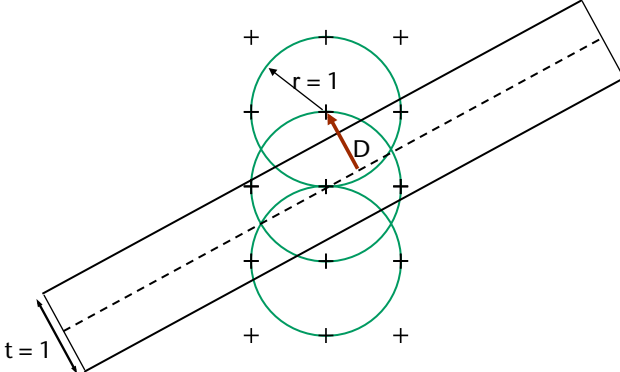


## Der Gupta-Sproull Algorithmus [1981]

- Variante des Standard-Midpoint-Algorithmus
- Berechnet inkrementell den Abstand  $D$  zwischen Pixelmittelpunkt und der Linie
- Bestimme Pixel-Intensität entsprechend dem Wert  $D$
- Führe dies für einige Pixel ober- und unterhalb der Linie durch
- Verwende einfache Lookup-Tables für die Intensität:  $Filter(D)$
- Beachte: Filterwert ist nur von  $D$  abhängig, nicht von der Steigung der Gerade

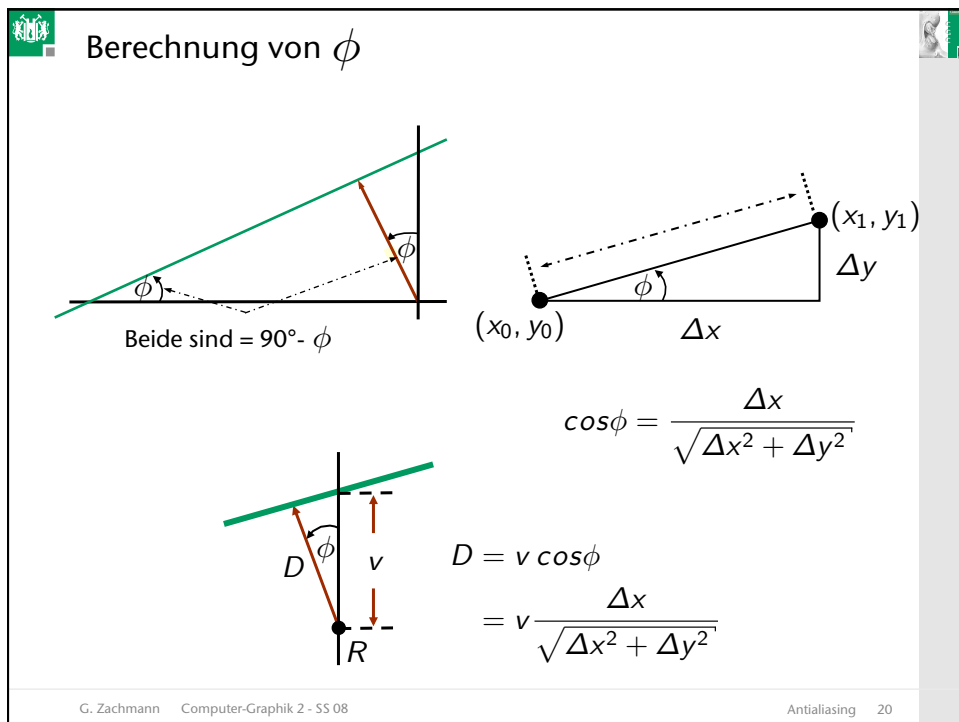
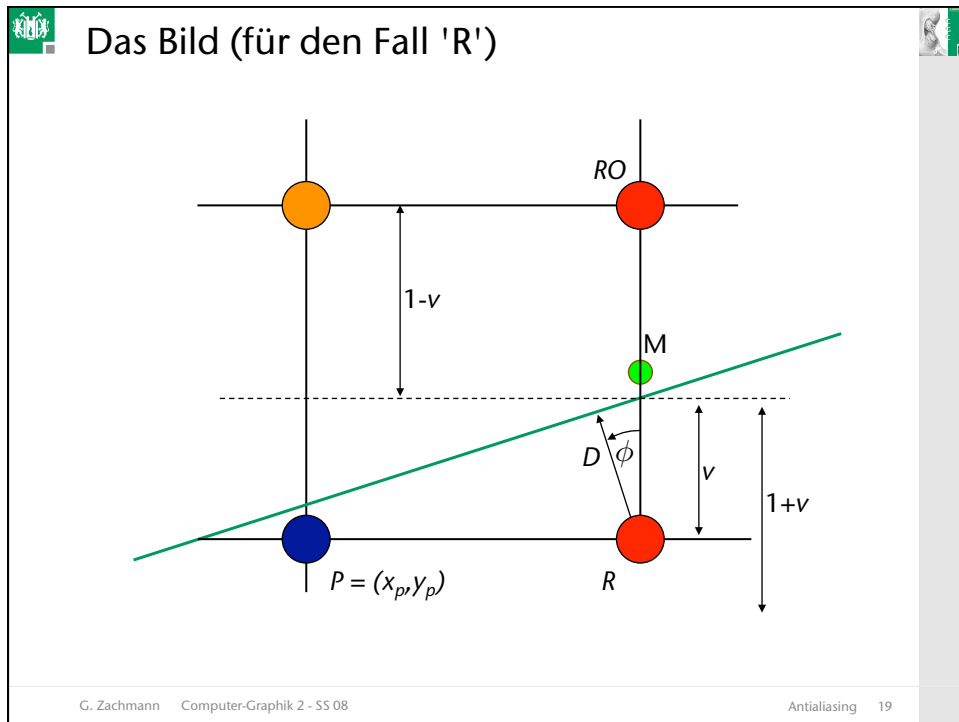
G. Zachmann Computer-Graphik 2 - SS 08 Antialiasing 17

- Eine Linie der Breite 1 Pixel überdeckt im Allgemeinen 3 Pixel-Filter-Kegel:



The diagram shows a 1-pixel wide line (shaded area) and a dashed line representing the ideal line. Three overlapping green circles represent filter kernels with radius  $r = 1$ . A red arrow labeled  $D$  indicates the distance from the pixel center to the dashed line. The label  $t = 1$  is at the bottom left of the shaded area.

G. Zachmann Computer-Graphik 2 - SS 08 Antialiasing 18



### Inkrementelle Berechnung von $D$

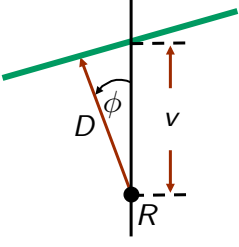
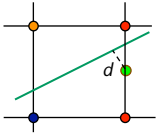
- $D = v \cos \phi$   

$$= v \frac{\Delta x}{\sqrt{\Delta x^2 + \Delta y^2}}$$
- Ziel:  $D$  **inkrementell** berechnen
- Idee:  $D$  aus  $d$  (Entscheidungsvariable) berechnen
- Erinnerung:

$$d = F(M) = F(x_p + 1, y_p + \frac{1}{2})$$

$$= n_1(x_p + 1) + n_2(y_p + \frac{1}{2}) + c$$

mit  $n_1 = \Delta y$ ,  $n_2 = -\Delta x$

G. Zachmann Computer-Graphik 2 - SS 08 Antialiasing 21

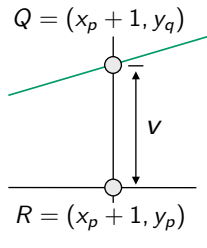
1.  $F(Q) = n_1(x_p + 1) + n_2 y_q + c = 0$   

$$y_q = \frac{-n_1(x_p + 1) - c}{n_2}$$
2.  $v = y_q - y_p = \frac{-n_1(x_p + 1) - c}{n_2} - y_p$   

$$n_2 \cdot v = -n_1(x_p + 1) - c - n_2 y_p \quad (n_2 = -\Delta x)$$

$$\Delta x \cdot v = n_1(x_p + 1) + n_2 y_p + c = F(M) - \frac{n_2}{2}$$

$$= d + \frac{\Delta x}{2}$$
3.  $D = \frac{d + \frac{1}{2} \Delta x}{\sqrt{\Delta x^2 + \Delta y^2}}$



G. Zachmann Computer-Graphik 2 - SS 08 Antialiasing 22

### Bei Entscheidung für "R" im Midpoint-Algo

$$D^R = \frac{d + \frac{1}{2}\Delta x}{\sqrt{\Delta x^2 + \Delta y^2}}$$

$$D_u^R = \frac{d + \frac{3}{2}\Delta x}{\sqrt{\Delta x^2 + \Delta y^2}}$$

$$D_o^R = \frac{d - \frac{1}{2}\Delta x}{\sqrt{\Delta x^2 + \Delta y^2}}$$

G. Zachmann Computer-Graphik 2 - SS 08
Antialiasing 23

### Analog für die Entscheidung 'RO'

■ Distanzen:

$$D^{RO} = \frac{d - \frac{1}{2}\Delta x}{\sqrt{\Delta x^2 + \Delta y^2}}$$

$$D_u^{RO} = \frac{d + \frac{1}{2}\Delta x}{\sqrt{\Delta x^2 + \Delta y^2}}$$

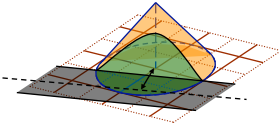
$$D_o^{RO} = \frac{3d - \frac{1}{2}\Delta x}{\sqrt{\Delta x^2 + \Delta y^2}}$$

G. Zachmann Computer-Graphik 2 - SS 08
Antialiasing 24

## Beispiel für Lookup-Table

- Precomputation: Berechne zu verschiedenen D's die Intensitäten (bzw. Gewichte)

Wert für D	Intensitätswert
0.00	0.70
0.25	0.40
0.50	0.25
0.75	0.05
1.00	0.00



$$= \int_L W(x, y) dx dy$$

G. Zachmann Computer-Graphik 2 - SS 08 Antialiasing 25

## Pseudo-Code für Gupta-Sproull Algorithmus

```

berechne  $n_1, n_2, c, d_1, d_2$ 
init  $x, y, d \sqrt{\Delta x^2 + \Delta y^2}$ 
setze  $z \leftarrow$ 
while  $x \leq x_1$ :
   $x_c, y_c = x, y$ 
  if  $d > 0$ : # RO
     $D \leftarrow (d - \frac{1}{2} \Delta x) / z$ 
    berechne  $D_u, D_o$ 
     $d += d_2$ 
     $y += 1$ 
  else: # R
     $D \leftarrow (d + \frac{1}{2} \Delta x) / z$ 
    berechne  $D_u, D_o$ 
     $d += d_1$ 
  zeichne Pixel( $x_c, y_c$ ) mit Intensität  $L(D)$ 
  zeichne Pixel( $x_c, y_c+1$ ) mit Intensität  $L(D_o)$ 
  zeichne Pixel( $x_c, y_c-1$ ) mit Intensität  $L(D_u)$ 

```

(L ist die Lookup-Table, die für gegebene Distanz die Gewichtung (gemäß Filter-Kernel) liefert.)

G. Zachmann Computer-Graphik 2 - SS 08 Antialiasing 26