأساسية
  - خوارزميات إزالة الأسطح المخفية
- **نماذج بيانيات الحاسوب**
  - الألوان ونماذج الألوان
  - المنحنيات والأسطح: النمذجة الهندسية
  - نماذج الإضاءة والتظليل
  - عناصر التركيب والنمذجة الإجرائية

خوارزميات بيانيات الحاسوب



- خوارزميات البيانات النقطية الأساسية Basic raster algorithms :
- خوارزمية Bresenheim (Bresenheim’s circle algorithm) لرسم دائرة
  - يمكن تعديل الخوارزمية الموصوفة لرسم مستقيم لرسم الدوائر
  - سيتم أولاً كتابة خوارزمية لدائرة نصف قطرها R متمركز في مبدأ الإحداثيات ثم سيتم تعميم الخوارزمية للحصول على دائرة عشوائية
  - نظرًا لتماثل الدائرة ، يكفي تنقيط 1/8 الدائرة فقط
    - يتوافق هذا الجزء من الدائرة مع قيم x ضمن المجال [0 ، √R/2] وقيمة y من √R/2 إلى R والتي سنقوم بتنقيطها أو رسمها
    - يمتلك هذا الجزء من الدائرة خاصية مهمة وهي:
      - يقع المنحدر عند أي نقطة في النطاق [1 ، 0]
    - سيتم رسم 8 بكسل لكل استدعاء (بكسل في كل جزء من الأجزاء الثمانية)
      - من خلال تطبيق التناظر للحصول على جميع وحدات البكسل من استخدام إحداثيات البكسل (من القوس) ومركز الدائرة (xc, yc)

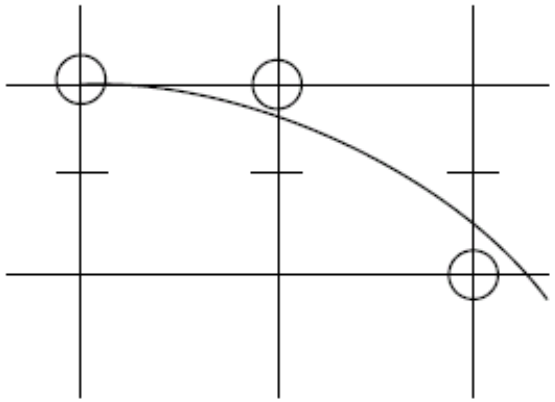
خوارزميات بيانيات الحاسوب

- خوارزميات البيانات النقطية الأساسية Basic raster algorithms :
  - خوارزمية Bresenheim (Bresenheim's circle algorithm) لرسم دائرة

```
void drawCirclePoints ( int xc, int yc, int x, int y,
                        int color )
{
  putPixel ( xc + x, yc + y, color );
  putPixel ( xc + y, yc + x, color );
  putPixel ( xc + y, yc - x, color );
  putPixel ( xc + x, yc - y, color );
  putPixel ( xc - x, yc - y, color );
  putPixel ( xc - y, yc - x, color );
  putPixel ( xc - y, yc + x, color );
  putPixel ( xc - x, yc + y, color );
}
```


- التحقق من نقاط التواجد داخل / خارج الدائرة:
  - $F(x, y) = x^2 + y^2 - R^2$
  - يساوي صفرًا على الدائرة ، وسالب داخل الدائرة وموجب خارجها
- بناء تسلسل للبكسل
  - البكسل الأول واضح: $x_0 = 0$، $y_0 = R$.

خوارزميات بيانيات الحاسوب



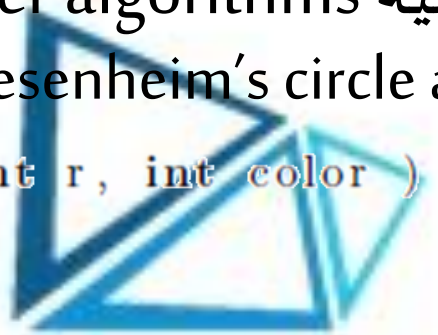
- خوارزميات البيانات النقطية الأساسية Basic raster algorithms :
  - خوارزمية Bresenheim (Bresenheim's circle algorithm) لرسم دائرة
    - يمكن الحصول على قيمة x للبكسل بناءً على العنصر السابق :
      - $x_{i+1} = x_i + 1$
    - بالنسبة إلى y_{i+1}، هناك خياران محتملان فقط
      - نظرًا لوجود المنحدر في نطاق [0 ، 1]
      - القيم الممكنة: y_i أو $y_i - 1$
    - التحقق من نقطة المنتصف بين هاتين القيمتين لمعرفة ما إذا كانت داخل الدائرة أو خارجها
      - استخدام متغير القرار
        - $di = F(x_i + 1, y_i - 1/2) = (x_i + 1)^2 + (y_i - 1/2)^2 - R^2$
      - إذا كانت $di < 0$، فإن نقطة المنتصف $(x_i + 1, y_i - 1/2)$ تقع داخل الدائرة
        - تحديد $y_{i+1} = y_i$
      - خلاف ذلك $(di \geq 0)$
        - وجعل $y_{i+1} = y_i - 1$.

خوارزميات بيانيات الحاسوب

- خوارزميات البيانات النقطية الأساسية Basic raster algorithms :
- خوارزمية Bresenheim (Bresenheim's circle algorithm) لرسم دائرة
  - الآن بحاجة لمعرفة كيف يتغير متغير القرار اعتمادًا على اختيارنا
    - إذا كانت قيمة di <0، فلدينا المعادلة التالية لـ d_{i+1}
      - $d_{i+1} = F(x_{i+2}, y_{i+1} - 1/2) = F(x_{i+2}, y_i - 1/2)$
      - $d_{i+1} - d_i = 2x_i + 3$
    - إذا كانت di ≥ 0 عندئذ $y_{i+1} = y_i - 1$
      - $d_{i+1} = F(x_{i+2}, y_i - 3/2) = (x_{i+2})^2 + (y_i - 3/2)^2 - R^2$
      - إعطاء زيادة لمتغير القرار
      - $d_{i+1} - d_i = 2(x_i - y_i) + 5$
  - القيمة الأولية (الإبتدائية) لمتغير القرار
    - $d_0 = F(0 + 1, R - 1/2) = 5/4 - R$

خوارزميات بيانيات الحاسوب

- خوارزميات البيانات النقطية الأساسية Basic raster algorithms :
- خوارزمية Bresenheim (Bresenheim's circle algorithm) لرسم دائرة




```
void drawCircle ( int xc, int yc, int r, int color )
{
  int   x = 0;
  int   y = r;
  float d = 1.25f - r;

  drawCirclePoints ( xc, yc, x, y, color );

  while ( y > x )
  {
    if ( d < 0 )
    {
      d += 2*x + 3;
      x++;
    }
    else
    {
      d += 2*(x-y)+5;
      x++;
      y--;
    }
    drawCirclePoints ( xc, yc, x, y, color );
  }
}
```

خوارزميات بيانيات الحاسوب

- خوارزميات البيانات النقطية الأساسية Basic raster algorithms :
  - خوارزمية Bresenheim (Bresenheim's circle algorithm) لرسم دائرة
    - يفضل للسهولة دائمًا تغيير متغير القرار الخاص بنا بقيمة عدد صحيح وبما أن قيمته الأولية تحتوي على الجزء الكسري من ¼
      - سيظل الجزء الكسري لمتغير القرار لكل بكسل ¼
      - ببساطة يمكن تجاهل الجزء الكسري من di
      - يصبح di متغير عدد صحيح
    - بالنتيجة نحصل على كود يحتوي عدد صحيح فقط:


```
void drawCircle ( int xc, int yc, int r, int color )
{
  int x      = 0;
  int y      = r;
  int d      = 1 - r;
  int delta1 = 3;
  int delta2 = -2*r+5;

  drawCirclePoints ( xc, yc, x, y, color );

  while ( y > x )
  {
    if ( d < 0 )
    {
      d      += delta1;
      delta1 += 2;
      delta2 += 2;
      x++;
    }
    else
    {
      d      += delta2;
      delta1 += 2;
      delta2 += 4;
      x++;
      y--;
    }

    drawCirclePoints ( xc, yc, x, y, color );
  }
}
```

خوارزميات بيانيات الحاسوب

- خوارزميات البيانات النقطية الأساسية Basic raster algorithms :
  - خوارزمية Bresenheim (Bresenheim’s circle algorithm) لرسم دائرة
    - يفضل للسهولة دائمًا تغيير متغير القرار الخاص بنا بقيمة عدد صحيح وبما أن قيمته الأولية تحتوي على الجزء الكسري من ¼
      - سيظل الجزء الكسري لمتغير القرار لكل بكسل ¼
      - ببساطة يمكن تجاهل الجزء الكسري من di
      - يصبح di متغير عدد صحيح
    - بالنتيجة نحصل على كود يحتوي عدد صحيح فقط:

- .