# Mitschrift
## Computergrafik

Martin Lenders

30. April 2010

# Inhaltsverzeichnis


**1 Einführung** **5**
- 1.1 Organisatorisches . . . 5
  - 1.1.1 Übungsblätter: . . . 5
  - 1.1.2 Programmierung . . . 5
- 1.2 Übersicht . . . 5
  - 1.2.1 Fahrplan . . . 5

**2 Koordinatensysteme, geometrische Transformationen** **7**
- 2.1 kartesische Koordinaten . . . 7
- 2.2 Geometrische Transformationen . . . 7
- 2.3 Homogene Koordinaten . . . 8
  - 2.3.1 Allgemeine affine Transformation in homogenen Koordinaten . . . 9
- 2.4 Die projektive Ebene . . . 9
  - 2.4.1 Geraden in der projektiven Ebene . . . 9
  - 2.4.2 Modelle der projektiven Ebene . . . 11
  - 2.4.3 Projektive Punkte zu karthesische Koordinaten . . . 12
- 2.5 Allgemeine projektive Transformationen . . . 13
- 2.6 Transformation im dreidimensionalen Raum . . . 15
  - 2.6.1 Affine Transformation im dreidimensionalen Raum . . . 15
  - 2.6.2 projektive Transformationen im dreidimensionalen Raum . . . 16
- 2.7 Projektionen und Perspektive . . . 16
- 2.8 Koordinaten in der Praxis . . . 17
- 2.9 „rendering pipeline“ – vom geometrichen Modell zum Rasterbild . . . 22

**3 Licht und Farben** **23**
- 3.1 Farbensehen im menschlichen Auge . . . 23

**4 Rasterung von Strecken und Kreisen** **25**
- 4.1 Strecken . . . 25
  - 4.1.1 Bresenham-Algorithmus . . . 27
  - 4.1.2 Midpoint Line Algorithmus . . . 28

# 1 Einführung

## 1.1 Organisatorisches

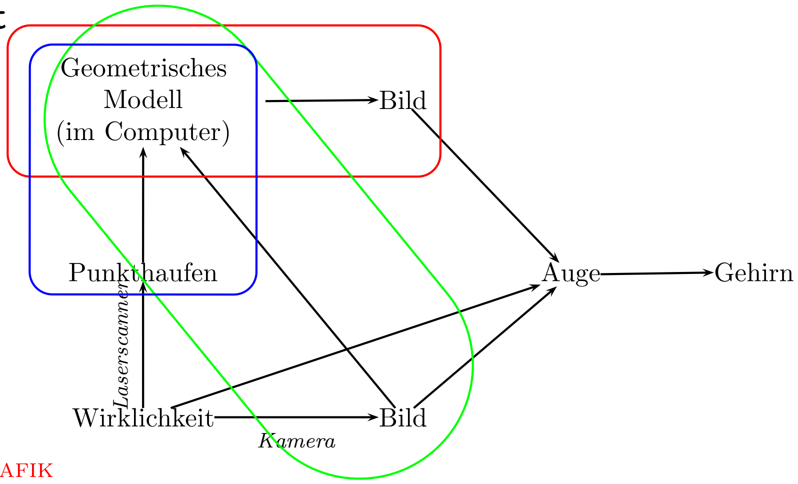
### 1.1.1 Übungsblätter:

- Ausgabe: Mittwoch, Abgabe: Freitag
- Abgabe in Zweiergruppen
- 60% der Punkte müssen erreicht werden
- min. einmal Vorrechnen

### 1.1.2 Programmierung

- Aufgaben in Java gestaltet
- mit OpenGL-Interface
- auf Nachfrage kann auch C/C++ verwendet werden

## 1.2 Übersicht



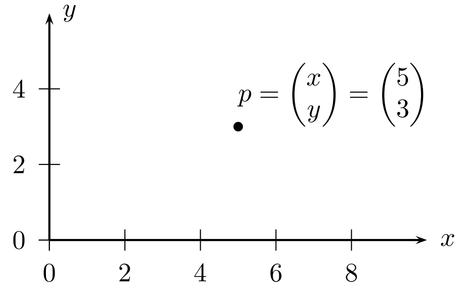

- Computergrafik
- Bildbearbeitung / Bilderkennung
- Geometrisches Rechnen / Geometrische Modellierung

### 1.2.1 Fahrplan

- Koordinatiensysteme, geometrische Transformationen
- Licht und Farben
- Rasterung
- Beleuchtung und Schattierung
- rendering-pipeline: vom Modell bis zum gerasterten Bildbearbeitung
- geometrische Modellierung: Kurven, Flächen und Splines
- Kein Anwendungskurs für OpenGL, JOGL, Javaview etc.!

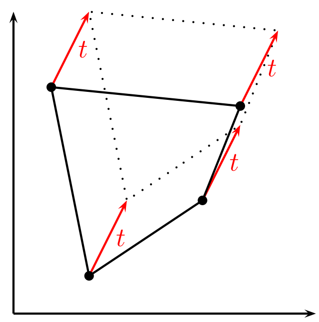
# 2 Koordinatensysteme, geometrische Transformationen

## 2.1 kartesische Koordinaten



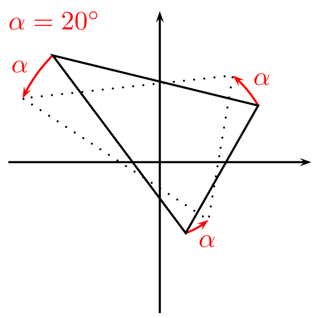

## 2.2 Geometrische Transformationen

- *Tranlation*: $p \mapsto p + t \qquad t \in \mathbb{R}^2$, Translationsvektor




- *Rotation* (um den Ursprung $\begin{pmatrix} 0 \\ 0 \end{pmatrix}$):

$$p \mapsto M \cdot p \qquad M = \begin{pmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{pmatrix}, \text{ Rotationsmatrix}$$




- *Rotation* um den Punkt $c$: $p \mapsto M(p - c) + c = Mp + (c - Mc), \qquad c \mapsto c$
- *gleichförmige Skalierung*:

$$p \mapsto \lambda \cdot p = \begin{pmatrix} \lambda & 0 \\ 0 & \lambda \end{pmatrix} \cdot p, \qquad \lambda \neq 0$$

$$\lambda = 1 \qquad p \mapsto -p = \begin{pmatrix} -1 & 0 \\ 0 & -1 \end{pmatrix} \cdot p = \text{Spiegelung am Ursprung} = \text{Rotation um } 180°$$

- *Ungleichförmige Skalierung*:

$$M = \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix} \qquad p \mapsto M \cdot p$$

$$\begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} \lambda_1 x \\ \lambda_2 y \end{pmatrix}$$

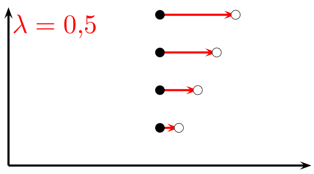
$$M = \begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix} \text{resultiert in der Spiegelung an der } x\text{-Achse}$$

$$M = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \text{resultiert in der Spiegelung an der } y\text{-Achse}$$

- *Scherung*

$$M = \overset{\text{Scherung auf der } x\text{-Achse}}{\begin{pmatrix} 1 & \lambda \\ 0 & 1 \end{pmatrix}} \left( \text{oder} \quad \overset{\text{Scherung auf der } y\text{-Achse}}{\begin{pmatrix} 1 & 0 \\ \lambda & 1 \end{pmatrix}} \right)$$

$$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \begin{pmatrix} 1 & \lambda \\ 0 & 1 \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} = \begin{pmatrix} x + \lambda y \\ y \end{pmatrix}$$




Flächeninhalt:

- Translationen, Rotationen, Scherungen und Spiegelungen ändern den Flächeninhalt <u>nicht</u>.
- Skalierung ändert den Flächeninhalt um den Faktor $\lambda_1 \cdot \lambda_2$

**Definition** Eine Verknüpfung mehrerer dieser Transformationen bildet eine **affine Transformation**. Allgemein ist diese:

$$p \mapsto M \cdot p = b, \qquad M \in \mathbb{R}^{2\times 2}, b \in \mathbb{R}^2, \det M \neq 0$$

Der Flächeninhalt ändert sich um den Faktor $\det M$

**Definition** Die Verknüpfung von Translation, Rotation und Spiegelung heißt **starre Bewegung** oder **Isometrie**. Allgemein ist diese:

$$p \mapsto Mp + t \text{ mit } \textbf{orthogonaler Matrix } M \text{ (d. h. } \det M = \pm 1)$$

die Isometrien zerfallen:

- **orientierungserhaltende** ($\det M = 1$) und
- **orientierungsumkehrende** ($\det M = -1$) Isometrien

## 2.3 Homogene Koordinaten

**Definition** **Homogene Koordinaten**: Statt $p = \begin{pmatrix} x \\ y \end{pmatrix}$ verwendet man eine dritte Koordinate $p = \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$

**Konvention** Die Koordinaten $\begin{pmatrix} x \\ y \\ z \end{pmatrix}$ und $\begin{pmatrix} \lambda x \\ \lambda y \\ \lambda z \end{pmatrix}$ stellen denselben Punkt dar ($\lambda \neq 0$)

Der Punkt $\begin{pmatrix} x \\ y \\ z \end{pmatrix}$ mit $z \neq 0$ hat die kartesischen Koordinaten $\begin{pmatrix} \frac{x}{z} \\ \frac{y}{z} \end{pmatrix}$

### 2.3.1 Allgemeine affine Transformation in homogenen Koordinaten

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} \mapsto \underbrace{\begin{pmatrix} m_{11} & m_{12} & b_1 \\ m_{21} & m_{22} & b_2 \\ 0 & 0 & 1 \end{pmatrix}}_{M'} \cdot \begin{pmatrix} x \\ y \\ z \end{pmatrix}$$

$$\begin{pmatrix} x \\ y \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} m_{11}x + m_{12}y + b_1 \\ m_{21}x + m_{22}y + b_2 \\ 1 \end{pmatrix}$$

Die Matrizen $M'$ und $\lambda M'$ beschreiben dieselbe Transformation ($\lambda \neq 0$)

$$p \mapsto M'p \text{ mit } M' = \begin{pmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ 0 & 0 & m_{33} \end{pmatrix} \text{ und } \det M' \neq 0$$

$$\det M' \neq 0 \Leftrightarrow m_{33} \neq 0 \wedge \begin{vmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{vmatrix} \neq 0$$

$\Rightarrow$ o. B. d. A. kann man auch $m_{33} = 1$ annehmen (Dann kann man die dritte Zeile auch weglassen).

## 2.4 Die projektive Ebene

**Definition** Die (reelle) **projektive Ebene** $P^2$ besteht aus den Äquivalenzklassen vo Punkten $\begin{pmatrix} x \\ y \\ z \end{pmatrix} \neq \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$, wobei $\begin{pmatrix} x \\ y \\ z \end{pmatrix}$ und $\begin{pmatrix} \lambda x \\ \lambda y \\ \lambda z \end{pmatrix}$ denselben Punkt darstellen ($\lambda \neq 0$)

### 2.4.1 Geraden in der projektiven Ebene

Gerade in $\mathbb{R}^2$ (karthesische Koordinaten):

$$y = ax + b \text{(Gerade darf nicht senkrecht sein)}$$

$$ax + bx = -c$$

$$\Updownarrow$$

Gerade in Homogenen Koordinaten

$$\begin{pmatrix} x \\ y \end{pmatrix} \longrightarrow \begin{pmatrix} x \\ y \\ 1 \end{pmatrix}$$

$$ax + by + c = 0 \Leftrightarrow \begin{pmatrix} a \\ b \\ c \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ 1 \end{pmatrix} = 0$$

Allgemeine Gleichung einer Geraden in $P^2$

$$\begin{pmatrix} a \\ b \\ c \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ z \end{pmatrix} = 0 \Leftrightarrow ax + by + cz = 0 \qquad \begin{pmatrix} a \\ b \\ c \end{pmatrix} \neq \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$$

Wenn $\begin{pmatrix} x \\ y \\ z \end{pmatrix}$ die Gleichung erfüllt, dann erfüllt auch $\begin{pmatrix} \lambda x \\ \lambda y \\ \lambda z \end{pmatrix}$ die Gleichung.

$\begin{pmatrix} a \\ b \\ c \end{pmatrix}$ und $\begin{pmatrix} \lambda a \\ \lambda b \\ \lambda c \end{pmatrix}$ stellen dieselbe Gerade dar.

**projektive Punkte** $\begin{pmatrix} x \\ y \\ z \end{pmatrix} \neq \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$ Skalierung egal.

**projektive Gerade** $\begin{pmatrix} a \\ b \\ c \end{pmatrix} \neq \begin{pmatrix} 0 \\ 0 \\ 0 \end{pmatrix}$ Skalierung egal

**Satz** Punkt $\begin{pmatrix} x \\ y \\ z \end{pmatrix}$ liegt auf der Geraden $\begin{pmatrix} a \\ b \\ c \end{pmatrix}$:

$$\begin{pmatrix} a \\ b \\ c \end{pmatrix} \cdot \begin{pmatrix} x \\ y \\ z \end{pmatrix} = 0$$

**Satz** Zwei verschiedene Geraden schneiden sich in genau einem Punkt.

**Beweis** Gerade $\forall\lambda : \begin{pmatrix} a_1 \\ b_1 \\ c_1 \end{pmatrix}, \begin{pmatrix} a_2 \\ b_2 \\ c_2 \end{pmatrix} \neq \lambda \begin{pmatrix} a_1 \\ b_1 \\ c_1 \end{pmatrix}$.

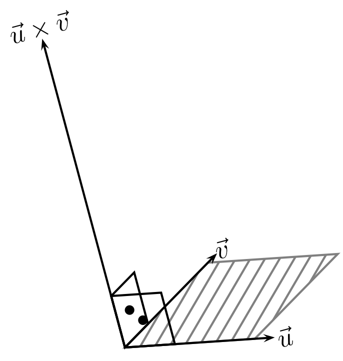
Schnittpunkt:

$$\begin{aligned} a_1x + b_1y + c_1z &= 0 \\ a_2x + b_2y + c_2z &= 0 \end{aligned}$$

Koeffizientenmatrix $A = \begin{pmatrix} a_1 & b_1 & c_1 \\ a_2 & b_2 & c_2 \end{pmatrix}$, $\operatorname{rg} A = 2$
$\Rightarrow$ Lösungsmenge ist eindimensional

$$L = \left\{ \lambda \begin{pmatrix} x_0 \\ y_0 \\ z_0 \end{pmatrix} \middle| \, y \in \mathbb{R} \right\} \text{ ist ein projektiver Punkt}$$

$$\begin{pmatrix} x_0 \\ y_0 \\ z_0 \end{pmatrix} \text{ kann als } \begin{pmatrix} a_1 \\ b_1 \\ c_1 \end{pmatrix} \times \begin{pmatrix} a_2 \\ b_2 \\ c_2 \end{pmatrix} = \begin{pmatrix} \begin{vmatrix} b_1 & b_2 \\ c_1 & c_2 \end{vmatrix} \\ \\ \begin{vmatrix} c_1 & c_2 \\ a_1 & a_2 \end{vmatrix} \\ \\ \begin{vmatrix} a_1 & a_2 \\ b_1 & b_2 \end{vmatrix} \end{pmatrix} \text{ berechnet werden (Kreuzprodukt)}$$



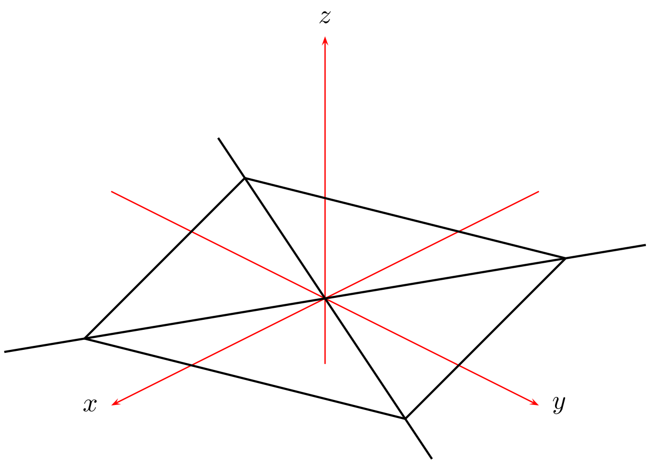

**Satz** Durch zwei verschiedene Punkte gibt es genau eine Geraden

**Beweis** gleich wie oben: $\begin{pmatrix} a \\ b \\ c \end{pmatrix}$ mit $\begin{pmatrix} x \\ y \\ z \end{pmatrix}$ vertauschen.

**Dualitätsprinzip** Man kann in einem Satz der projektiven Geometrie der Ebene „Punkte“ und „Geraden“ vertauschen und es bleibt ein gültiger Satz.

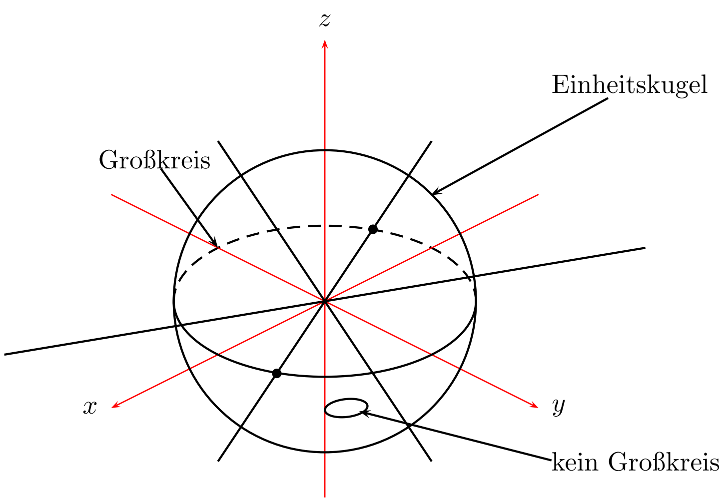
### 2.4.2 Modelle der projektiven Ebene

1. *Räumliches Modell der projektiven Ebene* $\left\{ \lambda \begin{pmatrix} x_0 \\ y_0 \\ z_0 \end{pmatrix} \middle| \lambda \in \mathbb{R} \right\}$... Geraden durch den Ursprung im $\mathbb{R}^3$ entsprechen den projektiven Punkten.



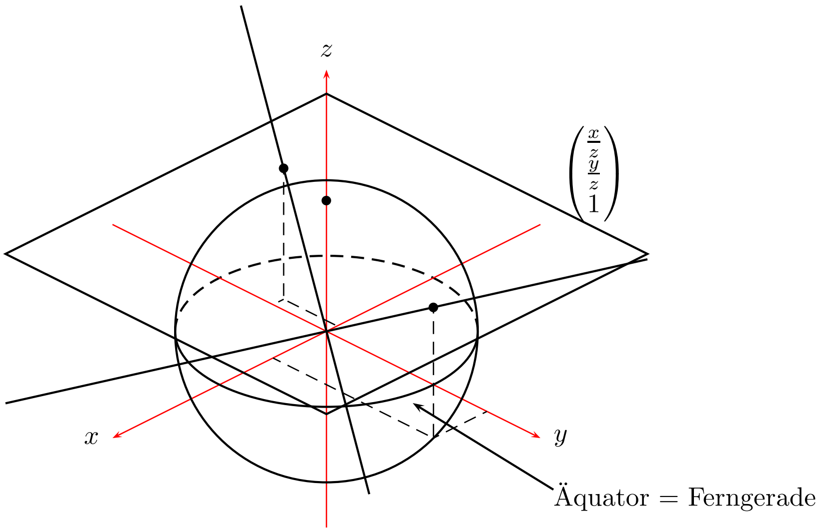

projektive Gerade $\equiv$ Ebene durch den Ursprung

2. *Kugelmodell der projektiven Ebene* entsteht durch Schnitt des räumlichen Modells mit der Einheitskugel $S^2 = \left\{ \begin{pmatrix} x \\ y \\ z \end{pmatrix} \middle| x^2 + y^2 + z^2 = 1 \right\}$




projektiver Punkt $\equiv$ Paar gegenüberliegender Punkte auf der Einheitskugel
projektive Gerade $\equiv$ Großkreise

### 2.4.3 Projektive Punkte zu karthesische Koordinaten




Schnitt der Geraden $\begin{pmatrix} x \\ y \\ z \end{pmatrix} \cdot \lambda$ im $\mathbb{R}^3$ mit Ebene $z = 1$: $z \cdot \lambda = 1 \Rightarrow \lambda = \frac{1}{z}$

$\rightarrow \begin{pmatrix} x \cdot \frac{1}{z} \\ y \cdot \frac{1}{z} \\ 1 \end{pmatrix}$

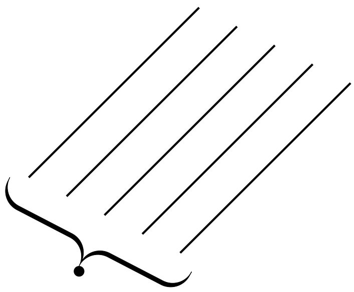
**Satz** Die Punkte $\begin{pmatrix} x \\ y \\ z \end{pmatrix}$ mit $z = 0$ haben *keine* Entsprechung in der euklidischen Ebene: Jede projektive Gerade hat als Bild in der euklidischen Ebene eine Gerade, mit einer Ausnahme: die Gerade $\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$

**Definition** Die Punkte des projektiven Raumes, die keine euklidische Entsprechung haben, heißen **Fernpunkte**. Die Gerade $\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$ **Ferngerade**.

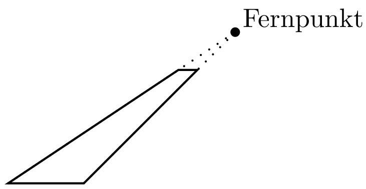
**Satz** Zwei Geraden der euklidischen Ebene sind genau dann *parallel*, wenn ihr Schnittpunkt ein Fernpunk ist.

**Satz** Die Punkte, die auf der Ferngeraden liegen, sind genau die Fernpunkte

**Satz** Es gibt zu jeder Schaar paralleler Geraden genau einen Fernpunkt.



Anschaulich ist ein Fernpunkt äquivalent zu perspektivischen Sammelpunkten:



## 2.5 Allgemeine projektive Transformationen

$\begin{pmatrix} x \\ y \\ z \end{pmatrix} \mapsto M \cdot \begin{pmatrix} x \\ y \\ z \end{pmatrix}$ mit $M = \mathbb{R}^{3\times 3}, \det M \neq 0$
(Punkte bleiben Punkte, Geraden bleiben Geraden, Inzidenz bleibt erhalten)

**Definition** Affine Transformationen sind jene Transformationen, bei denen die Fernpunkte Fernpunkte bleiben.

$$\begin{pmatrix} x \\ y \\ z \end{pmatrix} \mapsto M \begin{pmatrix} x \\ y \\ 0 \end{pmatrix} = \begin{pmatrix} x' \\ y' \\ z' \end{pmatrix} \stackrel{!}{=} \begin{pmatrix} x' \\ y' \\ 0 \end{pmatrix}$$

$$M = \begin{pmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ \cancel{m_{31}}0 & \cancel{m_{32}}0 & m_{33} \end{pmatrix}$$

$$\forall x, y : m_{31}x + m_{32}y + m_{33} \cdot 0 = 0 \Rightarrow m_{31} = m_{32} = 0$$

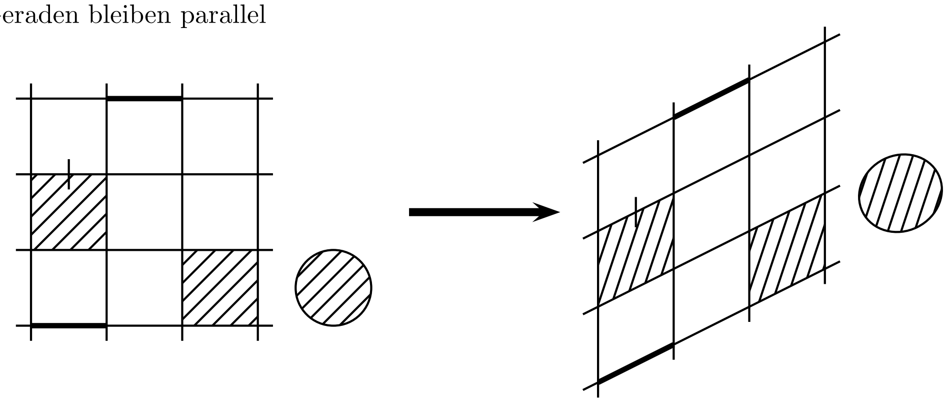
$$\det M \neq 0$$

$$\det M = \underbrace{m_{33}}_{\neq 0} \cdot \underbrace{\begin{vmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{vmatrix}}_{0} \qquad \Rightarrow \text{o. B. d. A. } m_33 = 1$$

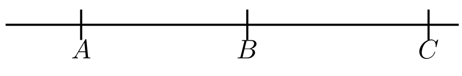
$$\begin{pmatrix} x \\ y \end{pmatrix} \mapsto \overbrace{\begin{pmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix}}^{\text{lineare Transformation}} + \overbrace{\begin{pmatrix} m_{13} \\ m_{23} \end{pmatrix}}^{\text{+ Translation}}$$

Affine Transformation:

- parallele Geraden bleiben parallel



- erhalten das Teilverhältnis auf parallelen Geraden



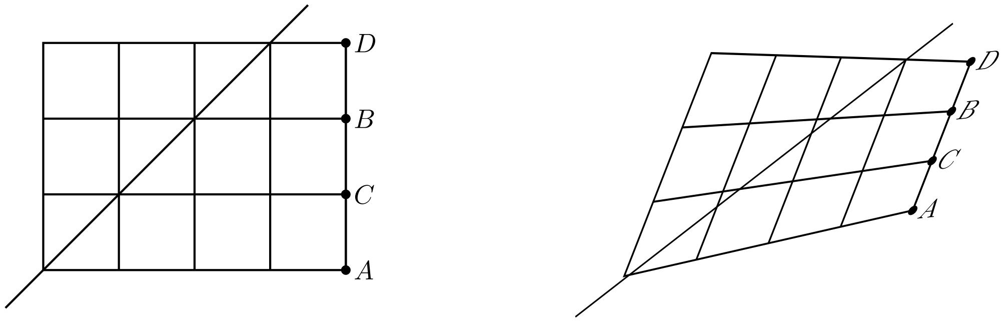

$$\frac{\overrightarrow{AB}}{\overrightarrow{AC}}$$

Starre Bewegungen (Isometrien, euklidische Transformationen):
$M = \begin{pmatrix} m_{11} & m_{12} \\ m_{21} & m_{22} \end{pmatrix}$ ist orthogonal $M^T = M^{-1}$ erhalten Längen, Winkel und Flächen

**Doppelverhältnis**

$$\boxed{\frac{\;\frac{\overrightarrow{AC}}{\overrightarrow{BC}}\;}{\;\frac{\overrightarrow{AD}}{\overrightarrow{BD}}\;}} = \text{Doppelverhältnis}$$

**Bemerkung** projektive Transformationen erhalten das sogenannte Doppelverhältnis

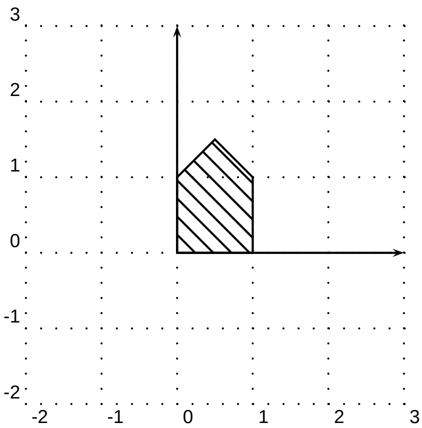



**Ausblick** projektiver Raum; wird beschrieben durch homogene Koordinaten $\begin{pmatrix} x \\ y \\ z \\ u \end{pmatrix} \neq \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}$. Karthesische Koordinaten $\begin{pmatrix} x \\ y \\ z \end{pmatrix}$ entsprechen homogenen Koordinaten $\begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix}$ oder $\begin{pmatrix} \lambda x \\ \lambda y \\ \lambda z \\ \lambda \end{pmatrix}$ ($\lambda \neq 0$, bel.).

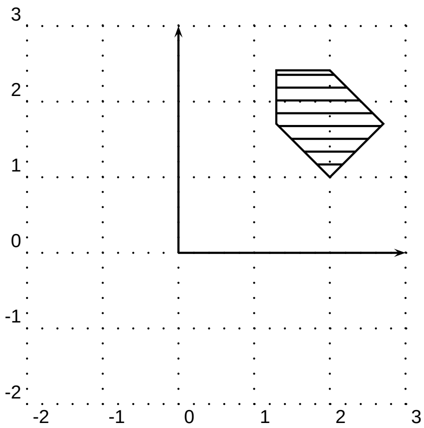
**Bemerkung 1** Transformation $x \mapsto Mx$ kann man auf zwei Arten interpretieren:

a) Wende die transformation $M$ auf Objekte an. Objekte werden bewegt, Standpunkt/Koordinatensystem bleibt fest.

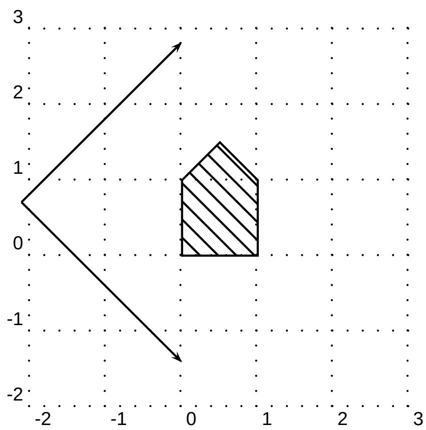
b) Drücke die unveränderte Lage eine Objektes in einem neuen Koordinatensystem aus.




$x \mapsto Mx$




$x \mapsto Mx$




Rechnerisch macht dies keinen Unterschied.

**Bemerkung 2** geometrische Transformationen können verknüpft; Reihenfolge ist wichtig!

$$y = M_1 x \qquad z = M_2 y \qquad z = \underbrace{M_2 M_1}_{\text{Matrizenmultiplikation}} x$$

Inverse Transformation wird dur die inverse Matrix ausgedrückt:

$$x = M_1^{-1} y$$

**Bemerkung 3** *Bei uns* stehen Koordinaten in *Spaltenvektoren* $\begin{pmatrix} x \\ y \\ z \end{pmatrix}$ $\begin{pmatrix} x \\ y \\ z \\ w \end{pmatrix}$

$\Rightarrow$ Transformation $\equiv$ Multiplikation mit einer Matrix *von links*

$$Mx = y$$

Alternative: Zeilenvektoren
$\Rightarrow$ Transformation $\equiv$ Multiplikation mit einer *von rechts* mit der *transponierten Matriz*

$$y^t = x^t M^t = (Mx)^t$$

Diese Schreibweise ist an sich intuitiver (da die Rechnung in der Reihenfolge der Anwendung aufgeschrieben wird), aber mathematisch unüblich:

$$M_2 M_1 x = z \Longleftrightarrow x^t M_1^t M_2^t = z^t$$

## 2.6 Transformation im dreidimensionalen Raum

### 2.6.1 Affine Transformation im dreidimensionalen Raum

- allgemeine affine Transformationen:

$$\begin{pmatrix} m_{11} & m_{12} & m_{13} & m_{14} \\ m_{21} & m_{22} & m_{23} & m_{24} \\ m_{31} & m_{32} & m_{33} & m_{34} \\ 0 & 0 & 0 & 1 \end{pmatrix} \text{ mit } \begin{vmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{vmatrix} \neq 0$$

- Isometrien (starre Bewegungen):

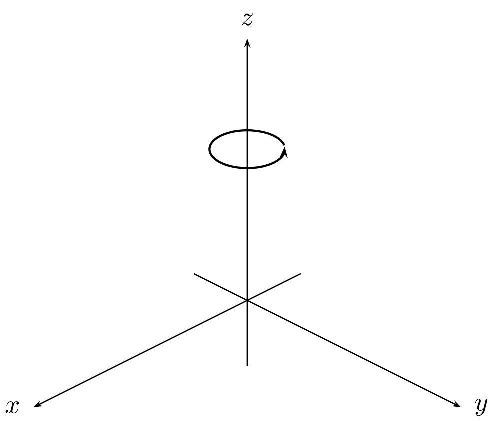
$$\begin{pmatrix} m_{11} & m_{12} & m_{13} \\ m_{21} & m_{22} & m_{23} \\ m_{31} & m_{32} & m_{33} \end{pmatrix} \text{ ist eine orthogonale Matrix}$$

  a) orientierungserhaltende $\det M = +1$ [Rotation um eine Achse (+ Translation)]
  b) orientierungsumkehrende $\det M = -1$ [Spiegelung an einer Ebene, Spiegelung an einem Punkt, Drehspiegelung ...]

#### Beispiele

- Drehung um die $z$-Achse:

$$M = \begin{pmatrix} \cos\varphi & -\sin\varphi & 0 & 0 \\ \sin\varphi & \cos\varphi & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$



- Spiegelung an der $xy$-Ebene:

$$M = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

- Spiegelung am Nullpunkt:

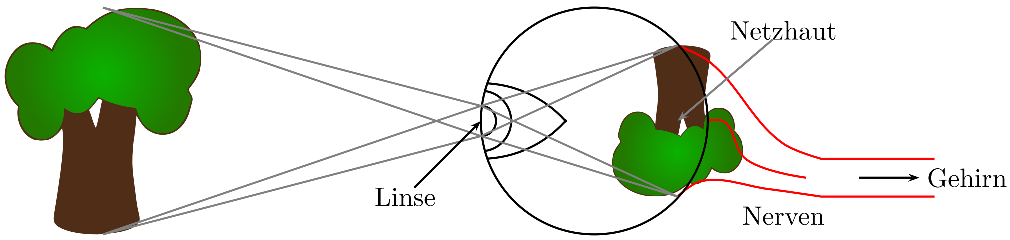
$$M = \begin{pmatrix} -1 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

### 2.6.2 projektive Transformationen im dreidimensionalen Raum

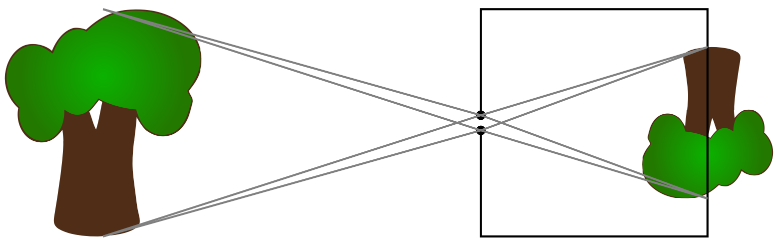
$x \mapsto Mx, \qquad M \in \mathbb{R}^{4\times 4}, \det M \neq 0$

## 2.7 Projektionen und Perspektive

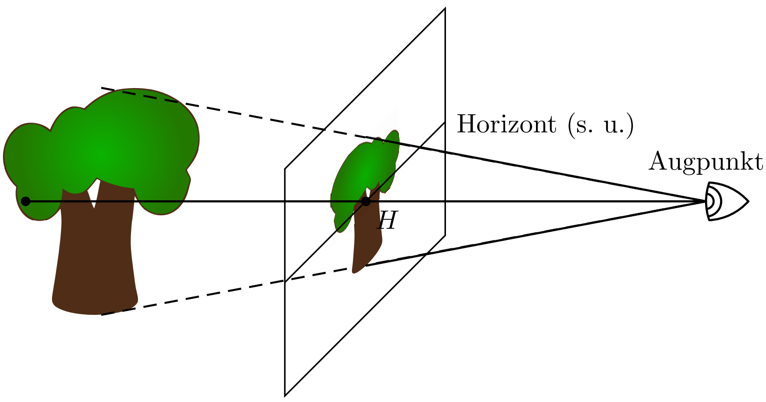
- Sehen mit dem menschlichen Auge




- Lochkamera



- Projektionen



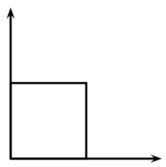

$H$ = Hauptpunkt (s. u.)

**Projektionen** Vebinde gegebene Punkte mit einem festen *Projektionszentrum* (kann auch ein Fernpunkt sein) und schneide die Strahlen mit einer Ebene (= *Projektionsebene*)

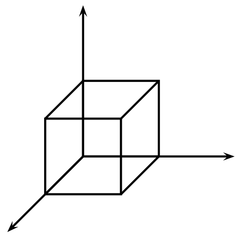
1. Projektionszentrum im Endlichen: Zentralprojektion

2. Projektionszentrum ein Fernpunkt: Parallelprojektion
(Parallele Geraden bleiben parallel)

a) Wenn die Projektion senkrecht auf den Projektionsstrahlen steht, spricht man von *orthographischer Projektion*

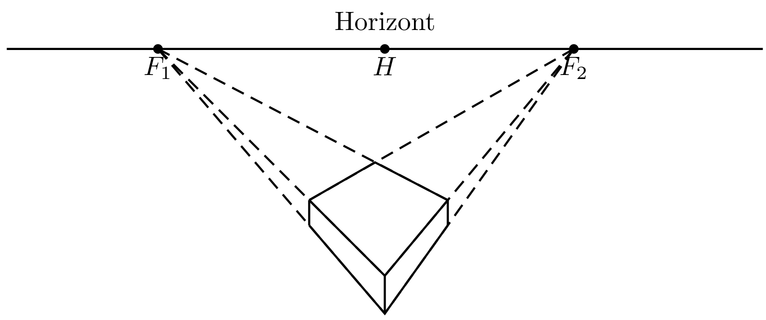


b) andernfalls von *schiefer Projektion*



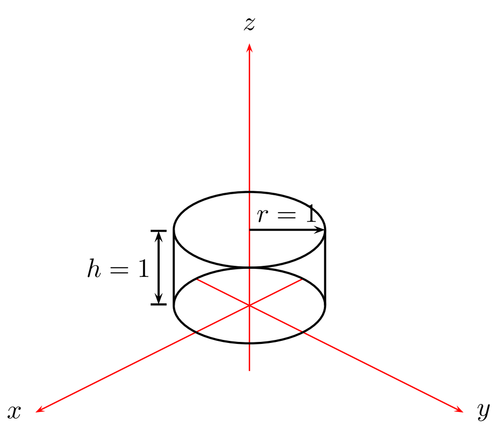
**zu 1. Zentralprojektion** Der *Hauptpunkt* ist der Punkt der Projektionsebene, der dem Auge am nächsten liegt. Ein projeziertes Bild vermittelt den exakten wirklichkeitsgetreuen Eindruck genau dann, wenn man sich so davor stellt, das das Auge direkt vor dem Hauptpunkt $H$ liegt und den richtigen Abstand $d$ und im richtigen Abstand zum Bild, mit dem das Bild berechnet wurde

- Parallele Geraden können in der Projektion zu schneidenden Geraden werden
- Das Bild des entsprechenden Fernpunktes heißt *Fluchtpunkt* (vanishing point)
- Die Fluchtpunkte der horizontalen Gerade liegen auf dem *Horizont* (die Fluchtgerade durch die alle horizontalen Ebenen gehen).
- Wenn die Projektionsgerade senkrecht ist, dann liegt der Hauptpunkt auf dem Horizont
  $\Rightarrow$ Senkrechte Geraden bleiben dann parallel (und senkrecht)




## 2.8 Koordinaten in der Praxis

- **Objektkoordinaten**

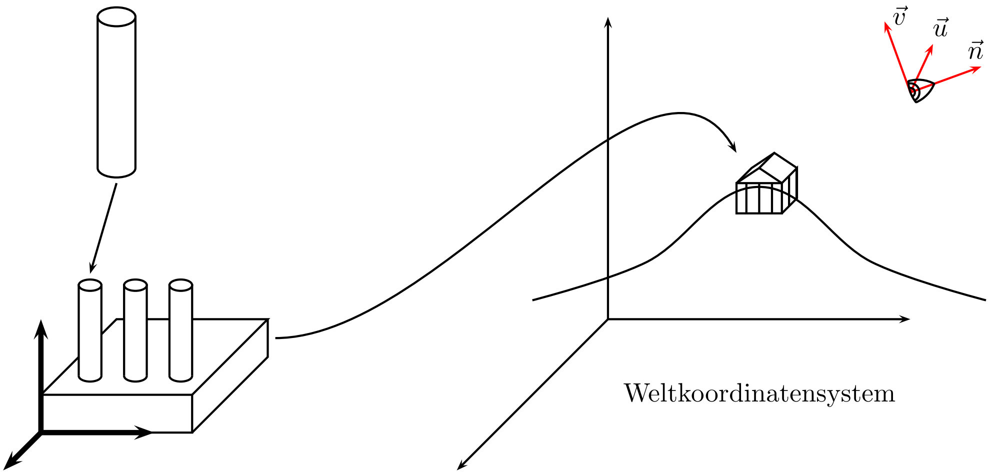



„Standardzylinder“ $x^2 + y^2 \leq 1, 0 \leq z \leq 1$

Durch die affinde Transformationen wird die Form des Zylinders angepasst und der Zylind an die passende Stelle (in einem größeren Modell / in der Umgebung) gesetzt

$$M = \begin{pmatrix} r & & & \\ & r & & \\ & & h & \\ & & & 1 \end{pmatrix} \text{...Skalierung} \rightarrow \text{Radius } r \text{ Höhe } h$$

*Translation* (+Rotation) von mehreren Kopien. Zylinder wird Teil eines größeren Objektes mit einem eigenen Koordinatensysteme




- **Weltkoordinaten**
  Ein globales Koordinatensystem, das für alle Berechnungen als Referenz dient.
- **Augenkoordinaten** (Kamerakoordinaten)
  - Ursprung = Augpunkt
  - 3 orthogonale Achsen:
    $\vec{n}$ = „Blickrichtung“ vom Objekt zum Betrachter
    $\vec{u}$ = „Horizontale Richtung“ von links nach rechts
    $\vec{v}$ = „Senkrechte Richtung“ von unten nach oben

  Die Projektionsebeneist orthogonal zu $\vec{n}$. Auf der Projektionsebene wir ein rechteckiges Bild erzeugt, dessen Kanten an $\vec{u}$ und $\vec{v}$ ausgerichtet sind.

$$\begin{pmatrix} x_{\text{Welt}} \\ y_{\text{Welt}} \\ z_{\text{Welt}} \\ w_{\text{Welt}} \end{pmatrix} \mapsto M_{AW} \begin{pmatrix} x_{\text{Welt}} \\ y_{\text{Welt}} \\ z_{\text{Welt}} \\ w_{\text{Welt}} \end{pmatrix} = \begin{pmatrix} x_{\text{Auge}} \\ y_{\text{Auge}} \\ z_{\text{Auge}} \\ w_{\text{Auge}} \end{pmatrix}$$

Weltkoordinaten $x, y, z$ bilden ein Rechtssystem. Augenkoordinaten $u, v, n$ bilden ein Rechtssystem.
$\Rightarrow M_{AW}$ ist Rotation+Translation

$$M_{AW}^{-1} \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} x_{\text{Auge}} \\ y_{\text{Auge}} \\ z_{\text{Auge}} \\ 1 \end{pmatrix}$$

$$M_{AW}^{-1} \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} x_u \\ y_u \\ z_u \\ 0 \end{pmatrix} = \text{Vektor } \vec{u} \text{ in Weltkoordinaten}$$

$$M_{AW}^{-1} \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} x_v \\ y_v \\ z_v \\ 0 \end{pmatrix} = \text{Vektor } \vec{v} \text{ in Weltkoordinaten}$$

$$M_{AW}^{-1} \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} x_n \\ y_n \\ z_n \\ 0 \end{pmatrix} = \text{Vektor } \vec{n} \text{ in Weltkoordinaten}$$

$$M_{AW}^{-1} = \begin{pmatrix} \overset{\text{orthogonal}}{\boxed{\begin{matrix} x_u & x_v & x_n \\ y_u & y_v & y_n \\ z_u & z_v & z_n \end{matrix}}} & \begin{matrix} x_{\text{Auge}} \\ z_{\text{Auge}} \\ y_{\text{Auge}} \end{matrix} \\ \begin{matrix} 0 & 0 & 0 \end{matrix} & 1 \end{pmatrix}$$

$$M_{AW} = \begin{pmatrix} x_u & y_u & z_u & * \\ x_v & y_v & z_v & * \\ x_n & y_n & z_n & * \\ 0 & 0 & 0 & 1 \end{pmatrix}$$

$$M_{AW} \begin{pmatrix} x_{\text{Auge}} \\ y_{\text{Auge}} \\ z_{\text{Auge}} \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix}$$

Der Punkt $\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$ in Augenkoordinaten wird auf $A + \vec{n}$ in Weltkoordinaten abgebildet.

$$M_{AW}^{-1} \begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} x_{\text{Auge}} + x_n \\ y_{\text{Auge}} + y_n \\ z_{\text{Auge}} + z_n \\ 1 \end{pmatrix} \qquad M_{AW}^{-1} \left[ \begin{pmatrix} 0 \\ 0 \\ 1 \\ 1 \end{pmatrix} - \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} \right] = \begin{pmatrix} x_n \\ y_n \\ z_n \\ 0 \end{pmatrix}$$

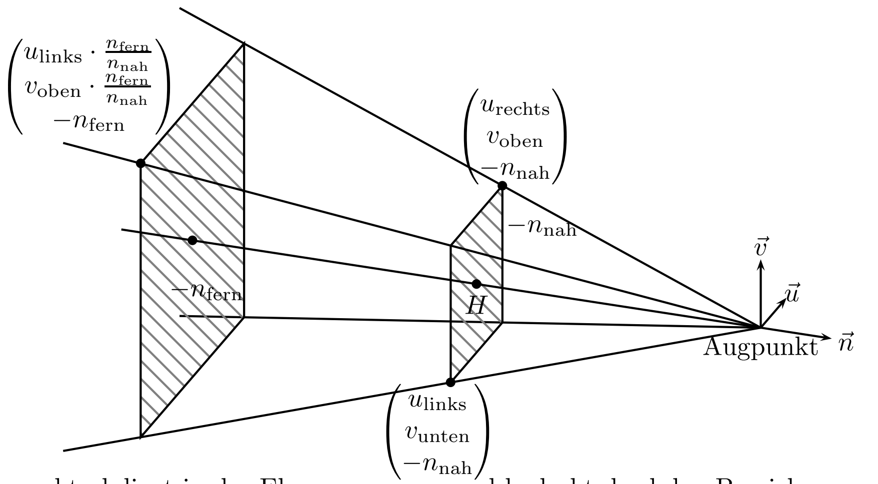
$$A = \begin{pmatrix} x_u & x_v & x_n \\ y_u & y_v & y_n \\ z_u & z_v & z_n \end{pmatrix} \text{ Spalten sind die kartesischen Weltkoordinaten von } \vec{u}, \vec{v}, \vec{n}$$

$$M_{AW}^{-1} = \left( \begin{array}{ccc|c} & A & & \begin{matrix} x_{\text{Auge}} \\ z_{\text{Auge}} \\ y_{\text{Auge}} \end{matrix} \\ \hline 0 & 0 & 0 & 1 \end{array} \right)$$

$$\boxed{M_{AW} = \left( \begin{array}{ccc|c} & A^T & & -A^T \begin{pmatrix} x_{\text{Auge}} \\ z_{\text{Auge}} \\ y_{\text{Auge}} \end{pmatrix} \\ \hline 0 & 0 & 0 & 1 \end{array} \right)}$$

Probe:

$$\left( \begin{array}{c|c} A & \begin{matrix} x_{\text{Auge}} \\ z_{\text{Auge}} \\ y_{\text{Auge}} \end{matrix} \\ \hline 0 & 1 \end{array} \right) \cdot \left( \begin{array}{c|c} A^T & -A^T \begin{pmatrix} x_{\text{Auge}} \\ z_{\text{Auge}} \\ y_{\text{Auge}} \end{pmatrix} \\ \hline 0 & 1 \end{array} \right) = \left( \begin{array}{c|c} A \cdot A^T = I' & 0 \\ \hline 0 & 1 \end{array} \right) = I \qquad \square$$

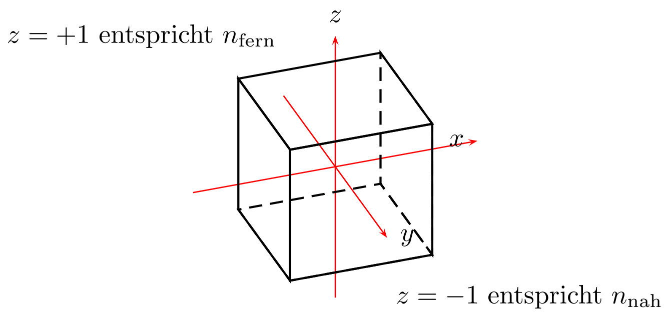



Projektionsrechteck liegt in der Ebene $n = n_{\text{nah}}$ und bedeckt dord den Bereich

$$[u_{\text{links}}, u_{\text{rechts}}] \times [v_{\text{unten}}, v_{\text{oben}}]$$

Der Sichtbare Bereich ist alles was hinter diesem Rechteck liegt. Zusätzlich wird alles abgeschnitten, was hinter der Ebene $n = n_{\text{fern}}$ liegt.
$\Rightarrow$ Pyramidenstumpf (view frustum)

- **Normalisierte Gerätekoordinaten** (normalized device coordinates, NDC) Der sichtbare Pyramidenstumpf wird durch projektive Transformation auf den Würfel $[-1, +1]^3$ transformatiert. $x, y, z$ bilden ein Linkssystem.




$$\begin{pmatrix} * \\ * \\ -n_{\text{nah}} \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} * \\ * \\ -1 \\ +1 \end{pmatrix} \cdot \lambda \qquad \text{(Ebene } z = -1)$$
$$\begin{pmatrix} * \\ * \\ -n_{\text{fern}} \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} * \\ * \\ 1 \\ 1 \end{pmatrix} \cdot \lambda_2$$

Fernpunkt auf der $z$-Achse:

$$\begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} 0 \\ 0 \\ * \\ 0 \end{pmatrix}$$

Horizontale Linien (Richtung $u$) bleiben parallel und horizontal (Richtung $x$):

$$\begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} \mapsto \begin{pmatrix} * \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

Vertikale Linien

$$\begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} \mapsto \begin{pmatrix} 0 \\ * \\ 0 \\ 0 \end{pmatrix}$$

Transformation

$$\begin{pmatrix} u \\ v \\ n \\ w \end{pmatrix} \mapsto \begin{pmatrix} * & 0 & * & 0 \\ 0 & * & * & 0 \\ 0 & 0 & * & * \\ 0 & 0 & -1 & 0 \end{pmatrix} \begin{pmatrix} u \\ v \\ n \\ w \end{pmatrix} = \begin{pmatrix} * & 0 & * & 0 \\ 0 & * & * & 0 \\ 0 & 0 & a & b \\ 0 & 0 & -1 & 0 \end{pmatrix} \begin{pmatrix} u \\ v \\ n \\ w \end{pmatrix}$$

$$\begin{pmatrix} * \\ * \\ -n_{\text{nah}} \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} * \\ * \\ -1 \\ 1 \end{pmatrix} \cdot \lambda = \begin{pmatrix} * \\ * \\ -n_{\text{nah}}a + b \\ n_{\text{nah}} \end{pmatrix} \Rightarrow -n_{\text{nah}}a + b = -n_{\text{nah}}$$
$$\begin{pmatrix} * \\ * \\ -n_{\text{fern}} \\ 1 \end{pmatrix} \mapsto \begin{pmatrix} * \\ * \\ 1 \\ 1 \end{pmatrix} \cdot \lambda = \begin{pmatrix} * \\ * \\ -n_{\text{fern}}a + b \\ n_{\text{fern}} \end{pmatrix} \Rightarrow -n_{\text{fern}}a + b = n_{\text{fern}}$$

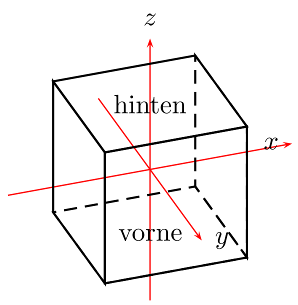
$$\begin{aligned} -n_{\text{nah}}a + b &= -n_{\text{nah}} \\ -n_{\text{fern}}a + b &= n_{\text{fern}} \\ \hline a(-n_{\text{nah}} + n_{\text{fern}}) &= -n_{\text{nah}} - n_{\text{fern}} \\ a &= -\frac{n_{\text{nah}} + n_{\text{fern}}}{n_{\text{nah}} - n_{\text{fern}}} \\ \Rightarrow \qquad b &= -\frac{2 \cdot n_{\text{fern}} \cdot n_{\text{fern}}}{n_{\text{fern}} - n_{\text{nah}}} \end{aligned}$$

$$\Rightarrow M = \begin{pmatrix} \frac{2n_{\text{nah}}}{u_{\text{rechts}} - u_{\text{links}}} & 0 & \frac{u_{\text{rechts}} + u_{\text{links}}}{u_{\text{rechts}} - u_{\text{links}}} & 0 \\ 0 & \frac{2n_{\text{nah}}}{v_{\text{oben}} - v_{\text{unten}}} & \frac{v_{\text{oben}} + v_{\text{unten}}}{v_{\text{oben}} - v_{\text{unten}}} & 0 \\ 0 & 0 & -\frac{n_{\text{fern}} + n_{\text{nah}}}{n_{\text{fern}} - n_{\text{nah}}} & -\frac{2n_{\text{fern}} \cdot n_{\text{nah}}}{n_{\text{fern}} - n_{\text{nah}}} \\ 0 & 0 & -1 & 0 \end{pmatrix}$$

$$M^{-1} = \begin{pmatrix} \frac{u_{\text{rechts}} - u_{\text{links}}}{2n_{\text{nah}}} & 0 & 0 & \frac{u_{\text{rechts}} - u_{\text{links}}}{2n_{\text{nah}}} \\ 0 & \frac{v_{\text{oben}} - v_{\text{unten}}}{2n_{\text{nah}}} & 0 & \frac{v_{\text{oben}} - v_{\text{unten}}}{2n_{\text{nah}}} \\ 0 & 0 & 0 & -1 \\ 0 & 0 & \frac{1}{2n_{\text{fern}}} - \frac{1}{2n_{\text{nah}}} & \frac{1}{2n_{\text{fern}}} + \frac{1}{2n_{\text{nah}}} \end{pmatrix}$$

- **Rasterkoordinaten -** Koordinaten auf dem Bildschirm
  von normalisierten Gerätekoordinaten (NDC) zu Rasterkoordinaten:

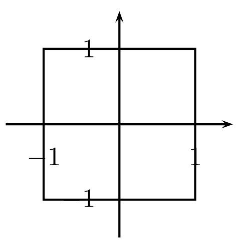
**zur Erinnerung** NDC Linkssystem

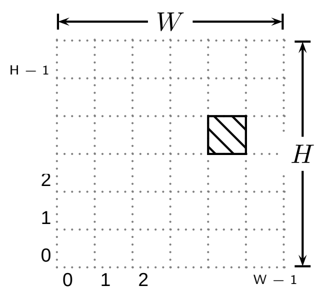



$[-1, 1] \times [-1, 1] \times [-1, 1]$

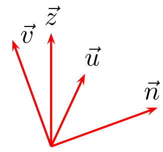
1. Projektion: $z$-Koordinaten weglassen. ($z$ gibt Informationen über die Tiefe, größerer $z$-Wert ist weiter hindent)

2. Skalierung des $x$-$y$-Quadrates und Runden auf $W$x$H$-Gitter




Pixelkoordinaten: $\begin{pmatrix} \left\lfloor (x+1) \cdot \frac{W}{2} \right\rfloor \\ \left\lfloor (y+1) \cdot \frac{H}{2} \right\rfloor \end{pmatrix}$

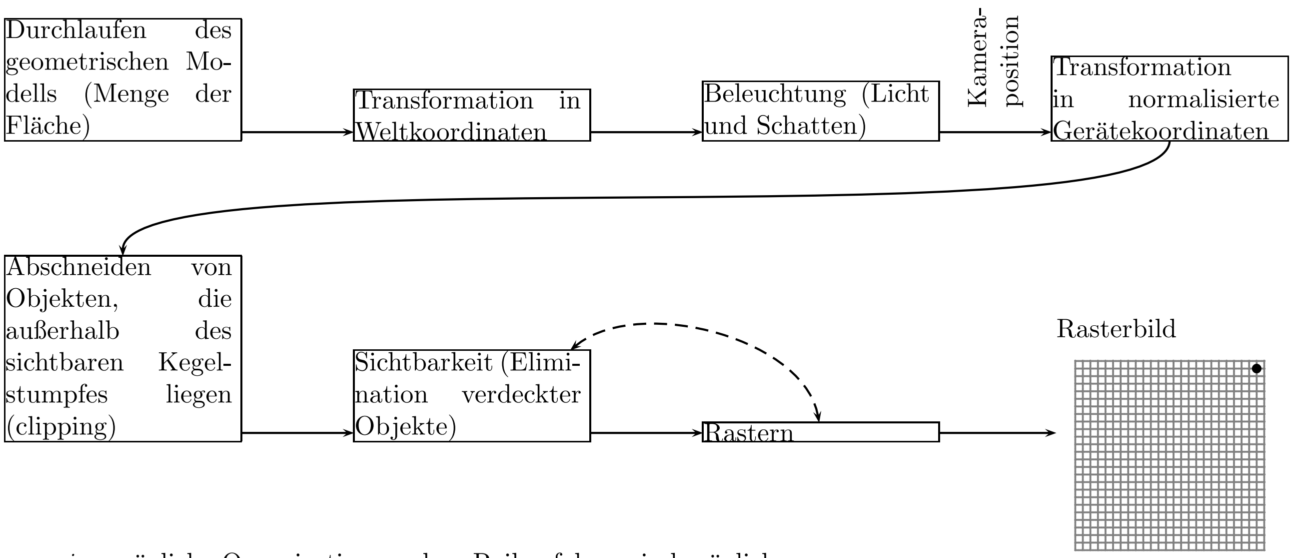



Berechnung des Augkoordinatensystems
Gegeben ist der Einheitsvektor $\vec{n}$ (und der Augpunkt)

Setze $\vec{u}_0 := \vec{z} \times \vec{n}$ $\qquad$ $\vec{z} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$ senkrecht nach oben

$$\vec{u} := \frac{\vec{u}_0}{||\vec{u}_0||}$$
$$\vec{v} := \vec{n} \times \vec{u}$$

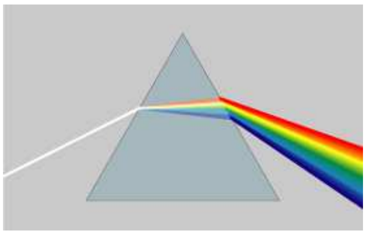
## 2.9 „rendering pipeline“ – vom geometrichen Modell zum Rasterbild




nur *eine* mögliche Organisation; andere Reihenfolgen sind möglich

# 3 Licht und Farben

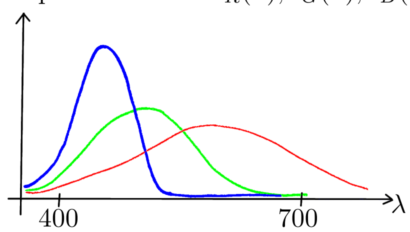
**Definition** (sichtbares) **Licht** sind elektromagnetische Wellen verschiedener Wellenlänge (ca. zwischen 400–700 nm) Die Wellenlänge $\lambda$ entscheidet über die **Farbe**. Das meiste Licht ist eine Mischung von verschiedenen Wellenlängen.



Wenn Licht auf einen Gegenstand trifft, dann wird es in unterschiedlichem Maß zurückgeworfen, je nach Wellenlänge Wenn Licht einen filter durchdringt, ist es analog (subtraktive Farbmischung).

## 3.1 Farbensehen im menschlichen Auge

Es gibt drei Arten von lichtempfindlichen *Zapfen* (R, G, B)



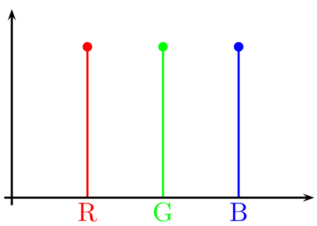

Erregung der „roten“ Zapfen bei einer Lichtquelle mit Intensitätsfunktion $f(\lambda)$

$$r = \int f(\lambda) \cdot e_R(\lambda)\ \mathrm{d}y \qquad f(\lambda)$$

analog „grün“: $g = \int f(\lambda) \cdot e_G(\lambda)\ \mathrm{d}y$
analog „blau“: $b = \int f(\lambda) \cdot e_B(\lambda)\ \mathrm{d}y$

- Verschiedene Lichtquellen mit verschiednen spektraler Zusammensetzung erzeugen den gleichen Farbeindruck, wenn sie die gleichen $(r, g, b)$-Werte hervorbringen.
- Dreidimensionaler Farbraum, aber nicht alle $(r, g, b)$-Werte erreichbar
(Wenn $g > 0 \Rightarrow r > 0$ oder $b > 0$, $(r, g, b) = (0, 1, 0)$ gibt es nicht)
- Wenn man $f(\lambda)$ mit einem Skalar $c > 0$ multipliziert, dann ändert sich nur die Helligkeit, nicht die Farbe. Entsprechend wird $(r, g, b)$ mit einem Skalar multipliziert.
- Normalisierung auf $r + b + g = 1$ führt auf einen zweidimensionalen Farbraum, bei dem die Helligkeit konstant ist.

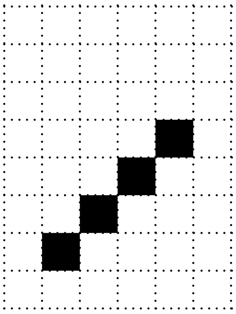
In der Computergrafik tut man so, als ob es nur *drei* verschiedene Wellenlängen (Grundfarben) gibt: Rot, Grün und Blau



# 4 Rasterung von Strecken und Kreisen

## 4.1 Strecken

- Rendering pipeline wurde durchlaufen
- Koordinaten sind ganzzahlige Pixelkoordinaten



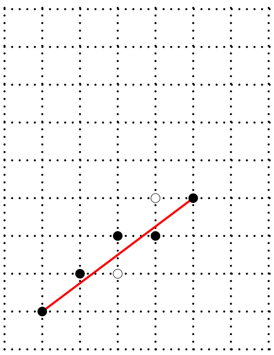
**Problemstellung** Gegeben sind zwei Punkte $p_1 = (x_1, y_1)$ und $p_2 = (x_2, y_2)$. Zeichne Strecke zwischen $p_1$ und $p_2$. $x_1, y_1, x_2, y_2$ sind ganzzahlig.

**Vereinfachung** Strecken die durch den Nullpunkt gehen $(0,0), p_1 = (x_1, y_1)$

- Beschränkung auf Strecken im ersten Quadranten des Koordinatensystems. Alle anderen Strecken (im Quadranten II, III, IV) können durch Speigelung an $x$- oder $y$-Achse erzeugt werden
- Beschränkung aufersten Oktanten, alle anderen Strecken erhält man wie oben durch Vertauschen von $x$- und $y$-Koordinate

**Vereinfachte Problemstellung** Gegeben ist ein Punkt $p_1 = (x_1, y_1)$ mit ganzzahligen Koordinaten. $x_1 \geq y_1, x_1 \geq 0, y_1 \geq 0$. Zeichne Strecke zwischen $(0,0)$ und $(x, y)$.

Geradengleichung: $g(x) = \frac{y_1}{x_1}x$ Steigung: $m = \frac{y_1}{x_1}, 0 \leq m \leq 1$



**Idee** Werte für jeden Wert $i$ zwischen 0 und $x_1$ die Funktion $g$ aus. Runde das Ergebnis $g(i)$ und nimm diesen Wer als $j$-Wert

```
double m = y1/x1;
for (i = 0; i <= x1, i++) {
        j = round(i*m); // auf- bzw. abrunden
        setPixel(i,j);
}
```

- erstetze `j= round(i * m);` durch `y=y+m_j; j = round(y);`
- im $i$-ten Schritt $y_i = m \cdot i$

- im $(i+1)$-ten Schritt $y_{i+1} = m \cdot (i+1)$
- $y_{i+1} - y_i = m(i+1-1) = m$
- $0 \leq m \leq 1 \Rightarrow y_{i+1} \leq y_i + 1$
- Wert von $j$ steigt pro Schleifendurchlauf um höchstens 1.

$$
\begin{aligned}
y_{i+1} &= y_i + m & m &= \frac{y_1}{x_1} \\
j &= round(y_i) \\
y_i &= j + r_i & \text{Rest: } -\frac{1}{2} &\leq r_i \leq \frac{1}{2} \\
y_{i+1} &= y_i + m = j + \underbrace{r_i + m}_{r_{i+1}}
\end{aligned}
$$

```
double m = y1/x1;
double r = 0;
int j = 0;
for (i = 0; i <= x1; i+1) {
        r = r + m;       // y = y + m;
        if (r >= 1/2) { // j = round(y)
                j = j+1,
                r = r - 1;
        }
        setPixel(i,j);
}
```

z. B:

$$
\begin{aligned}
y &= 0.6 & r_{alt} &= 0.6 \\
y &= j_{alt} + r_{alt} & j_{neu} &= j_{alt} + 1 \\
y &= j_{neu} + r_{neu} & r_{neu} &= r_{alt} + 1
\end{aligned}
$$

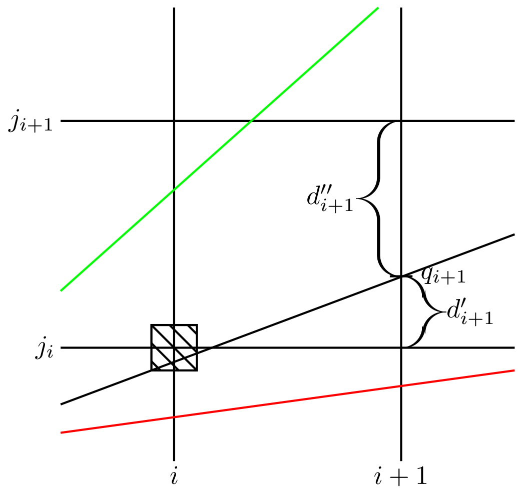
Immer noch ein Problem: **double**-Werte sind zu ungenau
Wir wissen, dass $x_1$ und $y_1$ ganzzahlig sind.
$\Rightarrow$ Multiplikation mit $2x_1$ liefert **int**-Werte

```
int m = 2*yi;
int r = 0;
int j = 0;
setPixel(0,0);
for(int i = 1; i <= x1; i++) {
        r = r + m;
        if (r >= x1) {
                j = j+1;
                r = r - 2*x1;
        }
        setPixel(i,j);
}
```

$$r_{i+1} + 2y_1 \geq x_1 \Leftrightarrow r \geq x_1 - 2y_1$$

```
int j = 0;
int m = 2*y1;
int r = 2*y1 - x1;
int c1 = m - 2*x1;
int c2 = m;
for (int i = 1; i <= x1; i++) {
        if (r >= 0) {
                j++;
                r = r + c1;
        } else {
                r = r + c2;
        }
        setPixel(i,j);
}
```

### 4.1.1 Bresenham-Algorithmus




Wähle eine Spalte $i+1$ das Pixel, das am nächsten an $q_{i+1}$ liegt, also

$$
\begin{aligned}
d'_{i+1} \leq d''_{i+1} &\Leftrightarrow \text{ wähle } j_{i+1} = j_i \\
d''_{i+1} < d'_{i+1} &\Leftrightarrow \text{ wähle } j_{i+1} = j_i + 1
\end{aligned}
$$

oder äquivalent

$$
\begin{aligned}
d'_{i+1} - d''_{i+1} \leq 0 &\Leftrightarrow j_{i+1} = j_i \\
d'_{i+1} - d''_{i+1} > 0 &\Leftrightarrow j_{i+1} = j_i + 1
\end{aligned}
$$

$y$-Koordinaten von $q_{i+1}$ $\quad \frac{y_1}{x_1}(i+1)$

$$
d'_{i+1} = \frac{y_1}{x_1}(i+1) - j_i \qquad d''_{i+1} = j_i + 1 - \frac{y_1}{x_1}(i+1)
$$

$$
d'_{i+1} - d''_{i+1} = 2\frac{y_1}{x_1}(i+1) - 2j_i - 1 \qquad \mid \text{ multipliziere mit } x_1
$$

(ändert nichts an der Bedingung $\underset{<}{\leq} 0$)
Wir erhalten eine Fehler-/Entscheidungsvariable für $(i+1)$-te Spalte

$$
\begin{aligned}
&e_{i+1} = x_i(d'_{i+1} - d''_{i+1}) = 2y_1(i+1) - 2j_i x_1 - x_1 \\
&e_{i+1} \leq 0 \Leftrightarrow j_{i+1} = j_i \\
&e_{i+1} > 0 \Leftrightarrow j_{i+1} = j_i + 1
\end{aligned}
$$

Betrachte Differenz zwischen aufeinanderfolgenden Entscheidungsvariablen

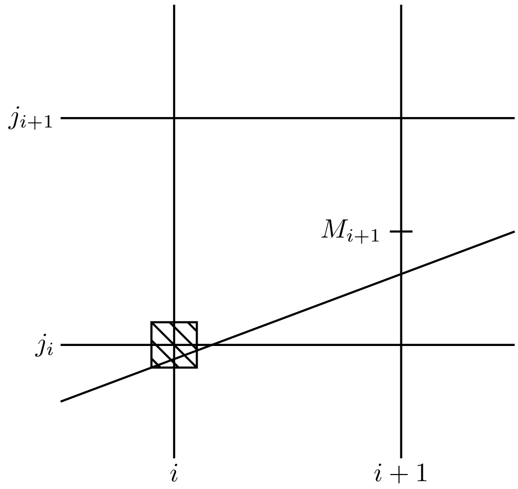
$$
\begin{aligned}
e_{i+1} - e_i &= \underline{2y_1(i+1)} - 2j_i x_1 - \underline{x_1} - \underline{2y_1}i + 2j_{i-1}x_1 + \underline{x_1} \\
e_{i+1} - e_i &= 2y_1 - 2x_1 \underbrace{(j_i - j_{i-1})}_{= \begin{cases} 0, & \text{falls } e_i \leq 0 \\ 1, & \text{falls } e_i > 0 \end{cases}}
\end{aligned}
$$

Also

$$
\begin{aligned}
e_i > 0, j_i = j_{i-1} + 1 &\text{ und } e_{i+1} = e_i + 2y_1 - 2x_1 \\
e_i \leq 0, j_i = j_{i-1} &\text{ und } e_{i+1} = e_i + 2y_1
\end{aligned}
$$

Anfangswerte: $i = 0, j_0 = 0, e_1 = 2y_1 - x_1$

### 4.1.2 Midpoint Line Algorithmus




$$M_{i+1} = \begin{pmatrix} i+1 \\ j_i + \frac{1}{2} \end{pmatrix}$$

Wenn $M_{i+1}$ über der Strecke liegt $\Rightarrow$ wähle $j_{i+1} = j_i$ Wenn $M_{i+1}$ unter der Strecke liegt $\Rightarrow$ wähle $j_{i+1} = j_i + 1$

$$\begin{aligned} \text{Geradengleichung: } y &= \frac{y_1}{x_1}x \Leftrightarrow y_1x - x_1y = 0 \\ (x,y) \text{ liegt über der Geraden } \Leftrightarrow y &> \frac{y_1}{x_1}x \Leftrightarrow y_1x - x_1y < 0 \\ (x,y) \text{ liegt unter der Geraden } \Leftrightarrow y &< \frac{y_1}{x_1}x \Leftrightarrow y_1x - x_1y > 0 \end{aligned}$$

Setze $F(x,y) = xy_1 - x_1y$, dann gilt also:

- $(x,y)$ über Geraden $\Leftrightarrow F(x,y) < 0$
- $(x,y)$ auf Geraden $\Leftrightarrow F(x,y) = 0$
- $(x,y)$ unter Geraden $\Leftrightarrow F(x,y) > 0$

Entscheidungsvariable

$$d_{i+1} = F(M_{i+1}) = (i+1)y_1 - x_i\left(j_i + \frac{1}{2}\right) \leq 0 \Leftrightarrow j_{i+1} = j_i$$
$$d_{i+1} = F(M_{i+1}) = (i+1)y_1 - x_i\left(j_i + \frac{1}{2}\right) = 0 \Leftrightarrow j_{i+1} = j_i + 1$$

$$\begin{aligned} d_{i+1} - d_i &= y_1(i+1) - x_1\left(j_i + \frac{1}{2}\right) - y_1 i + x_1\left(j_{i+1} + \frac{1}{2}\right) \\ &= y_1 - x_1 \underbrace{(j_i - j_{i-1})}_{= \begin{cases} 0, & \text{falls } d_i \leq 0 \\ 1, & \text{falls } d_i > 0 \end{cases}} \\ d_1 &= y_1 - \frac{x_1}{2} \end{aligned}$$

Für Ganzzahligkeit multiplitziere mit 2

$$e_i = 2d_i \Rightarrow e_{i+1} - e_i = 2y_1 - 2x_1(j_i - j_{i-1})$$
$$e_1 = 2y_1 - x_1$$