Script **generated by TTT**

Title: T?ubig: GAD (19.04.2012)

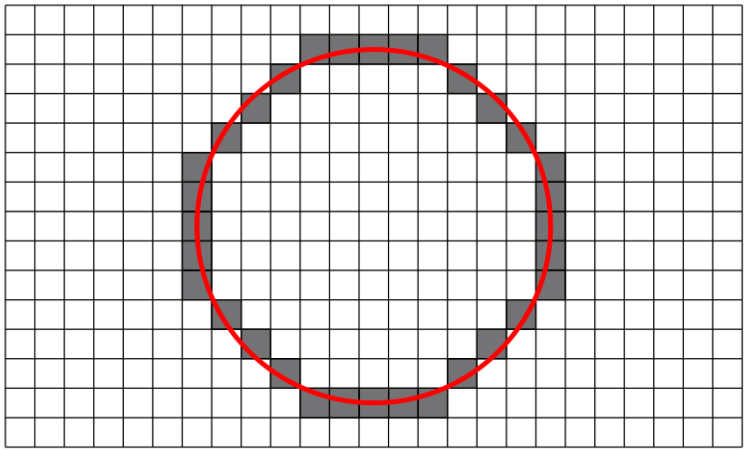
Date: Thu Apr 19 12:28:13 CEST 2012

Duration: 45:53 min

Pages: 21

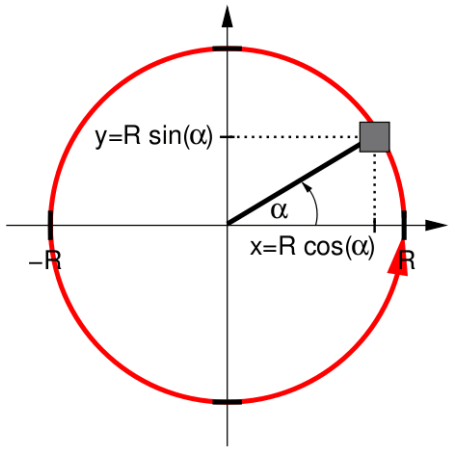
Kreis zeichnen

Wie kann ein Computer einen Kreis zeichnen?



Kreis zeichnen: mit Winkelfunktionen

Naiver Ansatz: eine Runde wie mit dem Zirkel



Verwendung von sin und cos für $\alpha = 0 \ldots 2\pi$

Kreis zeichnen: mit Wurzelfunktion

Algorithmus 3: Kreis2

Eingabe : radius R

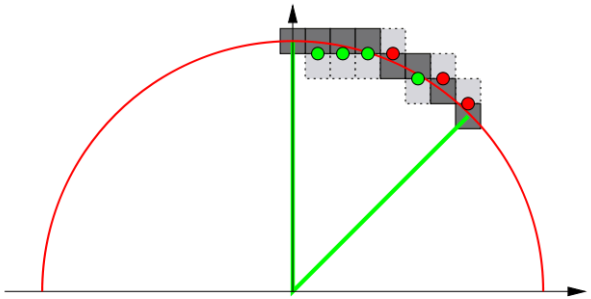
for $x = -R; x \leq R; x{+}{+}$ **do**

```
    y = sqrt(R * R - x * x);
    plot(x, y);
    plot(x, -y);
```

Problem: sqrt ist auch noch relativ teuer!

Kreis zeichnen: mit Multiplikation

- betrachtetes Kreissegment: Anstieg zwischen 0 und −1
- 2 Fälle für nächstes Pixel: nur rechts oder rechts unten
- Entscheidungskriterium:
  Grundlinienmittelpunkt des rechten Nachbarpixels innerhalb vom Kreis? ja: $x++$, nein: $x++; y--$



Kreis zeichnen: mit Multiplikation

- Test, ob (x, y) innerhalb des Kreises:

$$F(x,y) := x^2 + y^2 - R^2 < 0$$

- Mittelpunkt des ersten Quadrats: $(x, y) = (0, R)$
- Position seines Grundlinienmittelpunkts: $(0, R - \frac{1}{2})$
- Grundlinienmittelpunkt rechts daneben:
  $F(1, R - \frac{1}{2}) = 1^2 + (R - \frac{1}{2})^2 - R^2 = \frac{5}{4} - R < 0$?
- Update:

$$\begin{aligned} F(x+1,y) &= (x+1)^2 + y^2 - R^2 = (x^2 + 2x + 1) + y^2 - R^2 \\ F(x+1,y) &= F(x,y) + 2x + 1 \\ F(x+1,y-1) &= (x+1)^2 + (y-1)^2 - R^2 \\ &= (x^2 + 2x + 1) + (y^2 - 2y + 1) - R^2 \\ F(x+1,y-1) &= F(x,y) + 2x - 2y + 2 \end{aligned}$$

Kreis zeichnen: mit Multiplikation

- Test, ob (x, y) innerhalb des Kreises:

$$F(x,y) := x^2 + y^2 - R^2 < 0$$

- Mittelpunkt des ersten Quadrats: $(x, y) = (0, R)$
- Position seines Grundlinienmittelpunkts: $(0, R - \frac{1}{2})$
- Grundlinienmittelpunkt rechts daneben:
  $F(1, R - \frac{1}{2}) = 1^2 + (R - \frac{1}{2})^2 - R^2 = \frac{5}{4} - R < 0$?
- Update:

$$\begin{aligned} F(x+1,y) &= (x+1)^2 + y^2 - R^2 = (x^2 + 2x + 1) + y^2 - R^2 \\ F(x+1,y) &= F(x,y) + 2x + 1 \\ F(x+1,y-1) &= (x+1)^2 + (y-1)^2 - R^2 \\ &= (x^2 + 2x + 1) + (y^2 - 2y + 1) - R^2 \\ F(x+1,y-1) &= F(x,y) + 2x - 2y + 2 \end{aligned}$$

Kreis zeichnen: mit Multiplikation

Algorithmus 4: Bresenham1

$x = 0;\quad y = R;$
plot$(0, R)$; plot$(R, 0)$; plot$(0, -R)$; plot$(-R, 0)$;
$F = \frac{5}{4} - R;$
while $x < y$ **do**
  if $F < 0$ **then**
    $F = F + 2 * x + 1;$
  else
    $F = F + 2 * x - 2 * y + 2;$
    $y = y - 1;$
  $x = x + 1;$
  plot(x, y); plot$(-x, y)$; plot$(-y, x)$; plot$(-y, -x)$;
  plot(y, x); plot$(y, -x)$; plot$(x, -y)$; plot$(-x, -y)$;

Es geht sogar noch etwas schneller!

Kreis zeichnen: mit Addition / Subtraktion

- Ersetzung der Korrekturterme für F:

$$F = F + 2x + 1 \quad \rightarrow \quad F = F + d_E$$
$$F = F + 2x - 2y + 2 \quad \rightarrow \quad F = F + d_{SE}$$

mit $d_E = 2x + 1$ und $d_{SE} = 2x - 2y + 2$

- Anfangswerte:

$$\begin{aligned} d_E(0,R) &= 2 \cdot 0 + 1 = 1 \\ d_{SE}(0,R) &= 2 \cdot 0 - 2 \cdot R + 2 = 2 - 2 \cdot R \end{aligned}$$

- Updates nach rechts (E) und nach unten rechts (SE):

$$\begin{aligned} d_E(x+1,y) &= 2 \cdot (x+1) + 1 = d_E(x,y) + 2 \\ d_{SE}(x+1,y) &= 2 \cdot (x+1) - 2 \cdot y - 2 = d_{SE}(x,y) + 2 \\ d_E(x+1,y-1) &= 2 \cdot (x+1) + 1 = d_E(x,y) + 2 \\ d_{SE}(x+1,y-1) &= 2 \cdot (x+1) - 2 \cdot (y-1) + 2 = d_{SE}(x,y) + 4 \end{aligned}$$

Kreis zeichnen: mit Addition / Subtraktion

- Der Bruch $\frac{5}{4}$ kann durch 1 ersetzt werden, weil sich F immer um eine ganze Zahl ändert.
- D.h.

$$F = \frac{5}{4} - R + k < 0$$

ist äquivalent zu

$$F = 1 - R + k < 0$$

- Vorteil:

nur noch ganze Zahlen!

Kreis zeichnen: mit Addition / Subtraktion

Algorithmus 5: Bresenham2

$x = 0$; $y = R$; plot$(0,R)$; plot$(R,0)$; plot$(0,-R)$; plot$(-R,0)$;
$F = 1 - R$;
$d_E = 1$; $d_{SE} = 2 - R - R$;
while $x < y$ **do**
  if $F < 0$ **then**
    $F = F + d_E$;
    $d_{SE} = d_{SE} + 2$;
  else
    $F = F + d_{SE}$;
    $y = y - 1$;
    $d_{SE} = d_{SE} + 4$;
  $x = x + 1$; $d_E = d_E + 2$;
  plot(x,y); plot$(-x,y)$; plot$(-y,x)$; plot$(-y,-x)$;
  plot(y,x); plot$(y,-x)$; plot$(x,-y)$; plot$(-x,-y)$;

Bresenham-Algorithmus

- Ab Anfang der 1960er Jahre hat Jack Bresenham Algorithmen zur Linien- und Kreisdarstellung entwickelt.
- Diese verwenden nur einfache Additionen ganzer Zahlen.
- Sie sind damit deutlich schneller als die naiven Ansätze.

- Diese und weitere Beispiele:

Taschenbuch der Algorithmen (Springer, 2008)

Bresenham-Algorithmus

- Ab Anfang der 1960er Jahre hat Jack Bresenham Algorithmen zur Linien- und Kreisdarstellung entwickelt.
- Diese verwenden nur einfache Additionen ganzer Zahlen.
- Sie sind damit deutlich schneller als die naiven Ansätze.

- Diese und weitere Beispiele:

  Taschenbuch der Algorithmen (Springer, 2008)

Übersicht

Übersicht

Effizienzmessung

Exakte Spezifikation der Laufzeit eines Algorithmus (bzw. einer DS-Operation):

- Menge $\mathcal{I}$ der Instanzen
- Laufzeit des Algorithmus T : $\mathcal{I} \mapsto \mathbb{N}$

Problem: T sehr schwer exakt bestimmbar

Lösung: Gruppierung der Instanzen (meist nach Größe)

Eingabekodierung

Was ist die Größe einer Instanz?

- Speicherplatz in Bits oder Wörtern

Aber Vorsicht bei der Kodierung!

Beispiel: Primfaktorisierung

Gegeben: Zahl $x \in \mathbb{N}$

Gesucht: Primfaktoren von x (Primzahlen $p_1, \ldots, p_k$ mit $x = \prod_i p_i^{e_i}$)

Bekannt als hartes Problem (wichtig für RSA!)

Eingabekodierung - Beispiel Primfaktorisierung

Trivialer Algorithmus

Teste von $y = 2$ bis $\lfloor \sqrt{x} \rfloor$ alle Zahlen, ob diese x teilen und wenn ja, dann bestimme wiederholt das Ergebnis der Division bis die Teilung nicht mehr ohne Rest möglich ist

Laufzeit: $\sqrt{x}$ Teilbarkeitstests und höchstens $\log_2 x$ Divisionen

- Unäre Kodierung von x (x Einsen als Eingabe): Laufzeit linear, Anzahl der Teilbarkeitstests und Divisionen ist sogar sublinear, aber die Eingabe muss einmal ganz gelesen werden
- Binäre Kodierung von x ($\lceil \log_2 x \rceil$ Bits): Laufzeit exponentiell (bezüglich der Länge der Eingabe)

Eingabekodierung

Betrachtete Eingabegröße:

- Größe von Zahlen: binäre Kodierung
- Größe von Mengen / Folgen von Zahlen:
  Hier wird oft nur die Anzahl der Elemente betrachtet(!)

Beispiel (Sortieren)

Gegeben: Folge von Zahlen $a_1, \ldots, a_n \in \mathbb{N}$

Gesucht: sortierte Folge der Zahlen

Größe der Eingabe: n

Effizienzmessung

Für ein gegebenes Problem sei $\mathcal{I}_n$ die Menge der Instanzen der Größe n.

Effizienzmaße:

- Worst case:
$$t(n) = \max\{T(i) : i \in \mathcal{I}_n\}$$
- Average case:
$$t(n) = \frac{1}{|\mathcal{I}_n|} \sum_{i \in \mathcal{I}_n} T(i)$$

Effizienzmaße: Vor- und Nachteile

- worst case:
  liefert Garantie für die Effizienz des Algorithmus
  (auch wichtig für Robustheit)
- average case:
  nicht unbedingt übereinstimmend mit dem “typischen Fall” in der Praxis (evt. schwer formal zu erfassen), kann durch Wahrscheinlichkeitsverteilung noch verallgemeinert werden
- exakte Formeln für $t(n)$ meist sehr aufwendig

$\Rightarrow$ betrachte asymptotisches Wachstum

Asymptotische Notation

Intuition:

- Zwei Funktionen $f(n)$ und $g(n)$ haben die gleiche Wachstumsrate, falls für genügend große n das Verhältnis der beiden nach oben und unten durch Konstanten beschränkt ist, d.h.

$$\exists c, d \in \mathbb{R}_+ \quad \exists n_0 \in \mathbb{N} \quad \forall n \geq n_0 : \quad c \leq \frac{f(n)}{g(n)} \leq d$$

- $f(n)$ wächst schneller als $g(n)$, wenn es für alle positiven Konstanten c ein n_0 gibt, ab dem $f(n) \geq c \cdot g(n)$ für $n \geq n_0$ gilt

Beispiel

n^2 und $5n^2 - 7n$ haben das gleiche Wachstum, da für alle $n \geq 2$ $\frac{1}{5} < \frac{(5n^2-7n)}{n^2} < 5$ und $\frac{1}{5} < \frac{n^2}{(5n^2-7n)} < 5$ gilt.