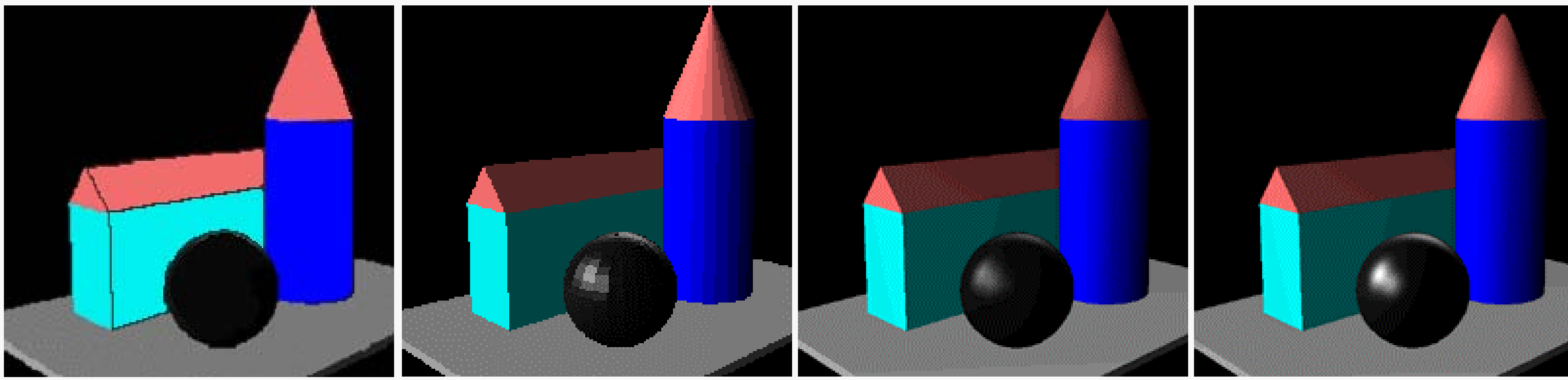
Exkurs: Fotorealismus, Echtzeit, Arithmetik

Prof. Dr. Aris Christidis

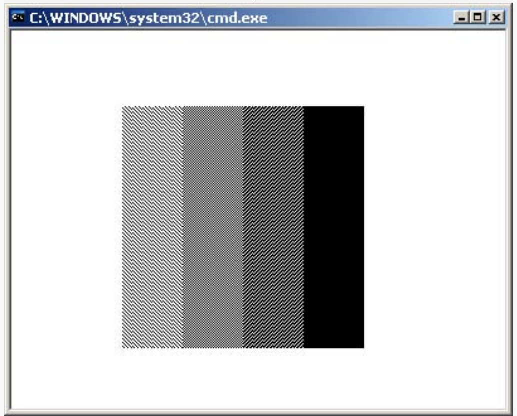
- Ziel vieler CG-Anwendungen: Fotorealismus in Echtzeit; wichtiges Mittel: stetige Helligkeits- / Farbübergänge – z.B.:

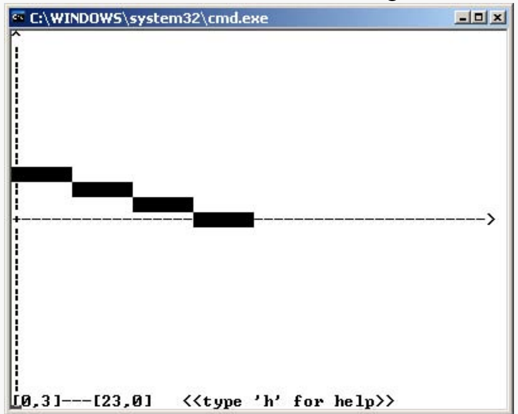


Bilder: www.glossar.de/glossar/z_shading.htm

Meistangewandte math. Methode: lineare Interpolation

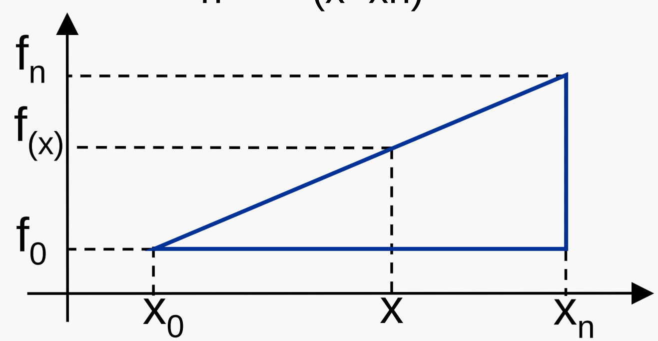
Zuordnung d.Werte einer abhängigen Var $y \in \mathbf{N}_0$ (Pixel-Helligkeit,-Farbe) d.Werten einer unabhängigen Variablen $x \in \mathbf{N}_0$ (Ort).





Exkurs: Fotorealismus, Echtzeit, Arithmetik

Zur Erinnerung: Lineare Interpolation im Wertebereich zwischen Anfangswert $f_0 = f_{(x=x0)}$ und Endwert $f_n = f_{(x=xn)}$ über den Definitionsbereich $x_0 \ldots x_n$
(typischer Funktionsname: **`lerp()`**)



$$(f_{(x)} - f_0) : (x - x_0) = (f_n - f_0) : (x_n - x_0)$$

$$\begin{aligned} f_{(x)} &= x \bullet (f_n - f_0) / (x_n - x_0) + f_0 - x_0 \bullet (f_n - f_0) / (x_n - x_0) \\ &= x \bullet m + b \qquad (m=\text{const.}; b=\text{const.}) \end{aligned}$$

$$\begin{aligned} f_{(x+1)} &= (x+1) \bullet (f_n - f_0) / (x_n - x_0) + f_0 - x_0 \bullet (f_n - f_0) / (x_n - x_0) \\ &= f_{(x)} + (f_n - f_0) / (x_n - x_0) \\ &= f_{(x)} + m \end{aligned}$$

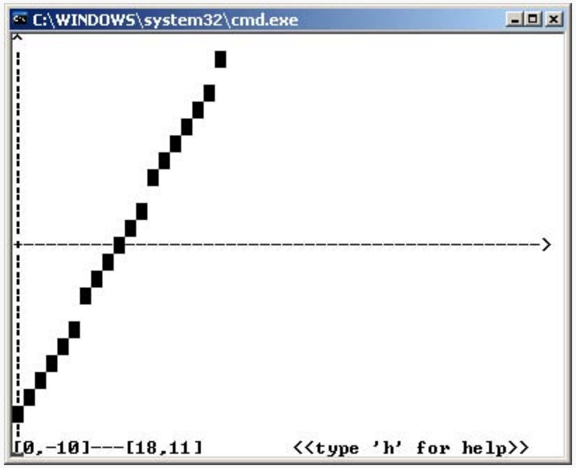
`float`/ **`int`**-Umwandlung:
`BruteForce`-Variante

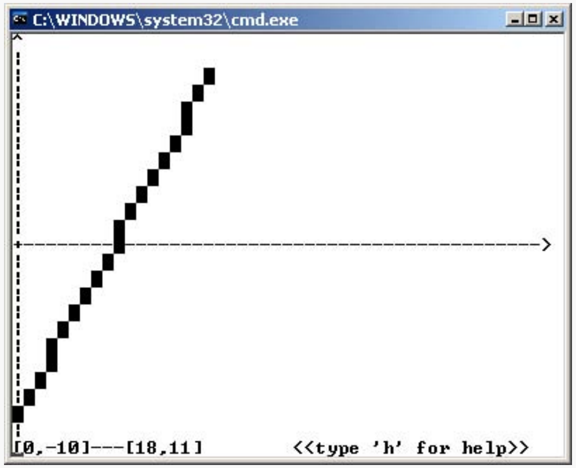
- Wunsch: Vermeidung (langsamer) Floating-Point- durch Einsatz geeigneter Ganzzahl-Arithmetik (vgl. Linien-Algorith.)

Exkurs: Fotorealismus, Echtzeit, Arithmetik

Unterscheidung zwischen Algorithmus zur Linienziehung und zur linearen Interpolation bei diskreten Systemen:

- Interpolation von (x_0,y_0) nach (x_n,y_n) ordnet (linear) jedem x-Wert im Definitionsbereich $x_0 \leq x \leq x_n$ genau einen y-Wert zu.
- Linie von (x_0,y_0) nach (x_n,y_n) ordnet (linear) jedem x-Wert im Definitionsbereich $x_0 \leq x \leq x_n$ max. so viele y-Werte zu, daß jedem y-Wert $y_0 \leq y \leq y_n$ ein x-Wert zugeordnet ist.



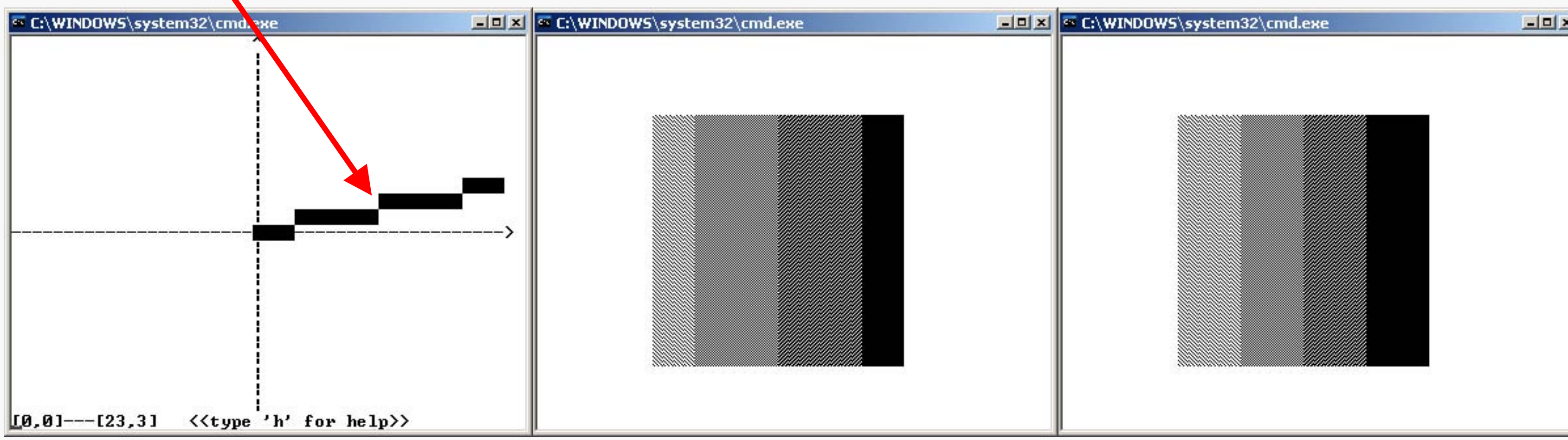


⇨ Bei Anpassung des Bresenham-Algorithmus: geänderte Behandlung des 2. und des 7. Oktanten; außerdem:  

Exkurs: Fotorealismus, Echtzeit, Arithmetik

Prof. Dr. Aris Christidis

- Weiterer Anpassungsbedarf des „Midpoint Algorithm“: „Plateau“-Bildung im mittleren Teil der Bresenham-Linie:



⇨ <u>Gegeben:</u> (n+1) Werte $x_0,...,x_n$ einer unabhängigen Variablen $x \in \mathbf{N}_0$ mit $x_{i+j}=x_i+j$ ($i,j \in \mathbf{N}_0$, $0 \leq i+j \leq n$ ⇨ $x_n=x_0+n$); zudem: x_0 u. x_n zugeordnete Werte y_0 u. y_n einer abhängigen Variablen $y \in \mathbf{Z}$

<u>Gesucht:</u> n-1 Werte $y_i \in \mathbf{Z}$ im abgeschlossenen Intervall $[y_0,y_n]$, die den Werten x_i ($i=1,...,n-1$) zugeordnet werden und eine auf ganzzahlige Werte gerundete lineare Interpolation zwischen (x_0,y_0) und (x_n,y_n) wiedergeben

Exkurs: Fotorealismus, Echtzeit, Arithmetik

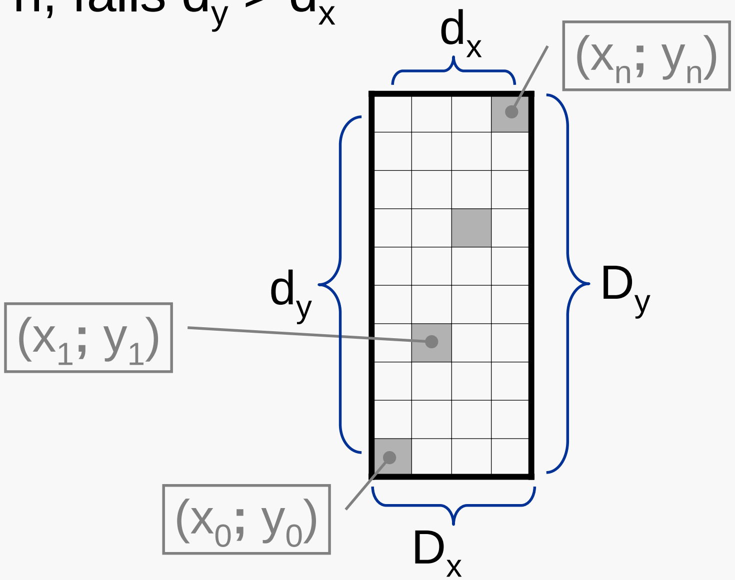
Vorbetrachtungen, Notation, Methodik:

- mit den max. Werte-Differenzen $d_x=x_n–x_0$ und $d_y=y_n–y_0$ gilt für die Anzahl möglicher x- bzw. y-Werte (Wertevorrat):

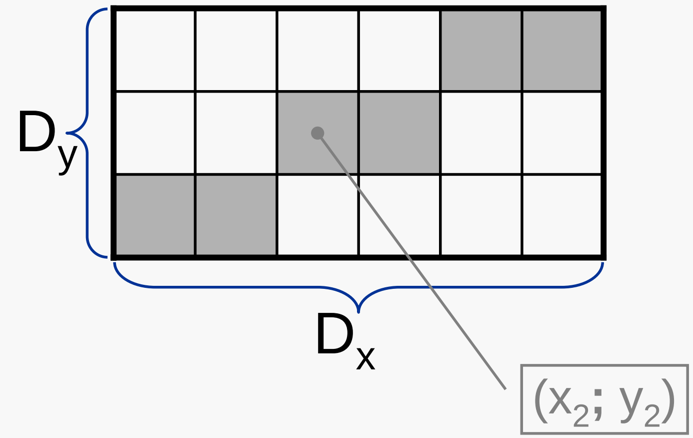
  Wertevorrat in x: $D_x = d_x+1 = x_n–x_0+1 = n+1$

  Wertevorrat in y: $D_y = d_y+1 = y_n–y_0+1 \begin{cases} < n, \text{ falls } d_y < d_x \\ = n, \text{ falls } d_y = d_x \\ > n, \text{ falls } d_y > d_x \end{cases}$

- Herleitung mit Ganzzahl-Arithmetik, ausgehend von den jeweils günstigsten Kostellationen (d.h.: solchen mit intuitiven, ganzzahligen Rechenergebnissen)



Exkurs: Fotorealismus, Echtzeit, Arithmetik



- <u>1. Oktant</u> ($d_x > d_y$)

  Günstige Konstellation bei:

  $D_x = p \cdot D_y$; $p \in \mathbf{N}$

  Zeilenwechsel zu $y_i = y_{i-1} + 1$, wenn x_i höchstens 0,5 hinter dem exakten Interpolationswert liegt:

  $(x_i - x_0 + ½) : (y_i - y_0) \geq D_x : D_y$ mit $y_i \neq y_0$, bzw.:

  $$\underbrace{\mathbf{2 \cdot D_y \cdot (x_i - x_0) - 2 \cdot D_x \cdot (y_i - y_0)}}_{\mathbf{dv}} \geq \mathbf{-D_y}$$

  <u>Algorithmus-Entwurf</u>, in Anlehnung an Bresenham:

  Eine „decision variable“ **dv** für den Zeilenwechsel wird eingerichtet; sie wird mit **`-2*Dx`** initialisiert (Check für y_1) und für jeden neuen **x**-Wert um **`2*Dy`** erhöht. Zeilenwechsel (**y++**) erfolgt, sobald **`dv≥-Dy`** wird; dann wird **dv** zudem um **`2*Dx`** reduziert.

Exkurs: Fotorealismus, Echtzeit, Arithmetik

- 2. Oktant ($d_x \le d_y$)

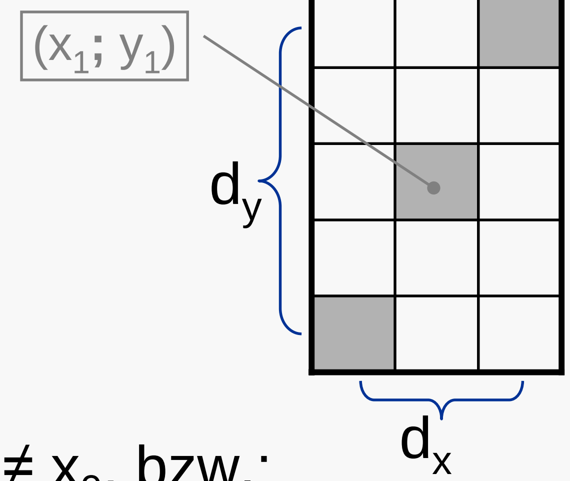
  Günstige Konstellation bei:

  $d_y = q \cdot d_x \;;\; q \in \mathbf{N}$

  Erhöhung von y_i solange, bis es höchstens 0,5 unterhalb des exakten Interpolationswertes liegt:

  $d_y : d_x \;\ge\; (y_i - y_0 + ½) : (x_i - x_0)$ mit $x_i \ne x_0$, bzw.:

  $$\underbrace{\mathbf{2 \cdot d_y \cdot (x_i - x_0) - 2 \cdot d_x \cdot (y_i - y_0)}}_{\mathbf{dv}} \;\ge\; \mathbf{d_x}$$



Algorithmus-Entwurf, in Anlehnung an Bresenham:

Eine „decision variable“ **dv** für den Spaltenwechsel wird eingerichtet (Initialisierung mit **0**); sie wird für jeden neuen **x**-Wert um **2*dy** erhöht. Solange **dv>=dx** gilt, werden **y** um **1** inkrementiert und **dv** um **2*dx** dekrementiert.

Übung:

Implementierung einer Funktion zum echtzeitfähigen Belegen eines Feldes mit den Werten einer ganzzahligen linearen Interpolation; die Funktion selbst soll nur Ganzzahl-Additionen und Subtraktionen enthalten (s. Übungsblatt).

```
int set_iLerp (int Dx, int y0, int yn, int *yStore)
```

BresenLerp