

## 2. EVC Test - Computervision

### Vorwort

---

Diese Stoffsammlung/Zusammenfassung enthält den Stoff, der in der EVC Vorlesung der TU Wien im Sommersemester 2025 vorgetragen wurde, der auch in den jeweiligen Skripten und Slides zu finden ist. Die Struktur dieser Zusammenfassung basiert demnach auch der des Skriptums.

#### Disclaimer

Vieles der Zusammenfassung wurde mit AI generiert, basiert allerdings nur auf Inhalten der Unterlagen. Die Stellen die mit AI generiert wurden, wurden von mir überprüft und mit den Unterlagen verglichen, aber auch ich kann Fehler machen.

Demnach, falls sich irgendwo Fehler befinden oder es Verbesserungsvorschläge gibt, bitte an [@xmozz](#) auf Discord wenden.

### Inhalt

---

- [9. Multiskalenrepräsentationen](#)
- [10. Stereo und Motion](#)
- [11. Deep Learning](#)
- [12. Computational Photography](#)

# 9. Multiskalenrepräsentationen

EVC\_Skriptum\_CV, p.46

- **Abbildung 38:** Beispielbild zur Gesichtserkennung auf mehreren Skalen



- **Nachbarschaftsoperationen:**
  - Sind ein erster Schritt in der Bildanalyse.
  - Extrahieren lediglich **lokale Merkmale** (Größenordnung von maximal einigen Pixeln).
- **Großkalige Information:**
  - Bilder enthalten auch Information über größere Bereiche.
  - Zur Extraktion sind **größere Filtermasken** notwendig.
  - **Problem:** Erhöhter Rechenaufwand bei großen Filtermasken (Verdopplung der Filtergröße  $\Rightarrow$  vierfacher Rechenaufwand bei 2D-Bildern).
- **Grauwertänderungen und Skalenfehlanpassung:**
  - Grauwertänderungen durch Kontrastunterschiede zwischen Objekten und Hintergrund können schwer zu bestimmen sein.
  - **Ursache:** Skalenfehlanpassung - Grauwerte ändern sich über größere Distanzen, die die detektierenden Operatoren nicht erfassen.
- **Abhängigkeit der Merkmalsdetektion von der Skala:**
  - Die Detektion bestimmter Merkmale in einem Bild hängt von der **richtigen Skala** ab.
  - Die richtige Skala hängt von den **charakteristischen Größen** im zu detektierenden Objekt ab (z.B. Größe der Gesichter bei Gesichtserkennung).
- **Multiskalenanalyse als Lösung:**

- Für die optimale Verarbeitung muss ein Bild in **unterschiedlichen Skalen** vorliegen.
- Dies erfordert eine Darstellung in **mehreren Auflösungsstufen**.
- **Definition:** Multiskalenanalyse ist ein mathematisches Konzept, das die Signalanalyse auf unterschiedlichen Auflösungsstufen beschreibt.
- **Vorteile der Multiskalenanalyse:**
  - **Feine Details:** Benötigen die maximale verfügbare Auflösung.
  - **Grobe Strukturen:** Können mit geringerem Aufwand bei **reduzierter Auflösung** analysiert werden.

## Abtastung

---

[EVC\\_Skriptum\\_CV, p.46, p.47](#)

- **Diskreteisierung von Signalen:** Bei der Bildaufnahme wird eine kontinuierliche Funktion in eine diskrete Funktion umgewandelt, was als **Abtastung (Sampling)** bezeichnet wird.
- **Definition:** Abtastung ist die Entnahme von Abtastwerten der kontinuierlichen Funktion an bestimmten Punkten in der Zeit oder im Raum, üblicherweise in regelmäßigen Abständen.
- **Formale Beschreibung mit der Impulsfunktion (Deltafunktion oder Dirac-Impuls)  $\delta(x)$ :**
  - Modelliert einen kontinuierlichen, "idealen" Impuls.
  - Ist überall null mit Ausnahme des Ursprungs, wo ihr Wert zwar ungleich null, aber undefiniert ist.
  - Ihr Integral ist eins:

$$\int_{-\infty}^{\infty} \delta(x) dx = 1$$

- **Interpretation von  $\delta(x)$ :** Kann man sich als einzelnen Impuls an der Position null vorstellen, der unendlich schmal ist, aber endliche Energie aufweist.
- **Modellierung der Abtastung mit der Impulsfunktion:**
  - Eine kontinuierliche Funktion  $g(x)$  wird mit der Impulsfunktion  $\delta(x)$  punktweise multipliziert.
  - Dadurch erhält man einen einzelnen, diskreten Abtastwert der Funktion  $g(x)$  an der Stelle  $x = 0$ .
- **Abtastung an beliebigen Stellen durch Verschieben der Impulsfunktion:**
  - Durch Verschieben der Impulsfunktion um eine Distanz  $x_0$  kann  $g(x)$  an jeder beliebigen Stelle  $x = x_0$  abgetastet werden (Multiplikation mit  $\delta(x - x_0)$ ).
- **Abtastung einer kontinuierlichen Funktion  $g(x)$  an einer Folge von  $N$  Positionen  $x_i = 1, 2, \dots, N$ :**
  - Kann als Summe der  $N$  Einzelabtastungen dargestellt werden:

$$g_s(x) = g(x) \cdot [\delta(x - 1) + \delta(x - 2) + \dots + \delta(x - N)]$$

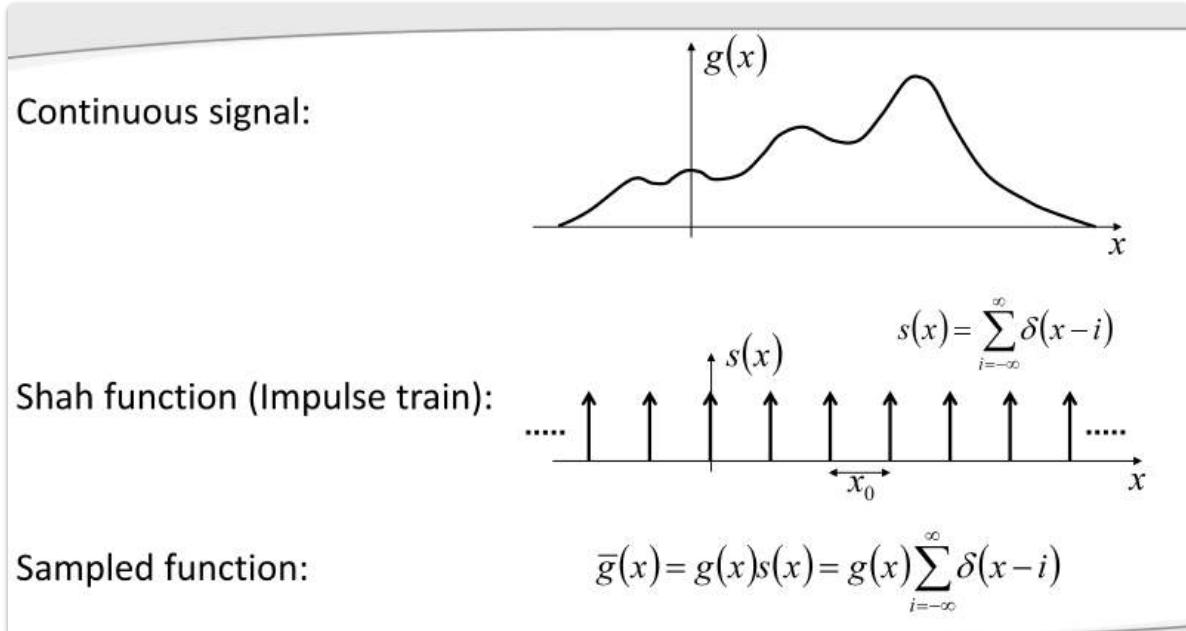
$$g_s(x) = g(x) \cdot \sum_{i=1}^N \delta(x - i)$$

- **Pulsfolge:** Diese Summe von verschobenen Einzelimpulsen wird als "Pulsfolge" bezeichnet.
- **Kammfunktion oder Shah-Funktion:** Wenn die Pulsfolge in beiden Richtungen ins Unendliche erweitert wird, erhalten wir die Kammfunktion oder Shah-Funktion:

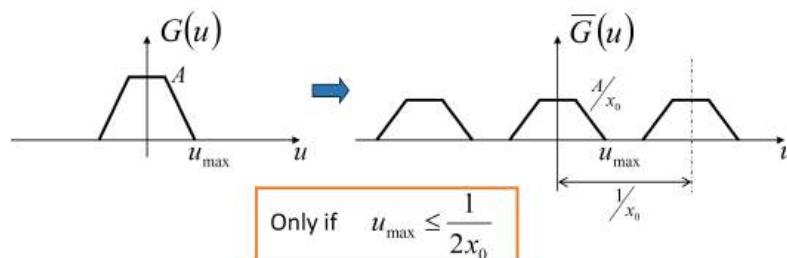
$$S(x) = \sum_{i=-\infty}^{\infty} \delta(x - i)$$

- **Auswirkungen der Abtastung auf das Frequenzspektrum:**
  - Die Abtastung einer kontinuierlichen Funktion hat auch Auswirkungen auf das Frequenzspektrum des resultierenden (diskreten) Signals.
  - Der Einsatz der Kammfunktion als formales Modell des Abtastvorgangs macht es einfacher, diese spektralen Auswirkungen zu interpretieren.
- **Eigenschaften der Kammfunktion im Frequenzbereich:**
  - Die Kammfunktion besitzt die seltene Eigenschaft, dass ihre Fouriertransformierte wiederum eine Kammfunktion ist, also den gleichen Funktionstyp hat.
  - Das Produkt zweier Funktionen in einem Raum (entweder im Orts- oder im Spektralraum) entspricht einer linearen Faltung im jeweils anderen Raum.
- **Spektrum des abgetasteten Signals:**
  - Das Fourierspektrum der Abtastfunktion (Kammfunktion im Zeitbereich) ist wiederum eine Kammfunktion im Frequenzbereich mit einem Impuls bei jeder ganzzahligen Frequenz.
  - Die Faltung einer beliebigen Funktion mit einem Impuls  $\delta(x)$  ergibt aber wieder die ursprüngliche Funktion:  $f(x) * \delta(x) = f(x)$ .
  - Das hat zur Folge, dass im Fourierspektrum des abgetasteten Signals  $\tilde{g}(u)$  das Spektrum  $G(u)$  des ursprünglichen, kontinuierlichen Signals unendlich oft, nämlich an jedem Puls im Spektrum der Abtastfunktion, repliziert wird.
  - Das daraus resultierende Fourierspektrum ist daher periodisch mit der Periodenlänge  $1/x_0$ , also im Abstand der Abtastfrequenz  $1/x_0$ .
- **Vermeidung von Aliasing:**
  - Solange sich die durch die Abtastung replizierten Spektralkomponenten in  $\tilde{g}(u)$  nicht überlappen, kann das ursprüngliche Spektrum  $G(u)$  - und damit auch das ursprüngliche, kontinuierliche Signal  $g(x)$  - ohne Verluste aus einer beliebigen Replika von  $G(u)$  aus dem periodischen Spektrum  $\tilde{g}(u)$  rekonstruiert werden.
- **Nyquist-Shannon-Abtasttheorem (impliziert):**
  - Zur Diskretisierung eines kontinuierlichen Signals  $g(x)$  mit Frequenzanteilen im Bereich  $0 \leq |u| \leq u_{max}$  benötigen wir eine Abtastfrequenz  $u_s$ , die mindestens doppelt so hoch wie die maximale Signalfrequenz  $u_{max}$  ist ( $u_s > 2u_{max}$ ).
- **Entstehung von Aliasing:**

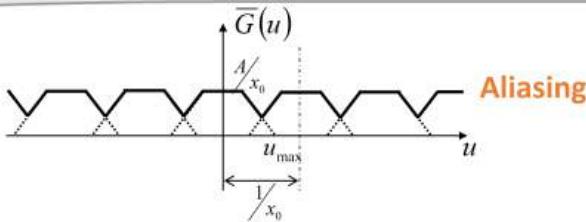
- Wird diese Bedingung nicht eingehalten, dann überlappen sich die replizierten Spektralteile im Spektrum des abgetasteten Signals, und das Spektrum wird verfälscht mit der Folge, dass das ursprüngliche Signal nicht mehr fehlerfrei aus dem Spektrum rekonstruiert werden kann.
- Dieser Effekt wird als **Aliasing** bezeichnet.



- Sampled function:
- $$\bar{g}(x) = g(x)s(x) = g(x)\sum_{i=-\infty}^{\infty} \delta(x - ix_0)$$
- Sampling frequency  $\frac{1}{x_0}$
- Fourier transformed:
- $$\bar{G}(u) = G(u) * S(u) = G(u) * \frac{1}{x_0} \sum_{i=-\infty}^{\infty} \delta\left(u - \frac{i}{x_0}\right)$$



$$\text{If } u_{\max} > \frac{1}{2x_0}$$



When can we recover  $G(u)$  from  $\bar{G}(u)$ ?

$$\text{Only if } u_{\max} \leq \frac{1}{2x_0} \text{ (Nyquist Frequency)}$$

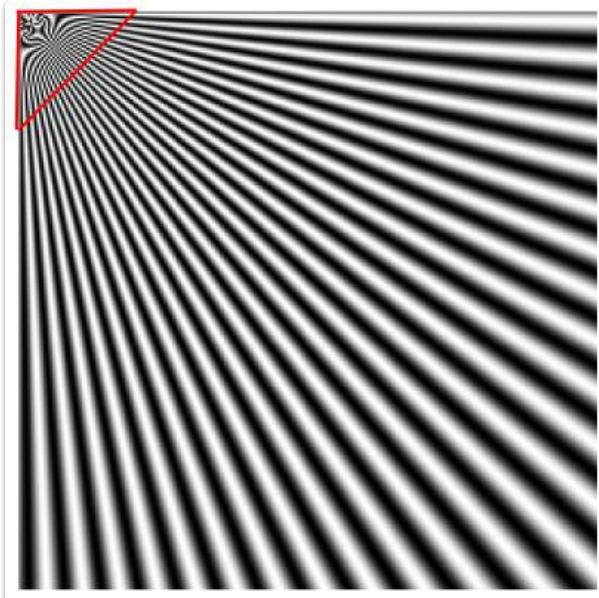
We can use

$$C(u) = \begin{cases} x_0 & |u| < \frac{1}{2x_0} \\ 0 & \text{otherwise} \end{cases}$$

$$\text{Then } G(u) = \bar{G}(u)C(u) \text{ and } f(x) = \text{IFT}[G(u)]$$

Sampling frequency must be greater than  $2u_{\max}$

Beispiel für Aliasing:

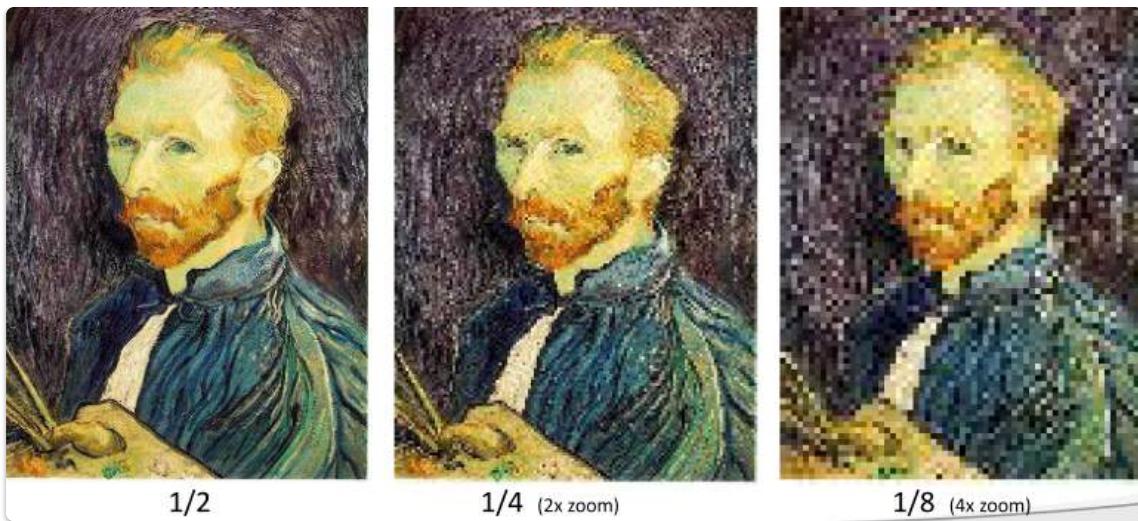


Das was wir hier machen zieht darauf ab dass wir nicht solche Bildfehler haben

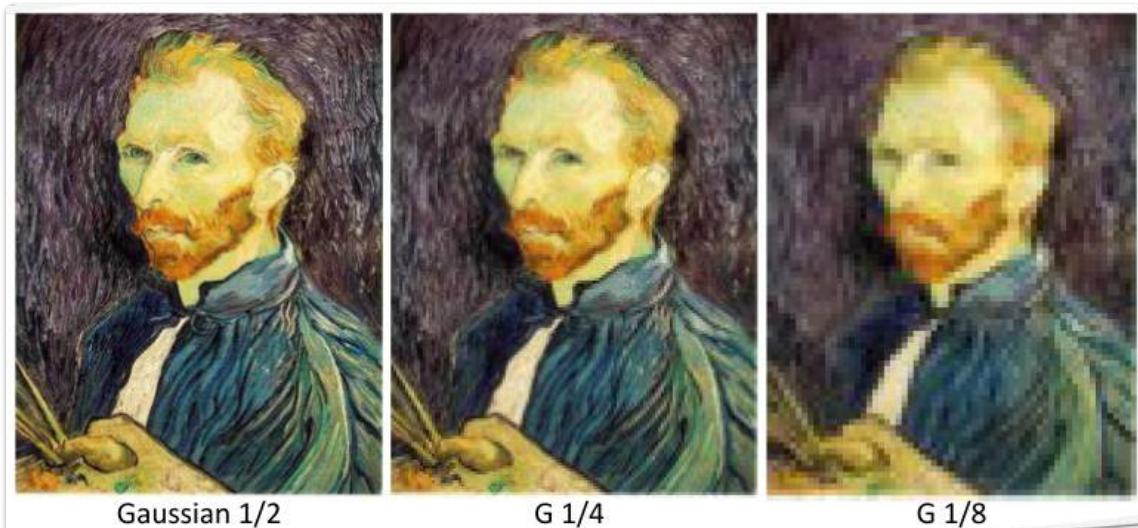
## Bildskalierung

[EVC\\_Skriptum\\_CV](#), p.47, p.48

- **Motivation für Bildreskalierung (Subsampling):** Wie können Bilder größengemäß angepasst werden, wenn ihre Originalgröße z.B. nicht auf den Bildschirm passt?
- **Beispiel für Subsampling:** Erzeugen eines Bildes, das nur ein Viertel so groß ist wie das Original.
- **Naives Subsampling (Löschen jeder zweiten Zeile und Spalte):** Führt zu einem Abtastproblem und erzeugt **Aliasing-Effekte**.
- Wenn man einfach nur Pixel steichen würde:



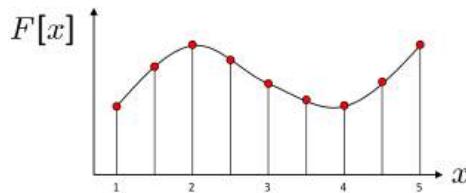
- **Korrekte Vorgehensweise für Subsampling:**
  1. Das Bild muss zuerst mit einem Gauß-Filter gefiltert werden.
  2. Anschließend wird das gefilterte Bild verkleinert.
  3. **Bedingung für den Filter:** Der Filter muss nach dem Abtasttheorem doppelt so groß wie die gewünschte Verkleinerung sein (bezogen auf die Frequenzen).
- **Grundlage für Multiskalenanalyse:** Dieser Theorie (des korrekten Subsamplings unter Berücksichtigung des Abtasttheorems) folgen im Weiteren auch alle Betrachtungen der Multiskalenanalyse.
- Hier das selbe Bild mit Gaußfilter:



## Resampling/Upsampling

- **Notwendigkeit:** Bei der Vergrößerung eines Bildes müssen neue Datenpunkte hinzugefügt werden.
- **Interpolation:** Die Werte dieser neuen Datenpunkte werden aus den benachbarten Pixeln des ursprünglichen Bildes interpoliert.

- What about arbitrary scale reduction?
- How can we increase the size of the image?



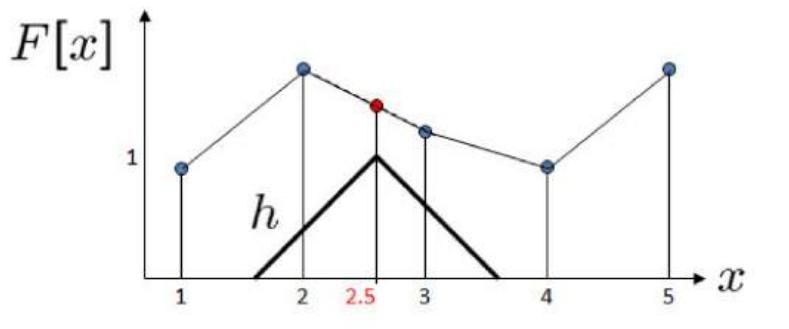
- Recall how a digital image is formed:

$$F[x, y] = \text{quantize}\{f(xd, yd)\}$$

- It is a discrete point-sampling of a continuous function
- If we could somehow reconstruct the original function, any new image could be generated, at any resolution and scale

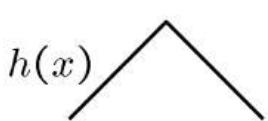
- **Grauwerte:** Die diskreten Grauwerte der Matrix des vergrößerten Bildes werden aus benachbarten Pixeln des Ausgangsbildes interpoliert.
- **Geometrie:** Die neue Matrix ist geometrisch durch das gewählte Bezugssystem definiert.
- **Neue Bildelemente:** Setzen sich aus Teilbereichen von Bildelementen der Eingabebildmatrix zusammen.
- **Filterkern:** Zur Grauwertzuweisung muss ein Filterkern definiert werden.
- **Gängige Resamplingverfahren:**

- **Bilineare Interpolation:** Berechnet den Wert einfach aus den Nachbarwerten (

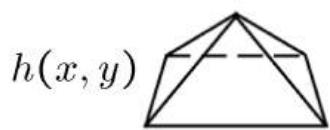


).

- **Filterfunktion:** Die Filterfunktion  $h(x)$  wird über  $F(x)$  gelegt und erzeugt den neuen Wert.
- **Bikubische Interpolation:** Geht von der vergrößerten neuen Bildmatrix aus und rechnet mit Hilfe von Transformationsgleichungen von den Mitten der dortigen Bildelemente in das Eingabebild zurück.

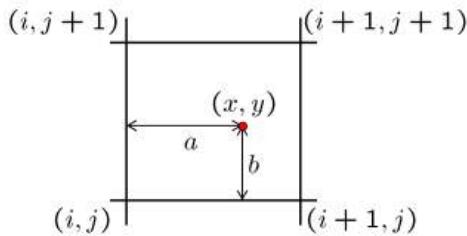


performs  
linear interpolation



performs  
bilinear interpolation

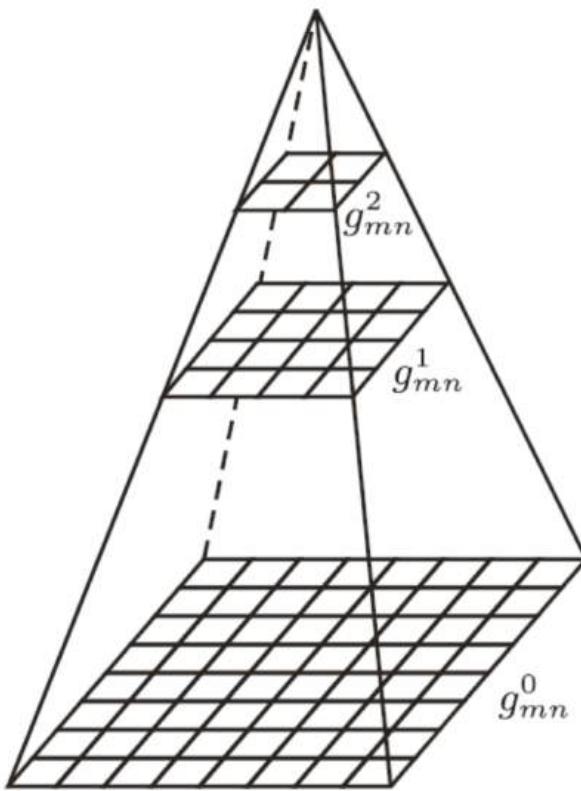
- A common method for resampling images



$$\begin{aligned}
 F(x, y) = & (1-a)(1-b) F(i, j) \\
 & + a(1-b) F(i+1, j) \\
 & + ab F(i+1, j+1) \\
 & + (1-a)b F(i, j+1)
 \end{aligned}$$

## Bildpyramiden zur effizienten Multiskalenanalyse

- **Motivation:**
  - Analyse von Strukturen auf verschiedenen Skalen für unterschiedliche Objekteigenschaften.
  - Erreichen einer gewissen Invarianz bezüglich der Objektgröße.
- **Problem naiver Multiskalenanalyse:** Beträchtlicher Rechenaufwand.
  - **Beispiel Korrelationsfilter:**
    - Bildgröße:  $N \times N$
    - Objektgröße:  $L \times L$
    - Aufwand Faltung im Ortsbereich:  $O(N^2 \cdot L^2)$
- **Lösung: Bildpyramiden**
  - **Konzept:** Mehrgitterdarstellung zur Verarbeitung von Signalen in unterschiedlichen Skalen.
  - **Effizienz:**
    - Feine Skalen → volle Auflösung erforderlich.
    - Grobe Strukturen → niedrigere Auflösung ausreichend.
  - **Aufbau:** Folge von Bildern, die von Stufe zu Stufe kleiner werden.
  - **Erzeugung:** Iterative Filterung und Unterabtastung des Bildes.
  - **Ergebnis:** Pyramide von Bildern, wobei jedes Bild eine bestimmte Skala repräsentiert.



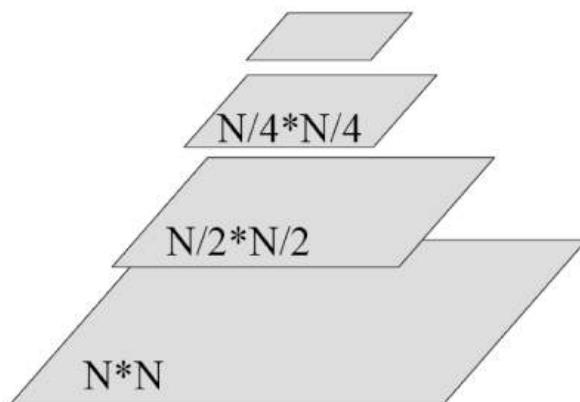
- **Problem einfache Abtastung:** Das einfache Entfernen jedes zweiten Bildpunkts in jeder zweiten Zeile führt zu Aliasing-Effekten.
- **Notwendigkeit Glättungsfilter:** Vor der Reduktion muss eine Glättung der höherfrequenten Strukturen durch einen passenden Glättungsfilter erfolgen.
- **Erzeugung des Skalenraums:** Größenreduktion muss mit einer angemessenen Glättung einhergehen.
- **Aufbau einer Bildpyramide:**
  - Start mit einem (o. B. d. A.) quadratischen Bild  $g_{mn}$  der Größe  $N \times N$  (Ebene  $v = 0$ ).
  - Berechnung der darüber liegenden Ebene mit reduzierter Größe (z.B. ein Viertel der Originalgröße).
  - **Schritt 1: Tiefpassfilterung** von  $g_{mn}$  auf halbe Bandbreite.
  - **Schritt 2: Unterabtastung** um den Faktor 2 (vermeidet Verletzung des Abtasttheorems durch vorherige Filterung).
  - **Ergebnis:** Gefiltertes und unterabgetastetes Bild  $g'_{mn}$  der Größe  $\frac{N}{2} \times \frac{N}{2}$  (Ebene  $v = 1$ ).
  - **Iteration:** Wiederholung des Vorgangs zur Erstellung der fehlenden Ebenen der Pyramide der Höhe  $K$ :  $v = 2, \dots, K; K = \log_2 N$ .

## Analyse auf verschiedenen Auflösungen in der Pyramide

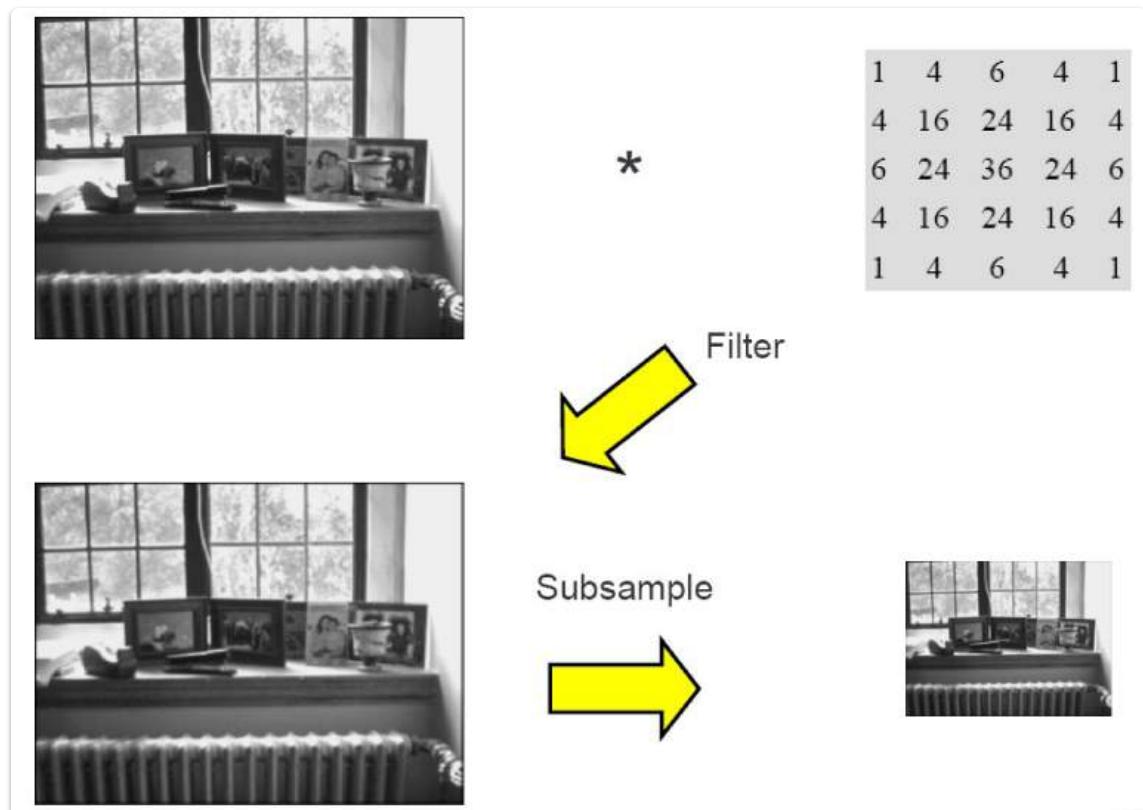
- **Vorteil:** Das Bild liegt in verschiedenen Auflösungen vor.
- **Analyse:** Sowohl grobe als auch feine Bildstrukturen können mittels kleiner lokaler Operatoren ausgewertet werden.
- **Äquivalenz:** Die Anwendung von Operatoren auf einer oberen Stufe der Pyramide ( $v > 0$ ) entspricht näherungsweise der Anwendung eines virtuell entsprechend vergrößerten

Operators auf das Originalbild ( $v = 0$ ).

- **Effiziente Detektion großer Strukturen:** Durch das Pyramidenkonzept gelangen grobe Strukturen sozusagen in Reichweite der lokalen Operatoren.
- **Reduzierter Aufwand:**
  - Betrachtet man die Detektionsaufgabe, so sinkt der Aufwand auf der Stufe  $v > 0$  der Pyramide auf  $O((\frac{N}{2^v})^2 \cdot (\frac{L}{2^v})^2)$ .
  - Die Abmessungen von Bild und Objekt verringern sich jeweils um den Faktor  $2^v$  auf  $\frac{N}{2^v} \times \frac{N}{2^v}$  bzw.  $\frac{L}{2^v} \times \frac{L}{2^v}$ .
  - Der Rechenaufwand verringert sich somit mit wachsendem  $v$  erheblich.
- **Kompromiss:** Die Detektion großer Objekte wird effizienter, allerdings auf Kosten des Verlusts feiner Details aufgrund der Tiefpassfilterung zwischen den Pyramidenstufen.

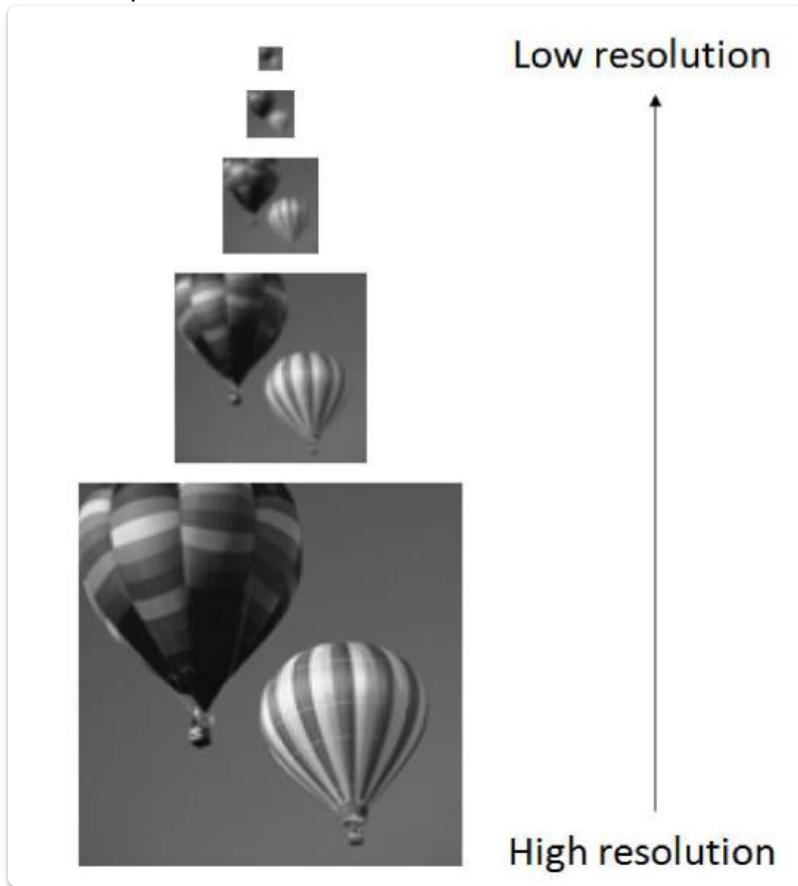


$$N^2 + \frac{1}{4}N^2 + \frac{1}{16}N^2 + \dots = N^2 + \frac{1}{3}N^2 = \frac{4}{3}N^2$$

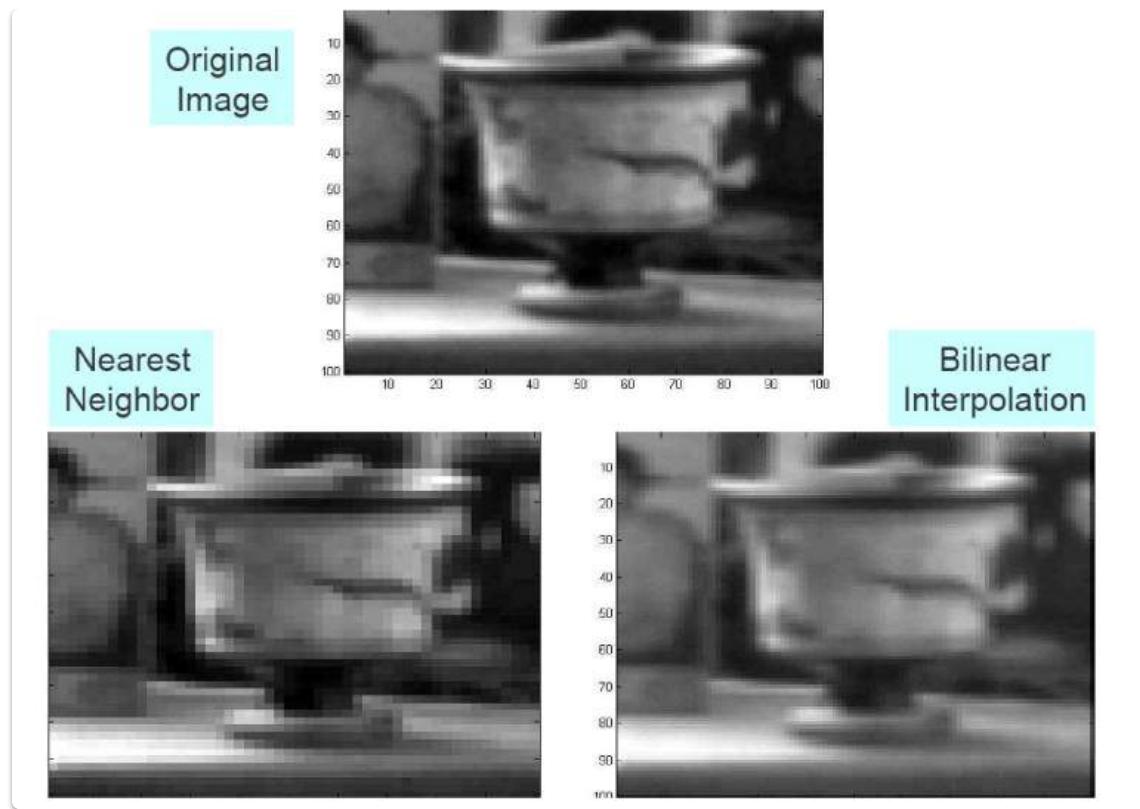


## Specheraufwand und Berechnungsaufwand von Bildpyramiden

- **Specheraufwand:** Trotz der Speicherung von Bildinformationen auf mehreren Skalen bleibt der Specheraufwand für eine Bildpyramide überschaubar.
  - Das unterabgetastete Bild  $g'_{mn}$  hat die Größe  $\frac{N}{2} \times \frac{N}{2}$ .
  - Das zweite unterabgetastete Bild hat die Größe  $\frac{N}{4} \times \frac{N}{4}$  usw.
  - Der Gesamtzahl der Bildpixel beträgt  $\sum_{i=0}^K (\frac{N}{2^i})^2 = N^2 \sum_{i=0}^K (\frac{1}{4})^i < N^2 \cdot \frac{1}{1-1/4} = \frac{4}{3} N^2$ .
  - Der Gesamtspeicherbedarf ist also nur ca.  $\frac{4}{3}$  des Speicherbedarfs des Originalbildes.
- **Berechnungsaufwand:** Ebenso effektiv ist die Berechnung der Pyramide.
  - Derselbe Glättungsfilter wird auf jede Ebene der Pyramide angewandt.
  - Damit erfordert die Berechnung der gesamten Pyramide lediglich  $\frac{4}{3}$  der Rechenoperationen für ein zweidimensionales Bild.



- **Nachbarschaftsoperationen:** Wenn die Pyramide einmal berechnet ist, können wir Nachbarschaftsoperationen mit großen Skalen in den oberen Ebenen der Pyramide durchführen.

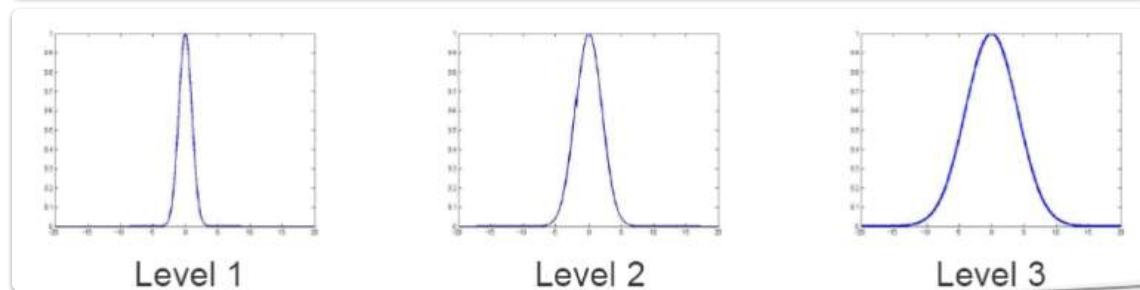
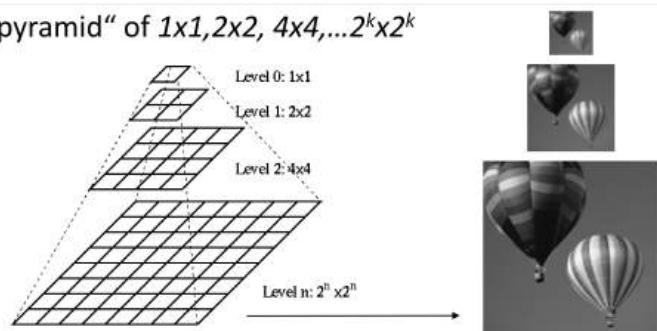


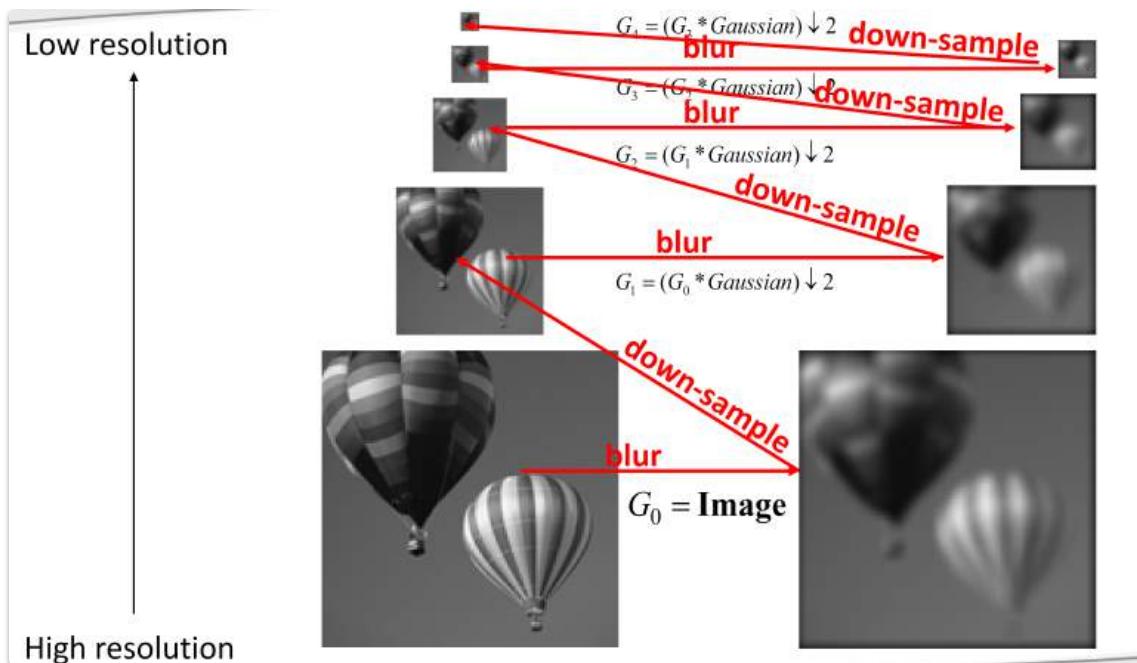
## Gaußpyramide

[EVC\\_Skriptum\\_CV](#), p.49, p.50

- **Definition:** Eine mit einem Tiefpass (Gaußfilter) konstruierte Bildpyramide heißt Gaußpyramide.

- **Idea:** Represent  $N \times N$  image as a „pyramid“ of  $1 \times 1, 2 \times 2, 4 \times 4, \dots 2^k \times 2^k$  images (assuming  $N = 2^k$ )





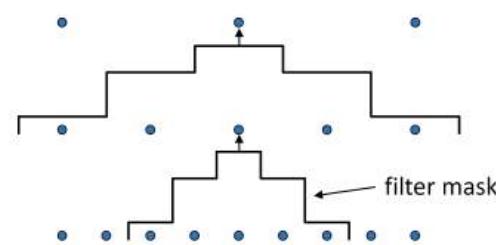
- **Erzeugung:**

- Tiefpassfilterung und Unterabtastung um den Faktor 2 an einem zweidimensionalen Signal  $g_m$ .
- Jeder abgeleitete Pixel wird jeweils aus fünf darunterliegenden Pixeln durch eine Gewichtung des  $1 \times 5$  Gaußfilters  $[1/16 \ 4/16 \ 3/8 \ 1/4 \ 1/16]$  erzeugt.
- Die kombinierte Glättung und Größenreduktion bei der Berechnung der  $(v+1)$ -ten Pyramidenebene aus der  $v$ -ten Ebene kann mit einem einzigen Operator ausgedrückt werden:

$$G^{(v+1)} = B \downarrow_2 G^{(v)}$$

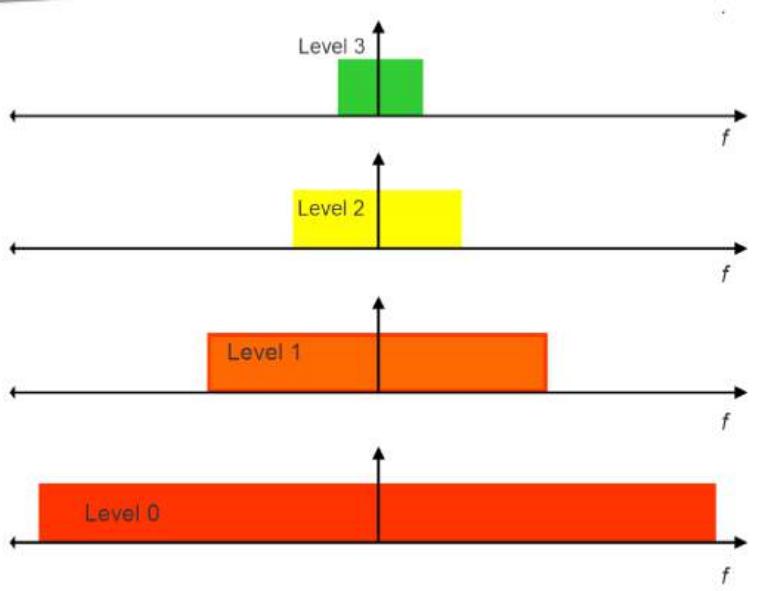
$\downarrow_2$  bezeichnet die Unterabtastung.

Die nullte Ebene ( $G^{(0)}$ ) ist das Originalbild.



- Repeat
  - Filter
  - Subsample
- Until minimum resolution reached
  - can specify desired number of levels (e.g., 3-level pyramid)
- The whole pyramid is only  $4/3$  the size of the original image!

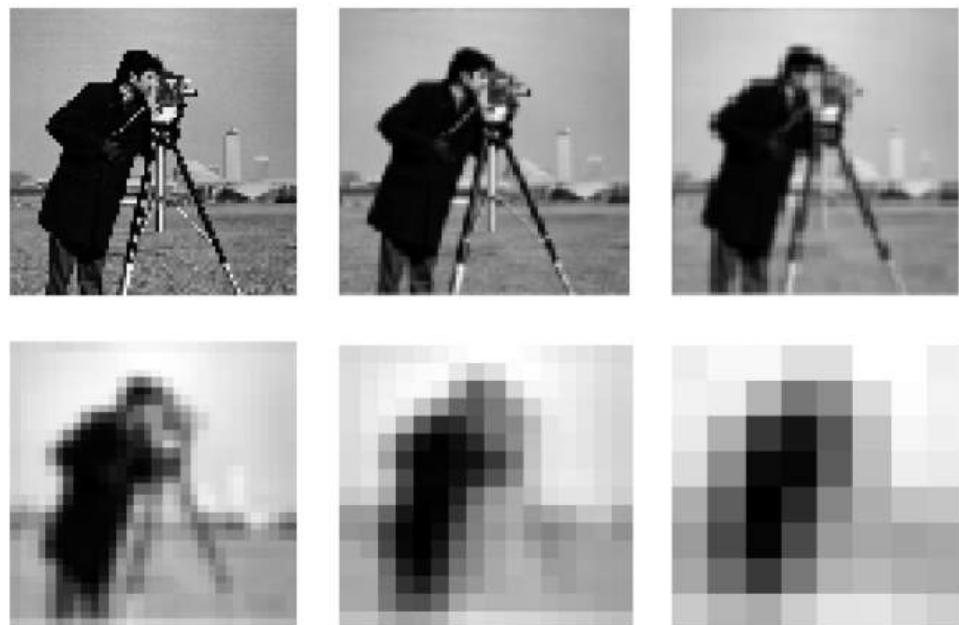
- **Struktur:** Die durch mehrmalige Anwendung dieses Operators erhaltene Bildserie wird als Gaußpyramide bezeichnet.
  - Von einer Ebene zur nächsten nimmt die Auflösung auf die Hälfte ab.
  - Man kann sich die Bildserie ebenfalls in Form einer Pyramide angeordnet vorstellen.



- **Eigenschaften:**

- Die Gaußpyramide stellt eine Serie von tiefpassgefilterten Bildern dar.
- Bei den oberen Ebenen nimmt die Grenzfrequenz von Ebene zu Ebene auf die Hälfte (eine Oktave) abnimmt.
- Damit verbleiben zunehmend gröbere Details im Bild.
- Um den gesamten Bereich der Frequenzen zu umspannen, sind nur wenige Pyramidenebenen erforderlich.
- Aus einem  $N \times N$ -Bild können wir eine Pyramide mit maximal  $\log_2(N) + 1$  Ebenen berechnen.
- Das kleinste Bild besteht nur aus einem Bildpunkt.

```
![[EVC_Skriptum_CV.pdf#page=50&rect=168,302,438,448|EVC_Skriptum_CV,  
p.49|500]]
```



- **Darstellung in Abbildung 42:** Jede Ebene wurde wieder auf die Originalauflösung skaliert, um Details an den oberen Ebenen zu zeigen.

## Anwendungsbeispiele

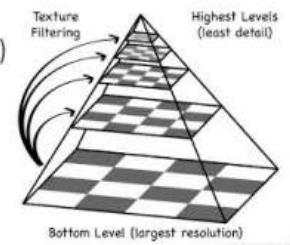
- **Improve Search**

- Search over translations: Classic coarse-to-fine strategy
- Search over scale: Template matching, e.g., find a face at different scales



- **Pre-computation**

- Need to access the image at different blur levels
- Useful for texture mapping at different resolutions (mip-mapping)



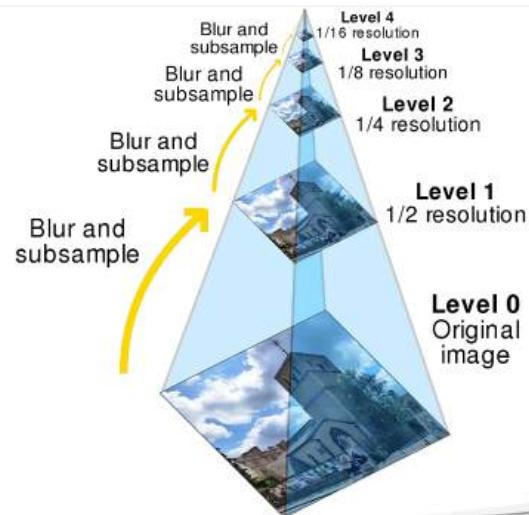
- **Image Processing**

- Editing frequency bands separately
- E.g., image blending...

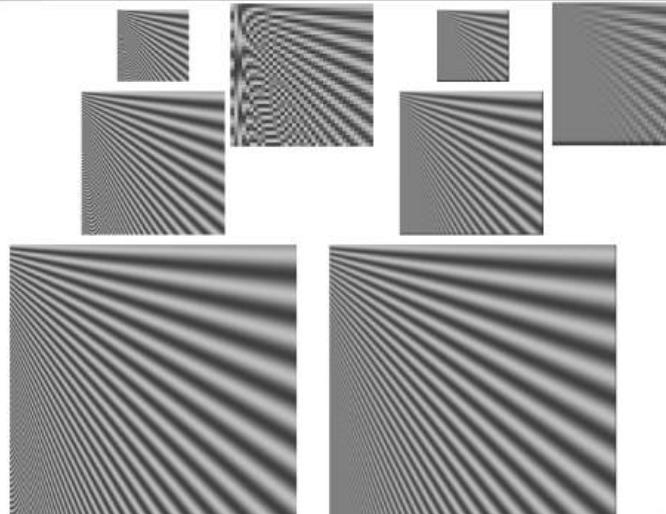
- **Up- or down-sampling images**

- **Multi-resolution image analysis**

- Look for an object over various spatial scales
- Coarse-to-fine image processing: from blur estimate or the motion analysis on a very low-resolution image, up-sample and repeat.
- Often a successful strategy for avoiding local minima in complicated estimation tasks.



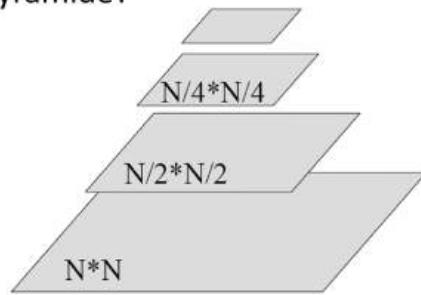
- A Gaussian pyramid (on the right) compared with a pyramid obtained by sampling without smoothing



## Testähnliches Beispiel

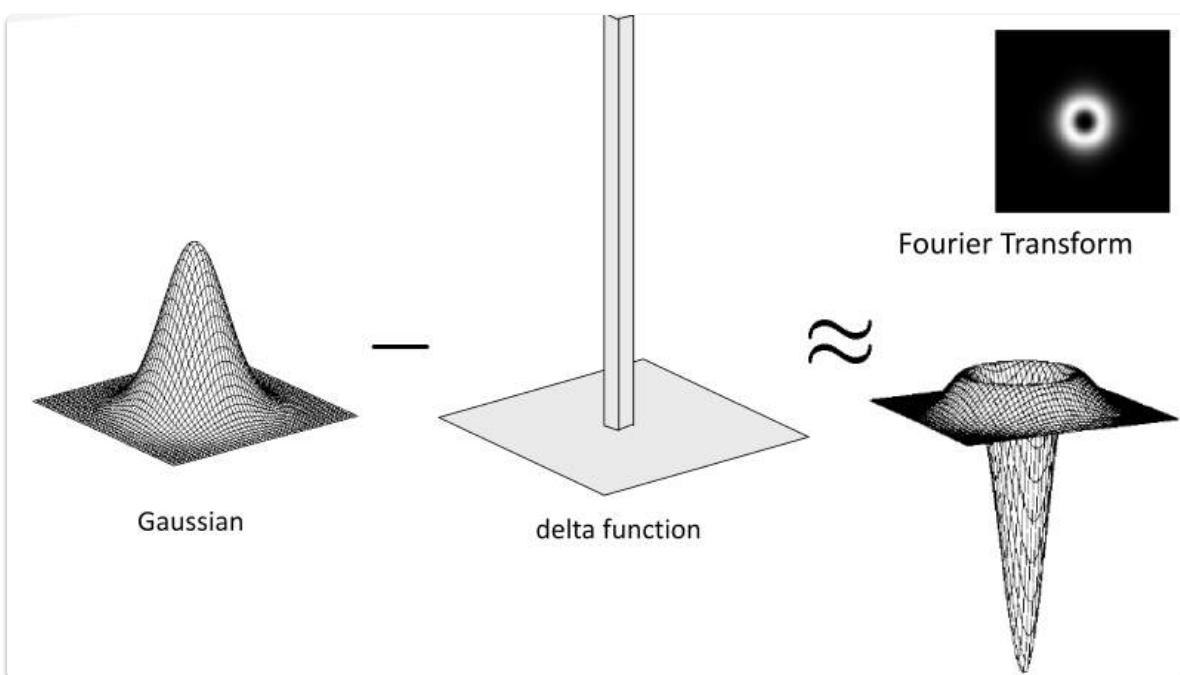
- Angenommen, ein Bild der 1. Ebene einer Gaußpyramide besteht aus 24.336 Pixeln ( $156 \times 156$ ), was der Originalauflösung des Bildes entspricht. Wie viele Pixel hat das Bild auf der 3. Ebene der Pyramide?

1. Ebene:  $156 \times 156$
2. Ebene:  $156/2 \times 156/2 = 78 \times 78$
3. Ebene:  $78/2 \times 78/2 = 39 \times 39 = \mathbf{1.521 \text{ Pixel}}$



## Laplacepyramide (Difference of Gaussians)

EVC\_Skriptum\_CV, p.50

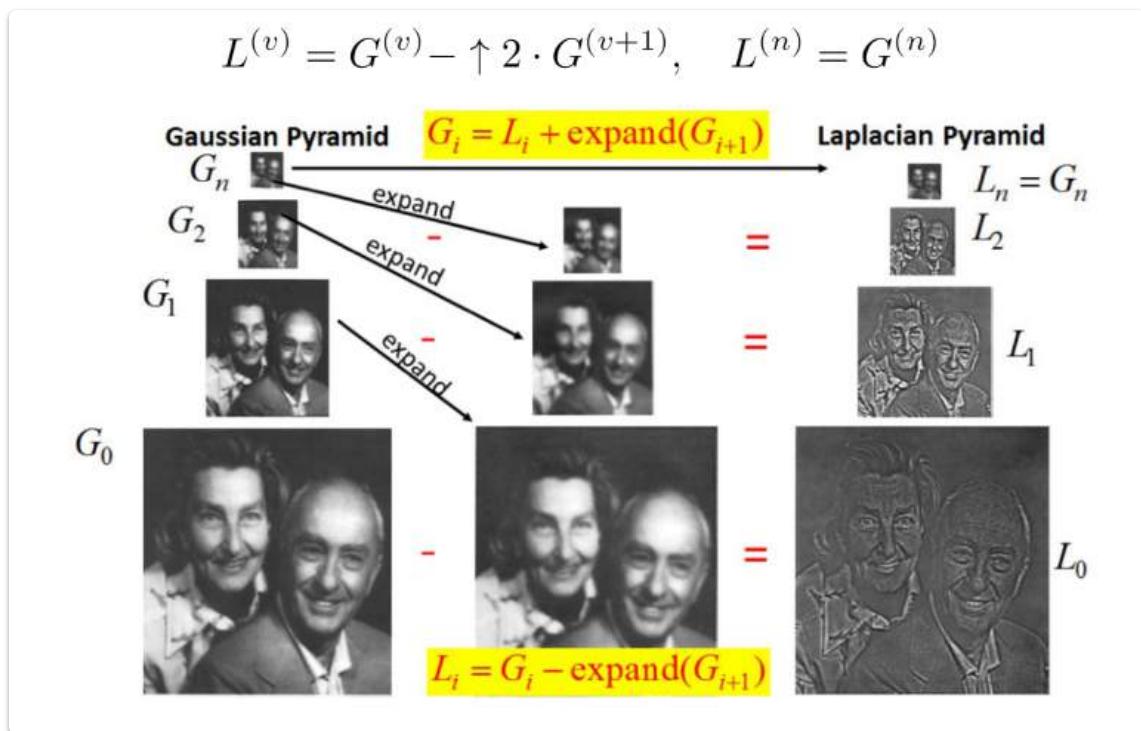


- **Konstruktion:**
  - Wird durch Differenzbildung mittels einer Tiefpasspyramide konstruiert.
  - In jedem Schritt werden Signalanteile mit hoher Ortsfrequenz herausgefiltert, sodass auf den oberen Stufen der Pyramide nur noch niedrige Ortsfrequenzen vorhanden sind.
- **Alternative Konstruktion:**
  - Ähnlich wie die Gaußpyramide konstruiert, aber jede Stufe ist das Ergebnis einer Bandpassfilterung des Originalbildes.
  - Eine einfache Möglichkeit zur Bandpassfilterung: Differenz zweier aufeinanderfolgender Bilder einer Gaußpyramide (Difference-of-Gaussians).
  - **Expansion:** Das Bild in der größeren Ebene wird zuerst expandiert.
    - **Expansionsoperator:**  $\uparrow_2$  durchgeführt.
    - **Reduzierung Glättungsoperator:** Grad der Glättung durch die Zahl nach dem  $\uparrow$ -Zeichen im Index angegeben.

- **Interpolation:** Der Expansion ist schwieriger als die Größenreduktion, da fehlende Information interpoliert werden muss.
- **Vergrößerung:** Um den Faktor zwei in allen Richtungen muss jeder zweite Bildpunkt in jeder Zeile interpoliert und dann jede zweite Zeile.
- **Erzeugung der  $v$ -ten Ebene der Laplacepyramide:**

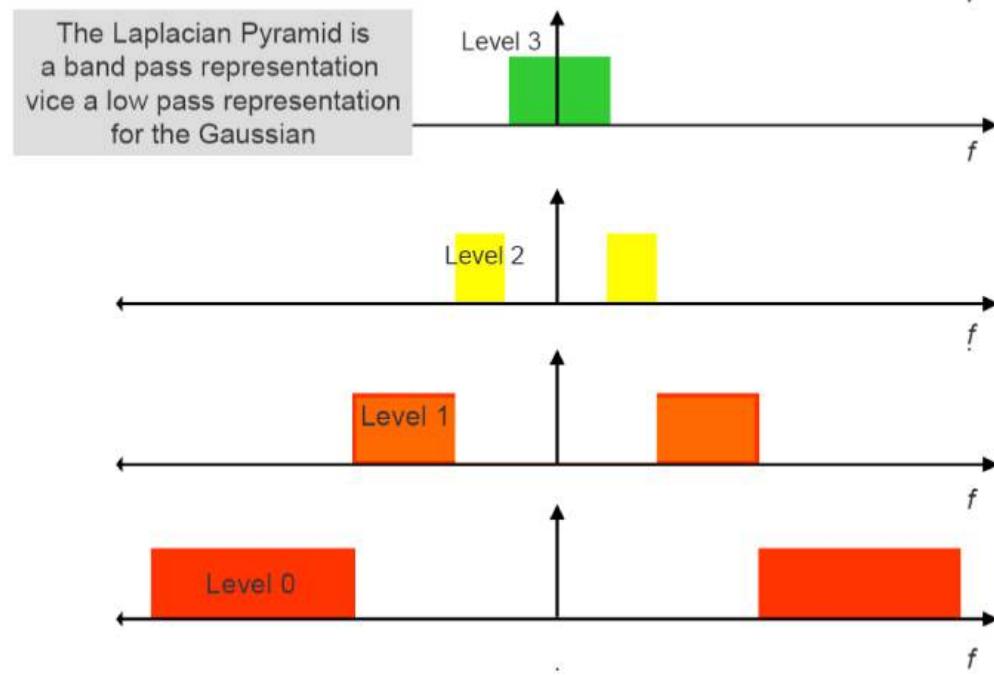
$$L^{(v)} = G^{(v)} - \uparrow_2 G^{(v+1)}, \quad L^{(n)} = G^{(n)}$$

- $G^{(v)}$ :  $v$ -te Ebene der Gaußpyramide.
- $\uparrow_2 G^{(v+1)}$ : Expandierte  $(v+1)$ -te Ebene der Gaußpyramide.
- $L^{(v)}$ :  $v$ -te Ebene der Laplacepyramide.
- Die oberste Ebene der Laplacepyramide ist die oberste (kleinste) Ebene der Gaußpyramide.



- **Eigenschaften der Laplacepyramide:**

- Stellt die Approximation der zweiten Ableitung des Originalbildes mit unterschiedlichen Glättungsparametern dar.
- Auf den ersten Stufen der Laplace-Pyramide werden feine Kantenstrukturen hervorgehoben.
- Während auf den höheren Stufen gröbere Kantenstrukturen des Originalbildes sichtbar sind.
- Die Laplacepyramide ist somit eine Zerlegung des Bildsignals in Bandpassbereiche mit logarithmischer Frequenzstaffelung.
- Das Originalbild kann aus der Laplace-Pyramide sowie dem obersten Bild der Gauß-Pyramide exakt rekonstruiert werden.
- Dies geschieht einfach durch eine Umkehrung des Konstruktionsschemas.



- **Vergleich zur Fouriertransformation:**

- Die Laplacepyramide ist lediglich zu einer groben Frequenzzerlegung ohne Richtungszerlegung.
- Alle Frequenzen innerhalb des Bereiches von ungefähr einer Oktave (Faktor 2) sind unabhängig von ihrer Richtung in einer Ebene der Pyramide enthalten.

## Anwendungen von Bildpyramiden

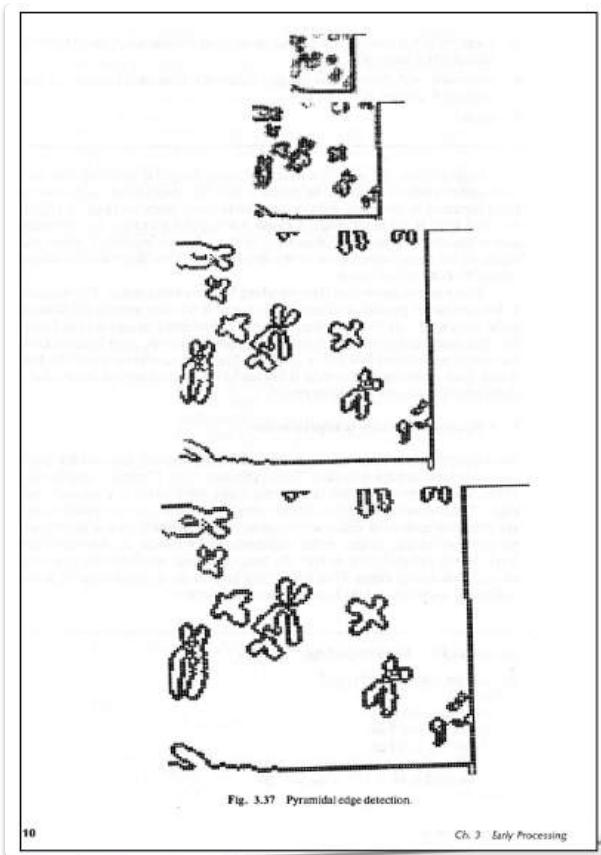
- **Coarse-to-Fine Strategien für Recheneffizienz:**

- **Suche nach Korrespondenzen:**
  - Suche auf groben Skalen, dann Verfeinerung mit feineren Skalen.
- **Kantenverfolgung (Edge Tracking):**
  - Eine "gute" Kante auf einer feinen Skala hat Eltern auf einer größeren Skala.
- **Kontrolle von Detail und Rechenaufwand beim Matching:**
  - Z.B. Finden von Streifen.
  - Sehr wichtig bei Texturerkennung.
- **Bildmischung und Mosaikbildung (Image Blending and Mosaicking).**
- **Datenkompression (Laplace-Pyramide).**

## Kantendetektion mit Pyramiden

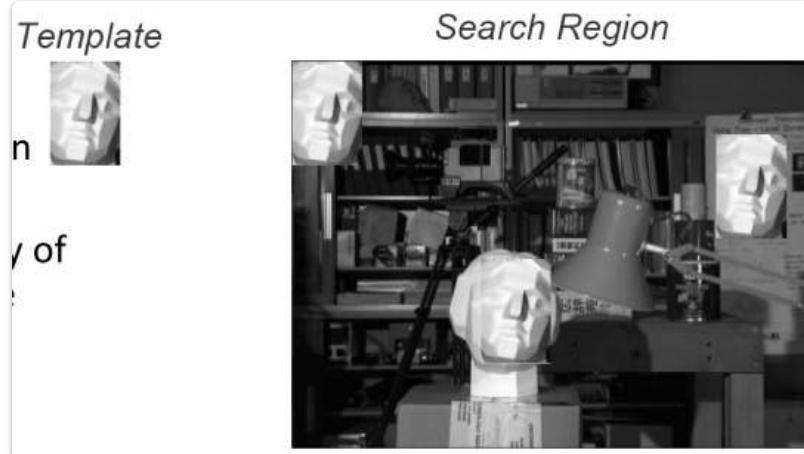
- **Coarse-to-Fine Strategie:**

- Kantendetektion auf einer höheren Ebene durchführen.
- Kanten feinerer Skalen nur in der Nähe der Kanten höherer Skalen berücksichtigen.



## Schnelles Template Matching

- Für ein  $m \times n$  Bild...
- Für ein  $p \times q$  Template...
- Die Komplexität der 2D Mustererkennungsaufgabe ist  $O(m \cdot n \cdot p \cdot q)$ .
- Dies wird noch schlimmer für eine Familie von Templates (z.B. um Skalierungs- und/oder Rotationseffekte zu berücksichtigen).

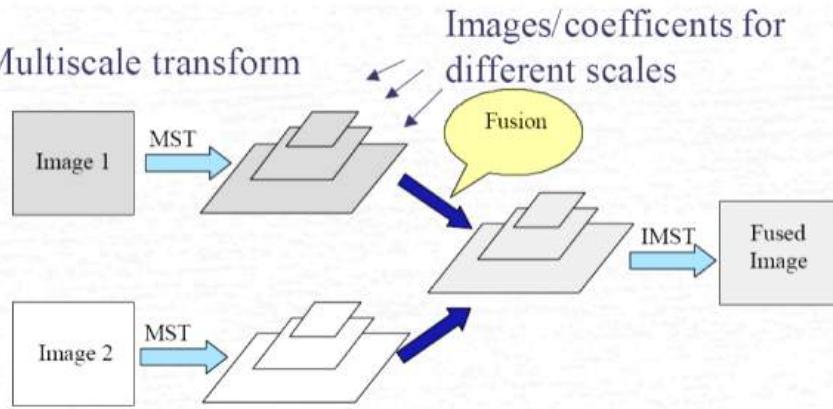


*Template**Search Region*

Original Image

**Image Fusion**

MST = Multiscale transform

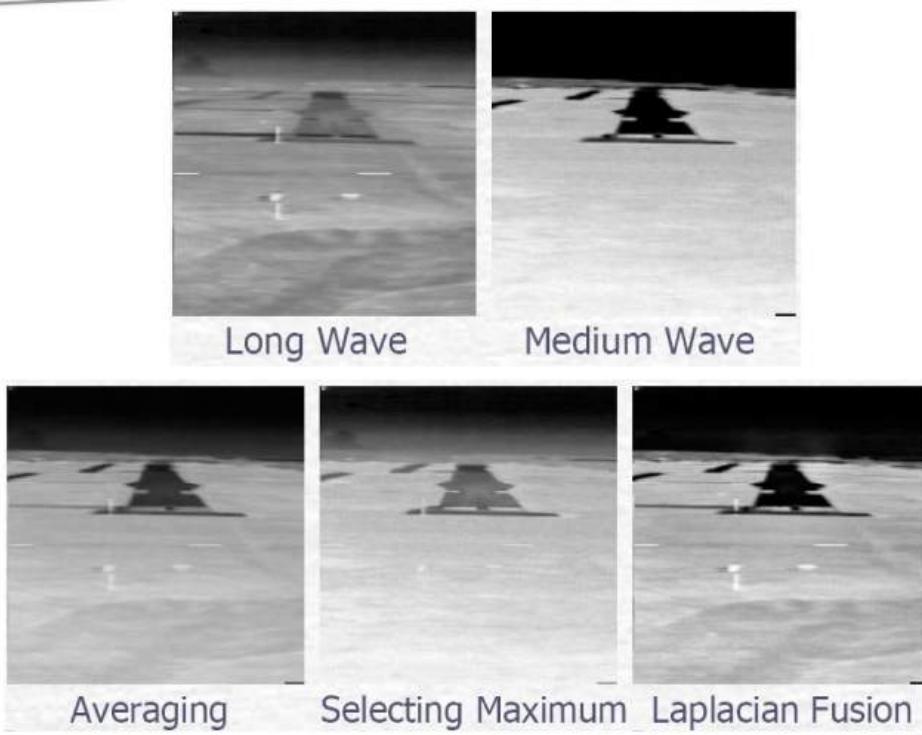


Multi-Scale Transform (MST)

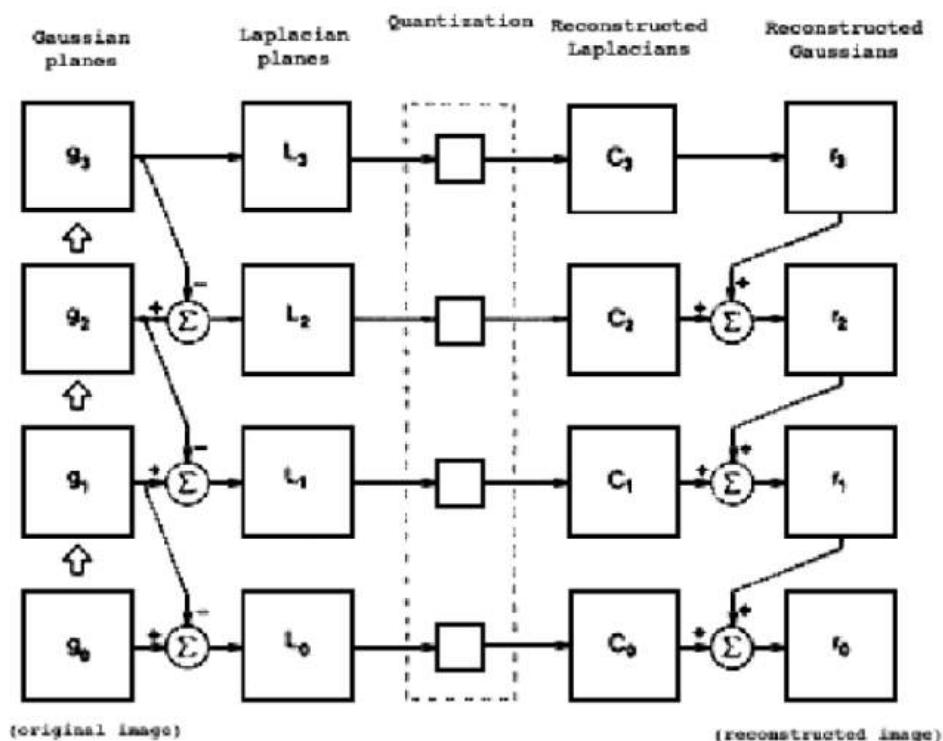
= Obtain Pyramid from Image

Inverse Multi-Scale Transform (IMST) = Obtain Image from Pyramid

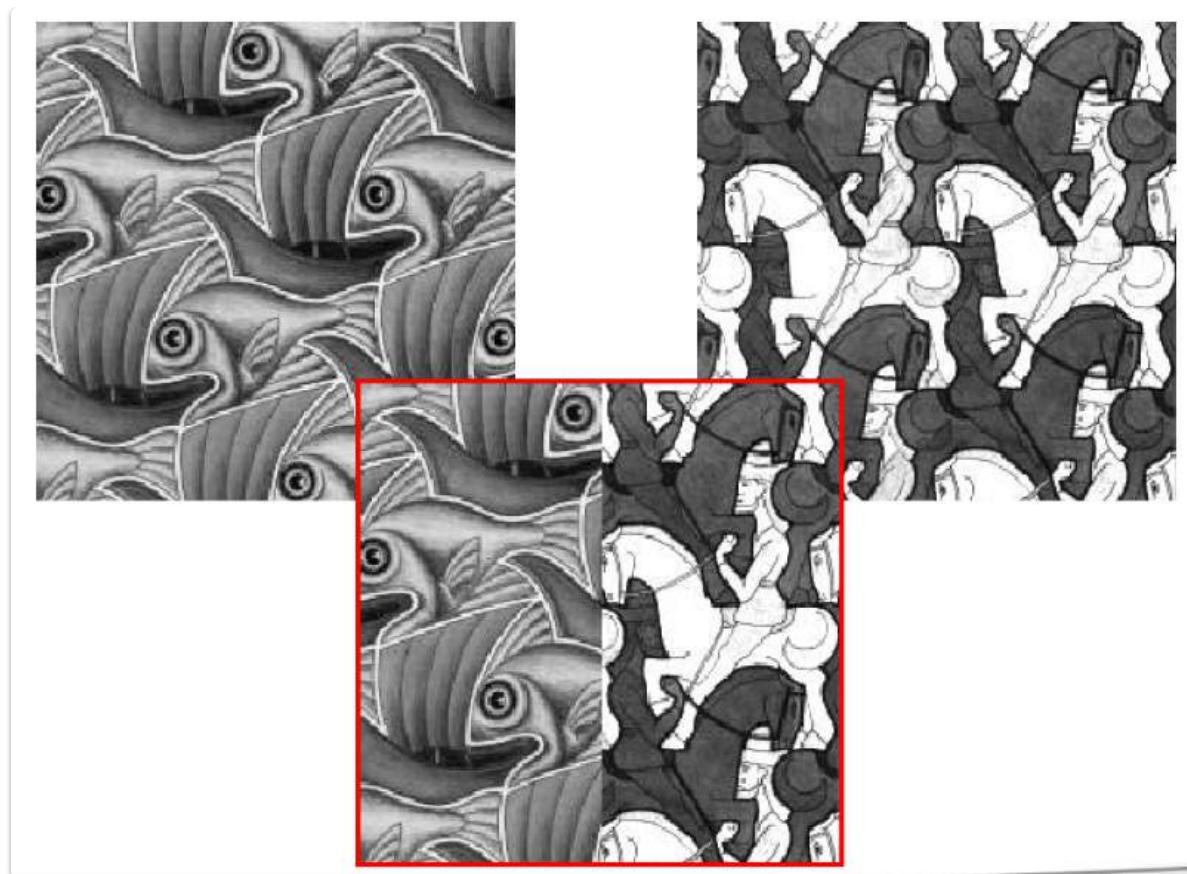
**Multi-Sensor Fusion**



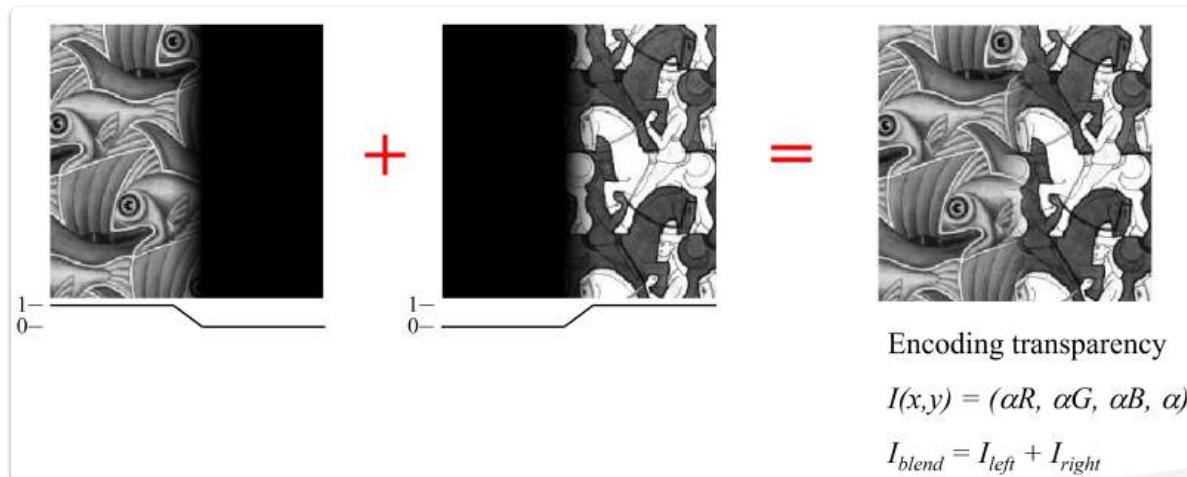
## Image Compression



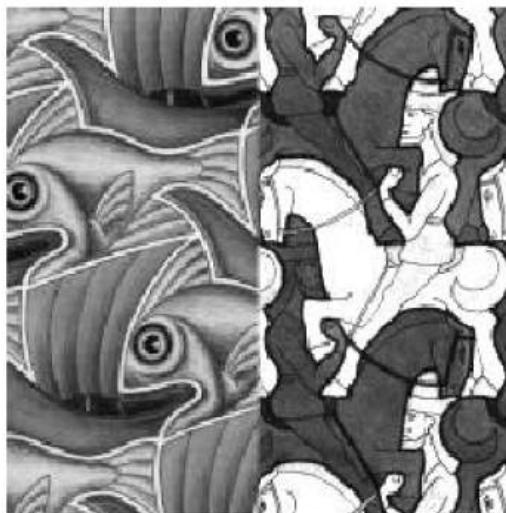
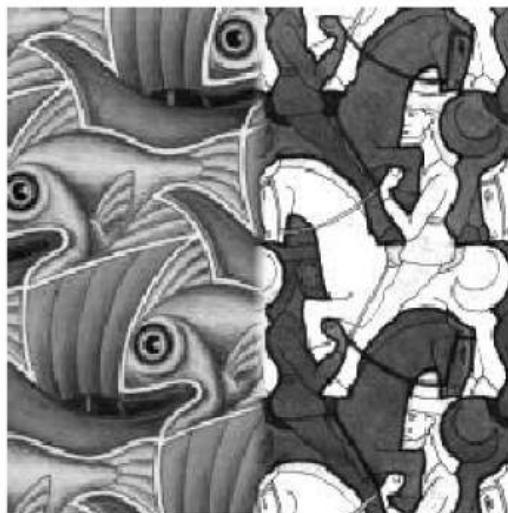
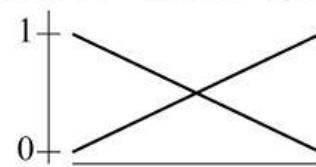
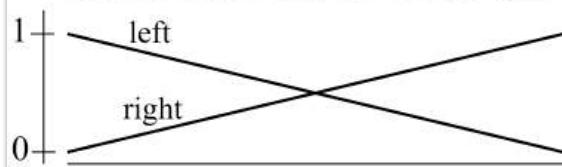
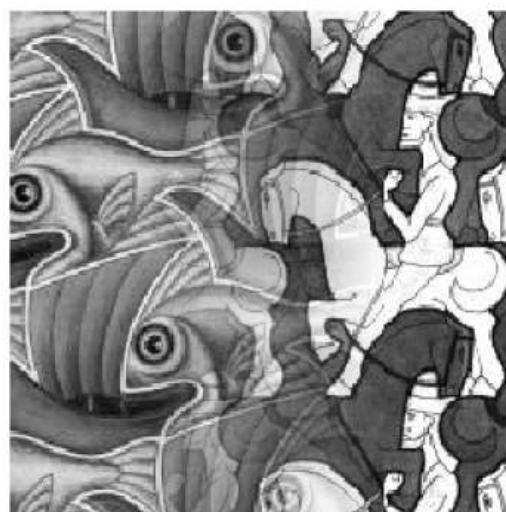
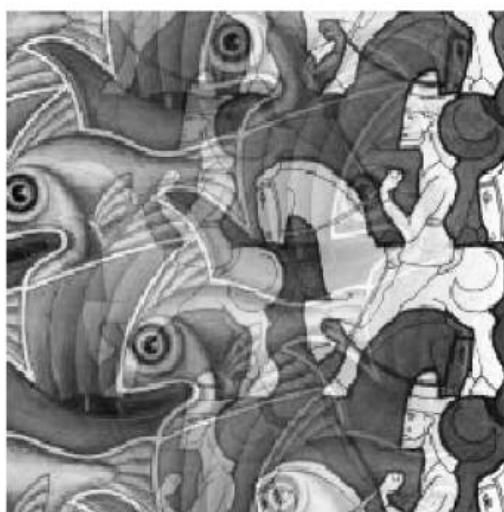
## Image Blending



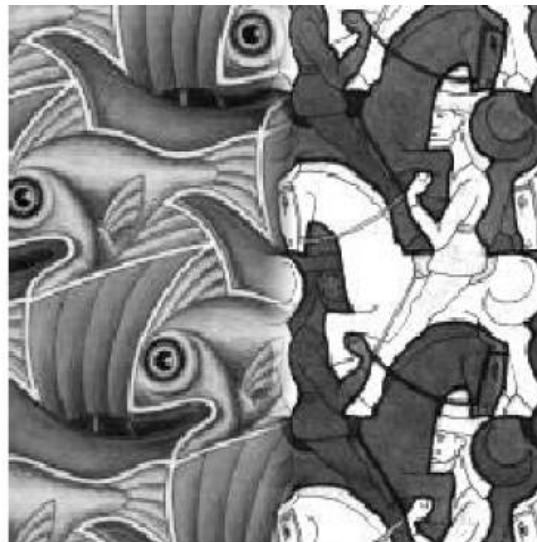
## Feathering



## Affect of Window Size



Gute Bildgröße:

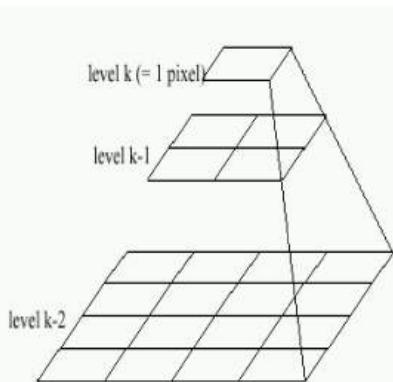


## “Optimal” Window: smooth but not ghosted

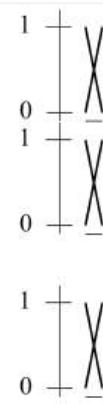
Was ist ein gutes Window?

- **Vermeidung von Nähten (Seams):**
  - Fenstergröße  $\geq$  Größe des größten prominenten Features.
- **Vermeidung von Geisterbildern (Ghosting):**
  - Fenstergröße  $\leq 2 \times$  Größe des kleinsten prominenten Features.

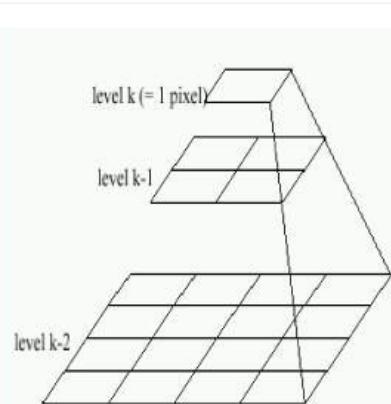
## Pyramid Blending



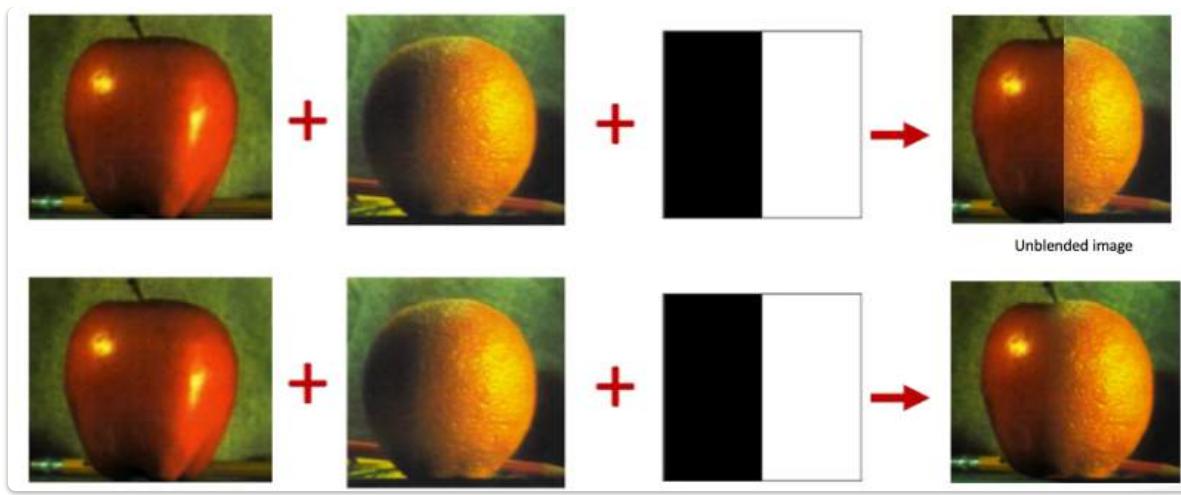
Left pyramid



blend



Right pyramid



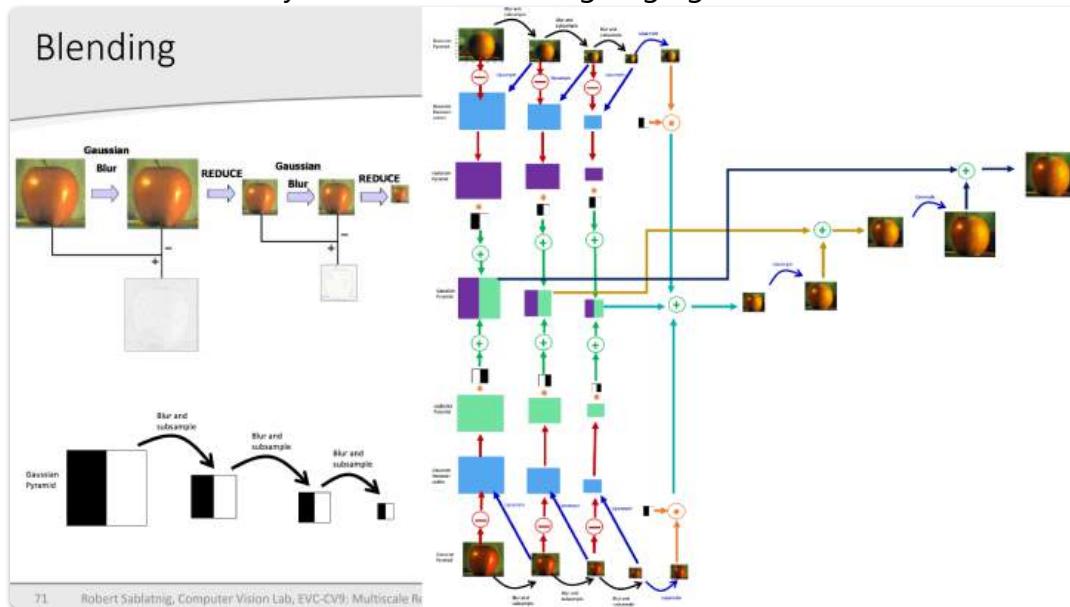
## Laplace-Pyramide: Bildmischung

- Genereller Ansatz:

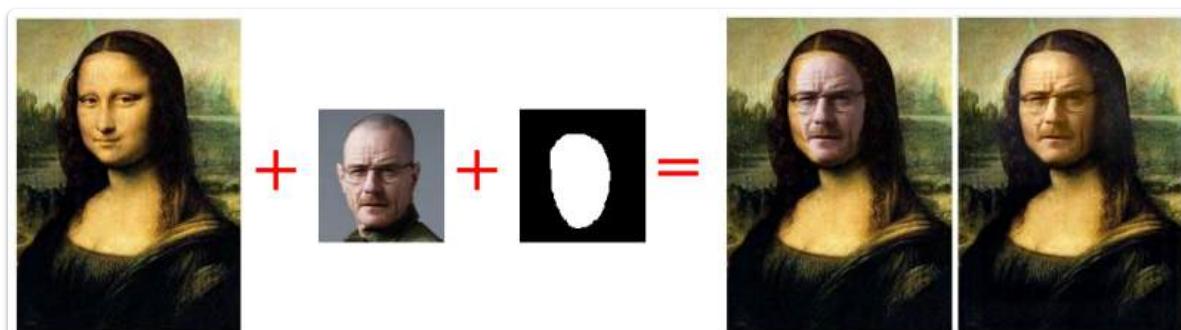
- Erstelle Laplace-Pyramiden  $LA$  und  $LB$  von den Bildern  $A$  und  $B$ .
- Erstelle eine Gauß-Pyramide  $GR$  von der ausgewählten Region  $R$ .
- Bilde eine kombinierte Pyramide  $LS$  aus  $LA$  und  $LB$  unter Verwendung der Knoten von  $GR$  als Gewichte:

$$LS(i, j) = GR(i, j) \cdot LA(i, j) + (1 - GR(i, j)) \cdot LB(i, j)$$

- Kollabiere die  $LS$ -Pyramide, um das endgültige gemischte Bild zu erhalten.



## Blending Regions



waltuh

# 10. Stereo und Motion

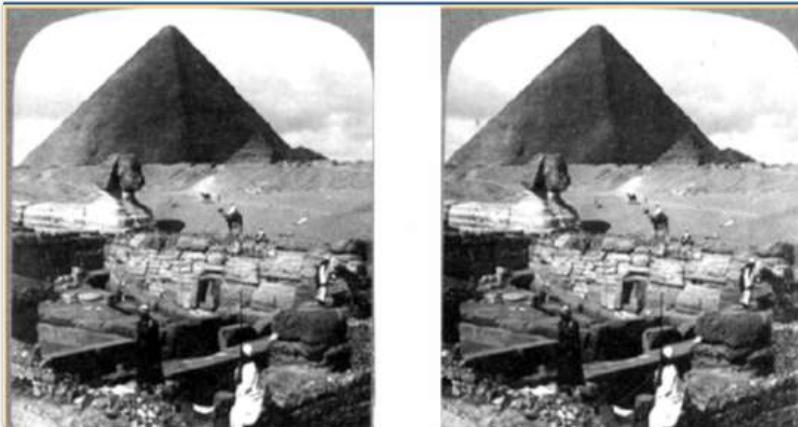
## Stereo Vision

[EVC\\_Skriptum\\_CV, p.51](#)

- Zusammengesetzt aus griechisch "stereos" (räumlich, fest) und lateinisch "videre" (sehen).
- Bezeichnet räumliches Sehen bzw. binokulares Sehen.
- Ziel: Erstellung eines Tiefenbildes aus zwei Bildern einer Szene.
- Bilder sind 2D-Projektionen einer 3D-Szene, wobei eine Dimension (die Tiefe) verloren geht.
- Der Mensch kann Tiefeninformationen aus seiner Umgebung durch binokulares Sehen gewinnen.

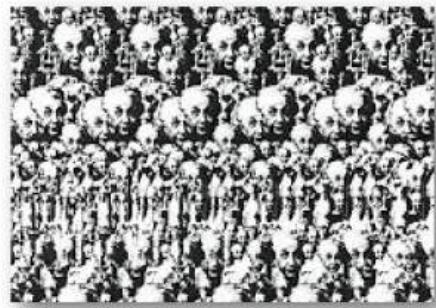
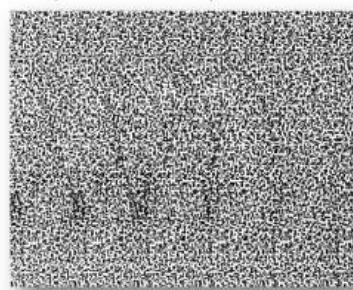
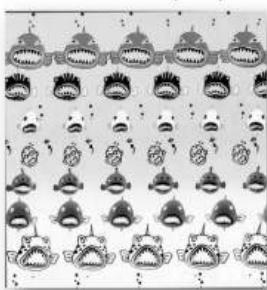
Prinzip der Stereo Vision:

- Nutzt zwei Kameras mit bekanntem Abstand zueinander.
- Anwendung geometrischer Prinzipien (Triangulation und Epipolargeometrie) zur Berechnung korrespondierender Bildpunkte.
- Rekonstruktion der Tiefenwerte.



For some people, Random Dot Stereograms are complicated to view:  
Autostereograms [Tyler77]:

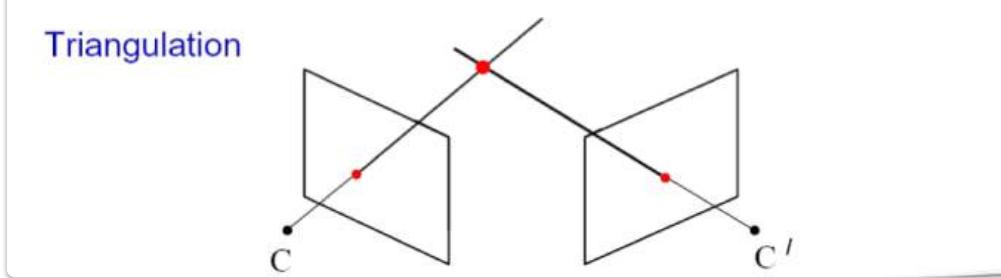
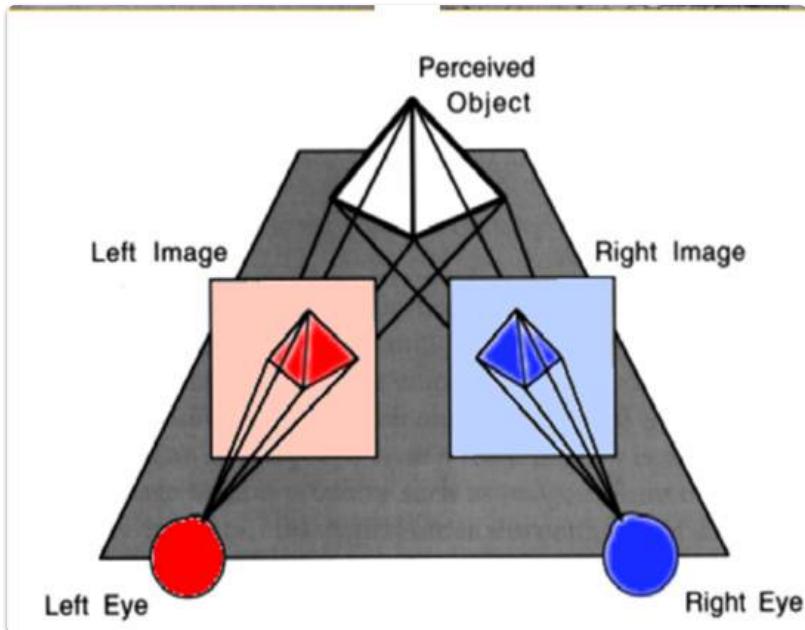
- Also called single image Stereogram
- Form of Art
- Is formed by repetition of patterns in specific intervals



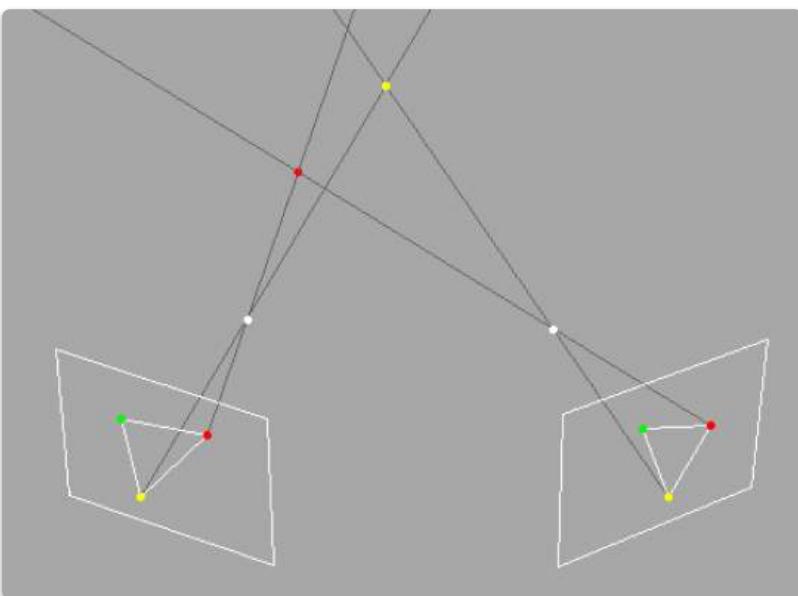
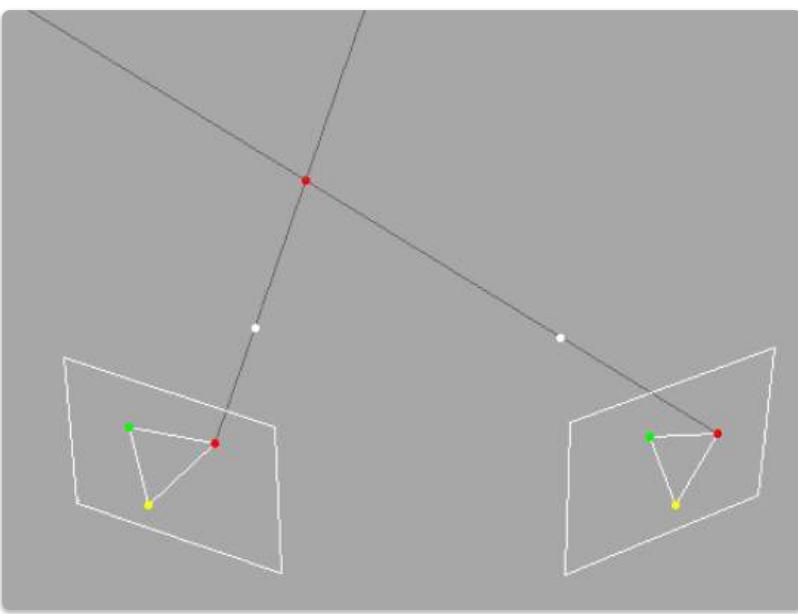
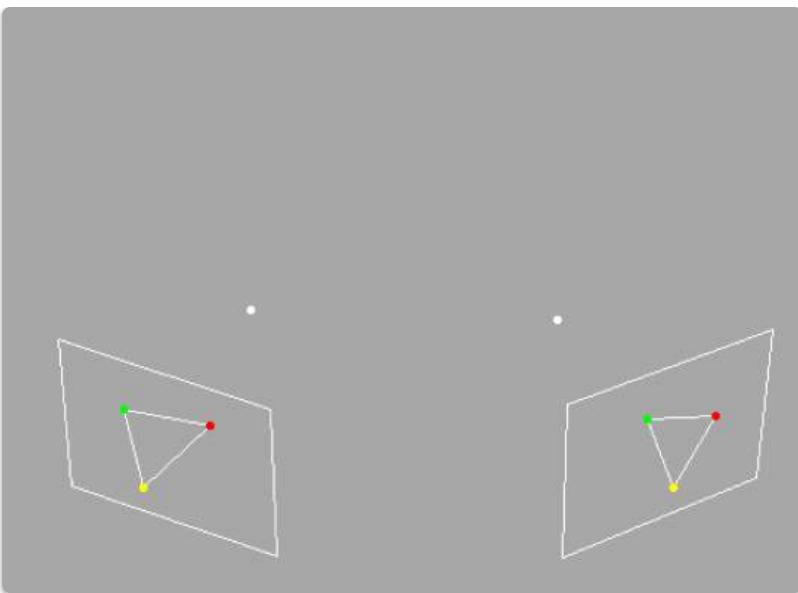
Tyler C.W. and Chang J.J. (1977) Visual echoes: The perception of repetition in quasi-random patterns. *Vision Res.* 17, 109-116.

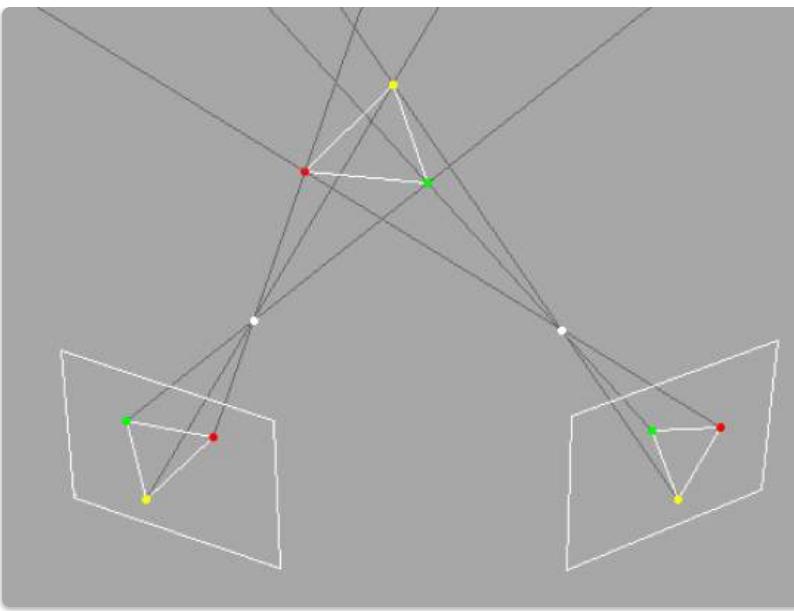
## Disparität:

- Leichter Versatz der beiden Kameras erzeugt unterschiedliche Bilder.
- Der Versatz in den Kamerabildern ist geringer für entfernte Objekte und größer für nahe Objekte.
- Dieser Versatz wird als Disparität bezeichnet.
- Durch Zuweisung der unterschiedlichen Disparitäten zu jedem Bildpunkt wird eine Disparitätsmatrix erstellt.
- Aus der Disparitätsmatrix lässt sich für jeden Bildpunkt die Tiefe ableiten.



**Triangulation: Fundament von Stereo Vision**

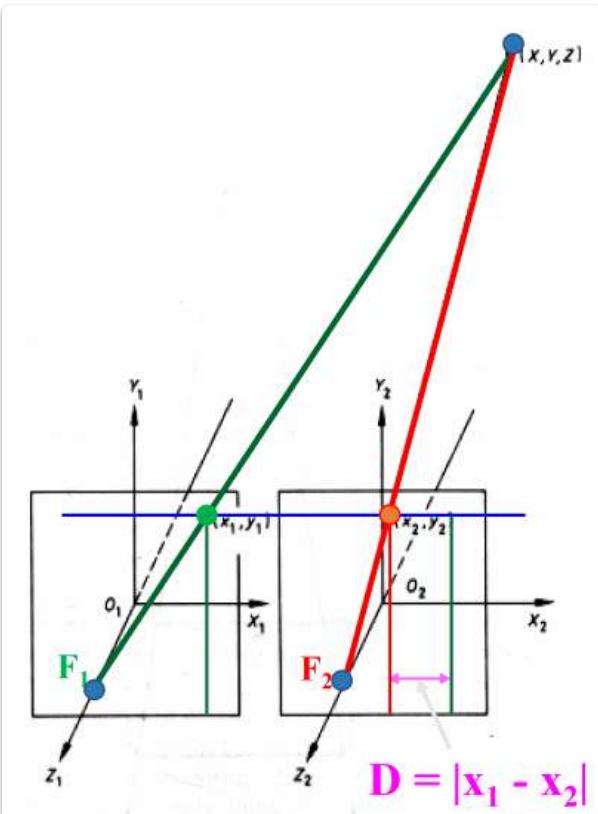




## Stereo Geometry

Grundlagen:

- Betrachtet werden *Perspektivische Projektionen* von Objektpunkten.
- **Spezifischer Fall (vereinfacht):** Beide Bildebenen sind parallel zueinander ausgerichtet.
  - Dieser spezifische Fall kann zur Erklärung des generellen Falls herangezogen werden.
- Durch die Nutzung der geometrischen Position der Bildebenen und der Information über die Tiefenprojektion des Objekts kann die Tiefe des Objekts berechnet werden.



Wichtige Elemente in der Abbildung:

- $(X, Y, Z)$ : Ein Punkt im 3D-Raum.
- $F_l$ : Projektionszentrum der linken Kamera.
- $F_r$ : Projektionszentrum der rechten Kamera.
- Bildebenen (parallel zueinander).
- $x'_l$ : Projektion des 3D-Punktes auf der linken Bildebene.
- $x'_r$ : Projektion des 3D-Punktes auf der rechten Bildebene.
- $D = |x'_l - x'_r|$ : Die *Disparität* – der horizontale Abstand zwischen den korrespondierenden Bildpunkten.

Zusammenfassend: Stereo Vision nutzt zwei leicht versetzte Kameras, um durch die Analyse der Disparität zwischen den beiden resultierenden Bildern Tiefeninformationen zu gewinnen. Die geometrische Beziehung zwischen den Kameras ermöglicht die Berechnung der 3D-Positionen der Punkte in der Szene.

## Stereoskopie

---

[EVC\\_Skriptum\\_CV, p.51](#)

Stereoskopie (griechisch "skopeo" = betrachten):

- Wiedergabe von Bildern mit einem räumlichen Eindruck von Tiefe.
- Die Tiefe ist physikalisch nicht vorhanden.
- Umgangssprachlich fälschlich als "3D" bezeichnet, obwohl es sich um zweidimensionale Abbildungen handelt, die einen räumlichen Eindruck vermitteln.

Prinzip des räumlichen Sehens:

- Bereits im 3. Jh. v. Chr. von dem griechischen Mathematiker Euklid beschrieben.
- Viele Wissenschaftler (u.a. Leonardo da Vinci) beschäftigten sich mit diesem Phänomen.

Geschichte der Stereoskopie:

- 19. Jh.: Charles Wheatstone entdeckte die Stereoskopie.
  - Hielt 1838 einen bahnbrechenden Vortrag über "einige merkwürdige und bisher nicht beobachtete Erscheinungen beim beidäugigen Sehen".
  - Berechnete und zeichnete Bildpaare.
  - Konstruierte das Stereoskop, ein Apparat, um diese Bildpaare räumlich betrachten zu können



Weitere Entwicklung:

- 1849: David Brewster stellte die erste Stereokamera vor.
  - Ermöglichte erstmals, ein bewegtes Motiv aufzunehmen (allerdings noch nicht für Fotografie im heutigen Sinne).
- 1851: Durchbruch auf der Weltausstellung in London.

- Hardware for Viewing (3D-TV sets):
  - Anaglyph
  - Polarized
  - Field-sequential (Active shutter)
  - Lenticular display

Anaglyph

Polarizing

Active shutter

Lenticular

Zusammenfassend: Die Stereoskopie erzeugt einen räumlichen Eindruck aus zwei leicht unterschiedlichen, zweidimensionalen Bildern. Historisch gesehen reicht die Beobachtung dieses Phänomens bis in die Antike zurück, die eigentliche Erfindung des Stereoskops und der Stereokamera erfolgte jedoch im 19. Jahrhundert.

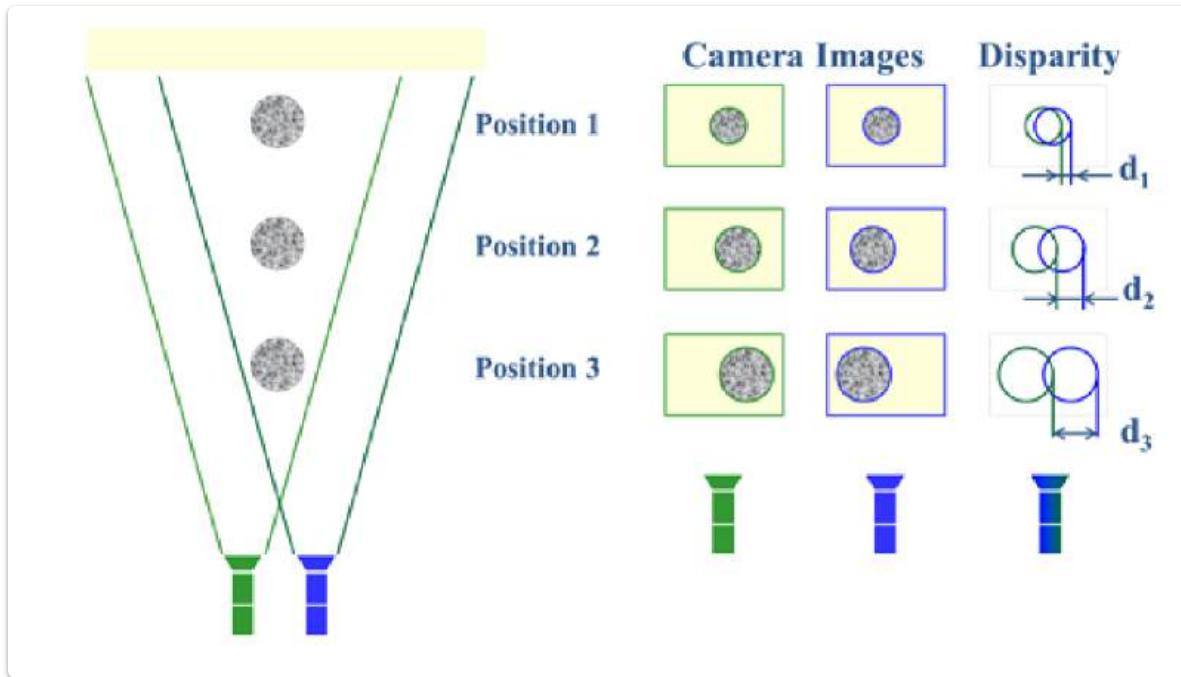
## Disparität

[EVC\\_Skriptum\\_CV, p.52](#)

Definition:

- Ein einzelner Punkt wird in beiden Bildern des Stereosystems auf unterschiedliche Bildkoordinaten abgebildet.

- Die **horizontale Differenz** zwischen diesen Bildkoordinaten nennt man *Disparität*.



Erläuterung anhand paralleler Kameras (siehe Abbildung 44):

- Eine Kugel wird an drei verschiedenen Positionen (Abständen zum Kamerapaar) aufgenommen.
- In der Überlagerung der beiden Kamerabilder kann die Disparität beobachtet werden.
- Position 1 (weit entfernt):**
  - Unterschied der Position der Kugel im überlagerten Bild ( $d_1$ ) ist klein.
- Position 2 und 3 (näher):**
  - Die Bilder der Kugel werden größer, da sie näher sind.
  - Der Unterschied der Punkte in den überlagerten Bildern ( $d_2, d_3$ ) wird größer.
  - Die Disparität wird größer ( $d_3 > d_2 > d_1$ ).

Zusammenhang zwischen Disparität und Tiefe:

- Je näher das Objekt zur Kamera ist, desto größer ist die Disparität.
- Im Unendlichen ist die Disparität 0.
- Die Disparität ist somit *umgekehrt proportional* zur Tiefe.

## Normalfall (Achsparalleles Stereosystem)

[EVC\\_Skriptum\\_CV, p.52](#)

Definition:

- Zeichnet sich durch zwei Kameras aus, die horizontal verschoben sind.
- Deren Koordinatensysteme sind nicht gegeneinander verdreht.
- Der Abstand zwischen den beiden optischen Zentren wird *Basislinie* ( $B$ ) genannt.

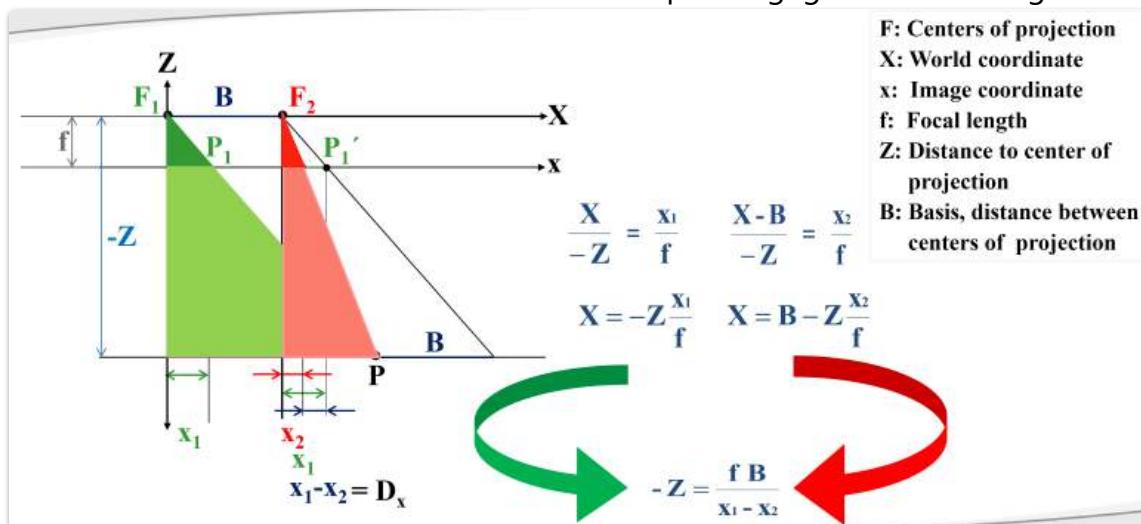
- Die Brennweite ( $f$ ) legt den Abstand der beiden Brennpunkte zu ihren Bildebenen fest und ist für beide Kameras als identisch vorausgesetzt.
- Ein 3D-Punkt  $X$  wird somit über die beiden optischen Zentren in den Abbildungen  $x$  und  $x'$  projiziert.
- Beim achsparallelen Stereosystem sind die Bildzeilen identisch, was für die unterschiedliche Perspektive der Kameras hinsichtlich des 3D-Punktes  $X$  nur zu einer horizontalen Disparität  $D$  in der Abbildung führt.
- Die Disparität wird im Allgemeinen in Bildkoordinaten berechnet, sodass die Einheit Pixel ist:  $D = |x_l - x_r|$ .

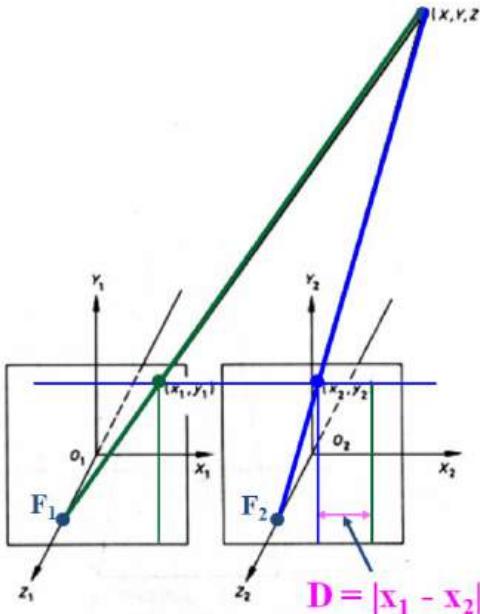
Tiefenberechnung:

- Der Abstand  $Z$  eines Punktes  $X$  von der Kamera lässt sich aus den bekannten konstanten Kameraparametern  $f, B$  sowie der Disparität  $D$  berechnen:

$$Z = \frac{B \cdot f}{D}$$

- Damit stellt die Disparität ein Maß für die Raumtiefe des 3D-Punktes  $X$  dar und verhält sich *umgekehrt proportional* zu ihr.
- Für Punkte im Unendlichen muss daher die Disparität gegen Null konvergieren.



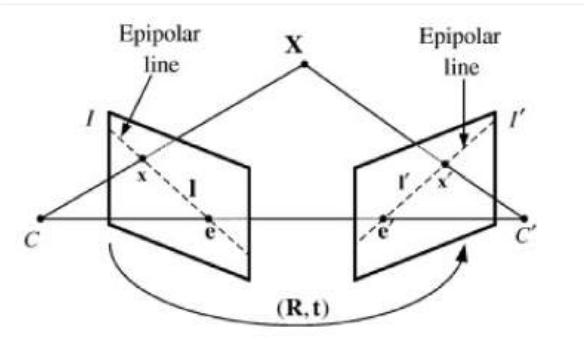


## Epipolargeometrie

[EVC\\_Skriptum\\_CV, p.52](#)

Kameraanordnungen mit zwei Kameras im Stereosystem:

- Können bezüglich ihrer räumlichen Anordnung in zwei grundlegende Klassen eingeteilt werden:
  - Das bereits vorgestellte *achsparallele Stereosystem (Normalfall)*.
  - Die *konvergente Anordnung* (Ausrichtung der optischen Achsen auf einen Konvergenzpunkt).
- Bei der allgemeineren Stereogeometrie, auch *Epipolargeometrie* genannt, sind die beiden Kameras nicht nur zueinander verschoben, sondern auch zueinander gedreht.



Wichtige Begriffe:

- **Epipole (e und e')**: Die Schnittpunkte der Verbindungsgeraden der beiden Kamerazentren (Basislinienebene) mit den jeweiligen Bildebenen. Die Epipole können auch als Projektion des optischen Zentrums der einen Kamera in der Bildebene der anderen Kamera aufgefasst werden.

- **Epipolarebene:** Die Ebene, die durch den 3D-Punkt  $X$  und die beiden Brennpunkte  $C$  und  $C'$  aufgespannt wird.
- **Epipolarlinien ( $l$  und  $l'$ ):** Die Schnittlinien der Epipolarebene mit den beiden Bildebenden. Betrachtet man die beiden Sehstrahlen des 3D-Punktes in den beiden Kameras als Gummiband, so bewegt man diesen Punkt innerhalb der Epipolarebene, wobei diese jedoch immer auf den Epipolarlinien liegen.

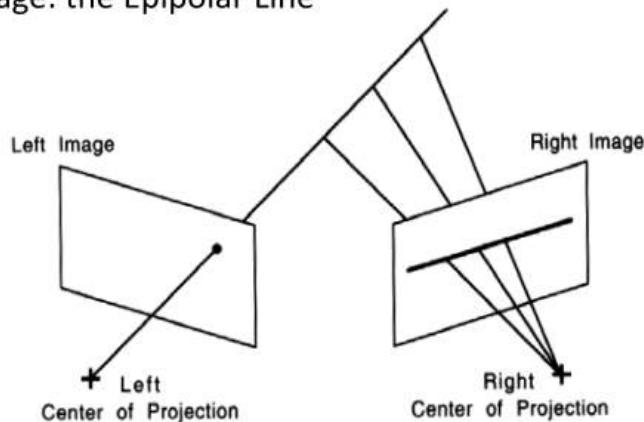
Bedeutung der Epipolarlinien:

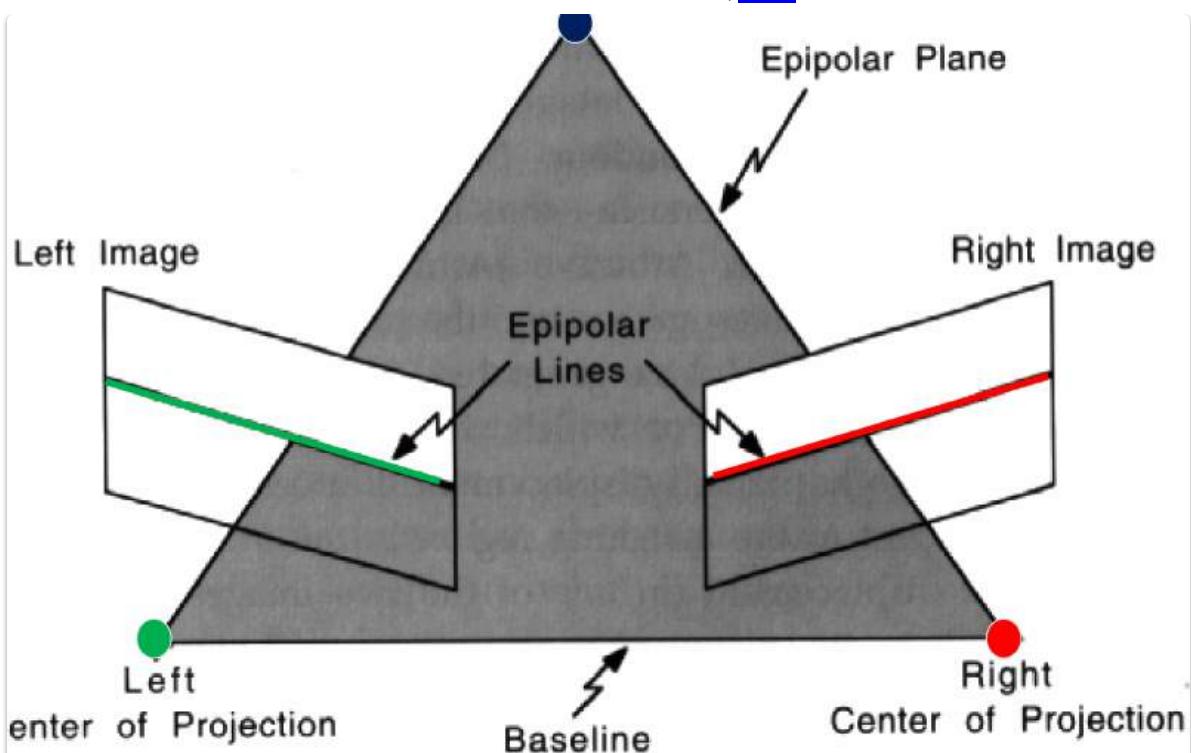
- Lässt man den 3D-Punkt  $X$  entlang seines Sehstrahles (z.B. in Richtung Kamera 1) laufen, so ergibt sich immer die gleiche Abbildung  $x$  in Kamera 1, während in Kamera 2 die Abbildung  $x'$  entlang der Epipolarlinie  $l'$  wandert.



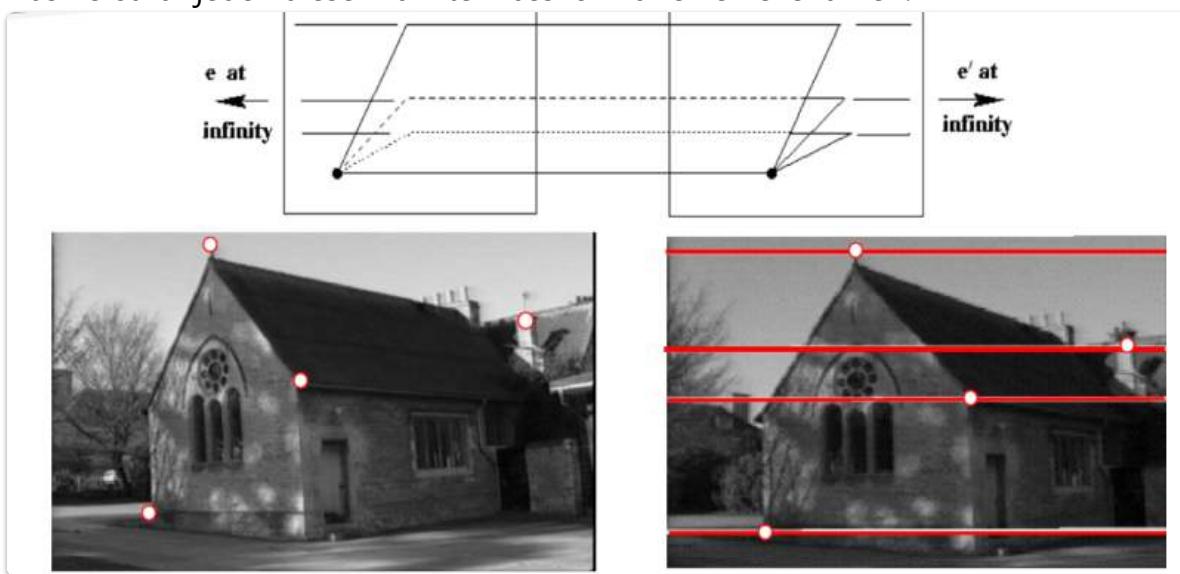
- Der Sehstrahl von jedem 3D-Punkt in einer Kamera liefert somit als Projektion in der anderen Kamera die entsprechende Epipolarlinie.
- Folglich muss auch für jeden Bildpunkt in einer Kamera der korrespondierende Punkt auf einer der Epipolarlinien in der anderen Kamera liegen.

- **Epipolar Constraint:** Each point of the left image can lie only on a specific line in the right image: the Epipolar Line





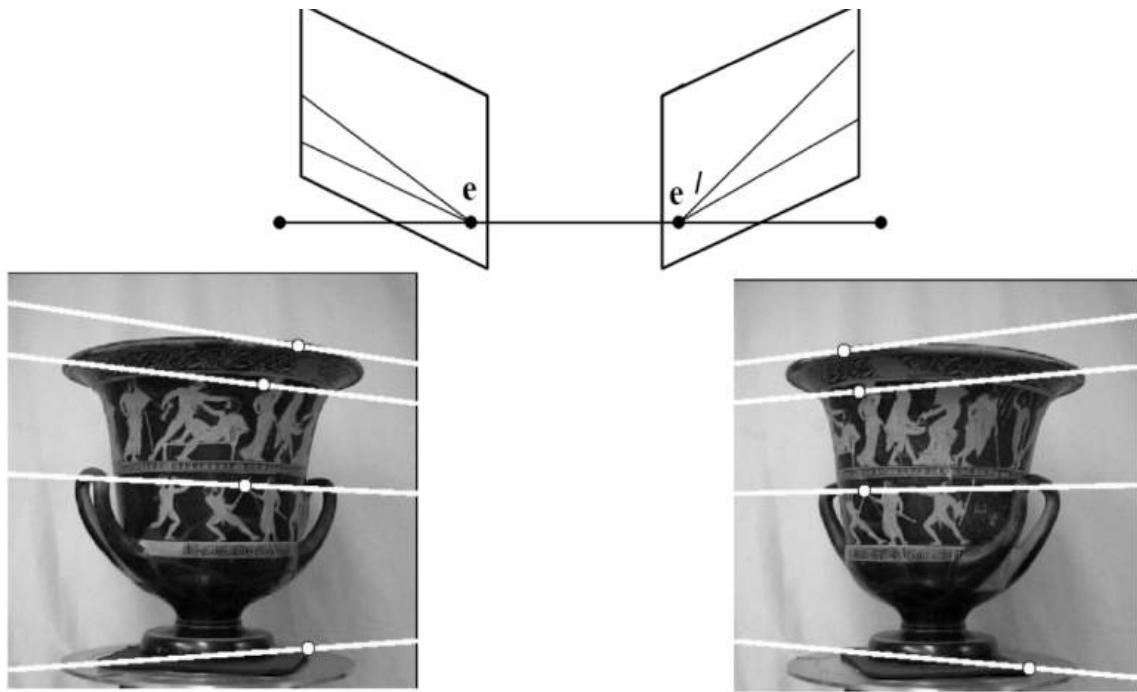
Das heißt für jeden dieser Punkte muss ich nur eine Zeile fahren:



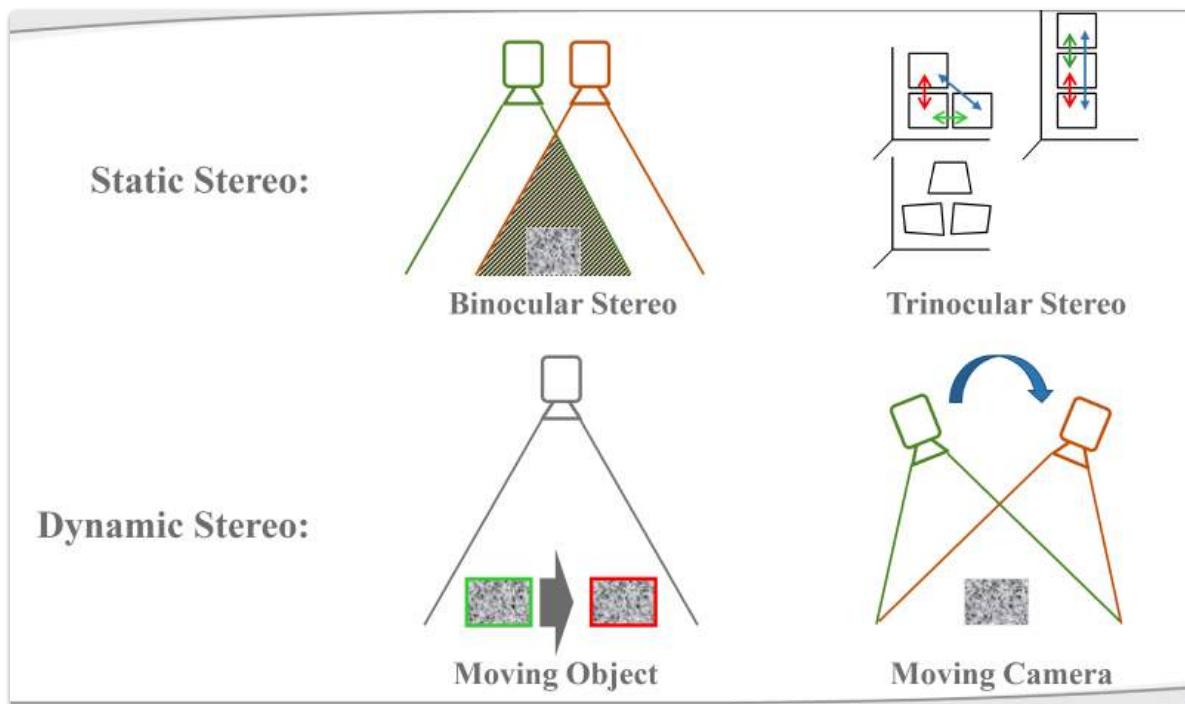
Wichtige Eigenschaften:

- Die Epipolargeometrie hängt **nur** von der relativen Pose (Position und Orientierung) und den internen Parametern der beiden Kameras ab.
  - Dazu gehören die Position der Kamerazentren und der Bildebenen.
- Sie hängt **nicht** von der Szenenstruktur ab (den 3D-Punkten außerhalb der Kameras).

Im Normalfall sind diese Epipole nicht parallel:

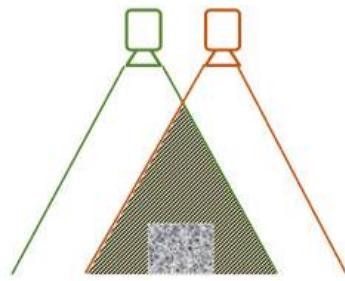


## Camera Setup



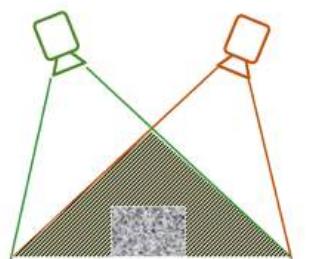
- Baseline

- Distance between cameras (focal points)



- Trade-off

- Small baseline: Matching easier
- Large baseline: Depth precision better

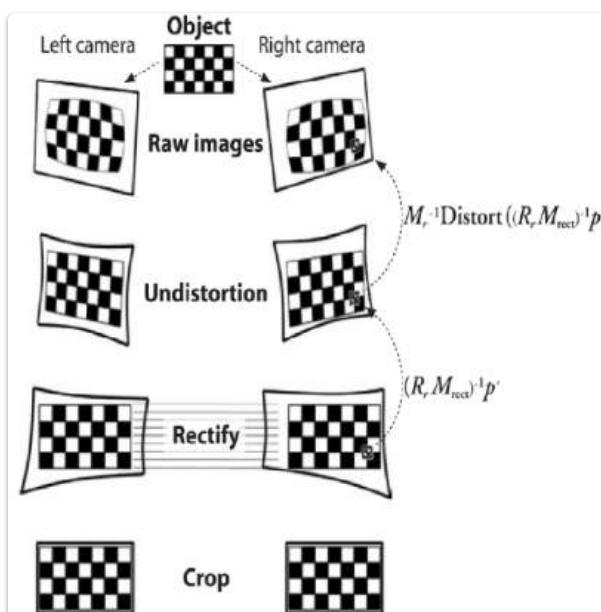


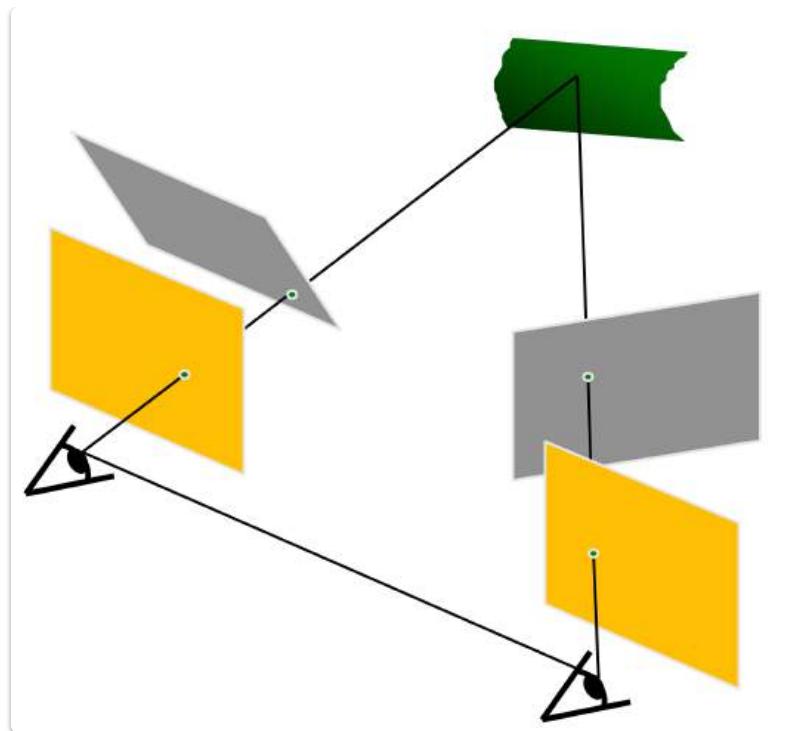
## Image Rectification

Ziel: Erreichen einer vereinfachten Stereo-Geometrie (ähnlich dem Normalfall) durch Bildrektifikation.

Image Re-projection:

- Die Bildebenden werden auf eine gemeinsame Ebene re-projiziert.
- Diese gemeinsame Ebene ist parallel zur Linie zwischen den optischen Zentren (Basislinie).
- Wichtig: Es ist zu beachten, dass hauptsächlich der Brennpunkt der Kamera relevant ist.





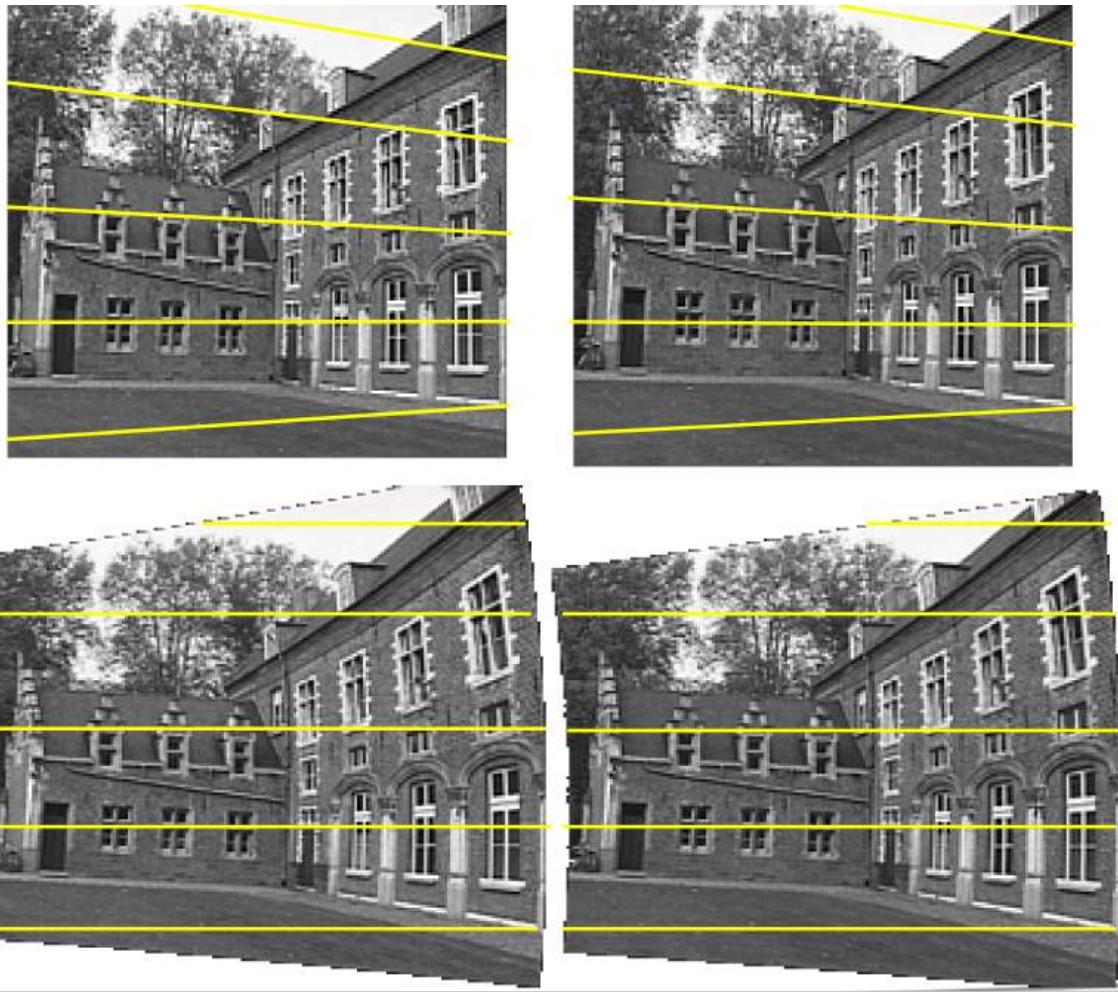
Prozess der Bildrektifikation (illustriert in der Abbildung):

1. **Raw Images:** Die ursprünglichen, möglicherweise verzerrten Bilder der linken und rechten Kamera.
2. **Undistortion:** Korrektur der Linsenverzerrung in den Rohbildern.
3. **Rectify:** Transformation der undistorrierten Bilder, sodass korrespondierende Punkte in den beiden Bildern auf der gleichen horizontalen Zeile liegen. Dies entspricht der Geometrie mit parallelen Bildebenen.
4. **Crop (optional):** Beschneidung der rektifizierten Bilder, um Bereiche ohne gültige Informationen zu entfernen.

Vorteil der Bildrektifikation:

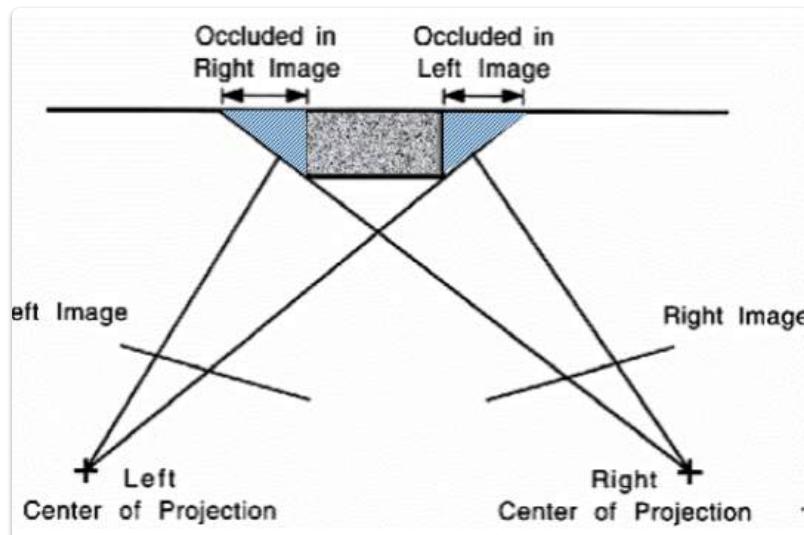
- Vereinfacht die Suche nach korrespondierenden Punkten erheblich, da diese nun auf horizontalen Epipolarlinien liegen (die zu horizontalen Zeilen im Bild werden).
- Ermöglicht die direkte Anwendung von Algorithmen, die für den Normalfall der Stereo-Geometrie entwickelt wurden.

**Beispiel:**



## Okklusionen

- **Ansichtsabhängig (View dependent):** Okklusionen treten auf, weil verschiedene Kameras unterschiedliche Perspektiven auf die Szene haben.
- **Verdeckte Punkte können nicht berechnet werden:** Bereiche, die in einem Bild verdeckt sind, liefern keine direkten Korrespondenzen im anderen Bild und somit keine Tiefeninformationen durch Stereo-Matching.



Erläuterung anhand der Abbildung:

- Ein Objekt (grauer Kasten) verdeckt einen bestimmten Bereich der Szene für die rechte Kamera (als "Occluded in Right Image" markiert).
- Gleichzeitig verdeckt das Objekt einen anderen Bereich der Szene für die linke Kamera (als "Occluded in Left Image" markiert).
- Diese verdeckten Bereiche können in den jeweiligen Bildern nicht mit korrespondierenden Punkten im anderen Bild abgeglichen werden.

Folge von Okklusionen:

- Führt zu fehlenden Tiefeninformationen in den Bereichen, die in mindestens einer der Kameras verdeckt sind.
- Die Größe und Position der okkludierten Bereiche hängen von der relativen Position und Orientierung der Kameras sowie der Geometrie der Szene ab.

## Testähnliches Beispiel

- Eine Szene wird mit einem Stereo-Setup aufgenommen. Der Abstand der beiden Kameras mit einer fokalen Länge von **450 Pixeln** beträgt **8cm**. Für einen Bildpunkt wird eine Disparität von **10 Pixeln** festgestellt. Wie weit ist der zugehörige Szenenpunkt entfernt?

$$\begin{aligned} \bullet & f = 450 \\ \bullet & B = 8\text{cm} \quad Z = \frac{f * B}{x_1 - x_2} \quad Z = \frac{450 * 8\text{cm}}{10} = 360\text{cm} \\ \bullet & D = 10 \end{aligned}$$

- Welche Disparität hat ein Szenenpunkt, der doppelt so weit entfernt ist?

$$D = \frac{f * B}{Z} \quad D = \frac{450 * 8\text{cm}}{720\text{cm}} = \frac{45}{9} = 5$$

## Korrespondenzproblem

[EVC\\_Skriptum\\_CV, p.53](#)

Definition:

- Das Korrespondenzproblem stellt das zentrale Problem der Stereo Vision dar.
- Es bezeichnet die Aufgabe, für jeden Bildpunkt im linken Bild jenen Punkt im rechten Bild zu finden, der denselben Objektpunkt abbildet.
- Entsprechende Suchverfahren werden als *Korrespondenzanalyse* oder auch als *Stereo Matching* bezeichnet.

Vereinfachung durch Epipolarkorrektur (Bildrektifikation):

- Aufgabe der Epipolarkorrektur ist es, Stereobildpaare anhand der Epipolargeometrie so zu transformieren, dass zusammengehörige Bildpunkte auf derselben horizontalen Linie liegen.

- Statt in zwei Dimensionen muss ein korrespondierender Punkt hier nur mehr entlang einer einzigen Scanline gesucht werden.
- Unter dieser Voraussetzung wird das Korrespondenzproblem wesentlich vereinfacht und die Korrespondenzanalyse somit beschleunigt.
- Gängige Stereo Matching Verfahren gehen in der Regel davon aus, dass die Bildpaare in *rektifizierter* Form vorliegen.

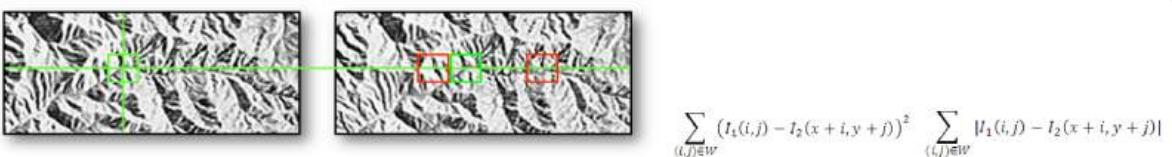
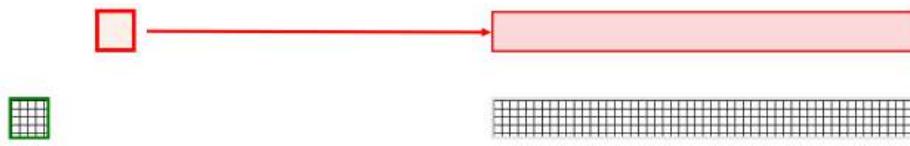
## Regionenbasiertes Matching (Area-Based Matching - ABM)

EVC\_Skriptum\_CV, p.53

Grundprinzip:

- Vergleicht kleine Ausschnitte (Beobachtungsfenster) zwischen den *rektifizierten* Bildern.
- Für jede Position eines Beobachtungsfensters im linken Bild wird das entsprechende Beobachtungsfenster im rechten Bild entlang der Epipolarlinie (jetzt eine horizontale Scanline) bewegt.
- Für jede Position wird geprüft, wie gut die Grauwerte mit den zu vergleichenden Grauwerten im linken Bild übereinstimmen.
- Dies geschieht durch eine Ähnlichkeitsmessung.

- The observation window in the left image is fixed
- For each position in the right image, the correlation function is calculated
- The window will be "slid" from left to right across the image
- Is performed for each position in the left image



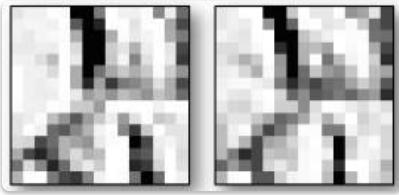
### Einfluss der Fenstergröße:

- **Zu kleine Fenster:** Beinhaltet zu wenig Information für eine korrekte Korrespondenzzuordnung, was zu erhöhten Fehlzuordnungen führen kann.
- **Zu große Fenster:** Erhöht die Rechenzeit.

### Gängige Ähnlichkeitsmaße:

- Summe der absoluten Differenzen (Sum of Absolute Differences - SAD)

- Summe der quadrierten Differenzen (Sum of Squared Differences - SSD)
- Normalisierte Kreuzkorrelation (Normalized Cross Correlation - NCC)



Formeln für Ähnlichkeitsmaße:

- **SSD (Sum of Squared Differences):**

$$SSD(\Delta m, \Delta n) = \sum_{i,j \in R} [I_1(i, j) - I_2(i - \Delta m, j - \Delta n)]^2$$

- $I_1(i, j)$ : Pixelintensität am Koordinaten  $(i, j)$  im ersten Fenster.
- $I_2(i - \Delta m, j - \Delta n)$ : Pixelintensität am verschobenen Koordinaten im zweiten Fenster.
- $R$ : Bereich des Fensters.
- $\Delta m, \Delta n$ : Verschiebung des zweiten Fensters relativ zum ersten.

- **CC (Cross-Correlation):**

$$CC(\Delta m, \Delta n) = \sum_{i,j \in R} [I_1(i, j) \cdot I_2(i - \Delta m, j - \Delta n)]$$

- Hier wird die Korrelation (Ähnlichkeit im Muster) direkt berechnet. Höhere Werte deuten auf größere Ähnlichkeit hin.

Anmerkung: Die gezeigte Formel für CC ist die unnormalisierte Kreuzkorrelation. Oft wird eine normalisierte Version (NCC) verwendet, um die Ergebnisse robuster gegenüber Helligkeits- und Kontrastunterschieden zu machen.

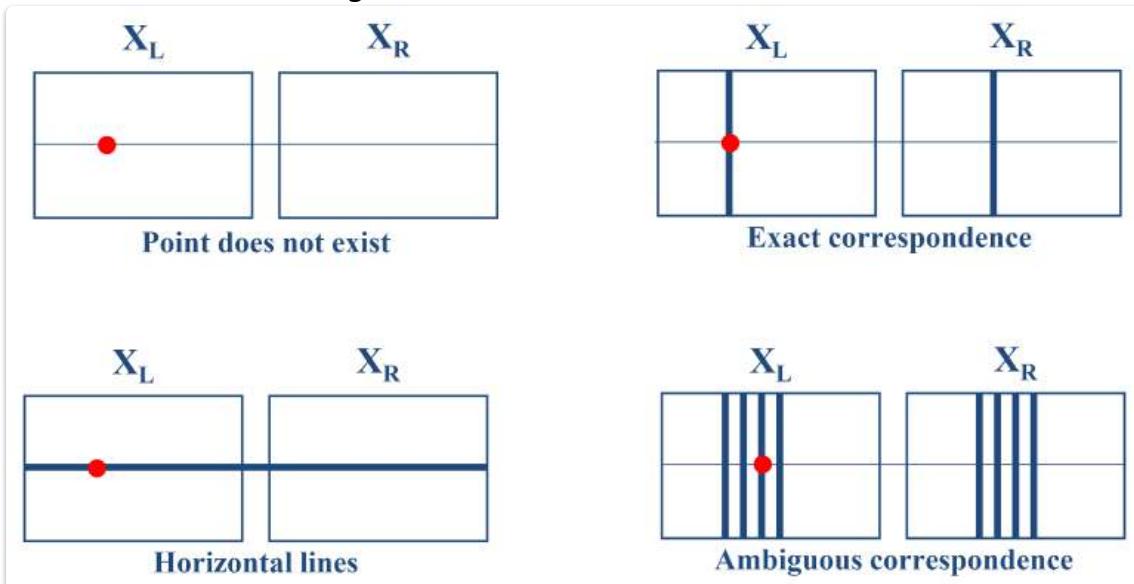
### Funktionsweise der Ähnlichkeitsmessung:

- Prinzipiell wird bei allen diesen Verfahren die Ähnlichkeit durch einen Vergleich von Pixeln innerhalb einer quadratischen Nachbarschaft zwischen dem linken und dem rechten Bild berechnet.
- Wenn das linke und rechte Bild exakt aufeinander passen, erhält man als Resultat ein Maximum (bei NCC) oder Minimum (bei SAD und SSD) in der Ähnlichkeitsfunktion.
- Mit Hilfe der Position des Maximums oder Minimums der Ähnlichkeitsfunktion wird die Position des korrespondierenden Punktes bestimmt (Disparität).

### Probleme:

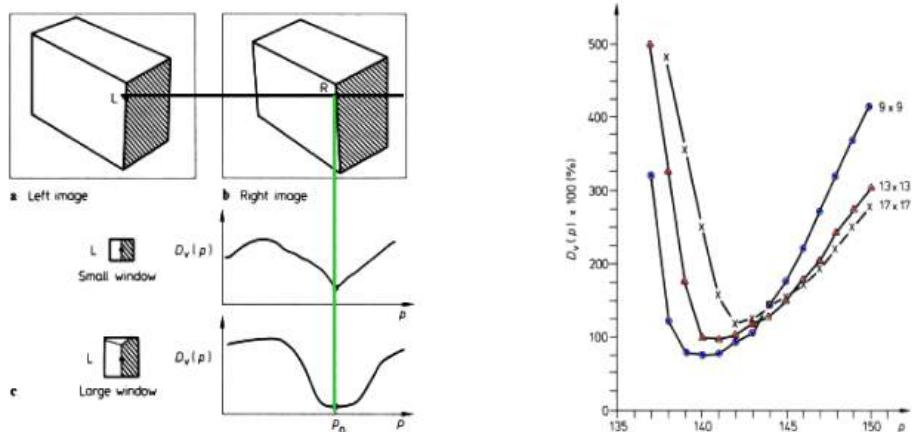
- Es ist möglich, dass kein eindeutiges Minimum oder Maximum gefunden werden kann.
- Dies kann zum Beispiel durch **Verdeckungen** passieren, wenn ein Punkt von einer Kamera aus sichtbar ist und von der anderen nicht.

- Intensitäten in den beiden Bildern müssen nicht zwingend passen, der korrespondierende Punkt ist nicht notwendig.

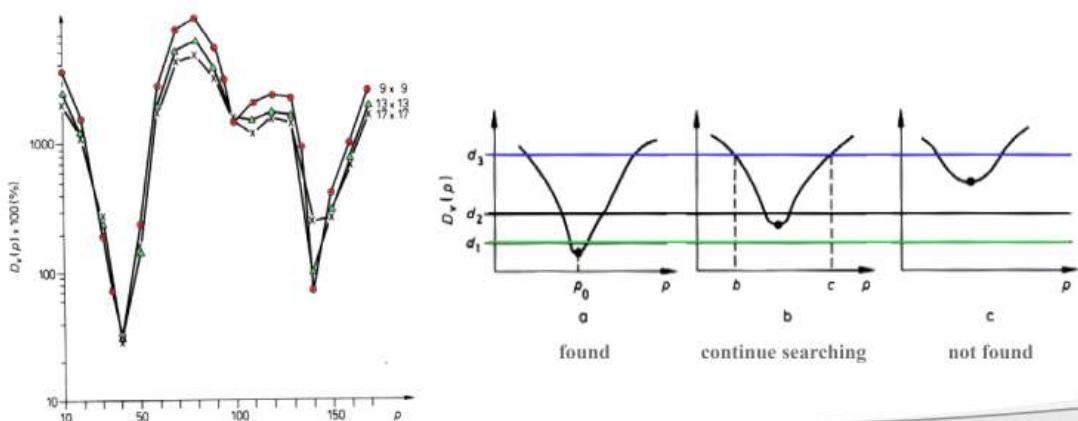


- Correspondence is strongly dependent on window size used

- Different algorithms like adaptive matching, pyramids ....



- Solution of ambiguity with threshold or additional constraints
- Different threshold values



Vorteil:

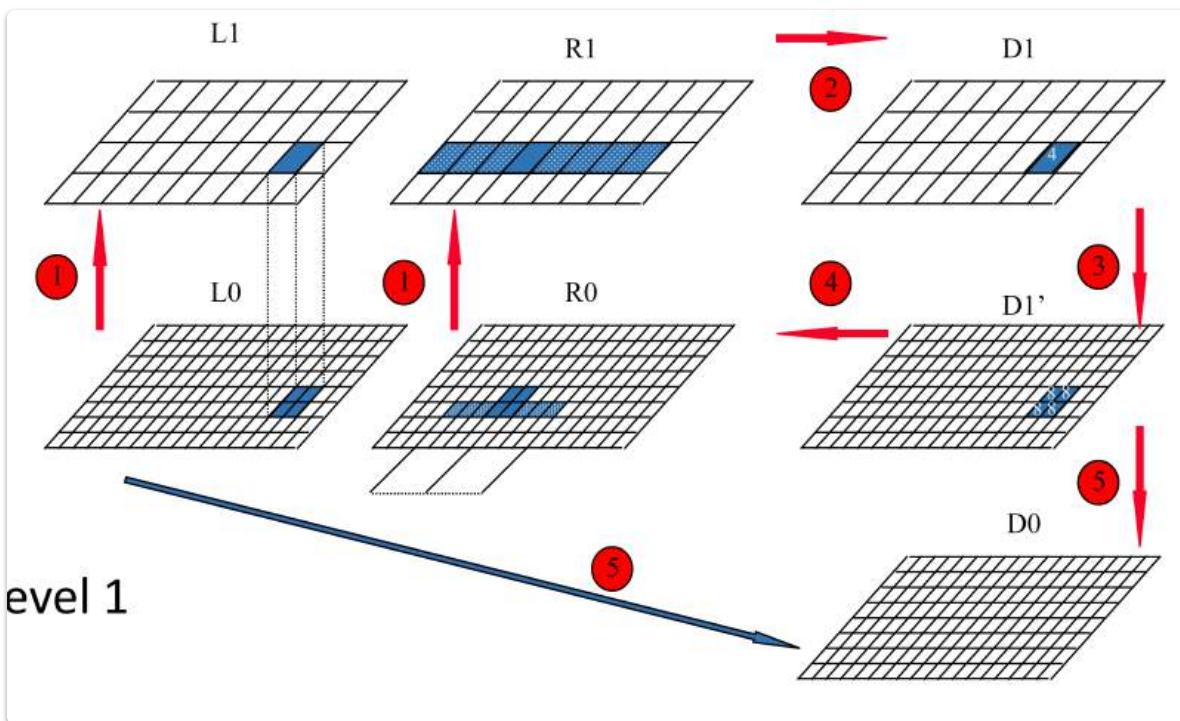
- Der Hauptvorteil des regionenbasierten Matching gegenüber den merkmalbasierten Verfahren ist ein dichteres Tiefenbild.
- Hier werden die Tiefenwerte für alle Pixel direkt ausgerechnet, nicht nur für einige ausgewählte Merkmalspunkte.

### Nachteil:

- Eine höhere Komplexität und ein entsprechend höherer Rechenaufwand.

### Hierarchical Matching

Ansatz: Verwendet Bildpyramiden (z.B. Gaussian Pyramids), um das Korrespondenzproblem effizienter zu lösen.

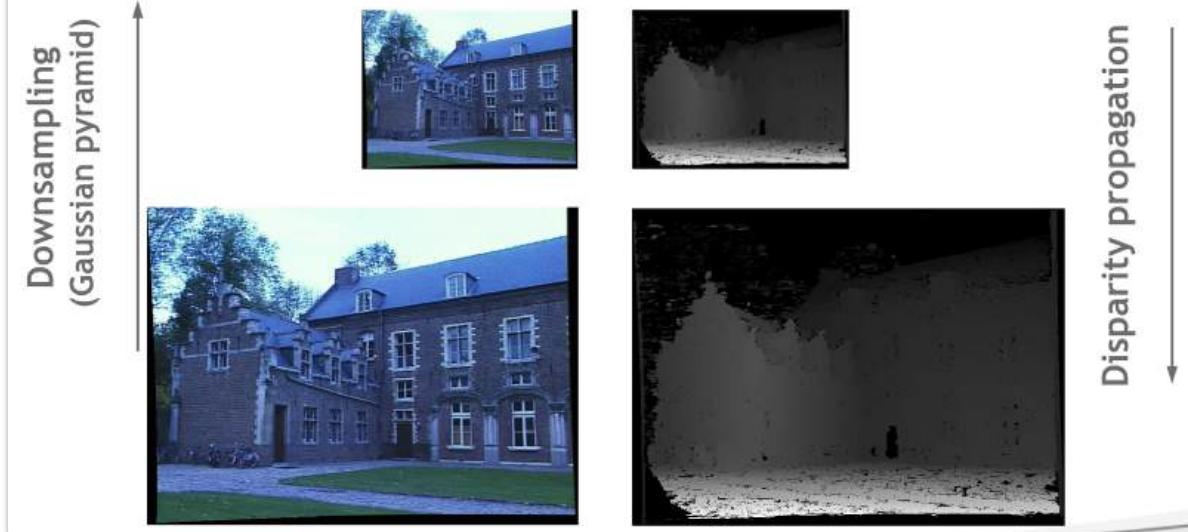


Schritte des hierarchischen Matchings (basierend auf der oberen Abbildung):

- Gaussian Pyramids:** Erstellung von Bildpyramiden für das linke und rechte Bild. Höhere Ebenen sind dabei niedrigere Auflösungsversionen der Originalbilder.
- Compute disparities of level 1:** Berechnung der Disparitäten auf der obersten (niedrigsten Auflösungs-) Ebene der Pyramide. Dies ist recheneffizienter und kann größere Disparitätsbereiche abdecken.
- Project values:** Die auf der höheren Ebene berechneten Disparitäten werden auf die nächstniedrigere Ebene projiziert und dienen dort als initiale Schätzwerte für die Disparitätssuche.
- Compute disparities of lower level:** Die Disparitäten werden auf der niedrigeren Ebene (mit höherer Auflösung) verfeinert, indem um die projizierten Schätzwerte herum gesucht wird. Dieser Prozess wird für alle Ebenen der Pyramide wiederholt, bis zur Originalauflösung.

tion

ity ranges



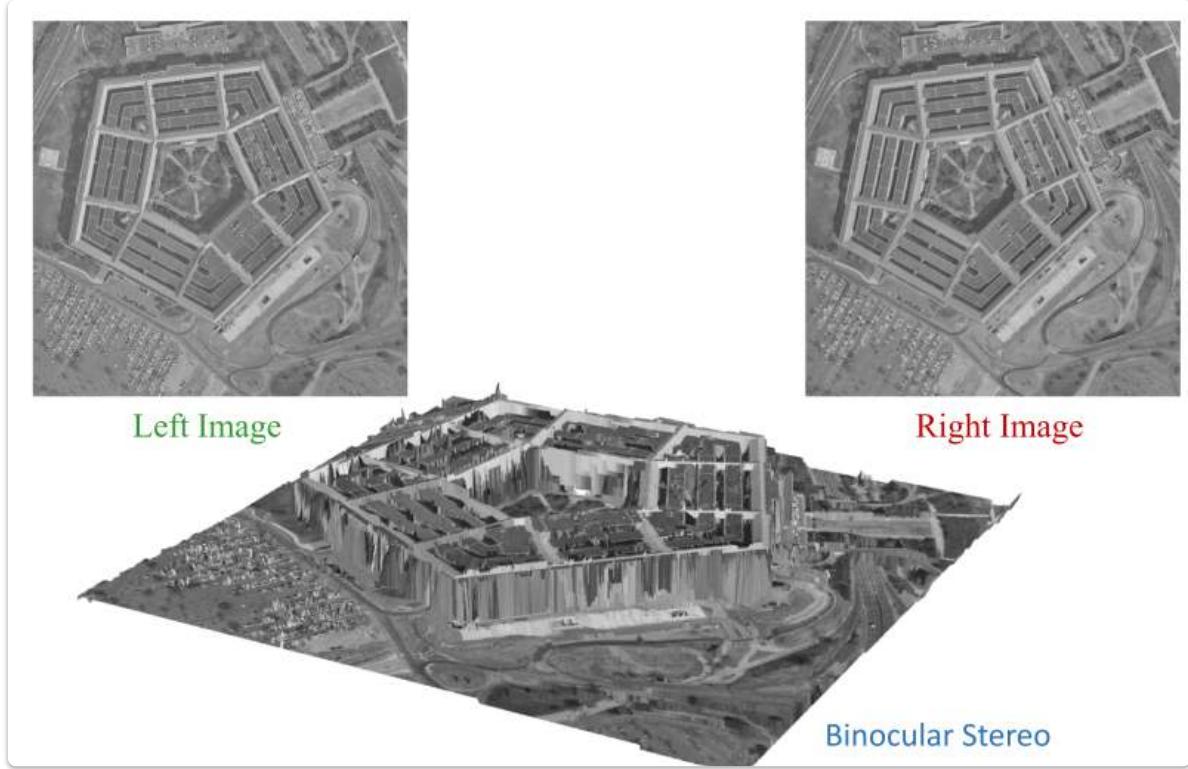
Vorteile des hierarchischen Stereo Matchings (basierend auf der unteren Abbildung):

- **Allows faster computation:** Die Suche nach Korrespondenzen beginnt auf einer niedrigen Auflösung, was die Anzahl der zu vergleichenden Pixel reduziert und somit die Berechnungszeit beschleunigt.
- **Deals with large disparity ranges:** Die grobe Suche auf niedriger Auflösung kann große Disparitätsunterschiede erfassen. Die anschließende Verfeinerung auf höheren Auflösungen ermöglicht eine präzisere Disparitätsschätzung.

Zusammenfassend: Hierarchisches Matching nutzt den Pyramidenansatz, um die Effizienz und den Suchbereich des Stereo Matchings zu verbessern. Durch die schrittweise Verfeinerung der Disparitäten von groben zu feinen Auflösungen können sowohl Rechenzeit reduziert als auch größere Disparitätsbereiche akkurat behandelt werden.

## Beispiele





## Merkmalbasiertes Matching

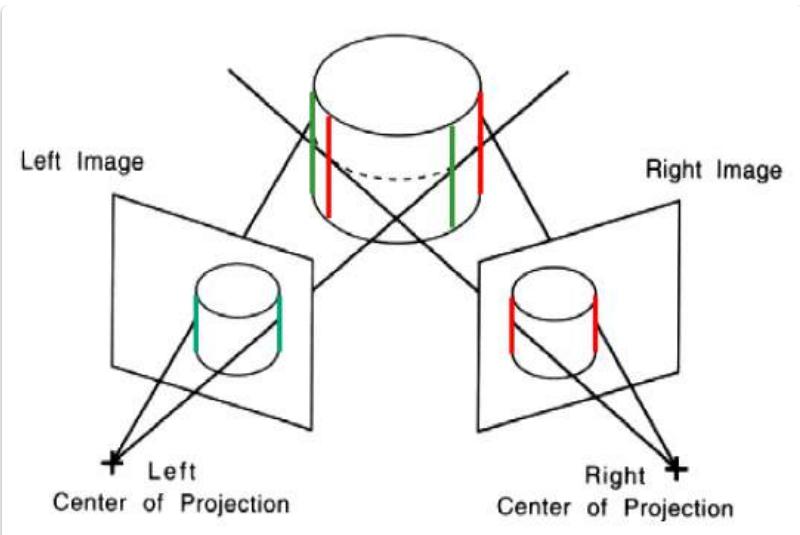
[EVC\\_Skriptum\\_CV, p.53](#)

Problem des ABM (Regionenbasiertes Matching):

- Homogene Bildbereiche enthalten sehr wenig Information.
- Werden aber in die Berechnung miteinbezogen und können zu Fehlern führen.

Ansatz des merkmalbasierten Matching:

- Verwendet einzelne, ausgewählte Pixel (Merkmale), die sich gut zuordnen lassen.
- Merkmale werden aus jedem Bild individuell extrahiert, bevor sie verglichen werden.
- Lokale Merkmale können Ecken, Kanten oder andere *Interest Points* sein.
- Diese Interest Points werden durch lokale Operatoren wie z.B. Moravec oder SIFT extrahiert.



Vorteil des merkmalbasierten Matching:

- Der eigentliche Korrespondenzvergleich kann schneller durchgeführt werden, da bei der Merkmalsextraktion eine wesentliche Datenreduktion stattfindet.

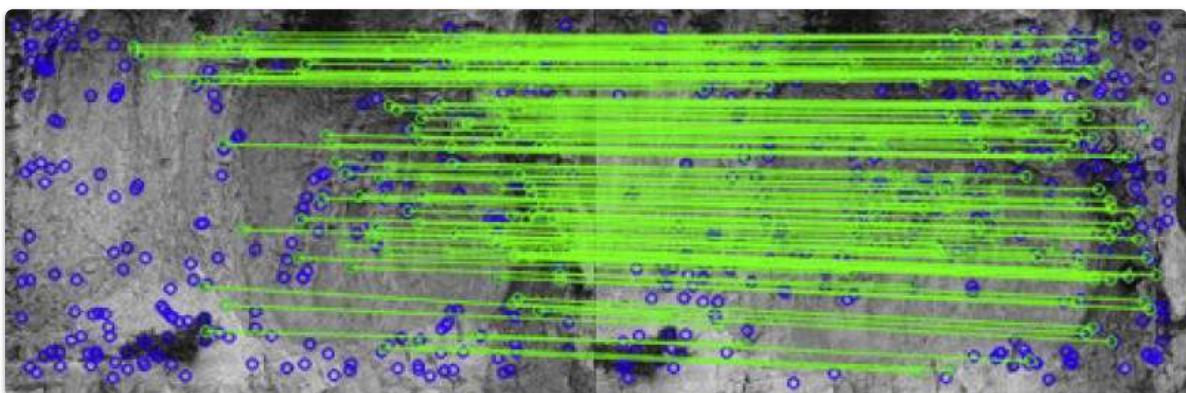
Nachteil des merkmalbasierten Matching:

- Der Hauptnachteil dieser Verfahren liegt aber darin, dass man nur zuverlässige Tiefeninformationen für diese ausgewählten Merkmale erhält (kein dichtes Tiefenbild wie bei ABM).
- Die Bereiche zwischen den Interest Points bleiben zunächst unberücksichtigt und müssen ggf. weiteren Verarbeitungen unterzogen werden (z.B. Interpolation).

### Matching Criteria (Merkmalbasiertes Matching)

Verwendete Merkmale:

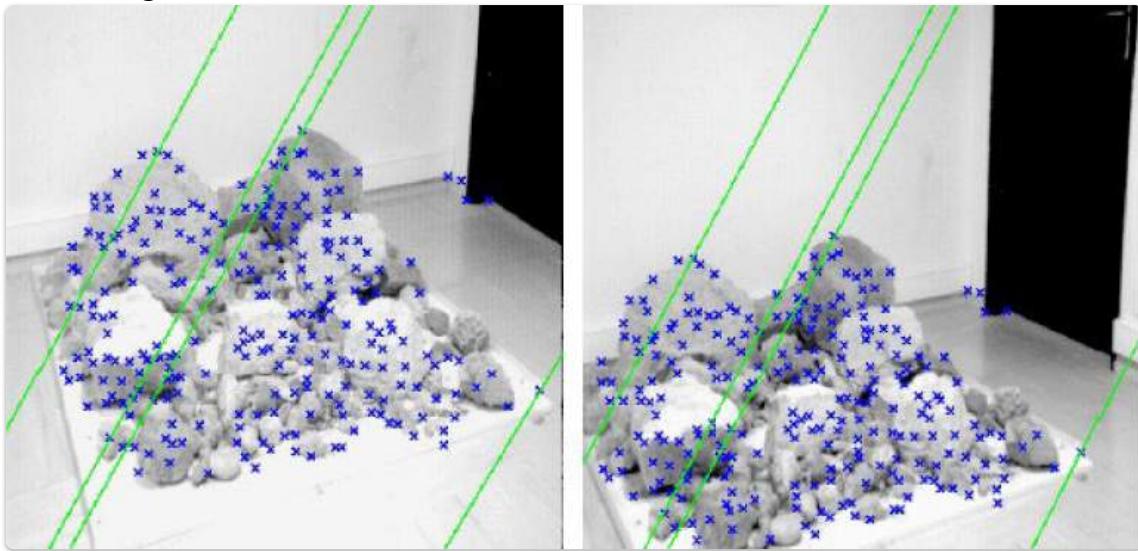
- "Corner"-artige Features (z.B. aus [Zhang, ...])
- Kanten (Edges) [viele Personen...]
- Gradienten [Seitz 89; Scharstein 94]
- Interest Points (z.B. SIFT)



### Feature-based Stereo

Prozess:

- **Match "corner" (interest) points:** Zuerst werden markante Punkte in beiden Bildern detektiert und einander zugeordnet (gematcht).
- **Interpolate complete solution:** Da nur für die Merkmale Tiefeninformationen vorhanden sind, muss die Tiefenkarte für die übrigen Bildbereiche interpoliert werden, um eine vollständige Tiefenkarte zu erhalten.



## Structure-from-Motion (SfM)

[EVC\\_Skriptum\\_CV, p.54](#)

Definition:

- Bezeichnet den Prozess der Gewinnung dreidimensionaler Informationen von Objekten oder einer ganzen Szene durch die Auswertung einer zeitlichen Folge von Bildern.

Zentrales Problem:

- Bestimmung der **Richtung** und des **Ausmaßes** der Kamerabewegung.
- Ansätze zur Lösung:
  - **Motion Field estimation:** Schätzung des Bewegungsfeldes der Punkte im Bild über die Zeit.
  - **Motion Field analysis:** Analyse des geschätzten Bewegungsfeldes zur Bestimmung der Kamerabewegung und der Szenenstruktur.
- Verschiebung des Blickpunkts führt zur scheinbaren Bewegung von Objekten in der Szene.

Unterschied zur Stereo Vision:

- Im Gegensatz zur Stereo Vision ist die Kamerageometrie (die relative Position und Orientierung der Kameras zueinander) zunächst **nicht bekannt**.

Vorgehensweise:

- Anhand der gefundenen Korrespondenzen werden Bewegungsfelder geschätzt.
- Diese Bewegungsfelder können dazu verwendet werden, die Kamerabewegungen zu bestimmen.
- Aus den Kamerabewegungen und den korrespondierenden Punkten kann dann die 3D-Struktur der Szene rekonstruiert werden.

Informationen, die aus der Bewegung gewonnen werden können:

- Information über die Bewegung des Betrachters (der Kamera).
- Tiefeninformationen der Umgebung (vgl. Stereo Vision).
- *Motion Parallax* = *Motion Disparity*: Nahe Objekte scheinen sich bei Bewegung des Betrachters schneller relativ zu entfernten Objekten zu bewegen.



## Bewegungsfeld

[EVC\\_Skriptum\\_CV, p.54](#)

Beobachtung:

- Fixiert ein Beobachter den Horizont, so scheinen sich Mond, Sterne und die gesamte obere Gesichtssphäre zu bewegen.
- Welt und Erdboden scheinen in einem kontinuierlichen Strom vorbeizuziehen.

Relativgeschwindigkeit und Entfernung (Helmholtz, 1950):

- Projiziert man die Umgebung auf eine Abbildungsebene vor dem Betrachter, so ist die relative Geschwindigkeit eines Objekts *umgekehrt proportional* zu seiner Entfernung vom Betrachter.
- Je weiter ein Objekt vom Betrachter entfernt ist, desto geringer ist dessen Bewegung.
- Sterne und der Mond bewegen sich nicht, die Straße direkt neben dem Beobachter bewegt sich sehr schnell.

Richtung des Bewegungsvektors:

- Die Richtung des Bewegungsvektors eines Ortspunktes ist abhängig von der Lage dieses Ortspunktes zum Betrachter.

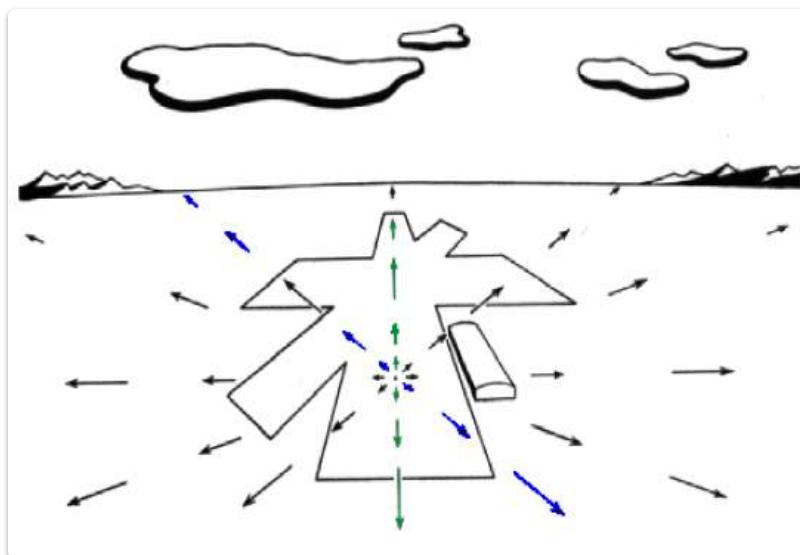
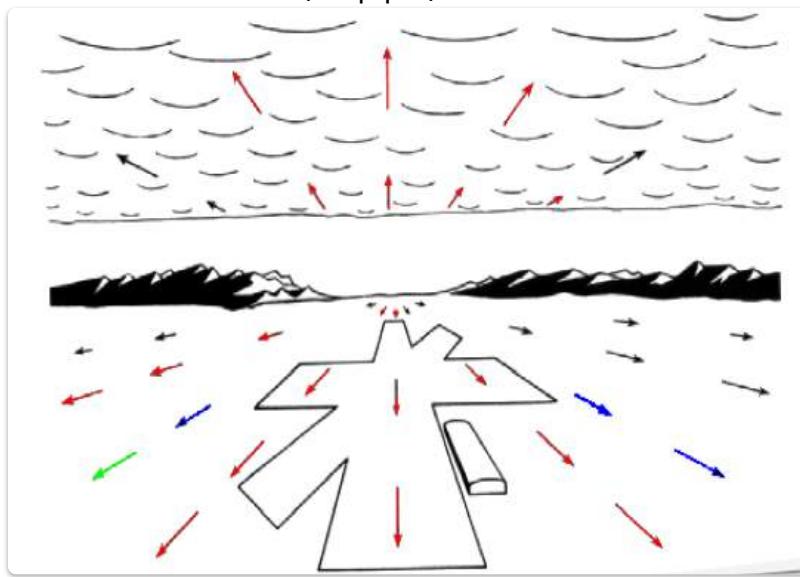
- In einer Umgebung, in der sich die Objekte zueinander nicht bewegen, kann die Eigenbewegung berechnet und eine relative Tiefenkarte der Umgebung erstellt werden.

Bestandteile des Bewegungsfeldes:

- Das Bewegungsfeld besteht aus den einzelnen Bewegungsvektoren aller Punkte im Bild.

Kamera ohne Rotation (Translation):

- Bewegt sich die Kamera ohne zu rotieren, bewirkt dies ein "nach außen" oder "nach innen" Zeigen aller Vektoren zu einem einzigen Punkt.
- Dieser Punkt wird *Focus of Expansion (FoE)* (bei Vorwärtsbewegung) oder *Focus of Contraction (FoC)* (bei Rückwärtsbewegung) genannt.
- Dieser Punkt befindet sich dort, wo sich die Verschiebungsvektor der Kamera mit der Bildebene schneidet (= Epipol).



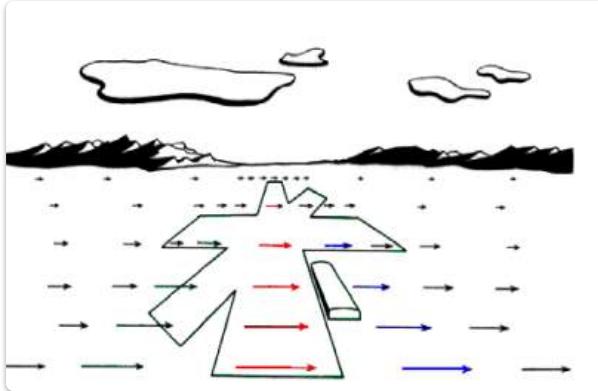
Größe der Bildbewegung eines Szenenpunktes (bei Kamerabewegung):

- Hängt *umgekehrt proportional* von der Entfernung eines Punktes zur Kamera ab.

- Hängt *direkt proportional* vom Sinus des Winkels zwischen der Richtung, in der dieser Szenenpunkt liegt, und der Richtung, in welcher die Kamera verschoben wird, ab.

Berechnung von Kamerabewegung:

- Damit können somit die Richtung der Kamerabewegung (FoE bzw. FoC) und der Betrag der Bewegung berechnet werden.
- Dies führt dann über die Epipolargeometrie zur Stereorekonstruktion (obwohl hier nur eine einzelne bewegte Kamera verwendet wird, nicht zwei gleichzeitig).



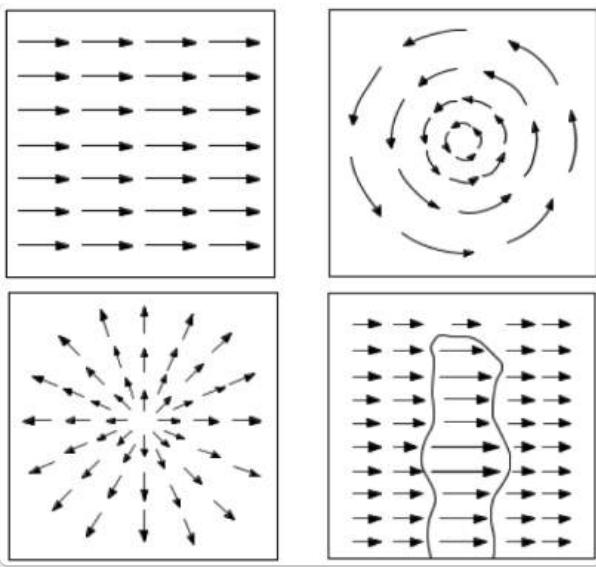
## Bewegungsfeld: Schätzung

Herausforderungen:

- **Sparsely occupied vector field:** Das resultierende Bewegungsfeld kann spärlich sein, d.h., nicht für jeden Pixel ist eine Bewegungsinformation vorhanden (insbesondere in homogenen Bereichen).
- **Same problem as stereo - just the moving direction of the camera is not known:** Das grundlegende Problem der Korrespondenzfindung ist ähnlich wie beim Stereo Matching, jedoch ist die relative Bewegung (die "Basislinie" und Ausrichtung) der Kamera zwischen den Zeitpunkten unbekannt.
- **Epipolar line not known in the beginning:** Da die Kamerabewegung unbekannt ist, sind auch die Epipolarlinien zunächst nicht bekannt, was die Suche nach korrespondierenden Punkten erschwert.

Ansätze zur Korrespondenzfindung zwischen Bildern in einer Sequenz:

- **High temporal sampling = low differences:** Bei einer hohen Bildrate (geringer Zeitabstand zwischen den Bildern) sind die Unterschiede zwischen aufeinanderfolgenden Bildern geringer, was die Korrespondenzsuche erleichtern kann.
- **Either unchanged intensities in both images or unchanged edges in both images:** Annahme, dass entweder die Intensitäten der korrespondierenden Punkte oder deren lokale Struktur (z.B. Kanten) sich zwischen den aufeinanderfolgenden Bildern nicht wesentlich ändern. Diese Annahme kann zur Einschränkung der Suche verwendet werden.



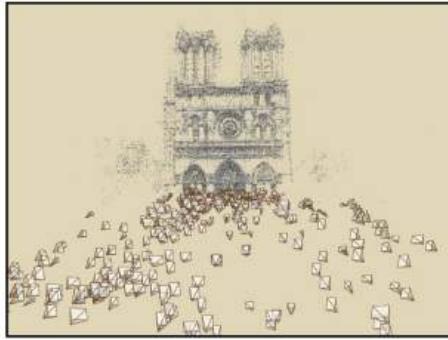
## Multi View Geometry

[EVC\\_Skriptum\\_CV, p.54](#)

### Definition:

- Eine weitere Möglichkeit zur 3D-Rekonstruktion, bei der mehr als zwei beliebige Bilder verwendet werden, um die Relation der Bildpunkte und der Punkte im 3D-Raum zu errechnen.

Structure from Motion (SfM)

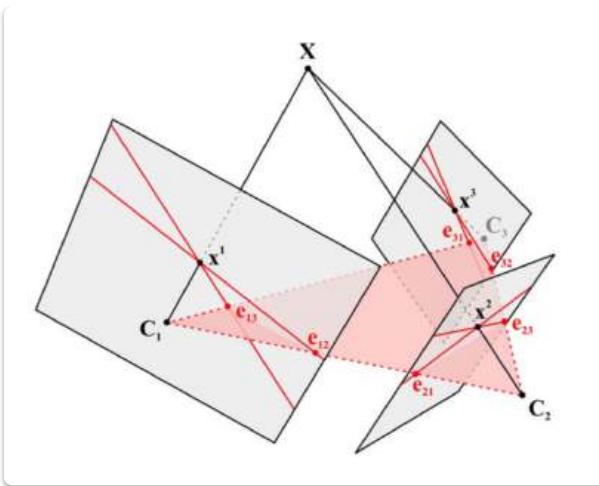


Dense multiview stereo



### Vorteil gegenüber Stereo mit nur zwei Bildern:

- Betrachtet man den Fall mit 3 Bildern: Wenn ein Objektpunkt in zwei Bildern identifiziert wurde, so kann seine geometrische Position im dritten Bild durch den Schnitt der entsprechenden Epipolar-Geraden vorhergesagt werden (siehe Abbildung 45).
- Im Unterschied zum Bildpaar existiert beim Bildtripel jedoch ein eindeutiges Ergebnis.
- Von den 2 oder 3 Basisbildern ausgehend, können bei der Mehrbildgeometrie dann stufenweise mehr Bilder der Szene aus mehreren von verschiedenen Orten aus aufgenommenen Bildern hinzugenommen werden.
- Durch Hinzunahme von mehr Bildern entsteht eine stabilere 3D-Punktwolke.
- Die Durch Bündelausgleich verbessert werden kann.



## Bündelausgleich (Bundle Adjustment):

- Stammt aus der Photogrammetrie.
- Erlaubt die gleichzeitige Bestimmung der internen und externen Kameraparameter sowie der 3D-Struktur der Szene aus mehreren verschiedenen Ansichten.
- Grundlegende Annahmen für den Bündelausgleich sind die Starrheit der Szene zwischen den einzelnen Ansichten und die Erfüllung der Kollinearitätsgleichung (Collinearity Condition).
- Kollinearitätsgleichung: Besagt, dass ein betrachteter 3D-Objektpunkt, sein zugehöriger Bildpunkt und das Projektionszentrum der Kamera auf einer gemeinsamen Gerade liegen müssen.

## Zielsetzung des Bündelausgleichs:

- Gleichzeitige Variation der 3D-Koordinaten der einzelnen Ansichten und der Transformationsparameter der einzelnen Ansichten.
- Ziel ist, eine möglichst gute Übereinstimmung zwischen erwarteten und gemessenen Bildpunkten zu erreichen.

## Zentralprojektion:

- Im Falle einer Zentralprojektion erzeugt jeder Szenenpunkt einen Strahl durch das Projektionszentrum.
- Für die Menge aller Punkte entsteht ein Strahlenbündel, welches im Projektionszentrum geschnürt wird.

## Structure from Motion

### Grundlagen:

- Gegeben viele korrespondierende Punkte über mehrere Bilder hinweg,  $\{(u_{ij}, v_{ij})\}$ .
- Ziel ist die simultane Berechnung der 3D-Positionen der Punkte  $\mathbf{x}_i$  und der Kamera- (oder Bewegungs-) Parameter  $(K, R_j, \mathbf{t}_j)$ .
  - $K$ : Intrinsische Kameraparameter (Kalibrierungsmatrix).

- $R_j$ : Rotationsmatrix der Kamera im  $j$ -ten Bild.
- $\mathbf{t}_j$ : Translationsvektor der Kamera im  $j$ -ten Bild.
- Die Bildkoordinaten  $(u_{ij}, v_{ij})$  sind eine Funktion der Kameraparameter und der 3D-Punktpositionen:

$$u_{ij} = f(K, R_j, \mathbf{t}_j, \mathbf{x}_i)$$

$$v_{ij} = g(K, R_j, \mathbf{t}_j, \mathbf{x}_i)$$

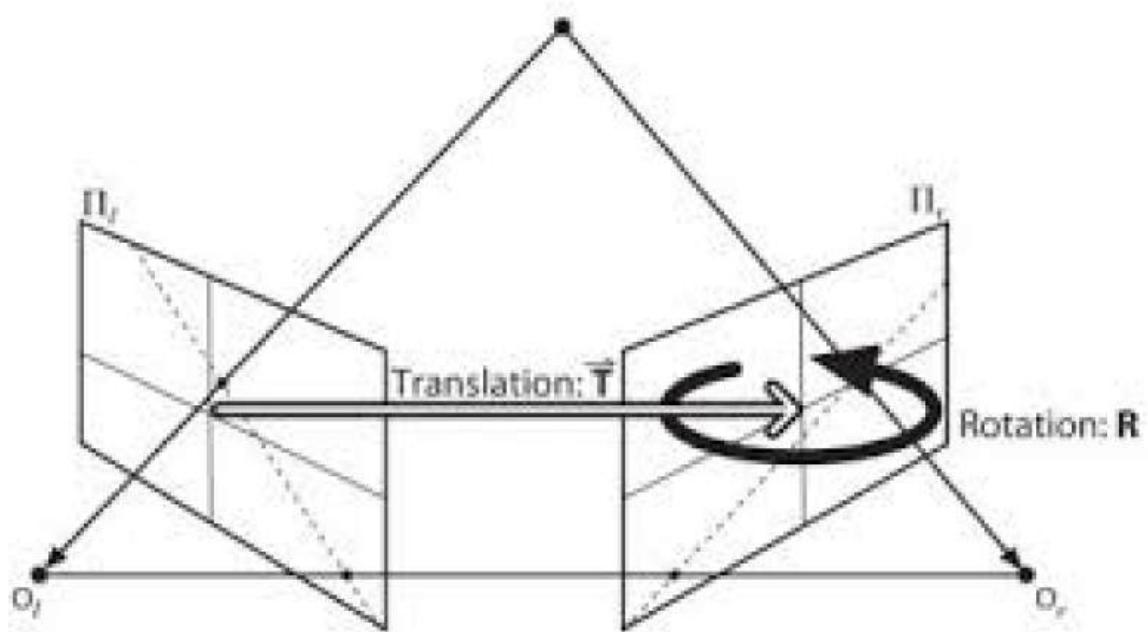
Hauptvarianten:

- **Kalibriert:** Die intrinsischen Kameraparameter  $K$  sind bekannt.
- **Unkalibriert:** Die intrinsischen Kameraparameter  $K$  sind unbekannt (manchmal assoziiert mit euklidischen und projektiven Rekonstruktionen).

## Automatische Berechnung von F (Fundamentale Matrix - typisch für unkalibrierte SfM)

Schritte:

1. **Interest points:** Detektion von markanten Punkten in den Bildern.
2. **Putative correspondences:** Erstellung initialer, potenzieller Korrespondenzen zwischen den Bildern (können fehlerhaft sein).
3. **RANSAC (RANdom SAmple Consensus):** Robustes Schätzverfahren zur Entfernung von Ausreißern (falschen Korrespondenzen) und zur Schätzung der Fundamentalen Matrix  $F$ .
4. **Non-linear re-estimation of F:** Nicht-lineare Optimierung zur Verfeinerung der Schätzung der Fundamentalen Matrix  $F$ .
5. **Guided matching:** Verwendung der geschätzten Epipolargeometrie (aus  $F$ ) zur Verbesserung der Korrespondenzsuche.
6. **Repeat (4.) and (5.) until stable:** Iterativer Prozess der Verfeinerung der Fundamentalen Matrix und der Korrespondenzen, bis eine stabile Lösung erreicht ist.



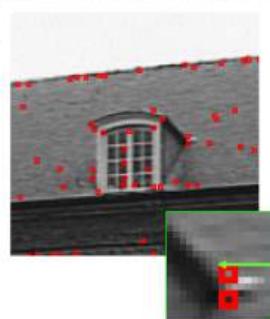
## Beispiel



Select strongest features (e.g. 1000/image)

Evaluate SSD and SAD for all features with similar coordinates

e.g.  $(x', y') \in [x - \frac{w}{10}, x + \frac{w}{10}] \times [y - \frac{h}{10}, y + \frac{h}{10}]$



Keep mutual best matches  
Still many wrong matches!



## RANSAC



Step 1. Extract features

Step 2. Compute a set of potential matches

Step 3. do

Step 3.1 select minimal sample (i.e. 7 matches)

Step 3.2 compute solution(s) for F

Step 3.3 determine inliers

(verify hypothesis)

}

(generate hypothesis)

until  $\Gamma(\#inliers, \#samples) < 95\%$

- Step 4. Compute F based on all inliers
- Step 5. Look for additional matches
- Step 6. Refine F based on all correct matches

$$\Gamma = 1 - \left(1 - \left(\frac{\#inliers}{\#matches}\right)^7\right)^{\#samples}$$

#Inliers	90%	80%	70%	60%	50%
#samples	5	13	35	106	382

# 11. Deep Learning for Computer Vision

Mehr dazu siehe auch [14. Machine Learning für 3D Graphics](#)

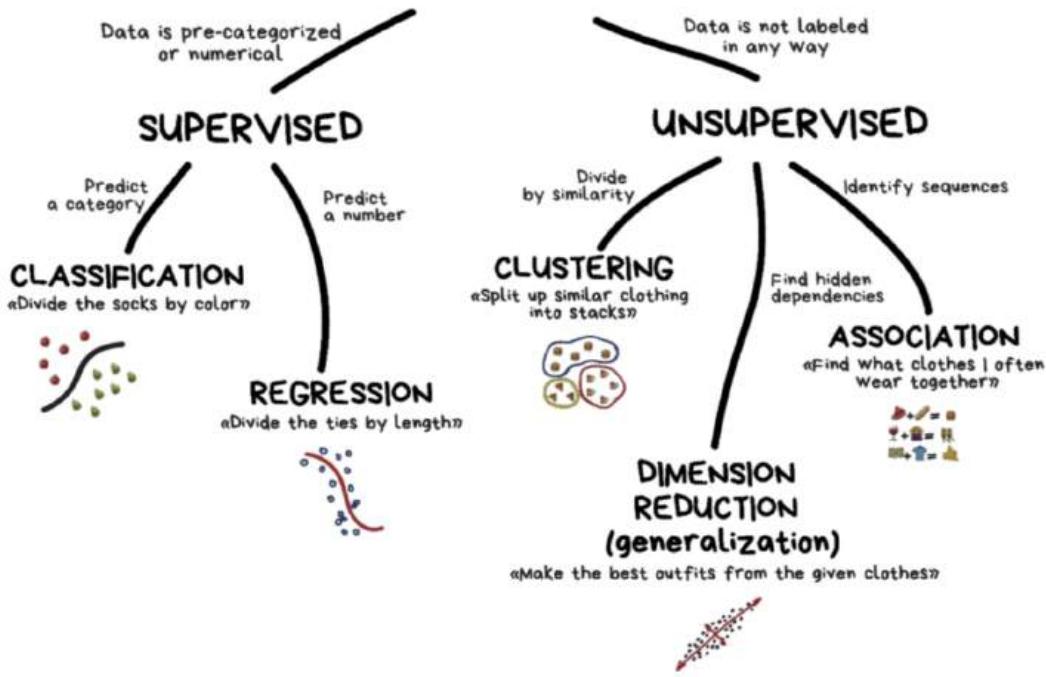
## Maschinelles Lernen

---

[EVC\\_Skriptum\\_CV, p.55](#)

- **Definition:** Maschinelles Lernen (ML) ist ein Teilgebiet der künstlichen Intelligenz (KI).
- **Geschichte:** Etabliertes Forschungsgebiet, geprägt durch rasante technologische Fortschritte seit Jahrzehnten.
- **Begriff der KI:** Erstmals 1955 von Minsky, McCarthy, Newell und Simon definiert: Maschinen, die sich verhalten, als würden sie über eine Art menschliche Intelligenz verfügen.
- **Allgemeines Prinzip des ML:** ML umfasst Lernprozesse, um Zusammenhänge in bestehenden Datensätzen zu erkennen und darauf basierend Vorhersagen zu treffen.
- **Wesentliche Bedeutung:** Modelle ziehen aufgrund von selbstlernenden Algorithmen und existierenden Daten zukunftsrelevante Rückschlüsse, ohne explizit programmiert zu werden.
- **Grundlage:** Lernprozesse dienen als Dateneingabe, die durch eine vordefinierte Reihe an Attributen charakterisiert ist.
- **Kategorien des Maschinellen Lernens:**
  - Überwachtes Lernen (Supervised Learning)
  - Unüberwachtes Lernen (Unsupervised Learning)
  - Bestärkendes Lernen (Reinforcement Learning)

# CLASSICAL MACHINE LEARNING



- **Überwachtes Lernen (Supervised Learning):**

- Daten sind vor-kategorisiert oder numerisch (Data is pre-categorized or numerical).
- Ziel: Vorhersage einer Kategorie (Klassifikation) oder einer Zahl (Regression).
- **Klassifikation:** Teilt die Daten nach Farben (Divide the socks by colors).
  - Beispiel: Vorhersage einer Kategorie.
- **Regression:** Passt eine Linie durch Datenpunkte (Predict a number).
  - Beispiel: Vorhersage eines Preises.

- **Unüberwachtes Lernen (Unsupervised Learning):**

- Daten sind in keiner Weise gelabelt (Data is not labeled in any way).
- Ziel: Struktur in den Daten finden.
- **Clustering:** Gruppert ähnliche Datenpunkte (Divide by similarity; spot similar clothing styles together).
  - Ziel: Gruppen ähnlicher Daten finden.
- **Assoziationsanalyse:** Findet häufige Abhängigkeiten (Find hidden dependencies; find what clothes I often wear together).
  - Ziel: Beziehungen zwischen Datenpunkten identifizieren.
- **Dimensionsreduktion (Generalisierung):** Reduziert die Anzahl der Merkmale (Create the best outfits from the given clothes).
  - Ziel: Wichtige Informationen extrahieren und Rauschen reduzieren.

## Überwachtes Lernen (Supervised Learning)

- **Zielvariable (dedizierte Ausgabewerte):** Wenn eine dedizierte AusgabevARIABLE definiert ist, kann durch Supervised Learning ein Modell darauf trainiert werden, diese für bisher

unbekannte Datensätze (Test Set) vorherzusagen.

- **Generelles Ziel:** Maximierung der Genauigkeit dieser Vorhersagen.
- **Anwendung:** Bei Datensätzen mit präzisen Ausgabewerten.
- **Lernprozess:** Algorithmus lernt die Beziehung zwischen Eingabewerten (Attributen) und den zugehörigen Ausgabewerten anhand des Trainingsdatensatzes.
- **Validierung:**
  - Der gesamte Datensatz wird in ein **Training Set** und ein **Test Set** aufgeteilt.
  - Der eigentliche Lernprozess zur Vorhersage der Zielvariable basiert auf dem **Training Set**.
  - Die Evaluation des trainierten Modells erfolgt mithilfe des **Testdatensatzes**.
  - Dadurch kann sichergestellt werden, dass die Bewertungsgrößen (Genauigkeit, Fehlerrate) anhand bisher unbekannter Daten bestimmt werden.

## Unüberwachtes Lernen (Unsupervised Learning)

- **Anwendung:** Vor allem dann, wenn ein Datensatz keine präzisen Ausgabewerte aufweist.
- **Ziel:** Durch Anwendung von Unsupervised Learning basierend auf den Eingabedaten bisher unbekannte Muster und Zusammenhänge abzuleiten.

## Bestärkendes Lernen (Reinforcement Learning)

- **Lernprozess:** Basiert auf Formen der Belohnung und Bestrafung.
- **Ziel:** Nutzen zu maximieren.
- **Hinweis:** Unsupervised und Reinforcement Learning werden im Folgenden nicht weiter behandelt.

## Ziel des überwachten Lernverfahrens

- Ausgabewerte anhand der vorhandenen Eingabewerte (den Attributen) mit möglichst hoher Genauigkeit vorherzusagen.

### Traditional Programming:



### Machine Learning:



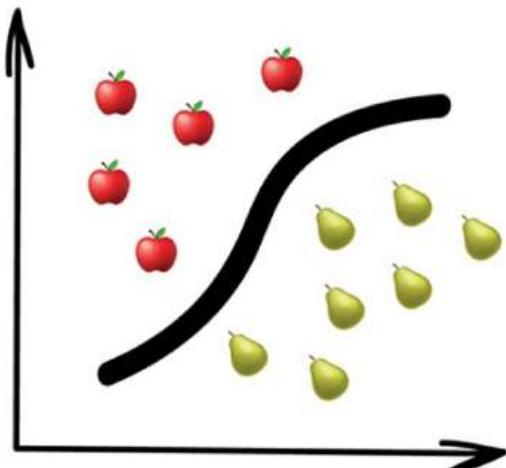
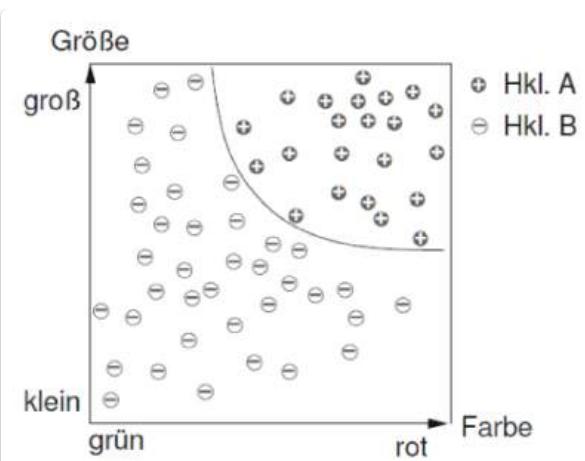
## Klassifikation

- **Häufigste Anwendung in der Computer Vision:** Klassifikation anhand von Merkmalen (Features).
- **Merkmale und Klassenzuordnung:** Im Vorhinein bekannt, die Daten sind "gelabelt".
- **Beispiel (Obstbauer):**
  - Geerntete Äpfel sollen automatisch in die Handelsklassen A und B eingeteilt werden.
  - Sortieranlage misst für jeden Apfel zwei Merkmale (Features): Größe und Farbe.
  - Aufgabe: Entscheidung, zu welcher der beiden Klassen der Apfel gehört.
  - **Typische Klassifikationsaufgabe.**
  -
- **Klassifikatoren (Classifier):** Systeme, welche in der Lage sind, Merkmalsvektoren in eine endliche Anzahl von Klassen einzuteilen.
- **Trainingsdaten-Erstellung:**
  - Zur Einstellung der Maschine werden Äpfel von einem Fachmann händisch verlesen (klassifiziert).
  - Die Messwerte (Größe und Farbe) werden zusammen mit der Klasse in einer Tabelle eingetragen.
  - Größe: Durchmesser in Zentimetern.
  - Farbe: Zahlenwert zwischen 0 (grün) und 1 (rot).
- **Aufgabe beim maschinellen Lernen:** Aus den gesammelten klassifizierten Daten eine Funktion zu generieren, die für neue Äpfel aus den beiden Features (Größe und Farbe) den Wert der Klasse (A oder B) berechnet.

Größe [cm]	8	8	6	3	...
Farbe	0.1	0.3	0.9	0.8	...
Handelsklasse	B	A	A	B	...

## Funktion zur Klassifikation

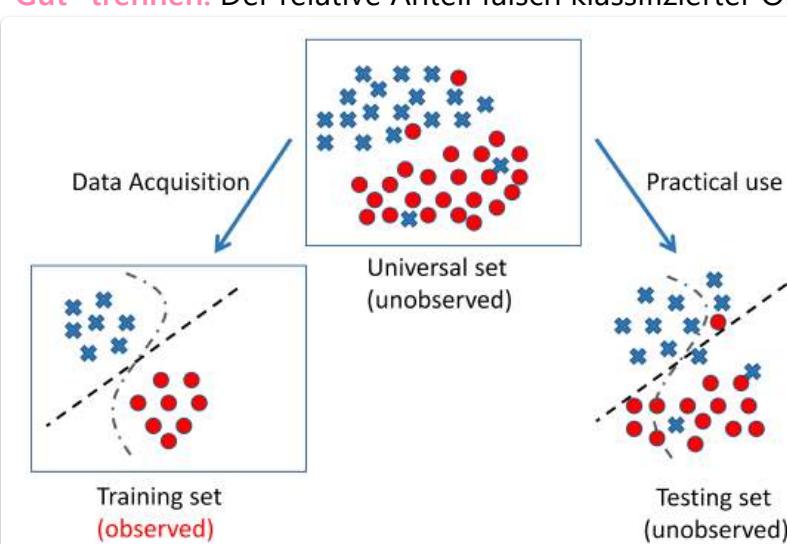
- Die in Abbildung 46 eingezeichnete Trennlinie repräsentiert eine Funktion, die die Klassifizierung vornimmt.
- **Klassenzuweisung:**
  - Äpfel mit Merkmalsvektor links unterhalb der Linie: Klasse B.
  - Alle anderen Äpfel: Klasse A.
- **Einfaches Beispiel:** Die Trennlinie ist in diesem Fall leicht zu finden.



**Classification**

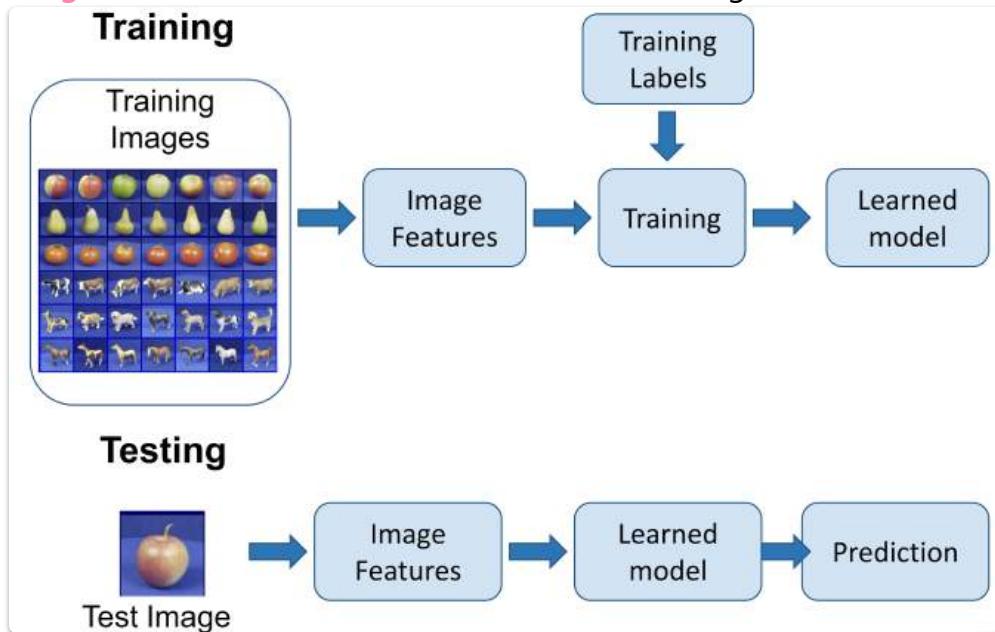
## Herausforderungen bei mehrdimensionalen Daten

- **Schwierigkeit bei vielen Merkmalen:** Die Aufgabe wird deutlich schwieriger und weniger anschaulich, wenn Objekte durch mehr als zwei Merkmale beschrieben werden.
- **Computer Vision:** Nutzt oft viele Merkmale, die aus Bilddaten abgeleitet werden.
- **n Merkmale:** Die Aufgabe besteht darin, im n-dimensionalen Merkmalsraum eine (n-1)-dimensionale Hyperfläche zu finden, welche die Klassen möglichst gut trennt.
- **"Gut" trennen:** Der relative Anteil falsch klassifizierter Objekte soll möglichst klein sein.



## Klassifikator als Abbildung

- Ein Klassifikator bildet einen Merkmalsvektor auf einen Klassenwert ab.
- **Feste Anzahl von Alternativen:** Meist eine kleine Anzahl möglicher Klassen.
- **Zielfunktion:** Diese Abbildung wird auch Zielfunktion genannt.
- **Aufgabe des Lernverfahrens:** Eine solche Abbildung zu lernen.

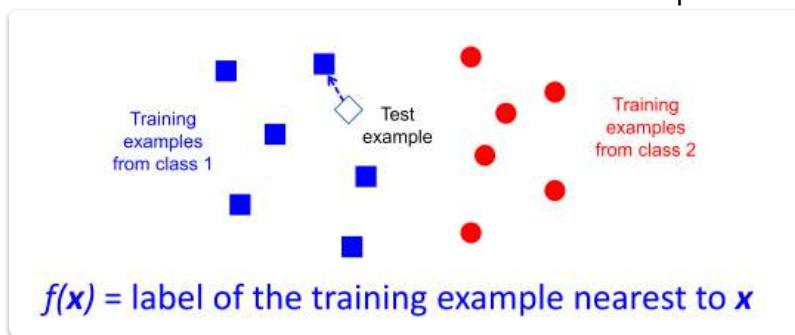


## Klassifikationsalgorithmen im Maschinellen Lernen

- **Vielfalt:** Es existiert eine Vielzahl von Klassifikationsalgorithmen im Bereich des maschinellen Lernens.
- **Überblick über geläufige überwachte Lernverfahren:**

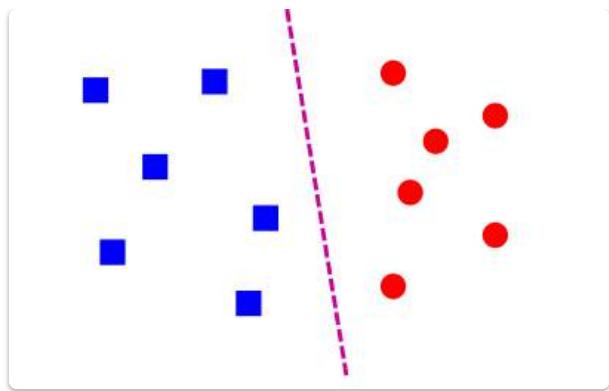
### Nearest Neighbor Algorithmus

- **Grundgedanke:** Ein Datenpunkt wird der Klasse zugeordnet, die durch den Mehrheitsentscheid seiner benachbarten Datenpunkte bestimmt wird.



### Lineare Gleichung

- Finden einer linearen Gleichung die die Klassen trennt



## Entscheidungsbäume (Decision Trees)

- **Klassenzuordnung:** Instanzen werden anhand ihres Pfades durch den Baum (Wurzelknoten und innere Knoten mit Entscheidungsregeln) einer Klasse zugeteilt.

## Random Forest Algorithmus

- **Erweiterung der Entscheidungsbäume.**
- **Klassenzuordnung:** Erfolgt durch den Mehrheitsbeschluss einer Vielzahl von unabhängigen Entscheidungsbäumen.

## Support Vector Machine (SVM)

- **Trennung der Klassen:** Verwendet eine Hyperebene, welche die Klassen mit dem möglichst größten Abstand voneinander trennt (Maximum-Margin-Klassifikator).

## Boosting-Verfahren (z.B. AdaBoost)

- **Ziel:** Steigerung der Gesamtleistung.
- **Funktionsweise:** Kombiniert eine Vielzahl von "schwachen" Klassifikatoren durch einen gewichteten Mehrheitsentscheid zu einem einzigen, stärkeren Klassifikator.

## Künstliche Neuronale Netze (NN)

- **Bedeutung:** Haben sich aufgrund der wachsenden Komplexität und Menge der Datensätze etabliert.
- **Weitere Behandlung:** Werden später noch genauer betrachtet.

## Generalisierung

---

[EVC\\_Skriptum\\_CV, p.56](#)



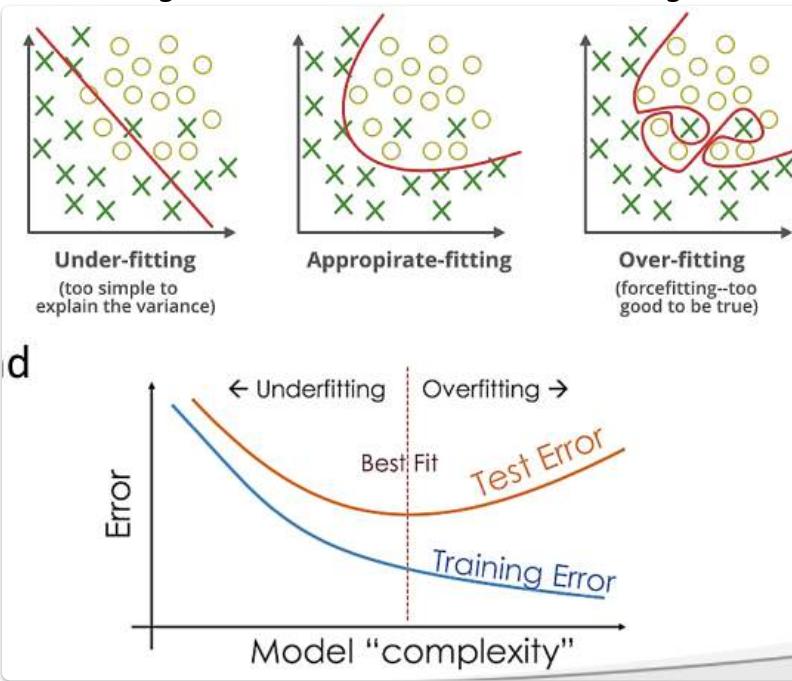
Training set (labels known)



Test set (labels unknown)

- **Überanpassung (Overfitting) - Auswirkungen:**

- Das Modell lernt nicht die generalisierbaren Muster, sondern die spezifischen Details und das Rauschen der Trainingsdaten.
- Führt zu einer hohen Genauigkeit auf den Trainingsdaten, aber einer geringen Genauigkeit auf neuen, ungesiehten Daten (Testdaten).
- Die Leistung des Modells auf realen Anwendungen ist schlecht.

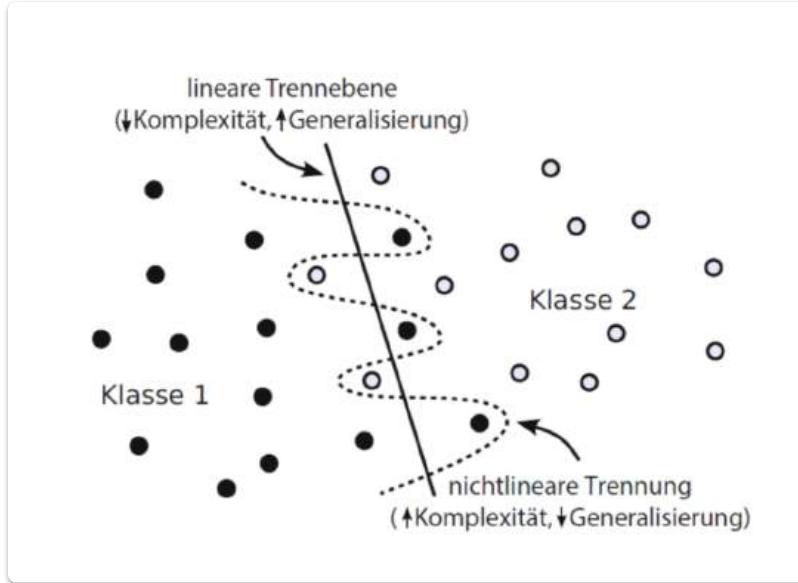


- **Generalisierung - Das Ziel:**

- Ein gutes Modell soll in der Lage sein, Muster zu erkennen, die in den Trainingsdaten vorhanden sind und diese Muster auch auf neue, unbekannte Daten anzuwenden.
- Eine hohe Generalisierungsfähigkeit ist das primäre Ziel beim Entwickeln von Machine-Learning-Modellen.

- **Modellkomplexität - Der Balanceakt:**

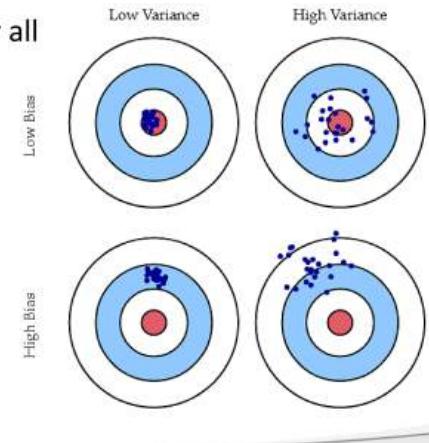
- **Zu einfache Modelle (Underfitting):** Können die zugrundeliegenden Zusammenhänge in den Daten nicht erfassen. Sowohl Trainings- als auch Testdaten werden schlecht vorhergesagt.
- **Zu komplexe Modelle (Overfitting):** Lernen die Trainingsdaten zu gut, inklusive Rauschen. Gute Leistung auf Trainingsdaten, schlechte Leistung auf Testdaten.
- **Die goldene Mitte:** Ein Modell mit der richtigen Komplexität, das die relevanten Muster erfasst, ohne sich zu stark an die Trainingsdaten anzupassen.
- **Praktische Implikationen:**
  - Die Wahl der Modellarchitektur und der Hyperparameter beeinflusst die Modellkomplexität maßgeblich.
  - Techniken wie Kreuzvalidierung (Cross-Validation) werden eingesetzt, um die Generalisierungsfähigkeit eines Modells auf unabhängigen Daten zu schätzen und Überanpassung zu erkennen.
  - Regularisierungsmethoden können verwendet werden, um die Komplexität eines Modells zu reduzieren und die Generalisierung zu verbessern.



## Generalisierung - Bias-Variance Tradeoff

- **Bias (Verzerrung):**
  - Maß für den Fehler aufgrund vereinfachter Annahmen im Modell.
  - Hoher Bias bedeutet, dass das Modell die zugrundeliegenden Muster in den Daten nicht gut erfassst (Underfitting).
  - Einfache Modelle haben tendenziell einen hohen Bias.
  - Ein komplexes Modell hat einen niedrigen Bias, da es flexibler ist und sich besser an die Trainingsdaten anpassen kann.

- **Bias:** how much does the **average model** over all training sets **differ** from the **true model**?
- **Variance:** how much **models** estimated from different training sets **differ from each other**



- **Variance (Varianz):**

- Maß für die Sensitivität der Modellschätzung gegenüber Schwankungen in den Trainingsdaten.
- Hohe Varianz bedeutet, dass das Modell stark auf spezifische Details und Rauschen in den Trainingsdaten reagiert (Overfitting).
- Komplexe Modelle haben tendenziell eine hohe Varianz.
- Ein einfaches Modell hat eine niedrige Varianz, da seine Schätzungen weniger stark durch Änderungen in den Trainingsdaten beeinflusst werden.

- **Bias-Variance Tradeoff:**

- Das Ziel ist es, ein Modell zu finden, das einen niedrigen Bias und eine niedrige Varianz aufweist, um eine gute Generalisierung zu erreichen.
- Es besteht ein Tradeoff, da das Reduzieren des Bias oft zu einer Erhöhung der Varianz führt und umgekehrt.
- **Gesamtfehler = Bias<sup>2</sup> + Varianz + Irreduzierbarer Fehler** (Der irreduzierbare Fehler ist durch die Daten selbst bedingt und kann durch das Modell nicht beeinflusst werden).

- **Komplexität und Bias-Varianz:**

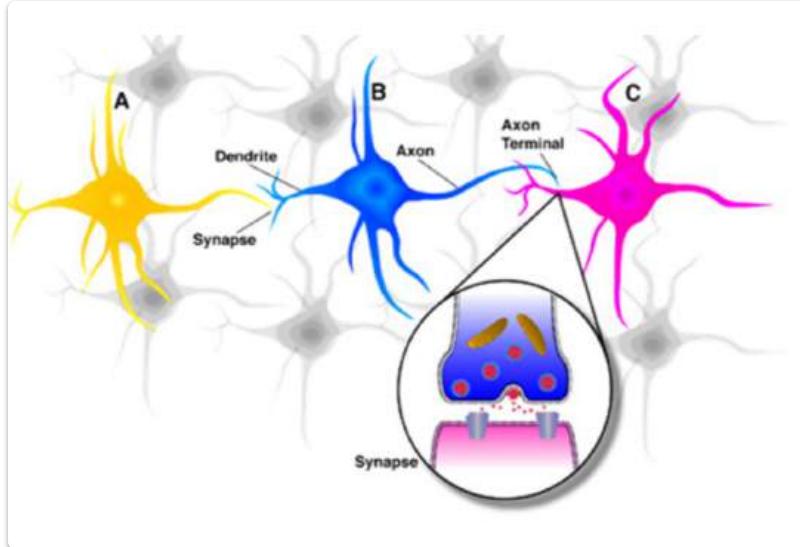
- **Geringe Komplexität:** Hoher Bias, niedrige Varianz (Underfitting).
- **Hohe Komplexität:** Niedriger Bias, hohe Varianz (Overfitting).
- **Optimale Komplexität:** Findet ein Gleichgewicht zwischen Bias und Varianz, was zu einer guten Generalisierung führt.

- **Praktische Konsequenzen:**

- Bei der Modellentwicklung müssen wir versuchen, die optimale Komplexität zu finden.
- Die Leistung des Modells auf unabhängigen Validierungsdaten ist ein guter Indikator für das Vorliegen von Bias oder Varianz Problemen.
- Techniken wie Regularisierung können helfen, die Varianz zu reduzieren, während komplexe Modelle den Bias reduzieren können (bis zu einem gewissen Punkt).

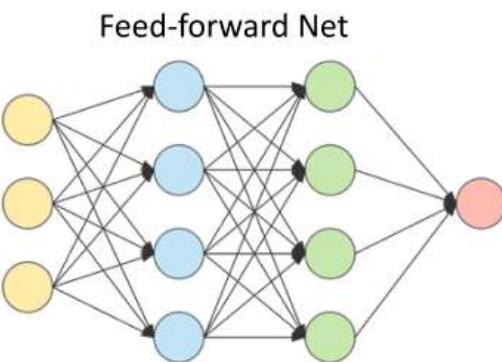
## Künstliche Neuronale Netze (NN) - Biologische Inspiration

- **Biologisches Vorbild:** Neuronale Netze im Gehirn von Menschen und Tieren.
- **Komplexität des Gehirns:** Ca. 10 bis 100 Milliarden Nervenzellen im menschlichen Gehirn.
- **Grundlage für Intelligenz:** Komplexe Verschaltung und Adaptivität der Neuronen ermöglichen Lernen und Anpassung.



- **Struktur und Funktion eines biologischen Neurons (vereinfacht):**
  - **Zellkörper (A, B, C):** Enthält den Zellkern und andere Organellen. Fungiert als Speicher für kleine elektrische Spannungen (ähnlich Kondensator/Akku).
  - **Dendriten:** Verzweigte Fortsätze, die eingehende Signale von anderen Neuronen über Synapsen empfangen.
  - **Axon:** Langer Fortsatz, der Ausgangssignale (Spannungsimpulse) zu anderen Neuronen über Synapsen weiterleitet.
  - **Feuern des Neurons:** Wenn die im Zellkörper gespeicherte Spannung einen bestimmten Schwellwert überschreitet, entlädt sich das Neuron und sendet einen Spannungsimpuls über das Axon.
  - **Synapsen:** Kontaktstellen zwischen dem Axon eines Neurons und den Dendriten eines anderen Neurons. Hier wird das Signal übertragen.
- **Lernen im biologischen neuronalen Netz:**
  - **Adaptivität der Synapsen:** Nicht die Neuronen selbst, sondern die Verbindungen (Synapsen) sind adaptiv.
  - **Veränderung der Verbindungsstärke:** Die Stärke der synaptischen Verbindungen kann sich verändern. Dies ermöglicht Lernen und Anpassung.
- **Mathematisches Modell der NN:**

- In short: Neural Networks (NN)
- Multi-layer fully-connected NN
- Elements:
  - Neurons (nodes)
  - Synapses (weights)
- Consist of:
  - Input layer,
  - Multiple hidden layers,
  - Output layer.
- Every node in one layer is connected to every other node in the next layer.

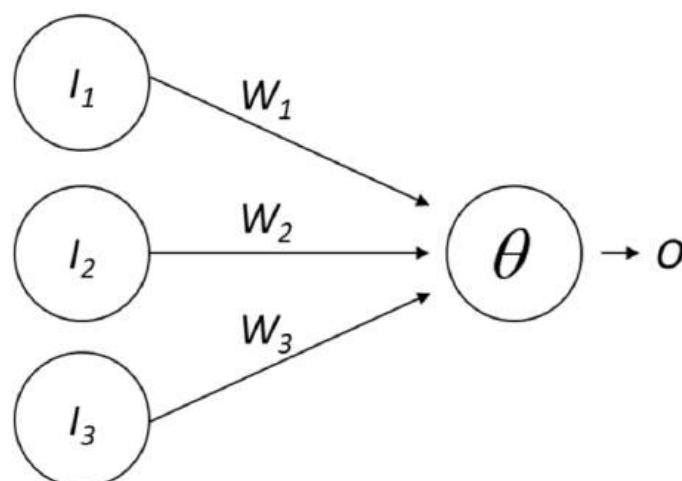


- **Inspiration:** Das biologische Modell diente als Vorbild.
- **McCulloch und Pitts (1943):** Stellten das erste mathematische Modell eines Neurons als grundlegendes Schaltelement vor.
- **Grundlage für NN:** Dieses Modell war die Basis für den Bau künstlicher neuronaler Netze.
- **Bestandteile des mathematischen Modells:**
  - **Neuronen (Knoten):** Entsprechen den Nervenzellen.
  - **Synapsen (Kanten):** Verbindungen zwischen den Neuronen.
  - **Gewichte:** Den Kanten (Synapsen) werden Gewichte zugewiesen.
- **Adaption:** Durch die Anpassung der Gewichte kann das "Verhalten" des künstlichen Neurons verändert werden (entspricht der Adaptivität der biologischen Synapsen).

## Das Perceptron - Ein frühes künstliches neuronales Netz

[EVC\\_Skriptum\\_CV, p.57](#)

- **Entwicklung:** 1958 von Frank Rosenblatt vorgestellt.
- **Struktur:** Besteht aus einem einzelnen künstlichen Neuron und einer Reihe von Eingängen ( $I_1$  bis  $I_k$ ).



- **Analogie zum biologischen Neuron:**

- **Eingänge ( $I_1$  bis  $I_k$ ):** Entsprechen den Dendriten.
- **Gewichte ( $w_1$  bis  $w_k$ ):** Werden mit den Eingangssignalen multipliziert. Entsprechen der Verstärkung oder Abschwächung natürlicher Signale durch Synapsen.
- **Gewichtete Summe der Eingaben:**  $\Phi(x) = \sum_i w_i I_i$
- **Aktivierungsfunktion ( $f$ ):** Wird auf die gewichtete Summe angewendet, um die Ausgabe des Neurons zu bestimmen:  $O = f(\sum_i w_i I_i)$ .
- **Ausgabe ( $O$ ):** Wird über "synaptische Gewichte" an nachgeschaltete Neuronen weitergegeben.

## Aktivierungsfunktionen

- **Vielfalt:** Es existieren verschiedene Möglichkeiten für die Aktivierungsfunktion  $f$ .
- **Einfachste Form: Identität**
  - $f(x) = x$
  - Das Neuron gibt lediglich die gewichtete Summe der Eingabewerte weiter.
  - **Problem:** Führt zu Konvergenzproblemen, da die Funktion nicht beschränkt ist und die Funktionswerte unbegrenzt wachsen können.

## Schwellwertfunktion im Perceptron

- **Motivation:** Um die unbegrenzte Ausgabe der Identitätsfunktion zu vermeiden, wird eine Schwellwertfunktion eingesetzt.
- **Funktionsweise:** Die gewichtete Summe der Eingangssignale  $\Phi(x) = \sum_i w_i I_i$  wird mit einem Schwellwert  $\theta$  verglichen.
- **Ausgabe ( $O$ ):**

$$O = \begin{cases} 1 & \text{falls } \sum_i w_i I_i + \theta \geq 0 \\ 0 & \text{sonst} \end{cases}$$

- **Abhängigkeit der Ausgabe:** Von den Eingangssignalen ( $I$ ), den zu lernenden Gewichten ( $w$ ) und dem Schwellwert ( $\theta$ ).
- **Ziel des Perceptrons (Klassifikation):** Trennung von Daten, die zu zwei unterschiedlichen Klassen gehören.
- **Lernaufgabe:** Anpassung der Gewichte  $w_i$ , sodass das Neuron bei Daten der ersten Klasse 0 und bei Daten der zweiten Klasse 1 ausgibt (oder umgekehrt).

## Perceptron-Lernalgorithmus ( $\delta$ -Regel / Widrow-Hoff-Regel)

- **Ziel:** Anpassung der Gewichte, um die gewünschte Ausgabe zu erzielen.
- **$\delta$ -Regel:** Die Gewichtsänderung  $\Delta w_i(t)$  zum Zeitpunkt  $t$  wird basierend auf der Differenz zwischen der gewünschten Ausgabe ( $T$ ) und der tatsächlichen Ausgabe ( $O$ ) berechnet:

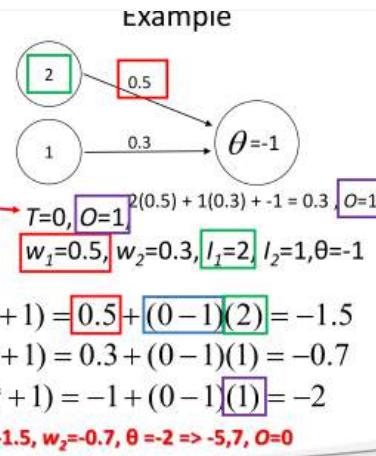
$$w_i(t+1) = w_i(t) + \Delta w_i(t)$$

$$\Delta w_i(t) = (T - O)I_i$$

- **Zwei Schlüsselkonzepte der  $\delta$ -Regel:**

1. **Fehlerbasierte Anpassung:** Die Gewichtsanpassung basiert auf dem Fehler ( $T - O$ ).
2. **Eingabeabhängigkeit:** Die Änderung hängt auch von der Eingabe  $I_i$  (der Ausgabe des vorherigen "Neurons") ab.

1. Randomly **assign weights** (between 0-1)
2. Present inputs from **training data**
3. **Get output  $O$ , nudge weights to give results toward desired output  $T$**
4. **Repeat;** stop when no errors, or enough epochs completed
  - Weights include Threshold
    - $T$ =Desired,  $w_i(t+1) = w_i(t) + \Delta w_i(t)$
    - $O$ =Actual output.  $\Delta w_i(t) = (T - O)I_i$
  - If we present this input again, output 0 instead  $w_1=-1.5, w_2=-0.7, \theta=-2 \Rightarrow -5.7, O=0$



## Konvergenz des Perceptron-Lernalgorithmus

- **Rosenblatts Theorem:** Der Lernalgorithmus des Perceptrons konvergiert in endlicher Zeit, wenn die Daten linear separierbar sind.
- **Implikation:** Das Perceptron kann alles lernen, was es repräsentieren kann, in endlicher Zeit.

## Beschränkungen des Perceptrons

- **Nicht-lineare Separierbarkeit:** Ein einschichtiges Perceptron kann keine Funktionen lernen, die nicht linear separierbar sind.
- **Lineare Separierbarkeit:** Die Eigenschaft, dass Daten im Raum durch Hyperebenen (in 2D: Geraden, in 3D: Ebenen) voneinander getrennt werden können.

## Mehrschichtige Perceptrons

- **Zweischichtige Perceptrons:** Können konvexe Mengen trennen.
- **Mehrschichtige Perceptron-Netze:** Können beliebige Mengen trennen. Dies ist ein wichtiger Schritt zur Überwindung der Beschränkungen des einfachen Perceptrons.

## Aktivierungsfunktionen für stetige Neuronen

- **Problem der Stufenfunktion:** Für binäre Neuronen (Ausgabe 0 oder 1) sinnvoll, erzeugt aber eine Unstetigkeit bei stetigen Neuronen (Aktivierungen zwischen 0 und 1).
- **Lösung: Sigmoid-Funktion:** Eine Möglichkeit, die Unstetigkeit zu glätten.
- **Beispiel einer Sigmoid-Funktion:**

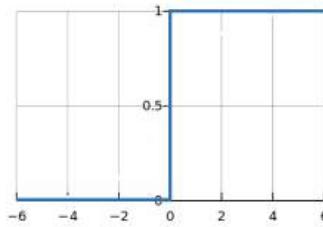
$$O = \frac{1}{1 + e^{-(\Phi(x)+\theta)}}$$

wobei  $\Phi(x) = \sum_i w_i I_i$  die gewichtete Summe der Eingaben und  $\theta$  der Schwellwert ist.

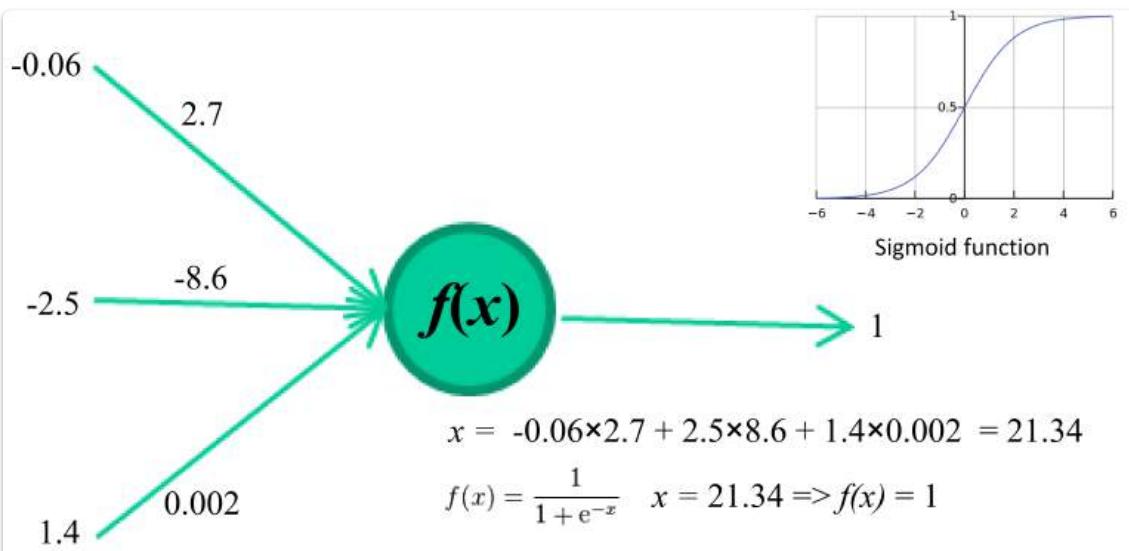
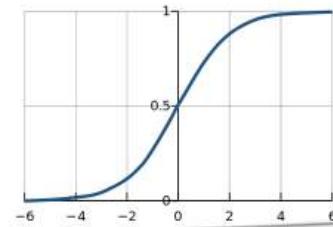
- **Eigenschaften der Sigmoid-Funktion:**
  - **Annähernd linear:** Verhält sich in der Nähe des kritischen Bereichs um den Schwellwert  $\theta$  annähernd linear.
  - **Asymptotisch beschränkt:** Die Ausgabe liegt immer zwischen 0 und 1. Dies verhindert das Problem des unbegrenzten Wachstums der Aktivierungen.
- **Vorteil gegenüber der Stufenfunktion:** Ermöglicht differenzierbare Ausgaben, was für viele Lernalgorithmen in neuronalen Netzen (insbesondere für das Training mit Gradientenabstieg) entscheidend ist.

$$\Delta w_k = cI_k(T_j - O_j)f'(ActivationFunction)$$

Old:  $O = \begin{cases} 1: \sum_i w_i I_i + \theta > 0 \\ 0: \text{otherwise} \end{cases}$



New:  $O = \frac{1}{1 + e^{-\sum_i w_i I_i + \Theta}}$



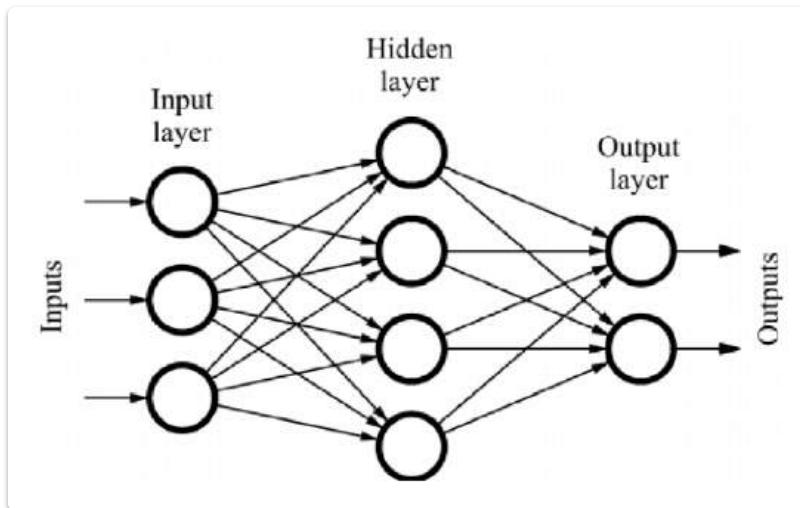
Ein animiertes Beispiel findet man in den Slides: [EVC-CV11-Deep Learning\\_2025S\\_Slides](#), p.42

## Multilayer Perceptron (MLP)

[EVC\\_Skriptum\\_CV](#), p.58

- **Typischer Aufbau:** Besteht aus mehreren Schichten von Verarbeitungseinheiten.
- **Beispiel (Abbildung 49):**
  - **Eingangsschicht (Input Layer):** Empfängt die Eingabedaten.
  - **Verborgene Schicht (Hidden Layer):** Eine oder mehrere Zwischenschichten, die komplexe Muster in den Daten verarbeiten.

- **Ausgabeschicht (Output Layer):** Liefert das Ergebnis der Verarbeitung (z.B. Klassifikation oder Regression).
- **Verbindungen:** Zwischen den Schichten befinden sich Lagen von Verbindungen, die mit Gewichten versehen sind.



- **Variationen der Struktur:**
  - **Shortcut-Verbindungen:** Direkte Verbindungen zwischen der Eingangsschicht und der Ausgabeschicht.
  - **Mehrere verborgene Schichten (Deep Neural Networks):** Ermöglichen die Modellierung noch komplexerer Beziehungen in den Daten.
- **Arbeitsweise: Vorwärtsvermittlung (Feed-forward Network):**
  - Die Aktivierung erfolgt schichtweise von der Eingabe- zur Ausgabeschicht.
  - **Keine Rückkopplungen:** Signale fließen nur in eine Richtung.

## Training von Multilayer Perceptron (MLP)

- **Überwachter Lernvorgang:** Die Gewichtsfaktoren zwischen den Verarbeitungseinheiten werden während des Trainings angepasst.
- **Lernzyklus (Epoche):** Alle Muster des Trainingsdatensatzes werden vom Netz klassifiziert (oder für Regression verwendet).
- **Fehlerberechnung:** Die Ergebnisse des Netzes werden mit den Soll-Werten (Labels) des Trainingsdatensatzes verglichen.
- **Gewichtsanpassung:** Die Gewichtsfaktoren werden so angepasst, dass der Fehler zwischen der Netzwerkausgabe und dem Soll-Wert minimiert wird.
- **Ziel:** Nach einer Reihe von Trainingszyklen soll der Trainingsdatensatz mit einer vorgegebenen Genauigkeit reproduziert werden.

## Backpropagation-Training (Fehler-Rückvermittlung)

- **Bekanntestes Trainingsverfahren für MLPs.**
- **Prinzip:** In jedem Lernschritt werden die Gewichte in Richtung des abnehmenden Fehlers verändert (Gradientenabstieg).
- **Ablauf eines Lernschritts:**

1. **Vorwärtslauf (Forward Pass):** Ein Eingangsmuster wird an das Netz angelegt und die Aktivierungen werden schichtweise bis zur Ausgabeschicht berechnet.
2. **Vergleich mit Soll-Output:** Der Zustand der Ausgabeschicht wird mit dem bekannten Soll-Output für dieses Muster verglichen.
3. **Fehlerberechnung:** Die Abweichung (der Fehler) zwischen der tatsächlichen und der gewünschten Ausgabe wird berechnet.
4. **Rückwärtslauf (Backward Pass):** Der Fehler wird vom Ende des Netzes (Ausgabeschicht) zurück zu den vorherigen Schichten propagiert.
5. **Gewichtsaktualisierung:** Die Gewichte jeder Verbindung im Netz werden proportional zum Beitrag dieser Verbindung zum Fehler angepasst.

## Generalisierte Delta-Regel und Fehlerrückvermittlung

- **Problem bei mehrschichtigen Netzen:** Für die verborgenen Schichten ist kein direkter Soll-Zustand bekannt. Die einfache Delta-Regel ist daher nicht direkt anwendbar.
- **Lösung:** Die Generalisierte Delta-Regel mit Fehlerrückvermittlung (Backpropagation).
- **Kernidee:** Der Fehler der Ausgabeschicht wird verwendet, um einen "virtuellen Fehler" für die Neuronen in den verborgenen Schichten zu berechnen. Dieser virtuelle Fehler gibt an, wie stark die Aktivität eines verborgenen Neurons zum Fehler in der Ausgabeschicht beigetragen hat.
- **Anwendung:** Mithilfe dieses virtuellen Fehlers können dann auch die Gewichte der Verbindungen, die in die verborgenen Neuronen führen, angepasst werden.

## Zielfunktion des Backpropagation-Algorithmus

- **Ziel:** Minimierung der Summe der quadratischen Fehler (Least Mean Squares, LMS) bei der Klassifikation aller Trainingsmuster.
- **Fehlerfunktion (LMS):**

$$Distance(LMS) = \frac{1}{n} \sum_{p=1}^n (T_p - O_p)^2$$

- $T_p$ : Soll-Ausgangswert für das  $p$ -te von  $n$  Trainingsbeispielen.
- $O_p$ : Tatsächlicher Ausgangswert des Netzes für das  $p$ -te Trainingsbeispiel.
- **Gesucht:** Das globale Minimum dieser Gesamtfehlerfunktion.

## Gradientenabstieg

- **Prinzip der Gewichtsanpassung:** Die Gewichte werden stets in Richtung des abnehmenden Fehlers verändert, d.h., entgegen dem Gradienten des Fehlers.
- **Ursprüngliche Delta-Regel:** Eine frühe Form dieser Idee.

## Kernidee des Backpropagation-Verfahrens

- **Fehler als Funktion der Gewichte:** Der Ausgangswert der Ausgabeeinheiten (und damit der Fehler  $F$ ) ist eine Funktion ihrer Eingabewerte. Diese Eingabewerte sind wiederum Funktionen der Ausgänge der vorhergehenden (verdeckten) Schicht und der Gewichte zwischen diesen Schichten.
- **Kettenregel:** Die Ausgänge der Zwischenschicht sind eine Funktion der Gewichte zwischen Eingangs- und Zwischenschicht. Daraus folgt, dass der Fehler  $F$  letztendlich auch eine Funktion dieser Gewichte ist.
- **Partielle Differentiation:** Durch Anwendung der Kettenregel ist es möglich, auch für die Zwischenschichten einen Fehler zu definieren.
- **Gradient für Zwischenschichten:** Aus diesem Fehler kann ein Gradient berechnet werden, um die Gewichte zwischen den Schichten anzupassen.

## Fehlerrückvermittlung (Error Backpropagation)

- **Berechnung des Fehlerwerts:** Zuerst für die Ausgabeeinheiten, dann für die verdeckten Einheiten.
- **"Backpropagation":** Die Fehleroptimierung pflanzt sich von hinten (Ausgabeschicht) nach vorne durch das Netz.
- **Implikation:** Die Information, wie die einzelnen Gewichte zum Fehler der Ausgabeschicht beigetragen haben, wird zurück durch das Netzwerk geleitet, um eine sinnvolle Anpassung der Gewichte in allen Schichten zu ermöglichen.

# Deep Learning

[EVC\\_Skriptum\\_CV, p.58, p.59](#)

### 1. What exactly is Deep Learning?

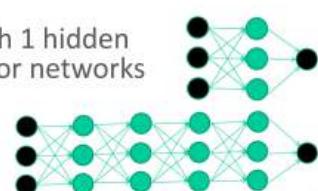
- Deep Learning means using a neural network with several layers of nodes between input and output

### 2. Why is it generally better than other methods?

- The series of layers between input & output do feature identification and processing in a series of stages, just as our brains seem to.

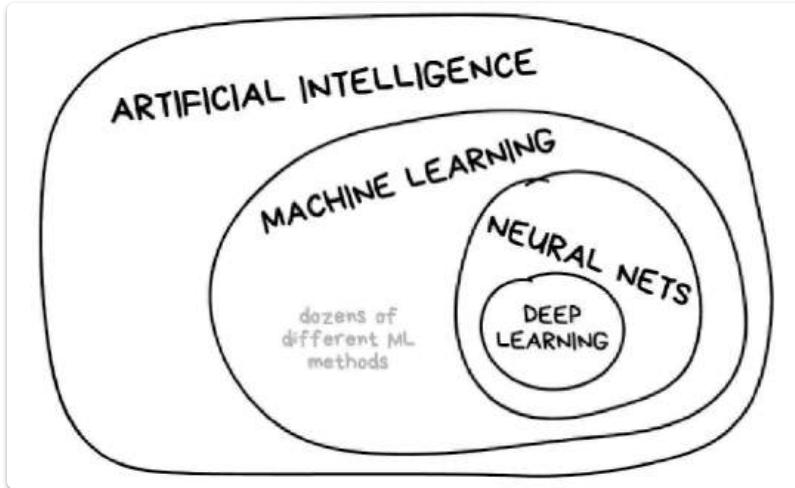
### 3. Multilayer neural networks have been around for 30 years. What's actually new?

- We had good algorithms for learning weights in networks with 1 hidden layer, but these algorithms are not good at learning weights for networks with more hidden layers.
- What's new is: algorithms for training many-layer networks



- **Bisherige Ansätze (Machine Learning):** Lernalgorithmen können aus Trainingsdaten komplexe Klassifikationsaufgaben lernen.
- **Manuelle Merkmalsgenerierung:** Die zur Klassifikation verwendeten Merkmale (z.B. Farbe, Kanten, SIFT) mussten bisher manuell von einem "Wissensingenieur" ausgewählt und generiert werden.
  - Ziel: Einen sinnvollen Satz von möglichst wenigen, aussagekräftigen Merkmalen finden.

- Diese Merkmale dienen dann als Eingabe für die Lernalgorithmen.



## Herausforderung: Direkte Verwendung von Sensordaten

- **Idee:** Warum nicht direkt alle verfügbaren Sensordaten verwenden (z.B. Pixel eines Bildes)?
- **Beispiel (RGB-Bild):** Ein Foto mit 10 Millionen Pixeln hätte einen Eingabevektor der Länge 30 Millionen (10 Mio. Pixel x 3 Farbwerte).
- **Problem: Skalierung mit der Dimension der Eingabedaten:**
  - **Trainingszeiten:** Wachsen sehr schnell, oft exponentiell, mit der Dimension der Eingabedaten.
  - **Rechenaufwand:** Um die Rechenzeiten in Grenzen zu halten, müssen die Eingabedaten zuerst auf kurze Merkmalsvektoren abgebildet werden.

## Traditionelle Merkmalsgewinnung

- **Frühe Merkmalsbestimmung:** Merkmale werden frühzeitig anhand manuell definierter Formeln bestimmt.
- **Grundlage:** Meist Expertenwissen, orientiert an der menschlichen Wahrnehmung ("Wie würde der Mensch die Klassifikationsaufgabe lösen?").

## Deep Learning: End-to-End-Learning

- **Herausforderung der manuellen Merkmalsgenerierung:** Für viele komplexe Anwendungen (z.B. Objektklassifikation in Bildern) ist es sehr schwierig bis unmöglich, manuell gute Merkmale zu definieren.
- **Prinzip von Deep Learning:** End-to-End-Learning.
  - **Simultanes Lernen:** Nicht nur die optimale Verarbeitung der Merkmale wird gelernt, sondern auch gleichzeitig die optimale Herleitung dieser Merkmale aus den Rohdaten.
  - **Automatisierte Merkmalsentwicklung:** Das Netzwerk lernt selbstständig, welche Merkmale für die jeweilige Aufgabe relevant sind.

## Deep Learning Architekturen

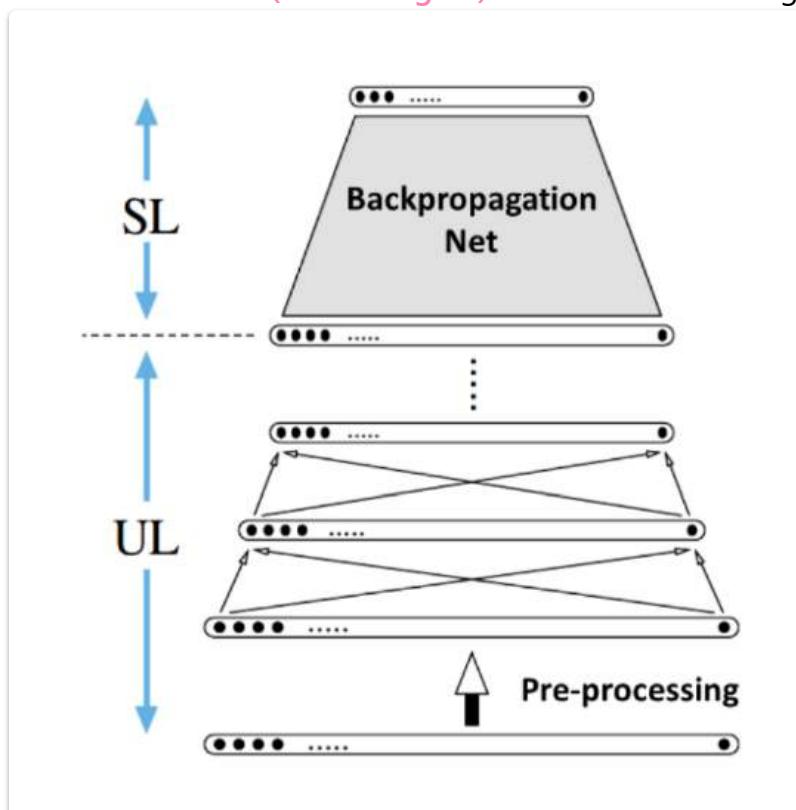
- **Computer Vision:**
  - **Convolutional Neural Networks (CNNs):** Kommen in der Regel zum Einsatz.
  - **Filterkoeffizienten:** Werden gelernt, um die Bilder zu Beginn zu falten (convolved). Diese Faltungsschritte extrahieren automatisch hierarchische Merkmale aus den Bildern (z.B. Kanten, Texturen, Objektteile).
- **Andere Deep Learning Verfahren:**
  - **Recurrent Neural Networks (RNNs):** Gut geeignet für sequenzielle Daten (z.B. Text, Zeitreihen).
  - **Generative Adversarial Networks (GANs):** Werden für generative Aufgaben eingesetzt (z.B. Erzeugung neuer Bilder, Texte).
  - **Variationen dieser Architekturen.**

## Komplexität tiefer neuronaler Netze

- **Anzahl der Schichten:** Architekturen mit bis zu fünfzig oder mehr Schichten sind möglich.
- **Hohe Komplexität:** Die genauen Architekturen sind oft sehr komplex und können hier nicht im Detail dargestellt werden.
- **Ressource für weitere Informationen:** Gute Einführungen finden sich auf [ufldl.stanford.edu/tutorial/](http://ufldl.stanford.edu/tutorial/).

## Deep Learning mit vielen Schichten

- **Aktueller Erfolg:** Erfolgreiche Deep-Learning-Lösungen verwenden viele Schichten von Neuronen.
- **Netzwerkstruktur (Abbildung 50):** Oft in zwei Teile aufgespalten.

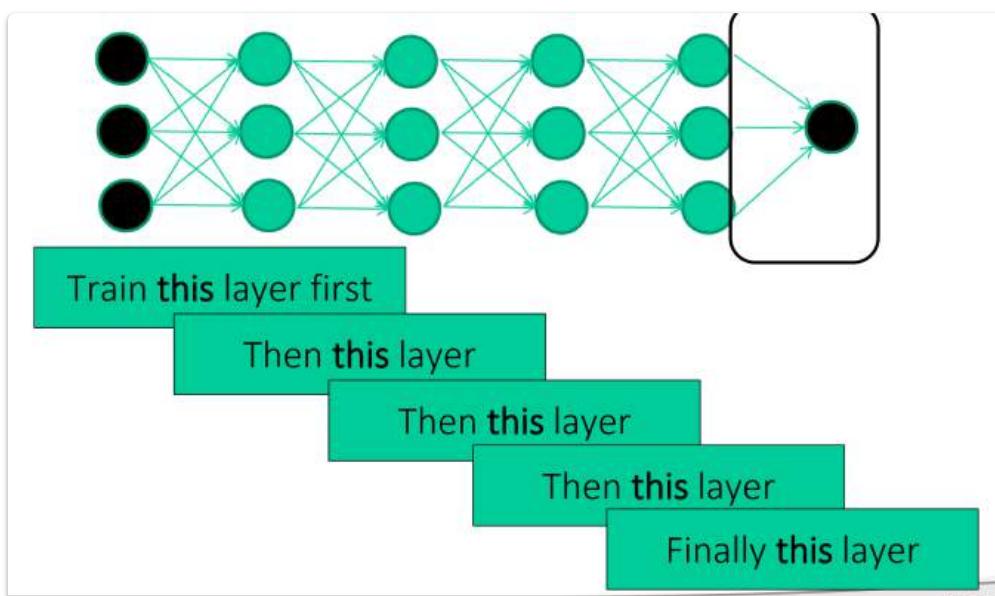


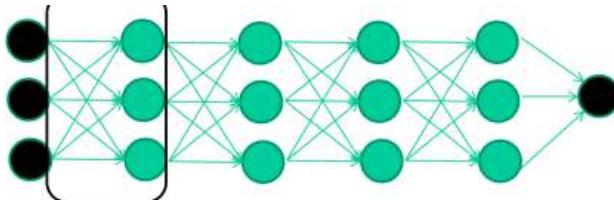
- **Unsupervised Learning (UL) zur Vorverarbeitung:**

- Nach einer Vorverarbeitungsschicht folgen mehrere Lagen, die mit unüberwachtem Lernen vorgenommen werden.
- **Merkmalsrepräsentation:** Jede Lage im UL-Netz repräsentiert Merkmale des Eingabemusters.
- **Hierarchische Merkmale:**
  - **Tiefe Lagen:** Repräsentieren einfache Merkmale (z.B. Kanten oder Linien in verschiedenen Ausrichtungen bei der Objekterkennung in Bildern).
  - **Höhere Lagen:** Können komplexe Merkmale bilden (z.B. das Vorhandensein eines Gesichts).
- **Supervised Learning (SL) zur Klassifikation/Regression:**
  - An das UL-Netz schließt sich ein klassisches überwachtes Lernen an.
  - **Training:** Kann beispielsweise mit Backpropagation trainiert werden.
- **Ablauf des Lernens:**
  1. **UL-Training:** Vortrainieren der Gewichte aller Merkmalschichten (Extraktion der Merkmale).
  2. **SL-Training:** Training des gesamten Netzes (einschließlich der vorgenommenen Schichten) mit überwachten Daten (z.B. Gradientenabstieg).

## Unsupervised Learning der Gewichte und Merkmalsextraktion

- **Ziel des UL-Trainings:** Bildung von Merkmalsgruppen durch das Lernen der Gewichte zu den Merkmalschichten. Hier findet die eigentliche Merkmalsextraktion statt.
- **Eigenschaft der Merkmalsextraktion:** Die Eingabedaten sollen in einen niedrigerdimensionalen Raum abgebildet werden, möglichst ohne (zu viel) Informationsverlust. Dies reduziert die Dimensionalität des Problems für die nachfolgende SL-Phase und kann zu robusteren und effizienteren Modellen führen.



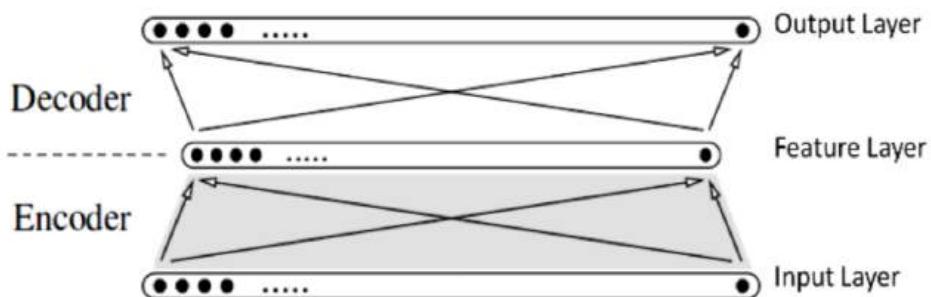


- EACH of the (non-output) layers is trained to be an auto-encoder
- Basically, it is forced to learn good features that describe what comes from the previous layer

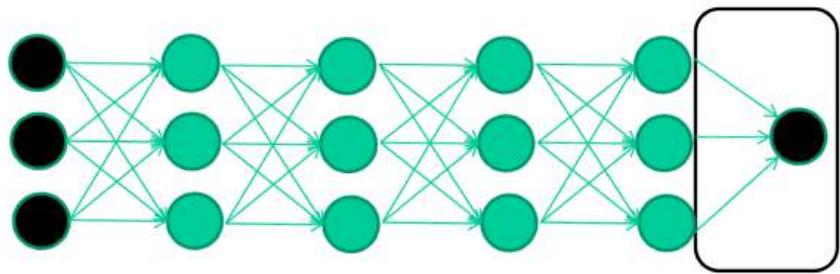
## Autoencoder

EVC\_Skriptum\_CV, p.59

- **Bestimmung der Gewichte der Merkmalschichten (UL):** Erfolgt bei sogenannten Autoencodern nach dem in der Abbildung gezeigten Verfahren.



- **Training der ersten verdeckten Merkmalschicht:**
  - Ein Autoencoder wird mit unüberwachtem Lernen (UL) trainiert.
  - **Ziel des Autoencoders:** Die identische Abbildung aller Eingabevektoren  $x$  auf sich selbst zu lernen (Identitätsfunktion).
  - **Merkmalschicht:** Dient hier als die erste verdeckte Lage (Feature Layer).
- **Nach dem Training des ersten Autoencoders:**
  - Die nicht benötigte Decoderlage (die zweite Lage von Gewichten) wird gelöscht.
  - Die erste Lage der Gewichte (Encoder) wird eingefroren und für die Berechnung der Merkmale in der ersten Lage des UL-Netzes übernommen.
- **Training der zweiten Merkmalschicht:**
  - Mit dieser festen ersten Schicht von Gewichten wird die zweite Merkmalschicht wieder mit dem Autoencoder-Verfahren trainiert.
  - Die Gewichte der Encoder-Seite des zweiten Autoencoders werden eingefroren.
- **Iterativer Prozess:** Dieser Vorgang wird so weitergeführt bis zur letzten Merkmalschicht.
- **Abschluss des unüberwachten Teils des Lernens:** Sobald alle Merkmalschichten auf diese Weise vortrainiert wurden, ist der unüberwachte Teil des Lernens beendet.



- Is trained to predict class based on outputs from previous layers

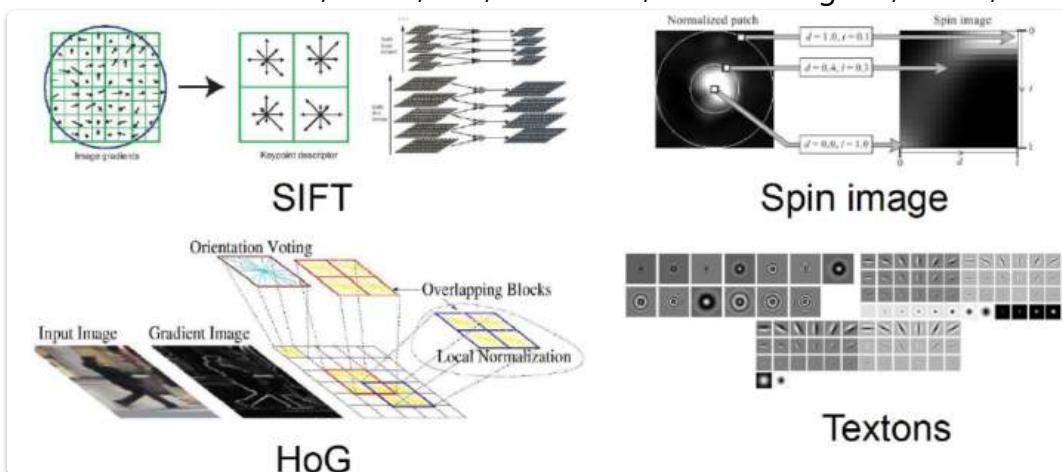
## Pros / Cons von Deeplearning

- |   |   |
|---|---|
| <ul style="list-style-type: none"> <li>• Pros           <ul style="list-style-type: none"> <li>• Not-domain specific</li> <li>• Supervised / Semi-supervised / Unsupervised</li> <li>• Classification / regression in last layer</li> <li>• Simple math</li> <li>• Hip</li> </ul> </li> </ul> | <ul style="list-style-type: none"> <li>• Cons           <ul style="list-style-type: none"> <li>• Lots of meta-parameters</li> <li>• Needs a lot of data</li> <li>• Very computational intensive</li> <li>• Hip</li> </ul> </li> </ul> |
|---|---|

## Weitere VO-Slides:

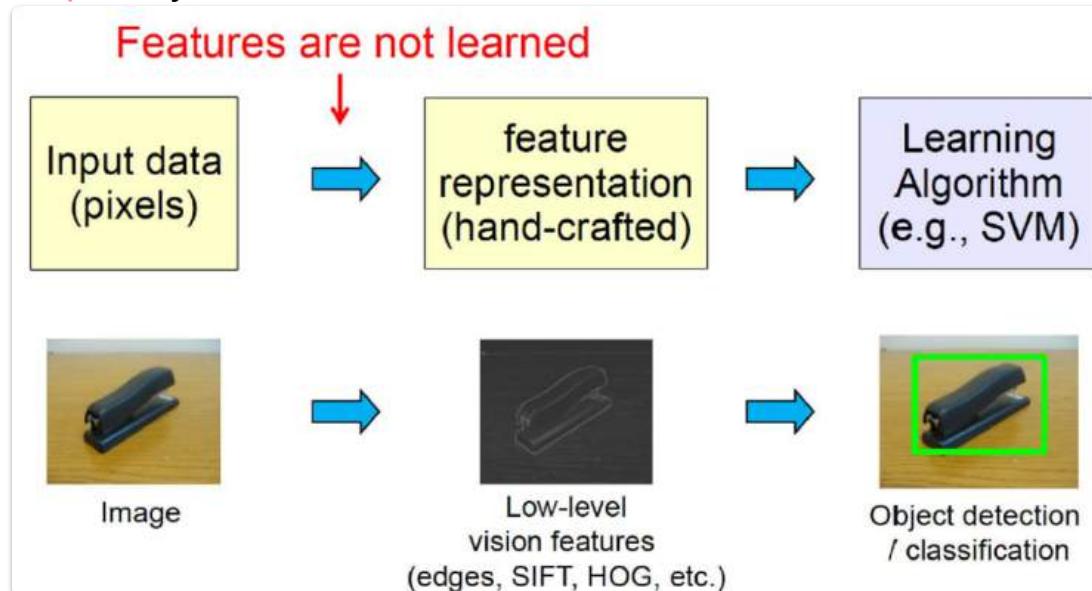
### Traditional Computer Vision Features

- Beispiele für handgefertigte Features:
  - SIFT (Scale-Invariant Feature Transform)
  - Spin image
  - HoG (Histogram of Oriented Gradients)
  - Textons
  - Viele andere wie SURF, MSER, LBP, Color SIFT, Color Histogram, GLOH, ...



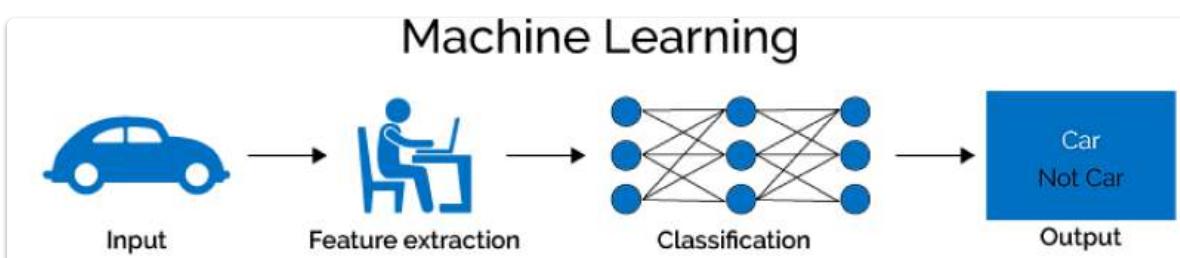
## Traditional Recognition Approach

- Merkmale werden nicht gelernt (Features are not learned).
- Ablauf:
  1. **Input data (pixels)**: Rohdaten, z.B. ein Bild.
  2. **Feature representation (hand-crafted)**: Manuell entworfene Merkmalsrepräsentation (Low-level vision features wie edges, SIFT, HoG, etc.).
  3. **Learning Algorithm (e.g., SVM)**: Ein Lernalgorithmus (z.B. Support Vector Machine) wird auf den extrahierten Merkmalen trainiert.
  4. **Output**: Objekt-Detektion / Klassifikation.



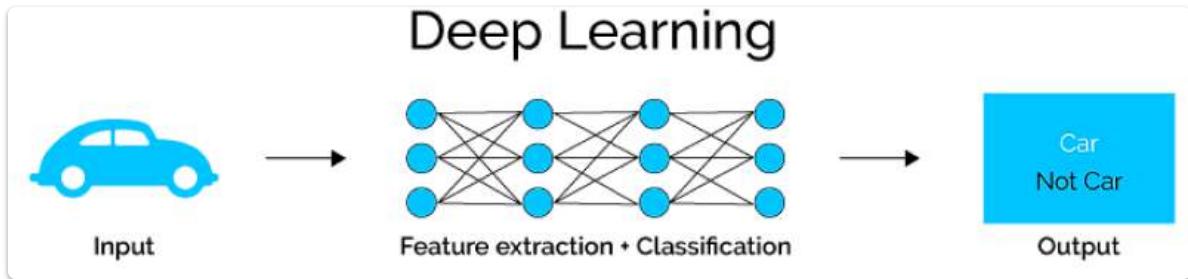
## Difference of Deep Learning to Classic Machine Learning

- **Machine Learning (Klassisch):**
  - Computes features.
  - Has simple and complex features.
  - **Ablauf:**
    1. **Input**: Rohdaten (z.B. ein Auto-Bild).
    2. **Feature extraction**: Manuelle oder algorithmische Extraktion von Merkmalen.
    3. **Classification**: Ein separater Klassifikator lernt, die extrahierten Merkmale den Klassen zuzuordnen.
    4. **Output**: Klassifikation (z.B. "Car" oder "Not Car").



- **Deep Learning:**
  - Does not need human interaction for feature design.
  - **Ablauf:**

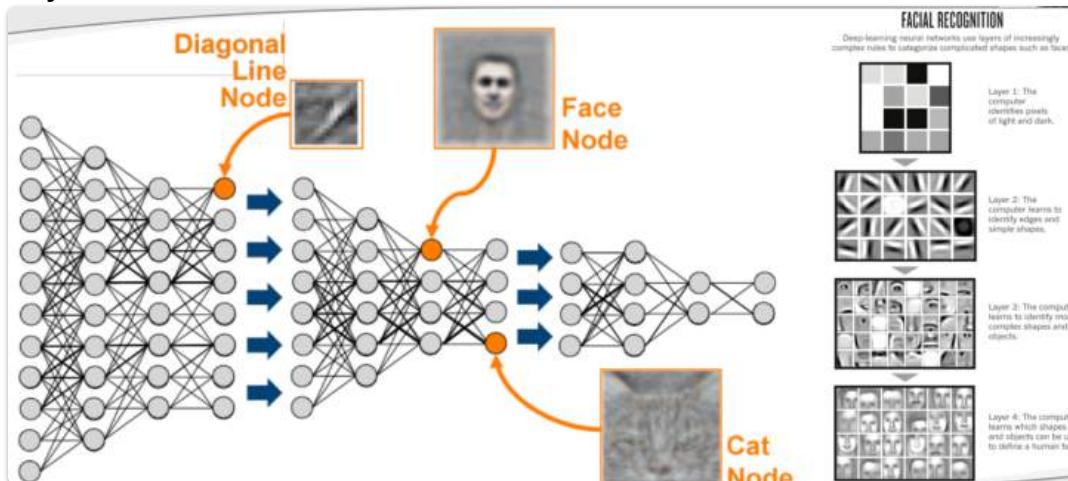
1. **Input:** Rohdaten (z.B. ein Auto-Bild).
2. **Feature extraction + Classification:** Die Merkmalsextraktion und die Klassifikation erfolgen gemeinsam und werden end-to-end gelernt in einem tiefen neuronalen Netzwerk.
3. **Output:** Klassifikation (z.B. "Car" oder "Not Car").



**Zusammenfassend:** Der Hauptunterschied besteht darin, dass traditionelles Machine Learning auf handgefertigten Merkmalen basiert, während Deep Learning die relevanten Merkmale direkt aus den Rohdaten lernt, was die Notwendigkeit menschlicher Expertise im Feature Engineering reduziert und oft zu besseren Ergebnissen bei komplexen Aufgaben führt.

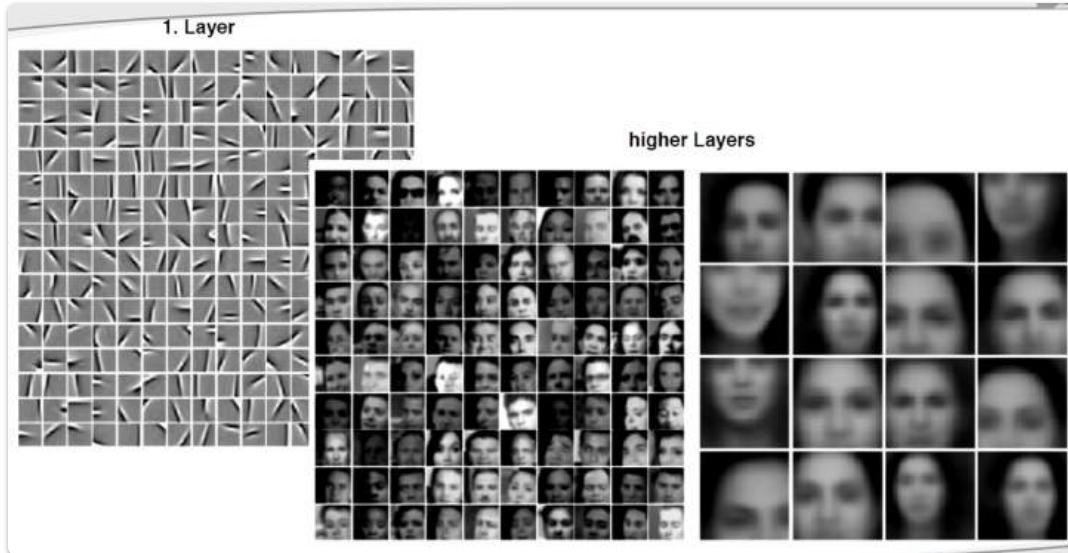
## CNN Features

- **Hierarchische Merkmalsrepräsentation:** Convolutional Neural Networks (CNNs) lernen hierarchische Merkmale.
- **Beispiel (Gesichtserkennung):**
  - **Diagonale Kanten (frühe Schichten):** Neuronen in frühen Schichten können auf einfache Merkmale wie diagonale Kanten reagieren.
  - **Gesichtsteile (mittlere Schichten):** Spätere Schichten kombinieren diese einfachen Merkmale, um komplexere Muster wie Augen, Nasen oder Münden zu erkennen.
  - **Gesicht (späte Schichten):** Noch höhere Schichten können das gesamte Gesicht als abstraktes Merkmal repräsentieren.
  - **Katze (anderes Beispiel):** Ähnliche hierarchische Merkmalsentwicklung für andere Objekte.



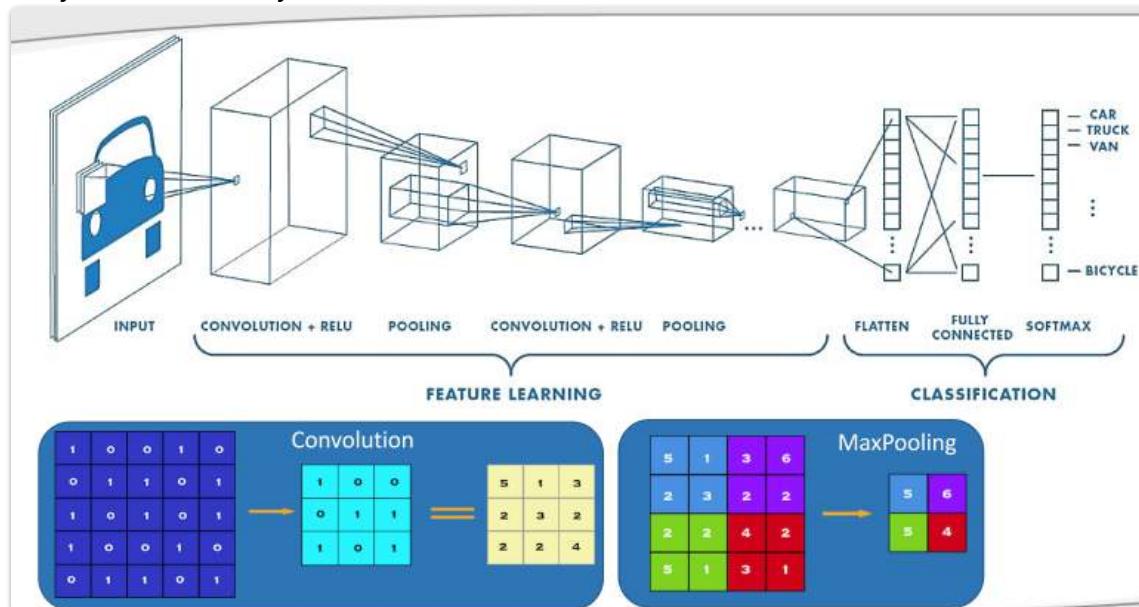
## CNN Visualization

- **Visualisierung gelernter Filter:** Die Abbildung zeigt beispielhafte Filter, die von CNNs in verschiedenen Schichten gelernt wurden.
  - **1. Layer:** Lernt oft einfache, lokale Muster wie Kanten und Orientierungen.
  - **Höhere Layer:** Lernen zunehmend komplexere und globalere Muster, die spezifische Objekte oder Teile davon repräsentieren können (z.B. Gesichtszüge).



## Inception Topologies

- **Beispiel einer komplexeren CNN-Architektur:** Die Inception-Architektur (hier vereinfacht dargestellt) verwendet verschiedene Filtergrößen und Operationen parallel, um Merkmale unterschiedlicher Skalen zu erfassen.
- **Feature Learning Pfad:** Zeigt typische Operationen wie Convolution und Pooling zur Merkmalsgewinnung.
- **Classification Pfad:** Führt die gelernten Merkmale zu einer Klassifikationsschicht (z.B. Fully Connected Layer mit Softmax).



## Deep Learning - Why does it work?

- **Can cope with vast amounts of data:** Tiefe Netze können von großen Datenmengen profitieren, um komplexe Muster zu lernen.
- **Learns small invariances:** Sie lernen, invariant gegenüber kleinen Variationen in den Eingabedaten zu sein (z.B. leichte Verschiebungen, Skalierungen, Rotationen).
- **Over-complete, sparse representations:** Können redundante und gleichzeitig spezialisierte Repräsentationen lernen.
- **Learn embedding:** Lernen, Eingabedaten in einen semantisch sinnvollen, niedrigerdimensionalen Raum einzubetten.
- **Lots of data:** Die Verfügbarkeit großer Trainingsdatensätze ist entscheidend für den Erfolg von Deep Learning.
- **Recent advance: it is actually computable!**: Fortschritte in der Hardware (z.B. GPUs) und in den Algorithmen haben das Training tiefer Netze in praktikabler Zeit ermöglicht.

## Weitere vo-slides zu ImageNet

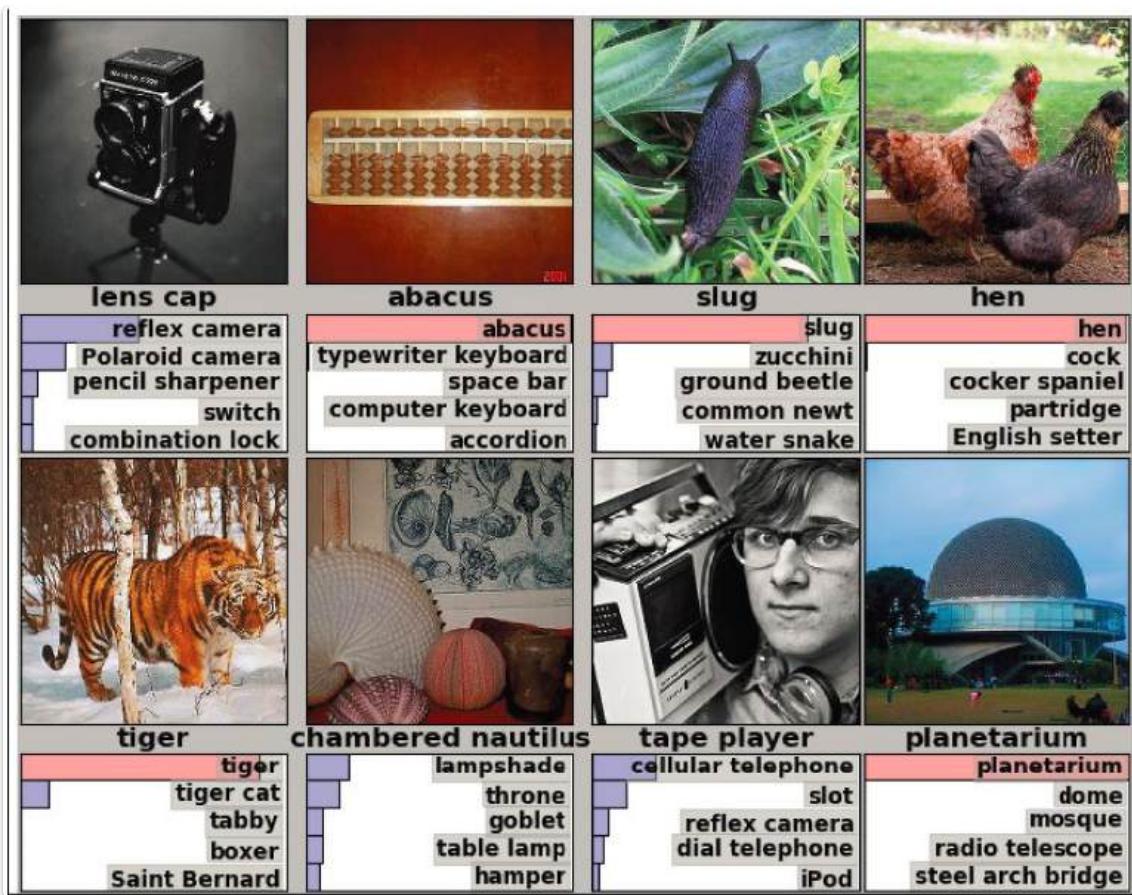
---

### CNN Success Story: ILSVRC

- **ImageNet database:**
  - 14 million labeled images.
  - 20,000 categories.

### ILSVRC: Classification

- **Computer Vision: International Large-Scale Visual Recognition Challenge (ILSVRC).**
- **Ziel:** Bildklassifizierung auf sehr großem Maßstab.
- **Beispielbilder aus dem ILSVRC Datensatz.**



# 12. Computational Photography

EVC\_Skriptum\_CV, p.60

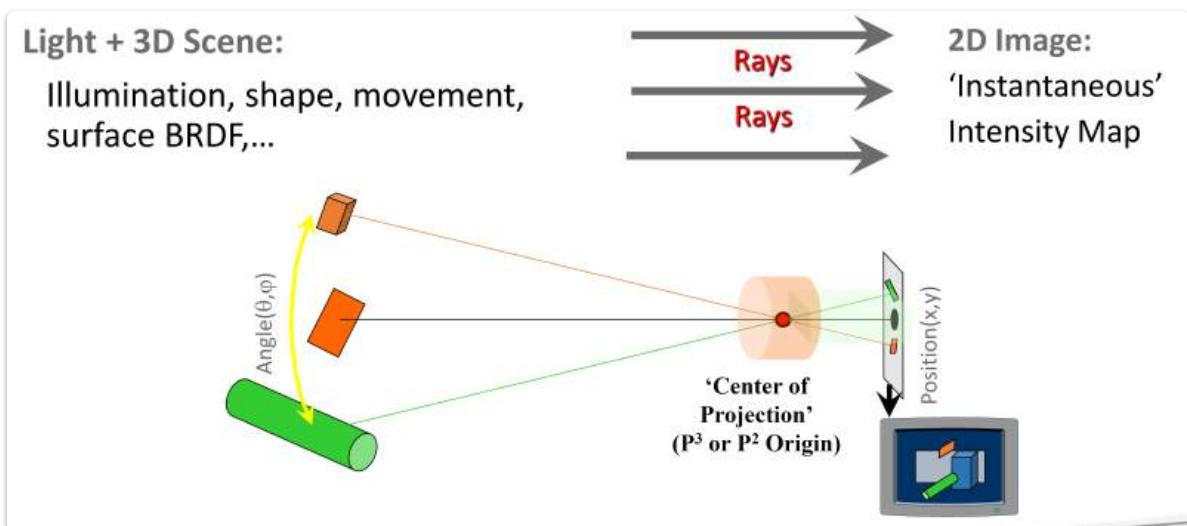
Computational Photography ist ein Forschungsgebiet, das die **Verbreitung von Digitalkameras** nutzt, um die alltägliche Fotografie zu verbessern.

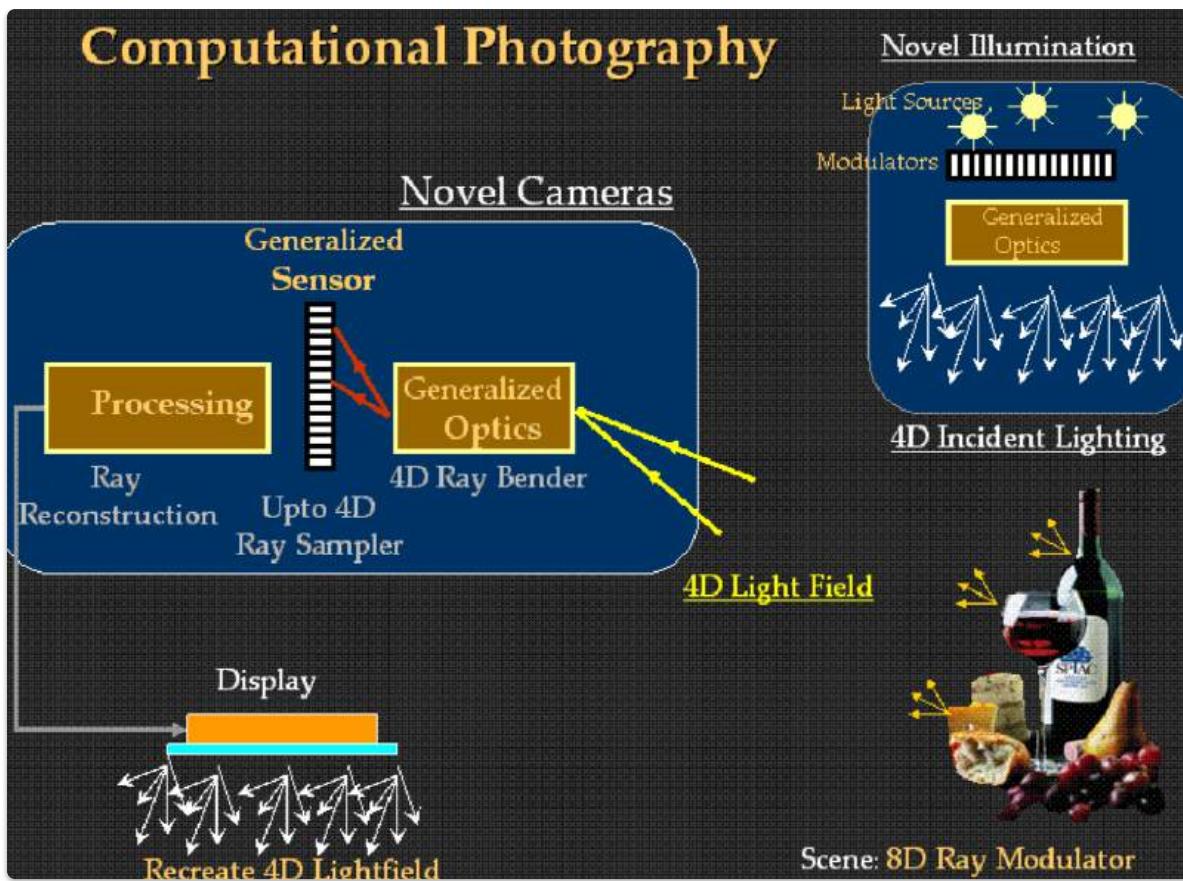
- **Ziele:**

- Neue Aufnahme- und Bearbeitungsmethoden entwickeln, die normalerweise fotografisch nicht möglich wären.
- Beispiele: Aufnahme bei extremem Kontrastumfang, verbesserte verwackelte Bilder, Komposition mehrerer Personen oder Objekte zu einem neuen Gesamtbild, Entfernen störender Personen oder Ähnliches.
- Übergeordnetes Ziel ist es, neue Funktionalitäten und Anwendungsbereiche für die Fotografie zu erschließen.

- **Grundlage:** Mittels algorithmischer ("Computational") Ansätze werden die physikalischen Grenzen traditioneller Projektionskameras überwunden.

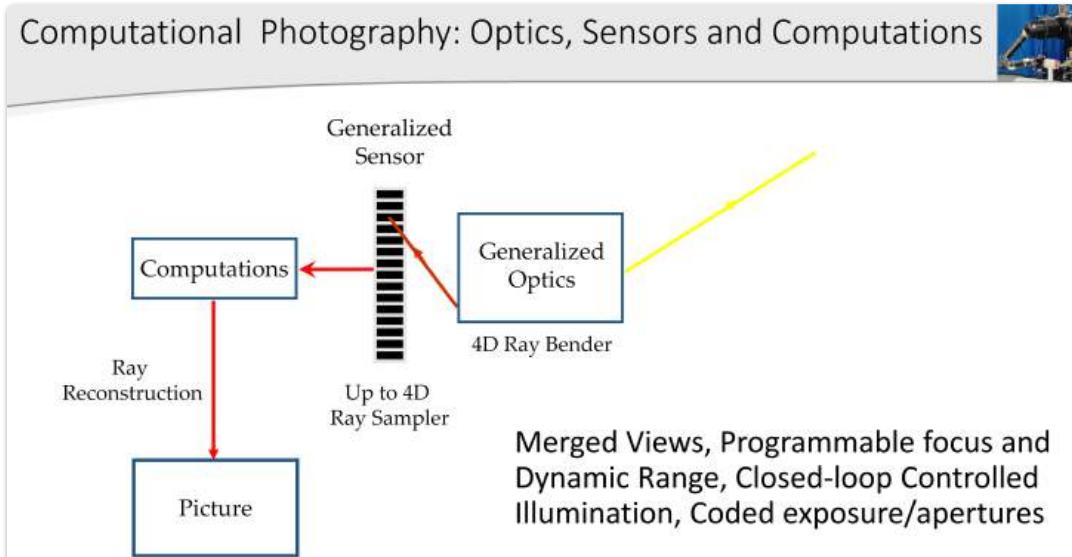
- Ermöglicht Bilderzeugung, die mit herkömmlichen Kameras nicht realisierbar wäre (z.B. große Tiefenschärfe, hohe Auflösung, geringer Verlust der dritten Dimension).
- Durch Kombination von Software, digitalen Sensoren, moderner Optik sowie intelligenter Beleuchtung verschiebt Computational Photography die Grenzen der traditionellen Fotografie.





## Moderne Kameras vs. Traditionelle Kameras

- **Moderne Kameras (Computational Photography):**
  - Komplexe Produkte, die eine über hundertjährige technische Evolution durchlaufen haben.
  - Grundlegender optischer Aufbau aus Linse(n), Blende und Sensor ist praktisch unverändert.
  - Aktuelle Systeme sind in der Lage, Leistungen zu erbringen, die das menschliche Auge übertreffen.



- **Traditionelle Kameras:**

- Nachteile, besonders bei der Bildaufnahme von traditionellen Kameras:  
*Führen dem Verlust einer Dimension der Szenengeometrie zu.*

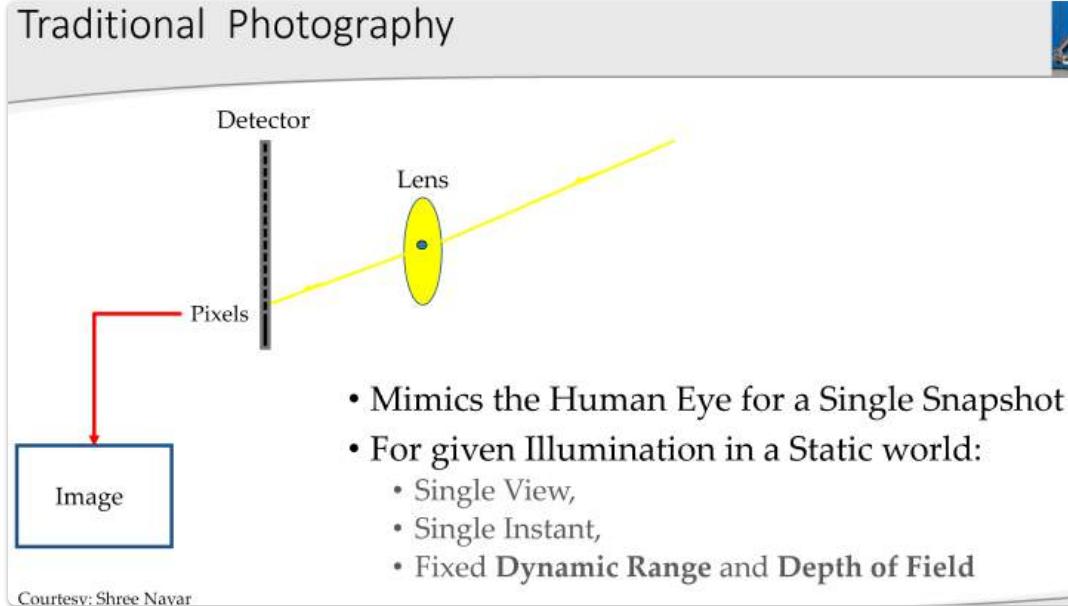
Der Sensor summiert einfallende Lichtstrahlen aus allen Einfallswinkeln zu einer Intensität.

*Informationen über die Reflexionseigenschaften von Szenenoberflächen (charakterisiert durch die Bidirectional Radiance Distribution Function, kurz BRDF) gehen verloren.*

Lassen sich aus der Bestrahlungsstärke auf der Bildebene nicht die strahlungsdichten Objekte bestimmen.

\* Die **Objektreffektivität** und der **unbekannten Bestrahlungsstärke** des Objekts werden zusammengesetzt dargestellt, was die Identifizierung und Klassifizierung von Objekten erschwert ("Helligkeits-Dilemma").

### Traditional Photography



## Zusätzliche optische Komponenten und deren Vorteile

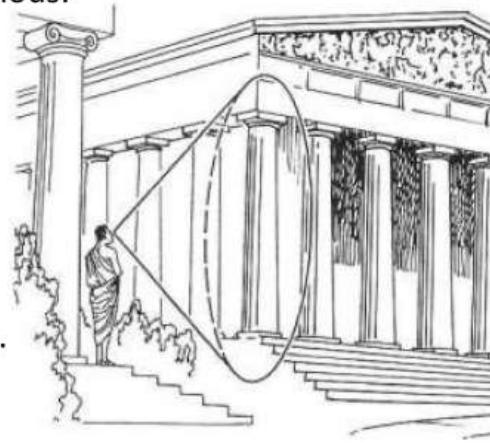
- **Abbildung in Kombination mit einer Blende zur Reduktion der Tiefunschärfe:**
  - Führt zu einem Verlust brauchbarer Informationen für weite Tiefenbereiche der Szene, da nur ein kleiner Bereich scharf abgebildet wird.
  - Verfügbare Sensoren integrieren über ein breites Spektrum der Wellenlängen einfallendes Licht.
  - Messung des spektralen Dimensions von Oberflächen (z.B. Farben) an unterschiedlichen Orten.
  - Zeit (Belichtungszeit) wird ebenfalls in die Dimension des Lichts integriert (zeitliche Auflösung geht verloren).
  - Räumliche Auflösung und die Quantisierung zu einer begrenzten Helligkeitsauflösung sind ebenfalls reduziert.
  - Dies führt zu einer **Reduktion des Signalumfangs (Dynamic Range)**.

- Core ideas are ancient, simple, and seem obvious:

- **Lighting:** ray sources
- **Optics:** ray bending / folding devices
- **Sensor:** measure light
- **Processing:** assess it
- **Display:** reproduce it

- **Ancient Greeks:**

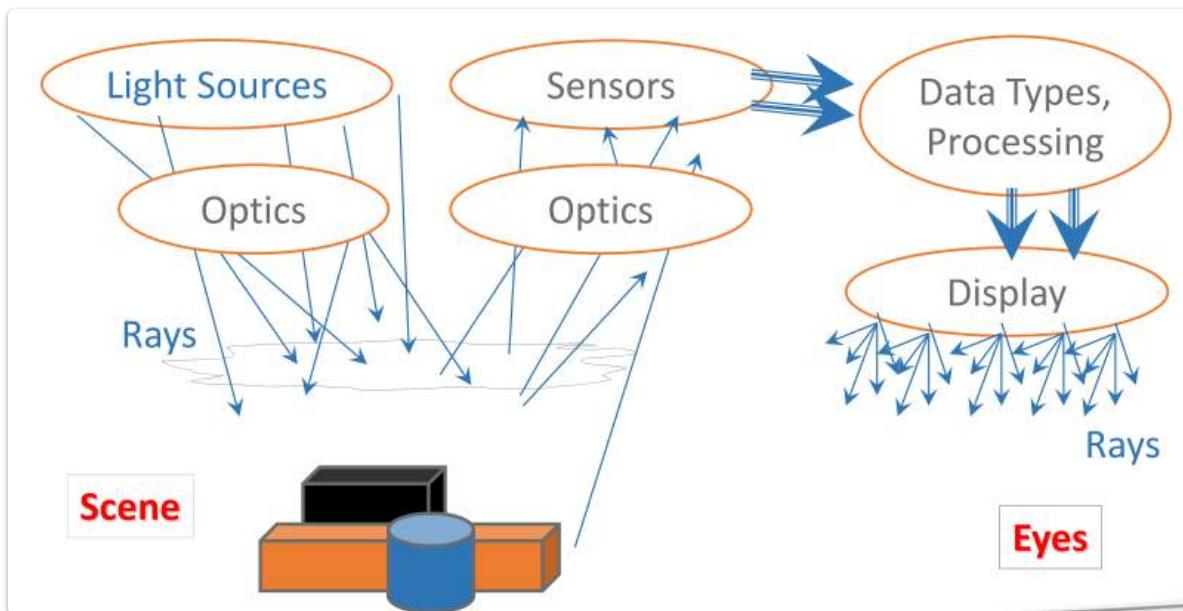
'Eye rays' wipe the world to feel its contents...



## 12.2 Lichtfeld

[EVC\\_Skriptum\\_CV](#), p.60, p.61

Das Lichtfeld ist eine Funktion, die die **Lichtmenge** beschreibt, die an jedem Punkt des dreidimensionalen Raums aus allen Richtungen einfällt.



- In der Computational Photography betrachtet man das Lichtfeld, das auf einen Punkt fällt, d.h., **welche Strahlen an diesen Punkt reflektiert werden**.
- **Objekte** werden als **Volumen** betrachtet.
- Einfallendes Beleuchtungsfeld wird in ein Beleuchtungsfeld umgewandelt, welches vom Objekt reflektiert wird.

### 4D-Lichtfeld-Parametrisierung

[EVC\\_Skriptum\\_CV](#), p.61

Einfallende Beleuchtung kann mittels eines **4D-Lichtfelds** parametrisiert werden.



- Man stellt sich um das Objekt eine Kugel vor, die das Objekt einschließt.
- **Position auf der Kugel:**  $(u_i, v_i)$
- **Richtung des eintreffenden Lichts:**  $(\theta_i, \phi_i)$
- **Einfallendes Lichtfeld:**  $R_i(u_i, v_i, \theta_i, \phi_i)$

Das ausgehende Licht kann ebenfalls mittels eines **4D-Lichtfelds** parametrisiert werden:

- **Was das Licht die umgebende Oberfläche verlässt:**  $R_r(u_r, v_r, \theta_r, \phi_r)$

Hier gabs dann noch ein Recap zur [PLenopischen Funktion: EVC-CV12-Computational Photography\\_S2025\\_Slides, p.11](#)

## Reflexionsfeld (8D-Funktion)

Die **Reflexionseigenschaften eines Objekts** können mittels einer **achtdimensionalen Funktion** - dem **Reflexionsfeld** - charakterisiert werden.

- Das Reflexionsfeld codiert für jeden einfallenden Lichtstrahl die 4D Reflexion des Lichtstrahls.
- Es beinhaltet die nötigen Informationen, um das Objekt unter verschiedenen Beleuchtungsgegebenheiten und von verschiedenen Blickwinkeln aus zu rendern.

## 4D-Lichtfeld-Kameras (Plenoptische Kameras)

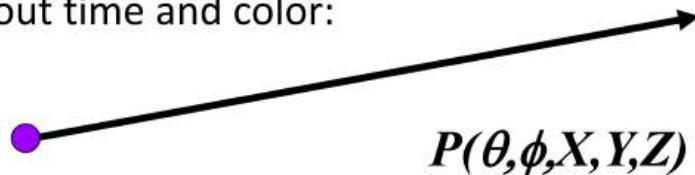
4D-Lichtfeld-Kameras sind nicht nur in der Lage, Ort und Intensität der Projektion der von einer Szene ausgehenden Lichtstrahlen zu detektieren, sondern auch den **Einfallswinkel** auf der Kamera-Ebene zu quantifizieren.



- Die so gewonnene **4D-Abtastung des Lichtfelds** einer Szene ermöglicht einen vielfältigen Informationsgewinn.
- **Aufbau:** 4D-Lichtfeld-Kameras verwenden ein **Mikrolinsen-Array**, das direkt auf dem Sensor aufgebracht wird.
- **Funktionsweise:** Die eintreffenden Lichtstrahlen werden durch die einzelnen Pixel im Sensorbereich unter jeder Mikrolinse nach ihrem Einfallswinkel quantifiziert, um ein 4D-Lichtfeld einer Szene zu rekonstruieren.
- **Nachteil:** Die direkt erreichbare **Ortsauflösung** entspricht der **Anzahl der Mikrolinsen**. Deren Durchmesser (in Pixeln) bestimmt wiederum die **Winkelauflösung**.
  - Je größer die Winkelauflösung, desto geringer ist die Ortsauflösung und umgekehrt.

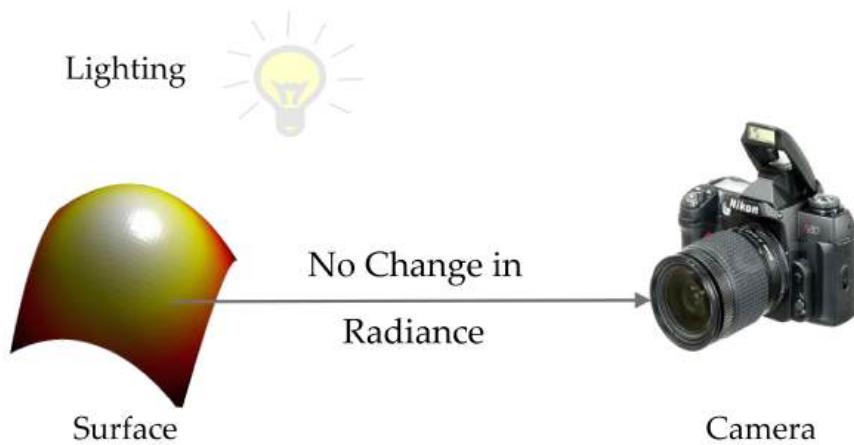
## Ray

- Let's not worry about time and color:



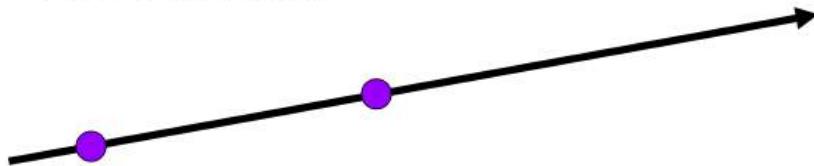
- 5D
  - 3D position
  - 2D direction

## How can we use this?



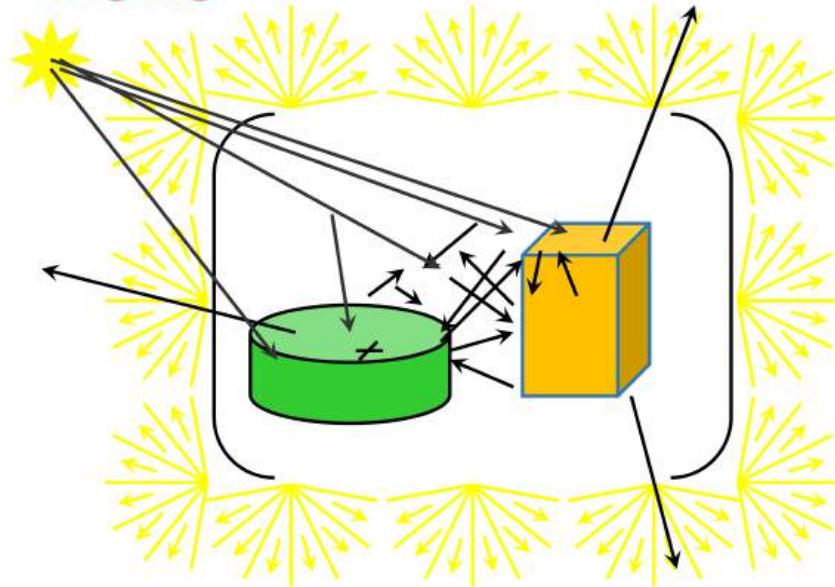
## Ray Reuse

- Infinite line
  - Assume light is constant (vacuum)



- 4D
  - 2D direction
  - 2D position
  - Non-dispersive medium

- Measure all the **outgoing** light rays.



$(u_i, v_i)$  indicate the position on the surface where the light enters,  
 $(\theta_i, \phi_i)$  indicate the direction in which it enters.

$(u_r, v_r)$  indicate the position on the surface where the light leaves,  
 $(\theta_r, \phi_r)$  indicate the direction in which it leaves.



$$(u_i, v_i, \theta_i, \phi_i ; u_r, v_r, \theta_r, \phi_r)$$

8D reflectance field

Since it is linear, we can represent as a matrix

$$R(u_i, v_i, \theta_i, \phi_i; u_r, v_r, \theta_r, \phi_r)$$

$360 \times 180 \times 180 \times 180 \times 360 \times 180 \times 180 \times 180$

= 4.4e18 measurements

x 6 bytes/pixel (in RGB 16-bit)

= 26 exabytes (billion GB)

= 1.3 million 20TB hard drives



(für nur ein Bild)

## High Dynamic Range (HDR)

[EVC\\_Skriptum\\_CV](#), p.61

**High Dynamic Range (HDR)** beschreibt eine Fülle an Techniken, die im Vergleich zu herkömmlichen Techniken für digitale Bildgebung oder fotografischen Methoden einen **größeren Bereich zwischen den hellsten und dunkelsten Regionen eines Bildes** darzustellen ermöglichen.

- **Ziel von HDR-Bildern:** Den Helligkeitsbereich in realen Szenen genauer zu repräsentieren.
- **Möglicher Helligkeitsbereich:** Erstreckt sich von direktem Sonnenlicht bis hin zu schwachem Sternenlicht.
- **Erzeugung:** Erzielt durch die Aufnahme von mehreren Bildern mit unterschiedlicher Belichtung der gleichen Szene.
- **HDR kompensiert Detailverlust**, der bei Nicht-HDR-Kameras mit einem einzigen Belichtungslevel auftritt (dort erhält man ein Bild mit Detailverlust in hellen oder dunklen Bildbereichen).
- **Ergebnis:** Ein Bild, dessen dunkle als auch helle Bildbereiche im Detail darstellbar sind.



## Anzeige von HDR-Bildern auf Geräten mit kleinem dynamischen Bereich

Um HDR-Bilder auf Geräten mit einem kleinen dynamischen Bereich anzuzeigen, werden **Techniken zur Abbildung von Farbtönen (engl. Tone Mapping)** benötigt.

- **Ziel von Tone Mapping:** Den gesamten Kontrast reduzieren.
- **Grund:** Ausdrücke, CRT- oder LCD-Monitore und Projektoren haben einen begrenzten dynamischen Bereich, der für die Darstellung der vollen Bandbreite der Lichtintensitäten ungeeignet ist.
- **Funktionsweise:** Tone Mapping behandelt die **Kontrastreduktion** der Lichtintensitäten der Szene zu den anzeigbaren Wertebereichen.
- **Wichtige Aspekte dabei:** Bilddetails und die Farbwirkung sollen möglichst erhalten bleiben, um den originalen Szeneninhalt wahrnehmen zu können.

Beispiel zu ToneMapping:

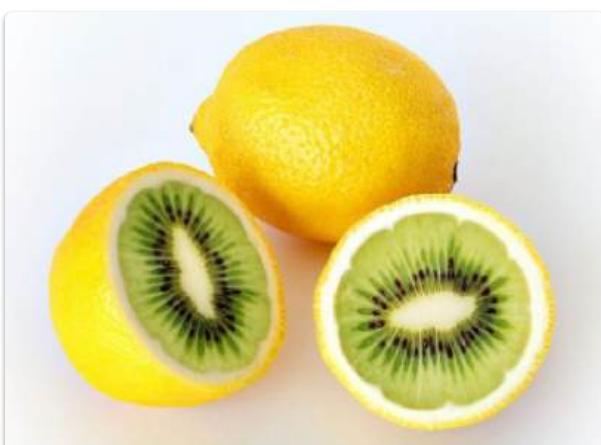


## Fotomontage/Komposition

EVC\_Skriptum\_CV, p.61

Fotomontage (auch bekannt als "Komposition" oder "image compositing") bezeichnet das Schneiden und Zusammenfügen von verschiedenen Fotos zu einem zusammengesetzten Bild.

- **Realisierung:** Heutzutage mittels moderner Bildbearbeitungssoftware.
- **Ziel:** Visuelle Elemente von unterschiedlichen Quellen werden in einem einzigen Bild kombiniert, um die **Illusion zu schaffen, dass alle diese Elemente Teil derselben Szene sind.**





- Bei Filmaufnahmen ist die Komposition unter verschiedenen Bezeichnungen bekannt:
  - "chroma key"
  - "blue screen"
  - "green screen" und andere Bezeichnungen.

## Image Inpainting

[EVC\\_Skriptum\\_CV](#), p.62

**Inpainting** ist ein Bildbearbeitungsprozess, der bei der **Rekonstruktion von schlecht erhaltenen oder nicht vorhandenen Bild- oder Videoteilen** Anwendung findet.

- **Analogie zur Malerei:** Im Bereich der Malerei wäre dies die Arbeit eines ausgebildeten Bildrestaurators.
- **In der digitalen Welt (auch bekannt als Bild- oder Videointerpolation):** Bezieht sich auf die Anwendung hochentwickelter Algorithmen, um **verlorene oder fehlerhafte Teile von Bilddaten zu ersetzen**.



- **Anwendungsbeispiele:**
  - **Entfernung eines Objekts aus einer Szene**, um ein beschädigtes Bild zu retuschieren.
  - **Für Fotografie und Film:**
    - Beseitigung von Zerfallsmerkmalen (z.B. Risse in Fotografien, Kratzer und Staubflecken im Film).
    - Hinzufügen oder Entfernen von Bildteilen (z.B. Aufnahmedatum).
- **Generelles Ziel:** Ein verändertes Bild zu produzieren, in dem die Inpaint-Region mit dem restlichen Bild so verschmolzen wird, dass ein gewöhnlicher Betrachter **nicht bemerkt**,

dass das Bild bearbeitet wurde.

## Warping

EVC\_Skriptum\_CV, p.62

**Warping** ist der Prozess der **Bildmanipulation**, bei dem jede im Bild vorkommende Form **räumlich signifikant verändert** wird.

Image filtering: change **range** of image

$$g(x) = T(f(x))$$

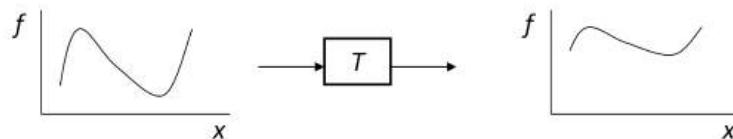


Image warping: change **domain** of image

$$g(x) = f(T(x))$$

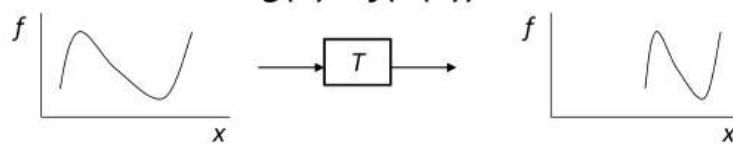


Image warping: change **domain** of image



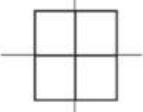
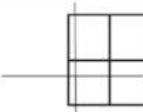
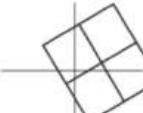
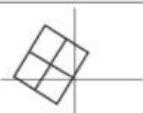
$$g(x) = f(T(x))$$



- **Anwendung:**

- **Verzerrung von Bildern korrigieren.**
- **Kreative Zwecke** (z.B. für Morphing).

- **Funktionsweise:** Beim reinen Warping wird der **Punkt auf einen anderen Punkt abgebildet, ohne die Farbe des Punktes zu ändern**.
- **Grundlage:** Warping kann auf jeder **Funktion** basieren, die (von einem Teil) einer Ebene auf eine Ebene abbildet.
- **Injektive Funktion:** Bei einer injektiven Funktion kann das Original wieder rekonstruiert werden.
- **Bijektive Funktion:** Wenn die Funktion bijektiv ist, kann das Bild invers transformiert werden.

<b>Identity</b>	$W(x) = x$	
<b>Translation</b>	$W(x; t) = x - t$	
<b>Rigid</b>	$W(x; R, t) = Rx - t$	
<b>Similarity</b>	$W(x; R, a, t) = aRx - t$	
<b>Affine</b>	$W(x; A, t) = Ax - t$	

where  $\alpha \in \mathcal{R}$ ,  $A \in \mathcal{R}^{2 \times 2}$ ,  $t \in \mathcal{R}^2$ ,  $R \in \mathcal{R}^{2 \times 2}$ ,  $|R| = 1$

- **Mathematische Beschreibung:**

- Gegeben sei ein gesampeltes Bild  $I(x)$ .
- **Warping** ist die **räumliche (geometrische) Deformation** des Quellbildes  $I_s(x)$ .
- Das Ergebnis  $I_d(x)$  wird **Zielbild** genannt.
- Das Warping wird mittels der **Warpingfunktion**  $W(x; p)$  mit den Parametern  $p$  bestimmt.
- Für jedes Pixel  $x$  in  $I_d$  gibt uns die Warpingfunktion an, woher dieses Pixel im Quellbild  $I_s$  stammt:  $I_d(x) = I_s(W(x; p))$ .

- **Zwei Formen von Transformationen:**

- **Affine Transformationen**
- **Projektive Transformationen** (auch perspektivische Transformation oder Homographie genannt).

## Affine Transformation

Die affine Transformation ist eine Transformation, die **gerade Linien und Distanzverhältnisse zwischen Punkten, die auf einer Linie liegen, erhält**.

- **Beispiel:** Der Mittelpunkt eines Liniensegments bleibt auch nach der Transformation dessen Mittelpunkt.
- **Wird nicht zwingend erhalten:** Längen und Winkel.
- **Typische affine Transformationen:** Translation (Verschiebung), Rotation (Drehung), Scherung und ähnliche Transformationen.
- **Kombinationen** von affinen Transformationen sind ebenfalls affin.

$$\begin{bmatrix} x' \\ y' \\ w \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$

## Projektive Transformation

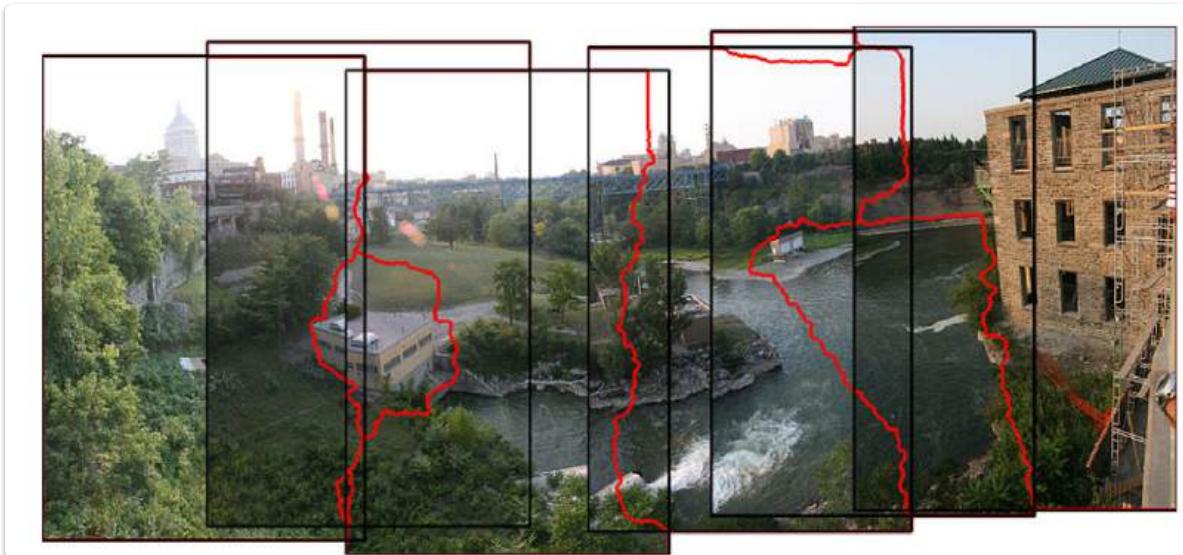
Die projektive Transformation einer Ebene ist eine Transformation, die in der **projektiven Geometrie** verwendet wird.

- **Wird nicht erhalten:** Größen oder Winkel.
- **Anwendung:** Jedes Bildpaar, das dieselbe ebene Fläche im Raum zeigt, steht mittels einer projektiven Transformation zueinander in Beziehung.
- **Praktische Anwendungen:**  
*Bildrektifizierung (Entzerrung von Bildern).*  
Bildregistrierung (Ausrichtung von Bildern zueinander).  
\* Berechnung der Kamerabewegung zwischen zwei Bildern.

$$\begin{bmatrix} x' \\ y' \\ w \end{bmatrix} = \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} \begin{bmatrix} x \\ y \\ w \end{bmatrix}$$

## Bildmosaik (Image Mosaicing / Stitching)

EVC\_Skriptum\_CV, p.63



Der Prozess zur Erstellung eines **Bildmosaiks** (auch oft als **Stitching** bezeichnet) kombiniert mehrere Bilder mit überlappenden Bereichen zu einem **hochauflösenden Panoramabild**.

- **Voraussetzung für korrekte Überlagerung (ohne Parallaxenfehler):** Die Kamera muss um den **Fokuspunkt** rotiert werden.
- **Der Prozess kann in drei Bestandteile aufgeteilt werden:**
  1. **Bildregistrierung**
  2. **Kalibrierung**

### 3. Blending

## 1. Bildregistrierung

Bei der Bildregistrierung werden **korrespondierende lokale Merkmale** zwischen den Bildern bestimmt, um die Bilder korrekt "übereinander legen" zu können.

## 2. Kalibrierung

Die Kalibrierung versucht, die **Differenzen zwischen einem idealen Linsenmodell und dem realen Linsensystem der Kamera(s)** zu minimieren. Das bedeutet, sie korrigiert:

- Optische Defekte (wie **Verzerrungen**).
- Unterschiedliche **Belichtungszeiten** der Bilder.
- **Vignettierung** (Randabdunkelung).
- **Chromatische Aberrationen** (Farbsäume).

## 3. Blending ("Verschmelzen")

Das Blending korrigiert die im Kalibrierungsschritt festgemachten Abweichungen und **bildet die einzelnen Aufnahmen auf ein gemeinsames Ausgabebild ab**.

- **Farben werden zwischen den Bildern angepasst**, um die Beleuchtungsunterschiede auszugleichen.
- Die Bilder müssen so miteinander verschmolzen werden, dass **keine Übergänge (engl. seams)** von einem Einzelbild zum nächsten sichtbar sind.

## Image Morphing

---

[EVC\\_Skriptum\\_CV](#), p.63

**Image Morphing** ist ein spezieller Bildeffekt, der ein Bild **nahtlos in ein anderes verwandelt**.

- **Anwendung:** Oft in surrealen Sequenzen oder im Fantasyfilm-Genre verwendet, um z.B. eine Person in eine andere zu verwandeln.
- **Traditionelle Methode:** Durch Aus- und Einblendtechniken (engl. *fading*) im Film erzielt.
- **Seit den frühen 1990ern:** Immer häufiger Computertechniken verwendet, die überzeugendere Ergebnisse lieferten.



- **Heutige Methode (Computertechniken):** Beim Überblenden werden die Bilder gleichzeitig anhand von **markierten korrespondierenden Punkten verzerrt**.
- **Beispiel (Gesichts-Morphing):**
  - Umwandlung eines Gesichts in ein anderes.
  - **Keypoints** im ersten Gesicht (z.B. die Kontur der Nase oder Position der Augen) werden markiert. Diese Keypoints müssen auch im zweiten Bild existieren.
  - Anschließend wird das erste Gesicht **verzerrt**, um die Form des zweiten Gesichts zu erhalten.
  - **Gleichzeitig** wird das erste Gesicht **ausgeblendet** und das zweite Gesicht **eingebettet**.
- **Höher entwickelte Überblendungstechniken (später):** Verschiedene Teile eines Bildes werden graduell auf das andere überblendet, anstatt das gesamte Bild auf einmal zu wandeln.

### Idea #1: Cross-Dissolve



- Interpolate whole images:  

$$\text{Image}_{\text{halfway}} = (1-t) * \text{Image}_1 + t * \text{image}_2$$
- This is called **cross-dissolve** in the film industry
- But what if the images are not aligned?

- Align first, then cross-dissolve
  - Alignment using global warp – picture still valid



## Testähnliches Beispiel

- Benennen Sie die dargestellten Verfahren der Computational Photography:

Eingabebild(er)	Ergebnis	Verfahren
		<u>Image Inpainting</u>
		<u>Image Morphing</u>
		<u>Panoramic Images</u>
		<u>HDR – High Dynamic Range Imaging</u>