Graustufendarstellung

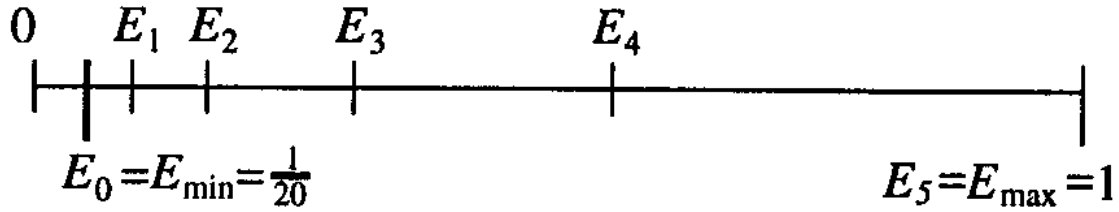
Diskrepanz zwischen (physikalischer) Luminanz und (wahrgenommener) Helligkeit von Graustufen:

Wahrnehmung reagiert in Relation zur empfangenen Lichtenergie in etwa logarithmisch

(+ weitere Einflüsse durch Monitor bzw. Art des Papiers)

$\Rightarrow$ *Transferfunktion* erforderlich, die aus theoret. Grauwert den adäquaten technischen Helligkeitswert berechnet

z.B. für 6 Graustufen E_0 bis E_5:



Die Transferfunktion kann an Eigenschaften der Hardware angepasst sein und tabellarisch (durch Stützstellen und Interpolation) spezifiziert sein.
Möglichkeit der Spezifikation in PostScript mit **`settransfer`** (siehe spätere Übung).

Kleinste realisierbare Intensität E_{min} (i. allg. > 0): abhängig vom Ausgabemedium.

Bildschirme	0,005–0,025
Foto-Abzüge	0,01
Dias	0,001
Zeitungspapier	0,1

(nach Bungartz et al. 1996)

Sinnvolle Anzahl von Graustufen: $n < 1 - \frac{\ln(E_{min})}{\ln(1{,}01)}$

Darstellung der Intensitätsstufen:

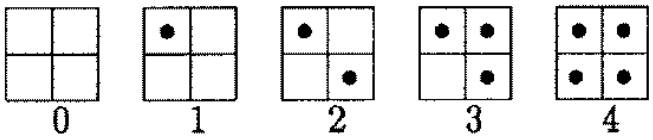
- auf Hardwareseite mehrere Intensitäten pro Pixel

oder

- <u>Halbtonverfahren</u>:

Einteilung des Bildschirms in Makropixel
Wahrnehmung fasst nahe zusammenliegende Pixel-Werte zu Graustufe zusammen

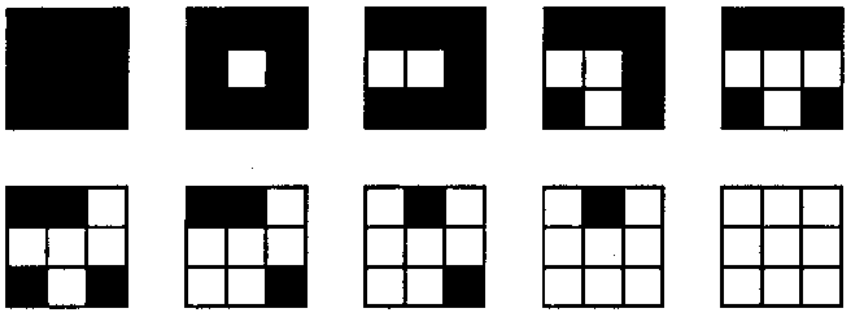
Beispiel: 2 x 2 - Halbtonmatrix



$$\begin{pmatrix} 1 & 3 \\ 4 & 2 \end{pmatrix}$$

3 x 3:

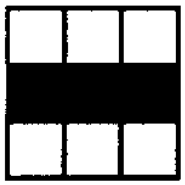
$$\begin{pmatrix} 6 & 8 & 4 \\ 1 & 0 & 3 \\ 5 & 2 & 7 \end{pmatrix} \triangleq$$



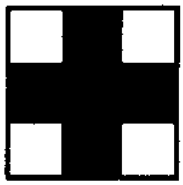
Wahl der Matrix ist nicht beliebig:

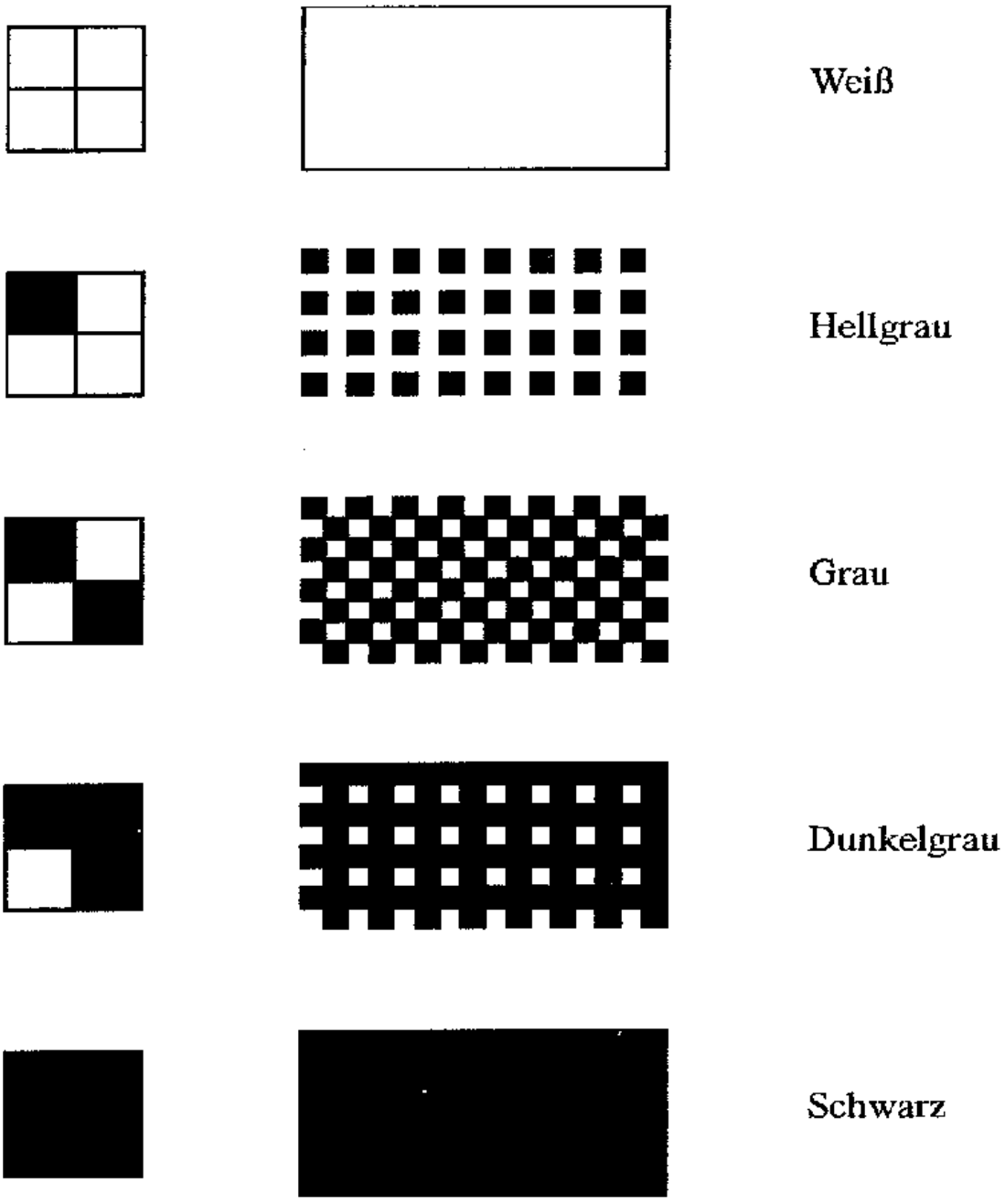
- (a) Vermeidung optischer Artefakte (Linienmuster)
- (b) hardwareabhängig: Vermeidung des Setzens benachbarter Pixel

a)



b)

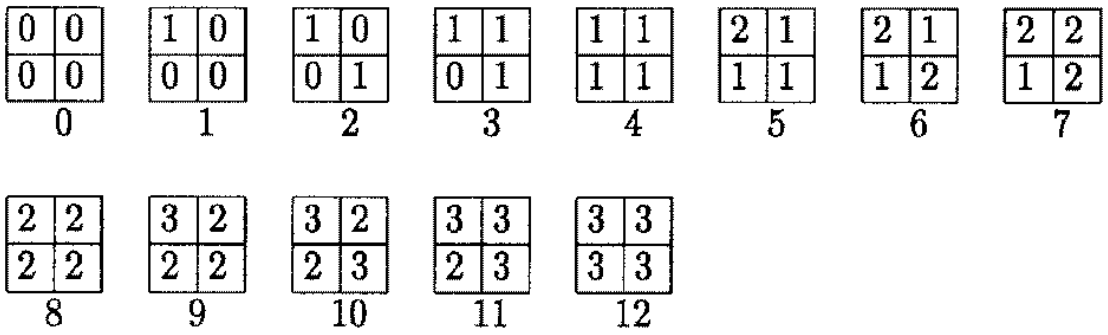




Vermeidung regelmäßiger Muster:

- Wechsel zwischen mehreren Halbtonmatrizen
- Zufallsauswahl der gesetzten Pixel

Verfahren der Halbtonsimulation auch für Ausgabegeräte mit mehr als 2 darstellbaren Intensitätsstufen anwendbar
z.B. bei 4 Intensitätsstufen 0–3:



Dither-Verfahren:

Intensitätswert wird nicht für Makro-Pixel berechnet, sondern für jedes einzelne Pixel. Darstellung in Abhängigkeit von der Position in einer Dither-Matrix (deren Kopien periodisch den Bildschirm überdecken).

```
for (y=0; y < rows; y++)
   {
   for (x=0; x < columns; x++)
      {
      i = x mod n; j = y mod n;
      if (intensity(x, y) > dither[i, j])
         set_pixel(x, y, 1);
      }
   }
```

Einträge in der Dither-Matrix = Schwellenwerte für die Bildschirmpixel

Wahl der Dither-Matrix:
Vermeidung artifizieller Muster in Bereichen konstanter Intensität

Beispiele:

2	6	4
5	0	1
8	3	7

0	8	2	10
12	4	14	6
3	11	1	9
15	7	13	5

21	10	17	14	23
15	2	6	4	9
20	5	0	1	18
12	8	3	7	13
24	18	19	11	22

(aus Rauber 1993)

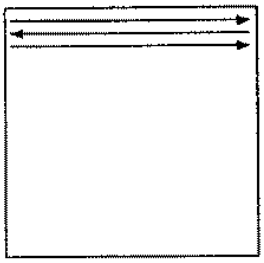
Vorteil gegenüber Halbton:
bei Vergrößerung von *n* erhöht sich Anzahl der darstellbaren Graustufen ohne merkliche Änderung der räumlichen Auflösung (bis zu Schwellenwert für *n*: bei Auflösung 1024 x 1024 ca. *n* = 10 bis 16).

<u>Fehlerverteilungsverfahren:</u>

Grundidee: Bei Darstellung des Intensitätswerts in einem Pixel, für den nur diskrete Intensitätswerte möglich sind, wird bei Rundung auf den nächstgelegenen möglichen Wert ein Fehler gemacht. Man versucht, diesen Fehler in den Nachbarpixeln auszugleichen.

Verschiedene Varianten: unterscheiden sich in Reihenfolge der Pixel und Auswahl der Pixel für die Fehler-(ausgleichs-) Verteilung.

Floyd-Steinberg-Verfahren: 3 benachbarte Pixel, Zickzack-Abarbeitung.



Lauf nach rechts

(x,y)	$\frac{3}{8}$
$\frac{3}{8}$	$\frac{1}{4}$

Lauf nach links

$\frac{3}{8}$	(x,y)
$\frac{1}{4}$	$\frac{3}{8}$

Vorteil:

- Verf. liefert oft gute Ergebnisse (besser als Halbton u. Dither)

Nachteile:

- sequentiell
- in ungünstigen Fällen Akkumulation des Fehlers, "Geisterbilder"

Vermeidung dieser Nachteile:

<u>Fehlerdiffusions-Verfahren</u>

Kombination von Dithering und Fehlerverteilung

Bildschirm wird von Kopien einer *Diffusionsmatrix* überdeckt (analog zum Dither-Verfahren) $\Rightarrow$ Einteilung der Pixel in n^2 Klassen

Pixel-Intensitätswerte werden klassenweise berechnet
Fehler wird auf die benachbarten Pixel verteilt, die noch nicht berechnet sind

Wahl der Diffusionsmatrix:
möglichst wenige Einträge, die keinen oder nur einen Nachbarn mit größerem Eintrag haben (weil dort der Fehler gar nicht oder nur an 1 Nachbarn verteilt werden kann).

Beispiele für Diffusionsmatrizen:

34	48	40	32	29	15	23	31
42	58	56	53	21	5	7	10
50	62	61	45	13	1	2	18
38	46	54	37	25	17	9	26
28	14	22	30	35	49	41	33
20	4	6	11	43	59	57	52
12	0	3	19	51	63	60	44
24	16	8	27	39	47	55	36

25	21	13	39	47	57	53	45
48	32	29	43	55	63	61	56
40	30	35	51	59	62	60	52
36	14	22	26	46	54	58	44
16	6	10	18	38	42	50	24
8	0	2	7	15	31	34	20
4	1	3	11	23	33	28	12
17	9	5	19	27	49	41	37

Diffusionsverf. besonders bei Druckeranwendungen dem Dithering und dem Fehlerverteilungsverf. überlegen.
Parallelisierbar, keine Geisterbilder.

Auswahl der Farbschattierungen (Helligkeitsstufen) für die Grundfarben R, G, B:

- logarithmische Wahrnehmungsskala ähnlich wie bei Graustufen
- zusätzlich Gamma-Korrektur des Bildschirms zu beachten (Nichtlinearität in Luminanzantwort des Bildschirm-Phosphors)

Belegung der Farbtabelle (Color Lookup Table):

Ziel: möglichst gute Annäherung der Farben des darzustellenden Bildes durch die Farben in der Tabelle

uniforme Quantisierung
(vgl. Kap. 3)
feste Zuordnung unabh. vom Bild, 3 Bit für Rotstufe, 3 Bit für Grünstufe, 2 Bit für Blaustufe

Standardbelegung der Tabelle bei wenigen darzustellenden Farben, z.B. bei grafischen Benutzungsoberflächen

Popularitätsalgorithmus

Belegung der Farbtabelle mit den 256 häufigsten Farbwerten des darzustellenden Bildes
Für im Bild auftretenden Farbwert wird der *nächstliegende* Farbwert der Tabelle verwendet

Farbabstände:

- euklidische Metrik im RGB-System

$$d(c_1, c_2) = \sqrt{(R_1 - R_2)^2 + (G_1 - G_2)^2 + (B_1 - B_2)^2}$$

zur Vermeidung der Wurzelberechnung häufig stattdessen:

- *Manhattan-Metrik*

$$d_m(c_1, c_2) = | R_1 - R_2 | + | G_1 - G_2 | + | B_1 - B_2 |$$

Einsparen der aufwändigen Minimumsuche für die Abstände:
Merken bereits aufgetretener Farbwerte und ihrer zugeordneten Tabelleneinträge in weiterer Tabelle.

Nachteil des Popularitätsalgorithmus:
farbliche Details in kleinen Bildbereichen können völlig falsch dargestellt werden!

Vermeidung:

Median-Schnitt-Algorithmus

unterteilt den RGB-Einheitswürfel sukzessive in Teilquader (Schnitte parallel zu Koordinatenebenen)

1. bestimme kleinsten Teilquader des Würfels, der alle Pixel des Bildes enthält
2. teile diesen so, dass in beiden Hälften (etwa) gleichviele Pixel enthalten sind ("Median-Schnitt")
3. kontrahiere beide Teilquader auf ihre Extremkoordinaten (vgl. Schritt 1)

4. wende hierauf wieder Schritt 2 an...
bis 256 Quader erzeugt sind oder kein Quader mehr geteilt werden kann
- berechne dann für jeden Teilquader Mischfarbe durch gewichtete Mittelwertbildung der enthaltenen Pixel-Werte
diese Farbe wird in die Farbtabelle eingetragen

Bildschirm-Darstellung: jedes Pixel erhält Farbwert des Quaders, in dem es liegt

Datenstruktur: BSP-Baum (*binary space partitioning*)
Bestimmung eines konkreten Tabellen-Wertes zu einem Farbwert des Bildes durch top-down-Lauf durch den BSP-Baum.

Vorteil: sehr gute Farbdarstellung
- noch weiter verbesserbar durch nachträgl. Anwendung des Floyd-Steinberg-Fehlerverteilungsverfahrens

Nachteil: hoher Speicherplatzbedarf, hohe Laufzeit
(alle Pixel werden im BSP-Baum gespeichert)

"Kompromisslösung": Octree-Quantisierung

regelmäßige rekursive Unterteilung des RGB-Würfels in 8 gleichgroße Unter-Würfel

- Aufbau des Octrees, bis jeder Teilwürfel nur noch 1 Pixel enthält
- Reduktion von den Blättern her (Mittelwertbildung für je 8 Tochter-Würfel), bis der Octree genau 256 Blattknoten hat

(Aufbau und Reduktion können speicherplatzsparend auch verschränkt werden)

Antialiasing

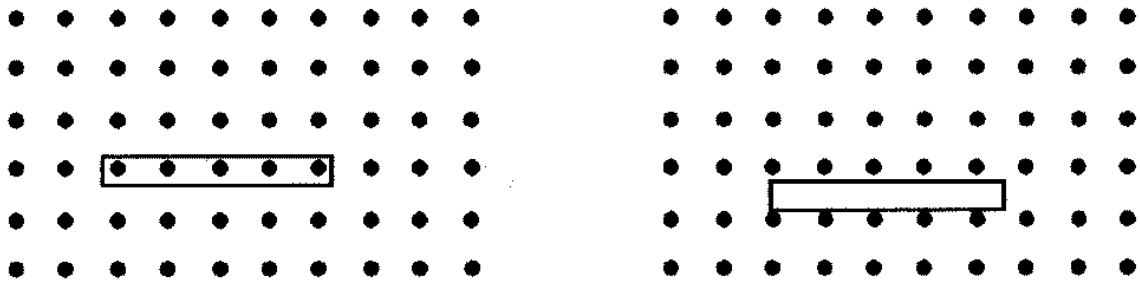
Aliasing (Aliasierung):
Sammelbez. für Verfremdungseffekte, die durch die Rasterkonvertierung stetiger Objekte entstehen

- Lattenzaun-Problem: Objekt mit regelmäßigem, periodischem Aufbau ist nicht verträglich mit der Rasterung

⇒ Teile verschwinden oder werden unregelmäßig dargestellt

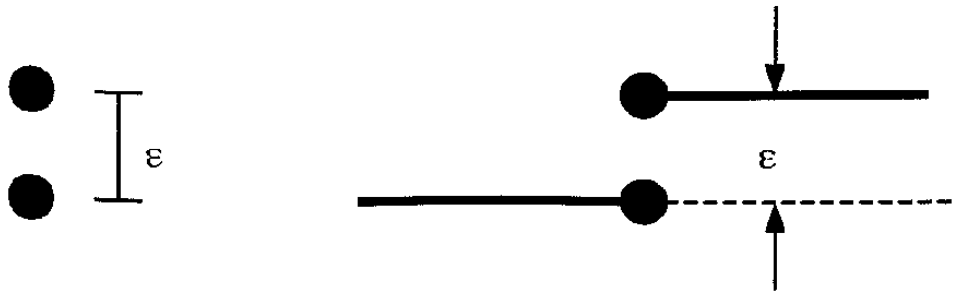
- *Moiree-Effekte* bei Objekte mit regelmäßiger Textur, Schraffierung etc.: Vortäuschen großflächiger Muster

- dünne Objekte werden "verschluckt" oder verfremdet



- Treppenstufen-Effekte bei schrägen Linien und bei Kurven

Problem hierbei: Abstands-Auflösung des visuellen Systems ist bei Linien exakter als bei einzelnen Punkten
⇒ Treppenstufen sichtbar, auch wenn Pixel nicht unterscheidbar!



Theoretische Grundlage für Aliasing-Probleme:
Signaltheorie

In der Fourieranalyse wird jede Funktion *f* als Überlagerung von trigonometrischen Funktionen (sin und cos) mit verschiedenen Frequenzen dargestellt.

$$f(x) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{+\infty} F(u)(\cos(ux) - i\sin(ux))du$$

$$= \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{+\infty} F(u)e^{-iux}du$$

Für viele Funktionen gibt es eine Grenzfrequenz u_G, nach der die Berechnung des Integrals abgebrochen werden kann:

$$f(x) = \frac{1}{\sqrt{2\pi}} \int_{-u_G}^{+u_G} F(u)e^{-iux}du$$

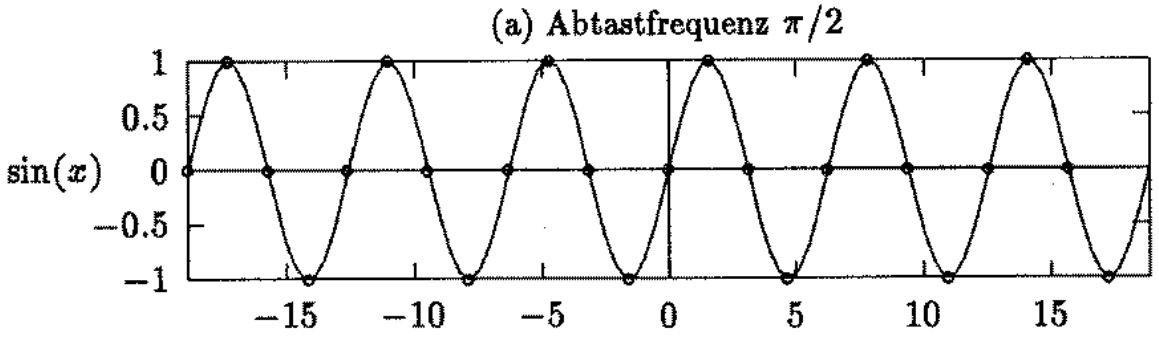
Eine kontinuierliche Funktion mit endlicher Grenzfrequenz kann vollständig durch *Abtastwerte* mit gleichem Abstand δx repräsentiert werden, wenn δx weniger als halb so groß wie die Periode der Grenzfrequenz ist:

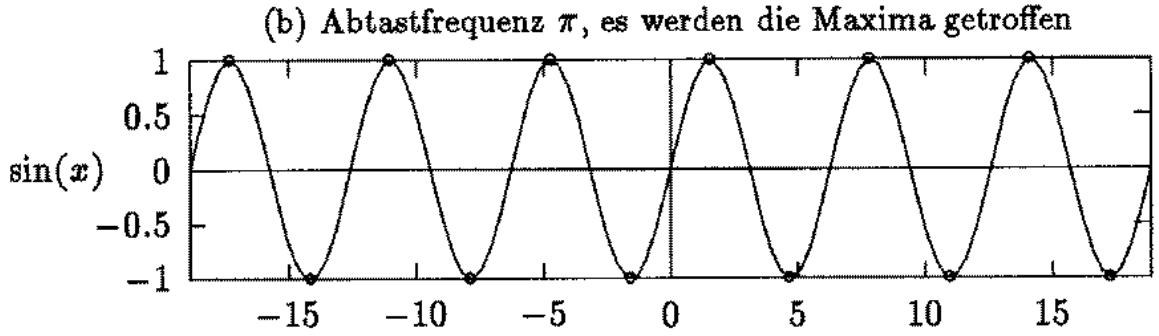
$$\delta x < \frac{1}{2u_G}$$

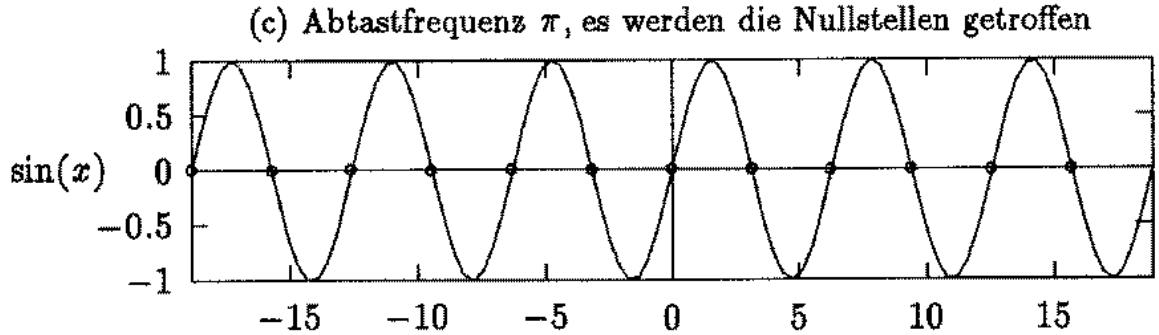
Die entspr. untere Grenze für die Abtast-Frequenz heißt *Nyquist-Frequenz*:

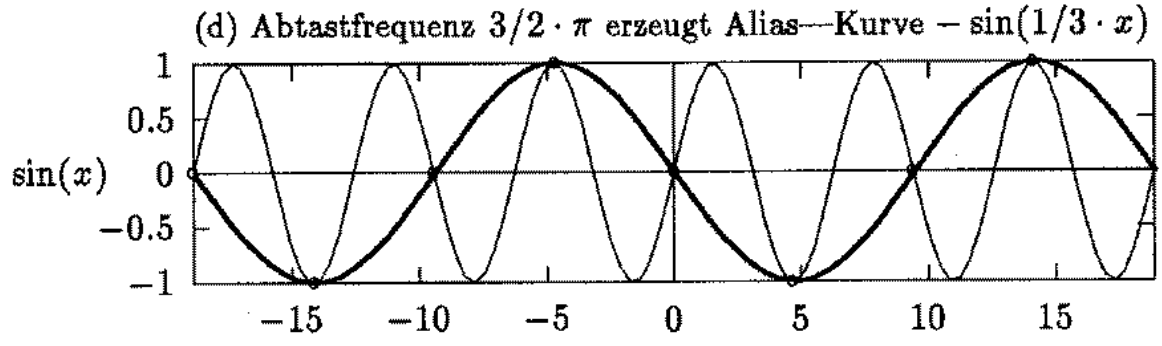
$u_N = 2\, u_G$.

Bei Abtastung mit einer Freq. > 2*u* kann eine Sinuskurve der Frequenz *u* rekonstruiert werden; bei kleineren Frequenzen kommt es zum Aliasing:







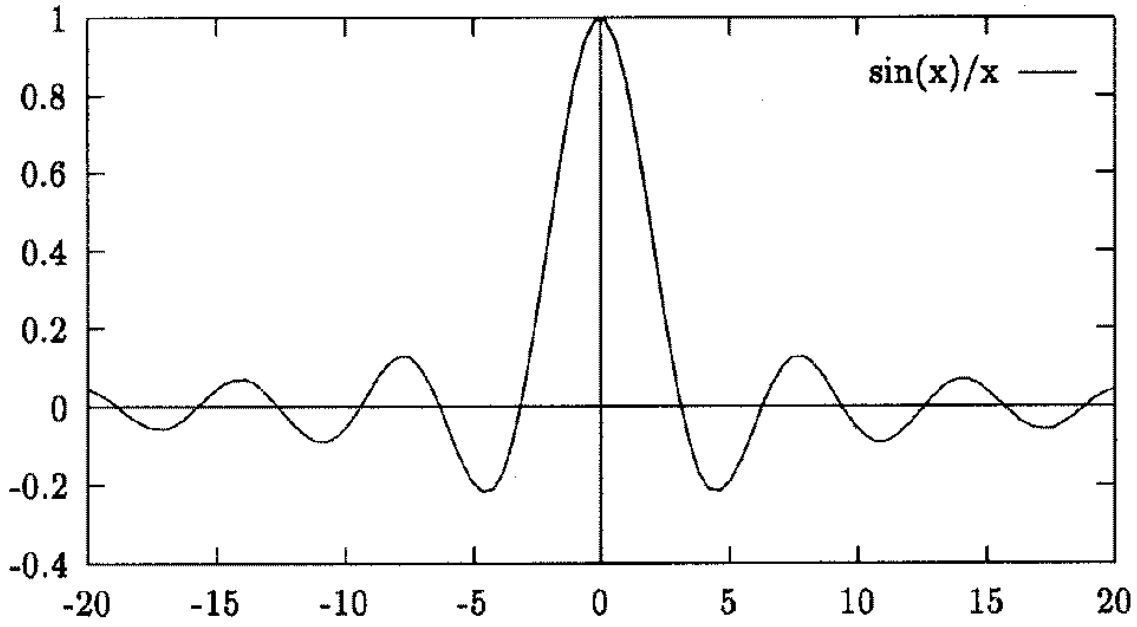


Antialiasing-Techniken:

- *Prefiltering* (Vorfilter-Techniken): Vermeidung des Aliasing *vor* der Bildschirmdarstellung durch geeignete Bild-manipulation
- *Postfiltering* (Nachfiltern): Abschwächung des Aliasing durch Nachbehandlung des bereits gerasterten Bildes
- Modifikation der Zeichenalgorithmen

<u>Prefiltering</u>:
Abschneiden der höheren Frequenzen in der Fourierdarstellung (vgl. JPEG-Kompression!)

rechnerisch durch Konvolution (Faltung) der Funktion mit einem Vielfachen der Funktion (sin *x*)/*x* (= der umgekehrten Fourier-Transformierten einer Rechteckfunktion):



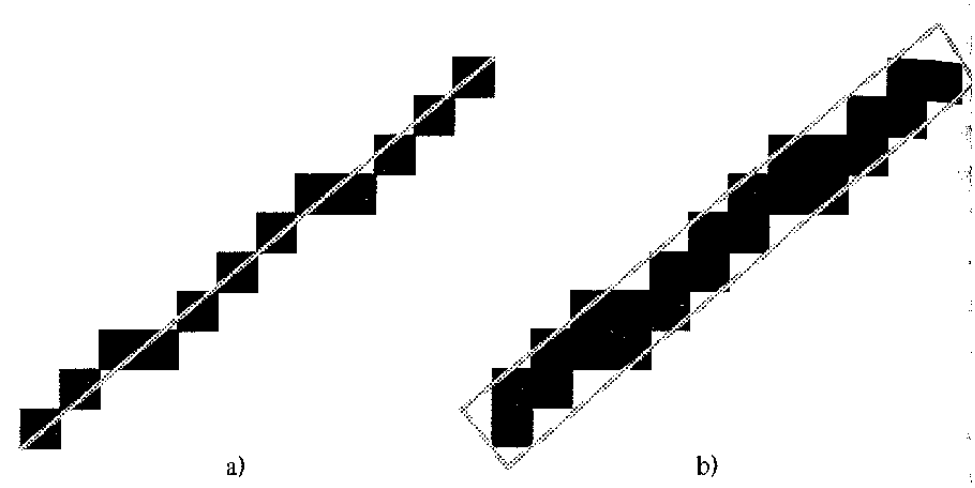
Abschneiden der Funktion erforderlich ⇒ nicht alle höheren Frequenzen verschwinden
dennoch sehr gute Qualitäten erreichbar
Nachteil: hoher Rechenaufwand
⇒ Verwendung anderer Filterfunktionen

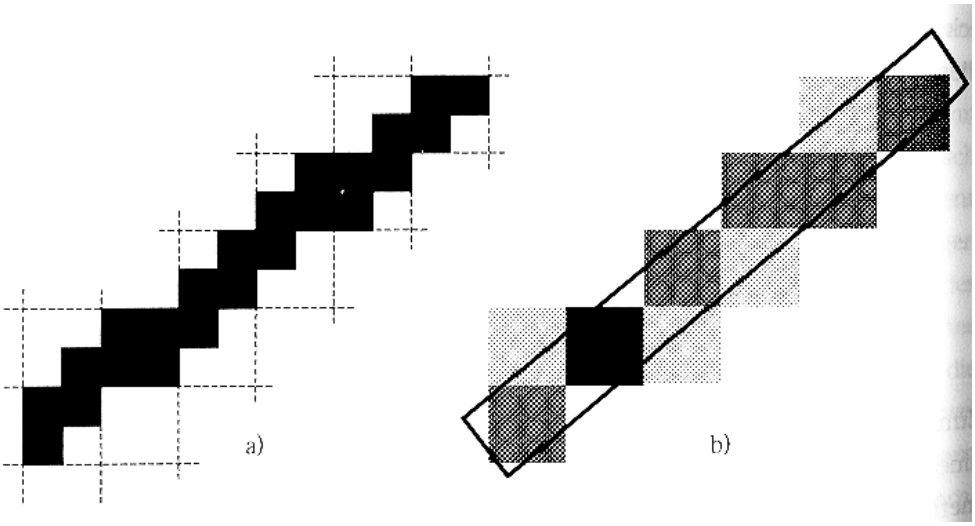
oder: heuristische Techniken

2 Techniken, die häufig für Linien angewandt werden:

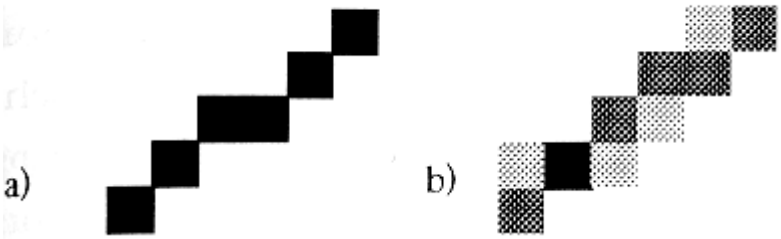
1.

- Zeichnen in doppelter Auflösung
- anschließende Verdopplung
- anschließende Zuordnung eines Grauwertes je nach Bedeckungsgrad der 4x4-Makropixel (die den echten Pixeln entsprechen)





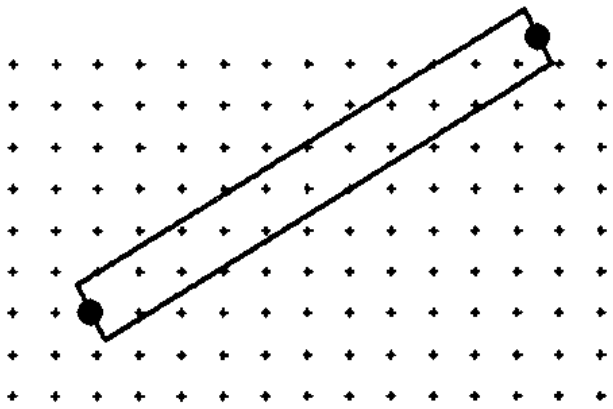
Ergebnis: "verschmierte" Linie, die in der Wahrnehmung aber "glatter" erscheint
Qualität: ca. entspr. doppelter Bildauflösung



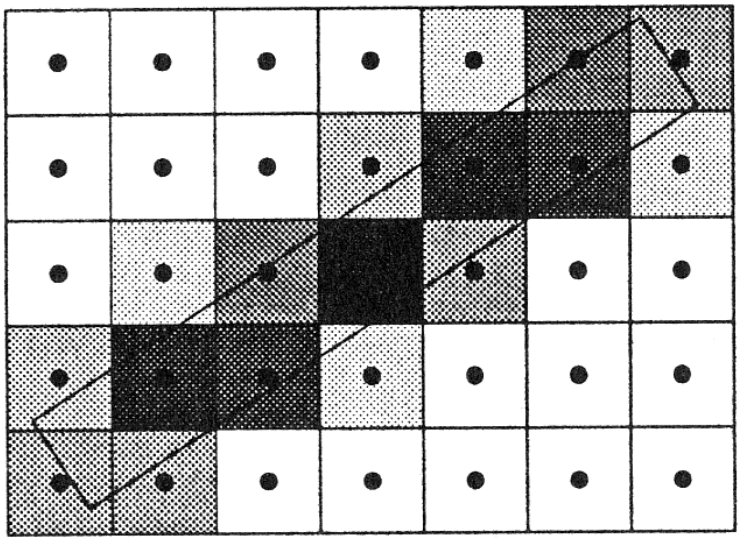
Gegenüberstellung der Rasterdarstellung einer Linie
(a) ohne Anti-Aliasing
(b) mit Anti-Aliasing
(c) mit doppelter Auflösung, ohne Anti-Aliasing:

2.
Area Sampling:

- Modellierung der Linie als rechteckiger Bereich



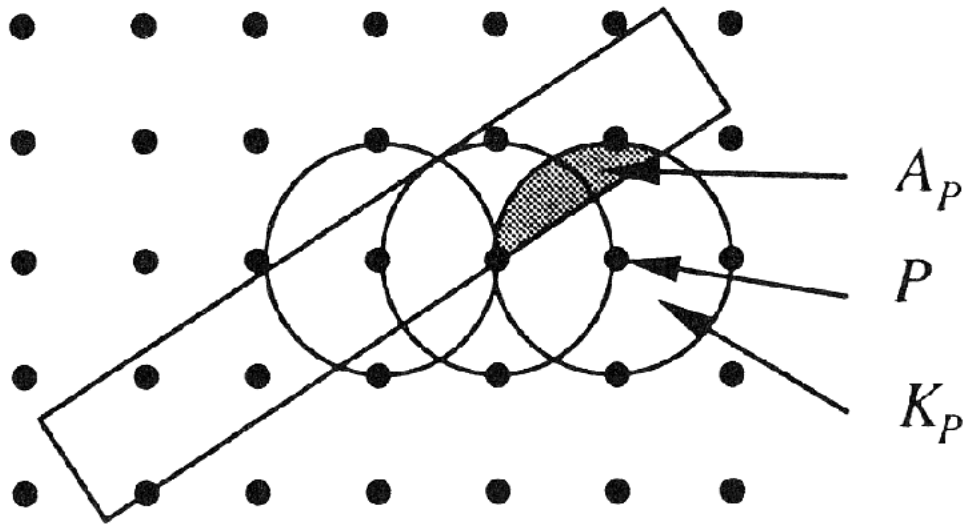
- Bestimmung der Graustufe eines Pixels nach dem Flächenanteil des Pixel(-Quadrats), den das Rechteck überstreicht (*unweighted area sampling*)



oder

- Berücksichtigung der Lage der überdeckten Fläche im Pixel bei der Festlegung der Graustufe:
  Weighted area sampling

den Pixeln werden überlappende Kreisscheiben zugeordnet:



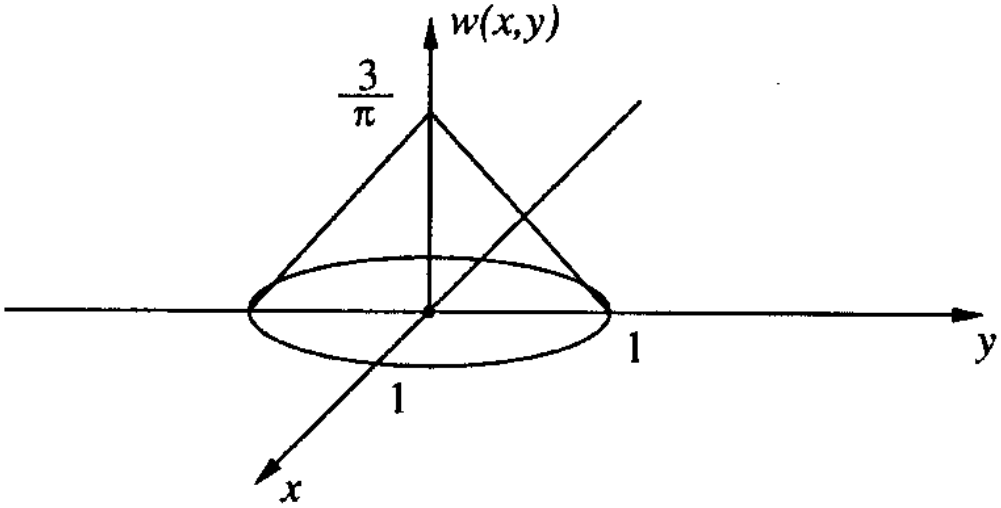
Pixel P, Kreisscheibe K_P, überdeckter Flächenteil A_P

Intensität des Pixels *P*:

$$I_P = \int_{A_P} w(x,y)\,dxdy$$

mit einer Gewichtsfunktion *w*, die $\int_{K_P} w(x,y)\,dxdy = 1$ erfüllt.

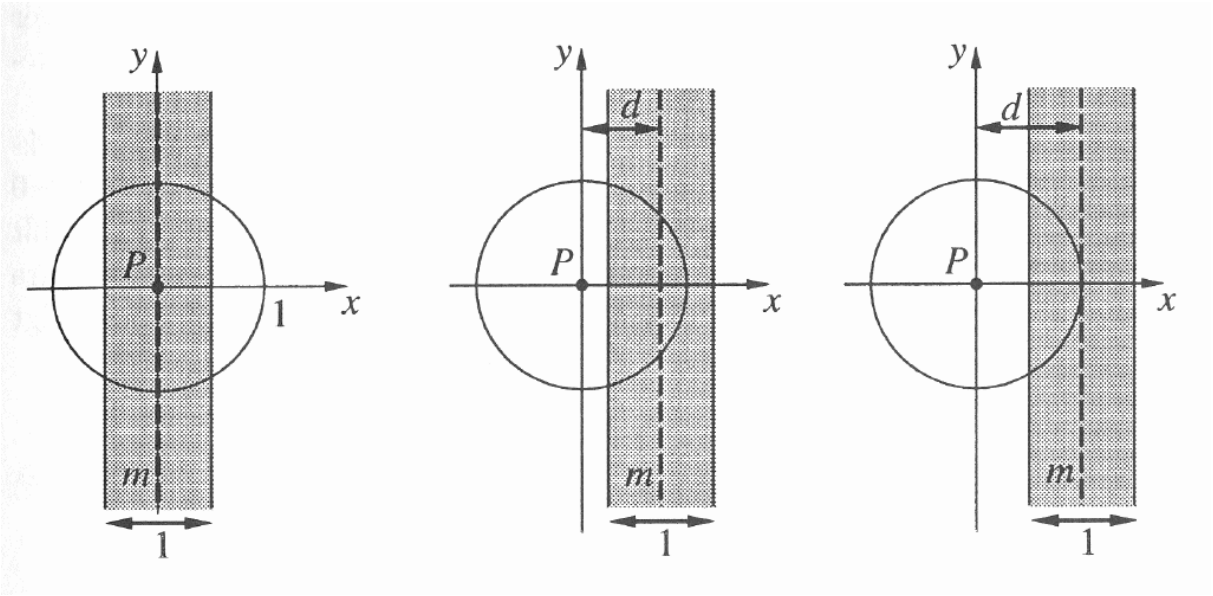
Beispielsweise *w* als Kegelfunktion:



Die Intensität hängt hier nur vom Abstand *d* der Pixelmitte zur Linien-Mittelachse ab ($0 \le d \le 1{,}5$).

Effiziente Implementation:

Algorithmus von Gupta und Sproull
Anstelle der Integralberechnung werden die Intensitätswerte in Abhängigkeit von *d* für 24 Werte von *d* ($d = i/16$, $i=0;...;23$) in einer Tabelle gespeichert. *d* kann beim Zeichnen inkrementell berechnet werden (vgl. Bresenham-Algorithmus).



Abhängigkeit der Intensität vom Abstand *d*.
Links: *d* = 0, Mitte: *d* = 0,75, rechts: *d* = 1,0.

Postfiltering-Techniken
(am häufigsten angewandte Antialiasing-Techniken)

Supersampling

Rasterung mit höherer Auflösung als der Bildschirm (Faktor 2 oder 4)
Intensitätswert eines Bildschirmpixels als Mittelwert der zugehörigen (4 oder 16) Pixel im errechneten Rasterbild

d.h.: Anwendung eines 2x2- oder 4x4-Filters auf das Rasterbild

$\frac{1}{16}$	$\frac{1}{16}$	$\frac{1}{16}$	$\frac{1}{16}$
$\frac{1}{16}$	$\frac{1}{16}$	$\frac{1}{16}$	$\frac{1}{16}$
$\frac{1}{16}$	$\frac{1}{16}$	$\frac{1}{16}$	$\frac{1}{16}$
$\frac{1}{16}$	$\frac{1}{16}$	$\frac{1}{16}$	$\frac{1}{16}$

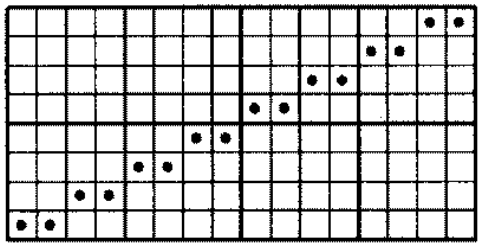
ungewichteter Filter

$\frac{1}{36}$	$\frac{2}{36}$	$\frac{2}{36}$	$\frac{1}{36}$
$\frac{2}{36}$	$\frac{4}{36}$	$\frac{4}{36}$	$\frac{2}{36}$
$\frac{2}{36}$	$\frac{4}{36}$	$\frac{4}{36}$	$\frac{2}{36}$
$\frac{1}{36}$	$\frac{2}{36}$	$\frac{2}{36}$	$\frac{1}{36}$

Filter mit stärkerer Gewichtung des mittleren Pixelblocks

Nachteile:

- Schwäche des Supersamplings bei Bildern mit dünnen Linien



0	0	$\frac{4}{16}$	$\frac{4}{16}$
$\frac{4}{16}$	$\frac{4}{16}$	0	0

Ergebnis des Supersamplings einer Linie mit 4 x 4-Filter mit gleichmäßiger Gewichtsverteilung. Die resultierenden Bildschirmpixel zeigen nur 1/4 der max. Intensität und sind ebenso gestuft wie das Original.

- hoher Rechenzeit- und Speicherplatzaufwand

Abhilfe: *adaptives Supersampling*. Gröbere Rasterung in Bereichen konstanter Intensität.

Filter-Methode (ohne Supersampling)

Es wird mit der Auflösung des Bildschirms gearbeitet. Die endgültige Darstellung wird durch gewichtete Mittelung der Intensitäten benachbarter Pixel erzeugt:
"Verschmieren" der Intensitäten der Pixel.

3 x 3 - Filter:

$\frac{1}{36}$	$\frac{1}{9}$	$\frac{1}{36}$
$\frac{1}{9}$	$\frac{4}{9}$	$\frac{1}{9}$
$\frac{1}{36}$	$\frac{1}{9}$	$\frac{1}{36}$

Für höhere Bildschirm-Auflösungen: 5 x 5 - Filter
(hier: *Bartlett-Filter*):

$\frac{1}{81}$	$\frac{2}{81}$	$\frac{3}{81}$	$\frac{2}{81}$	$\frac{1}{81}$
$\frac{2}{81}$	$\frac{4}{81}$	$\frac{6}{81}$	$\frac{4}{81}$	$\frac{2}{81}$
$\frac{3}{81}$	$\frac{6}{81}$	$\frac{9}{81}$	$\frac{6}{81}$	$\frac{3}{81}$
$\frac{2}{81}$	$\frac{4}{81}$	$\frac{6}{81}$	$\frac{4}{81}$	$\frac{2}{81}$
$\frac{1}{81}$	$\frac{2}{81}$	$\frac{3}{81}$	$\frac{2}{81}$	$\frac{1}{81}$

Die Summe der Gewichtsfaktoren ergibt 1.
Der Filter wird "über die Pixel des Rasterbildes geschoben".

Vor der Anwendung: Replikation der Randzeilen und -spalten.