Echtzeit–Videoverarbeitung

Codierung
Überblick und Bewegungsschätzung

Paul Berschin

Überblick

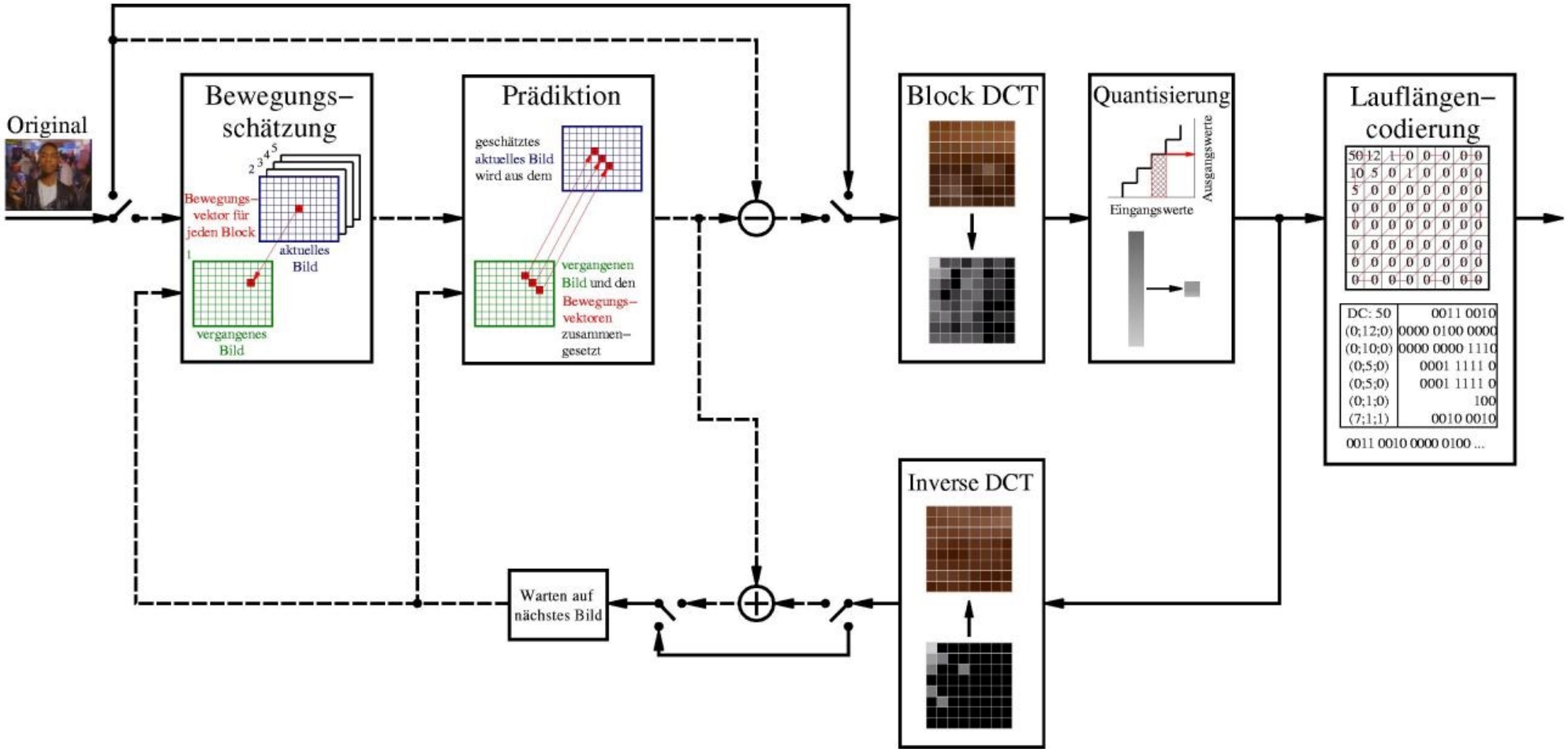
Vorstellung der Kernelemente am Beispiel ITU H.263

- Transformation
- Quantisierung
- Lauflängencodierung

Bewegungsschätzung

- Prinzip
- Bewertungskriterien
- Verfahren
- Codierungseffizienz

Videocodierung nach ITU H.263



Videocodierung nach ITU H.263 (2)

Farben sind unterabgetastet gemäß 4:2:0

5 Bildformate:

Format	Lum. Breite	Lum. Höhe	Cr. Breite	Cr. Höhe
SQCIF	128	96	64	48
QCIF	176	144	88	72
CIF	352	288	176	144
4CIF	704	576	352	288
16CIF	1408	1152	704	576

Einteilung des Bildes in Makroblöcke:
16 x 16 Pixel Luminanz (4 Blöcke) und je
8 x 8 Pixel Farbdifferenzsignal C_b und C_R.

Transformation

Blockbasierte Diskrete Cosinus Transformation (DCT)

Dekorrelierte, spektrale Repräsentation des Bildsignals

Blockgröße 8x8

$$DCT_{uv} = \frac{2 \cdot C_u \cdot C_v}{N} \cdot \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} f(i,j) \cdot \cos\left[\frac{\pi \cdot u \cdot (2 \cdot i + 1)}{2 \cdot N}\right] \cdot \cos\left[\frac{\pi \cdot v \cdot (2 \cdot j + 1)}{2 \cdot N}\right]$$

f(i,j): Wert des Pixels an der Stelle (i,j)

$$C_u = \begin{cases} \frac{1}{\sqrt{2}} & \text{falls } u = 0 \\ 1 & \text{sonst} \end{cases}, \quad C_v = \begin{cases} \frac{1}{\sqrt{2}} & \text{falls } v = 0 \\ 1 & \text{sonst} \end{cases}$$

Quantisierung

31 Quantisierungsstufen
gleichförmige Stufenhöhe
gleicher Quantisierer für alle Koeffizienten
unterschiedlicher Quantisierer je nach Typ des Koeffizienten:
Quantisierer für INTER Koeffizienten:

$$|LEVEL| = (|COF| - QUANT / 2) / (2 \cdot QUANT)$$

Quantisierer für INTRA non−DC Koeffizienten

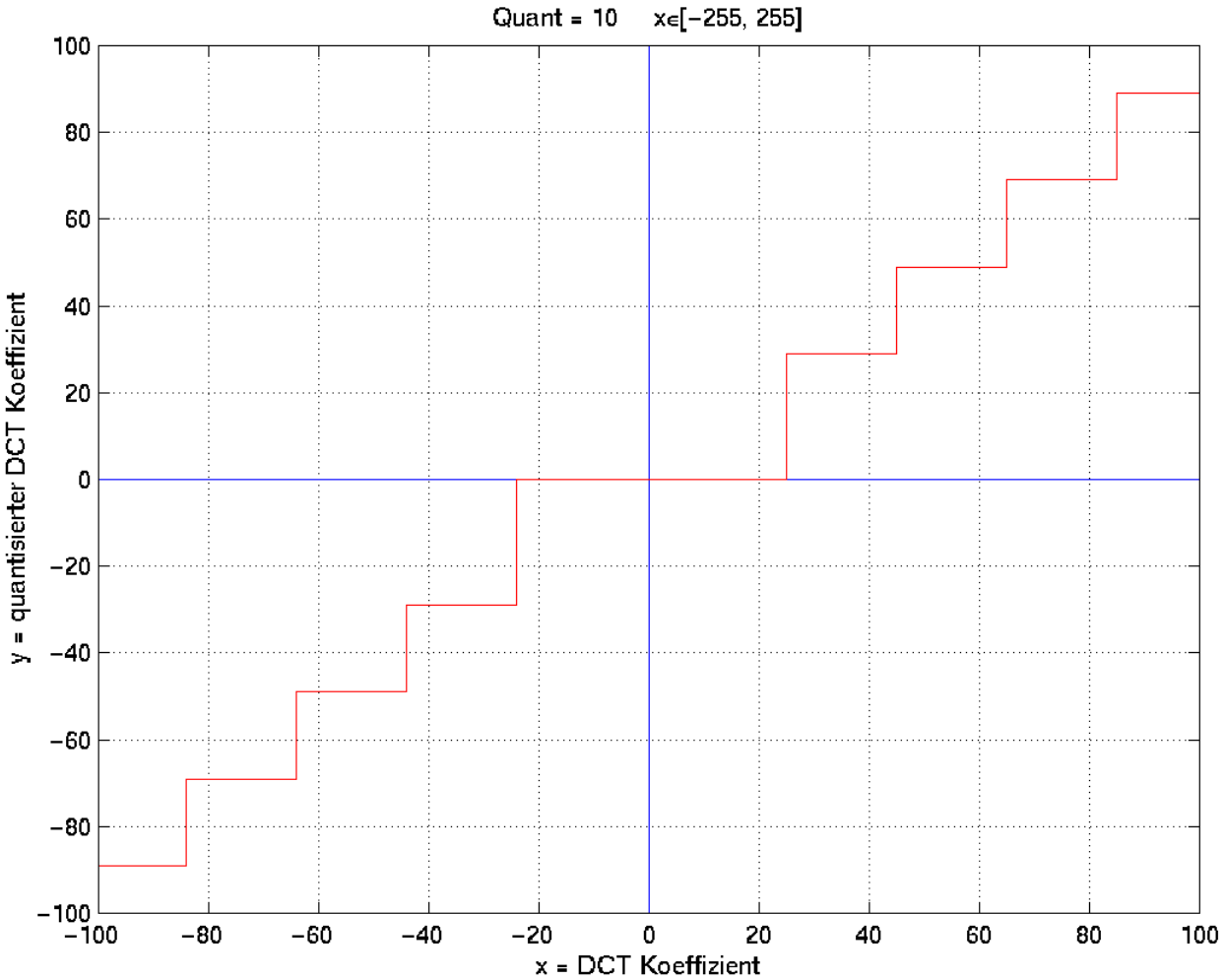
$$|LEVEL| = |COF| / (2 \cdot QUANT)$$

Quantisierer für INTRA DC Koeffizienten

$$LEVEL = (COF + 4) / (2 \cdot 4)$$

Hier entsteht der „Verlust“ bei verlustbehafteter Codierung

Quantisierung

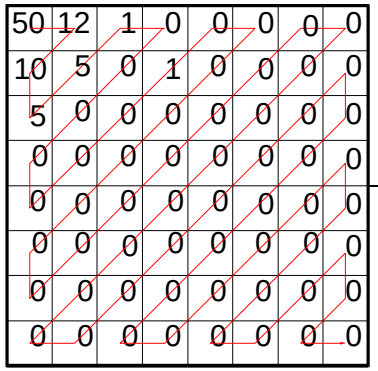


Lauflängen−und Huffmancodierung

Nach Transformation und Quantisierung sind nur wenige Koeffizienten ungleich 0.

- Zig−Zag−Scan
- Lauflängencodierung in run−level−last
- Entropiecodierung mit Huffman Tabellen

Lauflängen−und Huffmancodierung



50, 12, 10, 5, 5, 1, 0, 0, 0, 0, 0, 0, 0, 1, 0, 0, 0, . . .

(0,50)(0,12)(0,10)(0,5)(0,5)(7,1) LAST

(7;1)	1010
	.
(0;5)	10101
(0;10)	10110010
(0;12)	10110011
(0;50)	10110100

Bitstream → 1011010010110011...

Bewegungskompensation

Dient zum Auffinden translatorischer Bewegungen

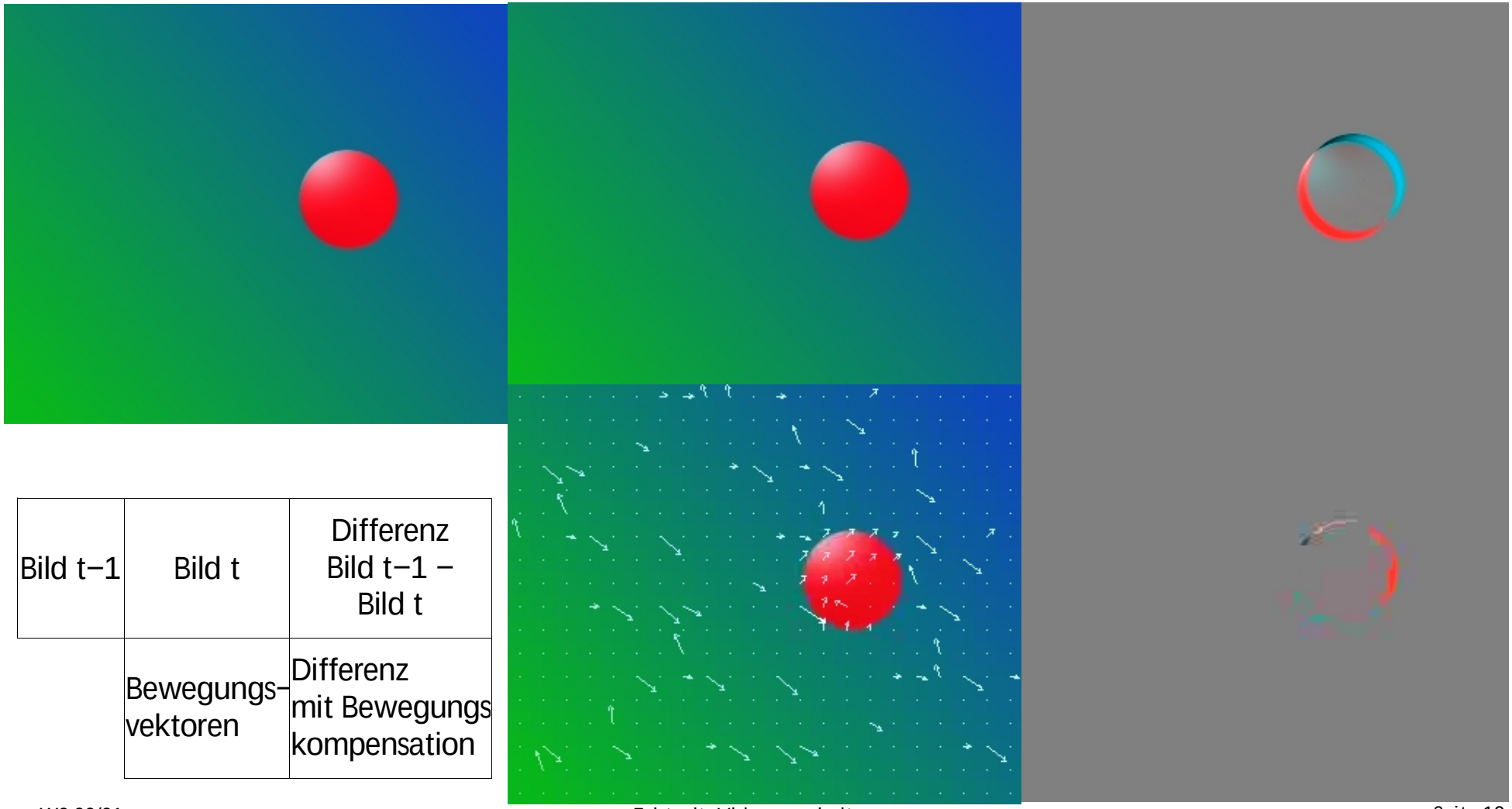
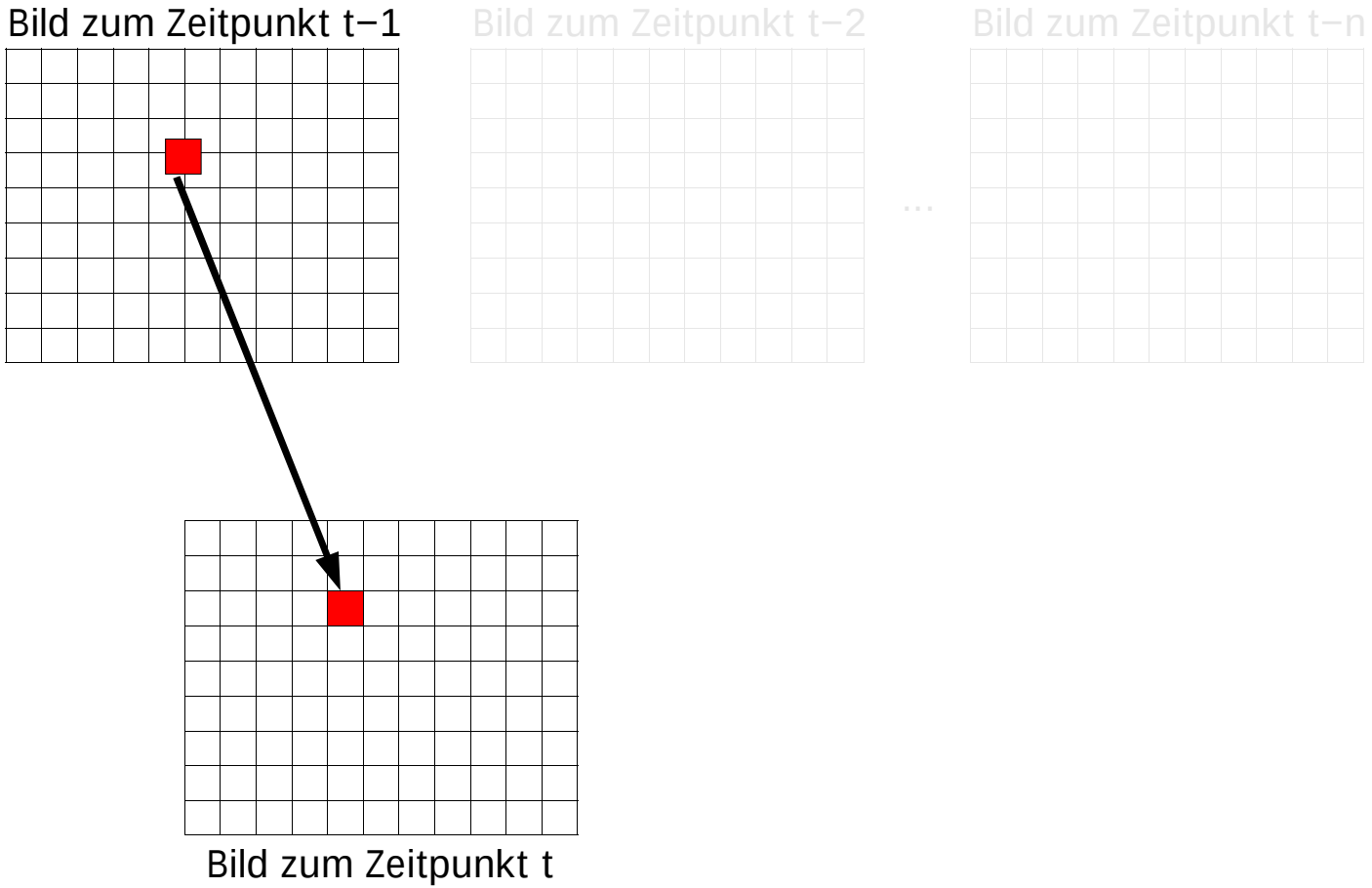
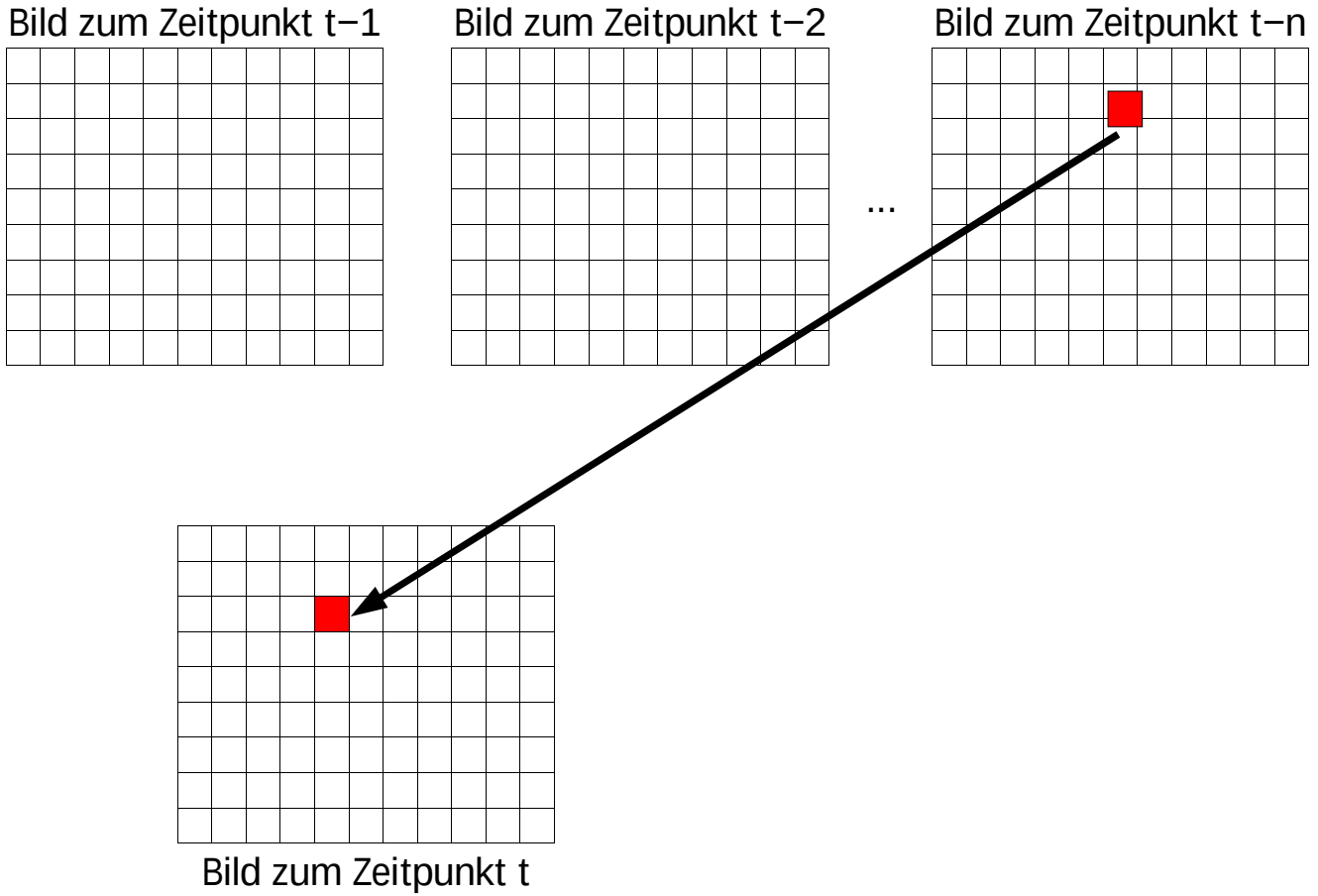


Bild t−1	Bild t	Differenz Bild t−1 − Bild t
	Bewegungs-vektoren	Differenz mit Bewegungs kompensation

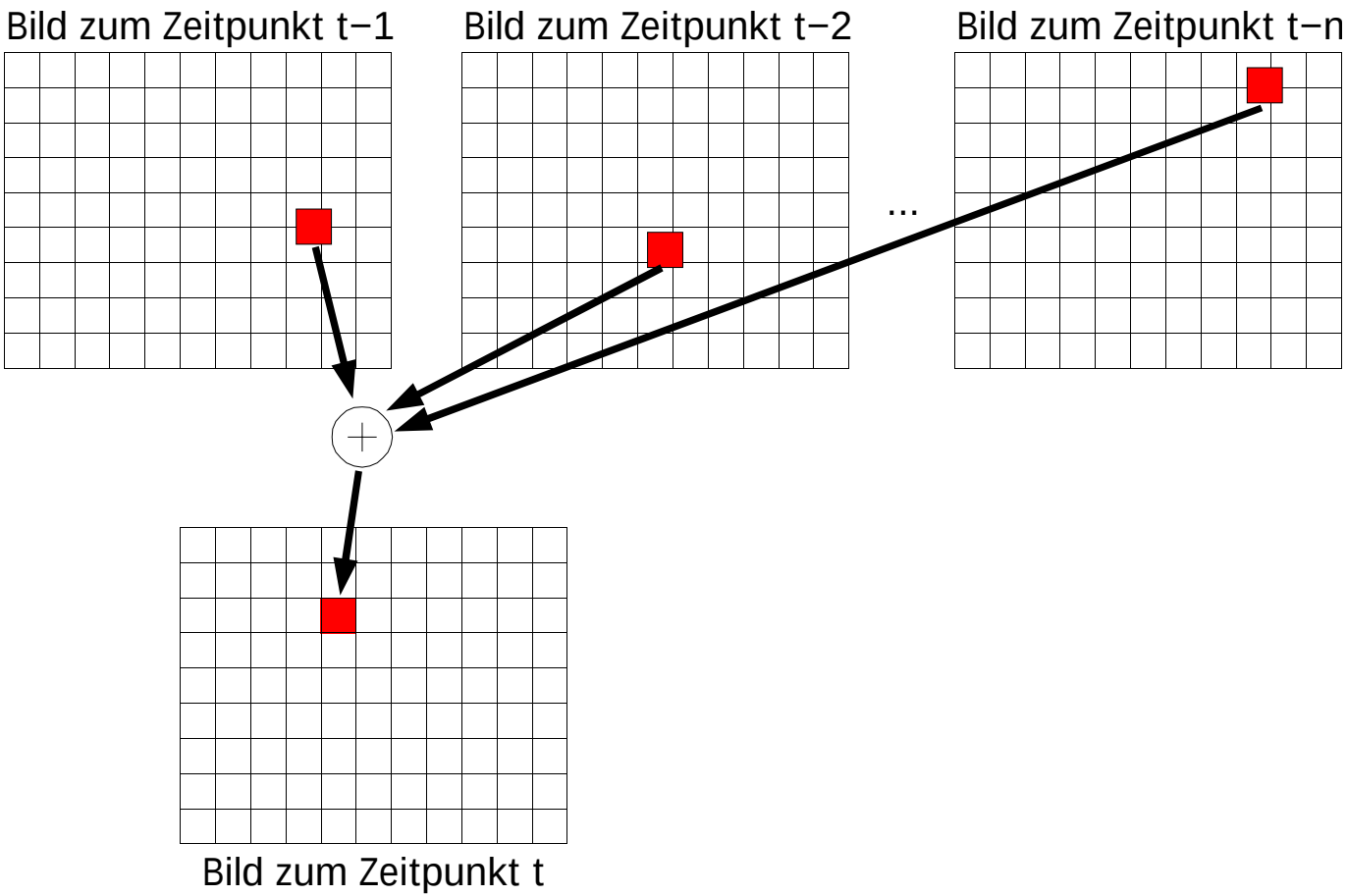
Einfache Bewegungsschätzung



Long−Term Motion Estimation



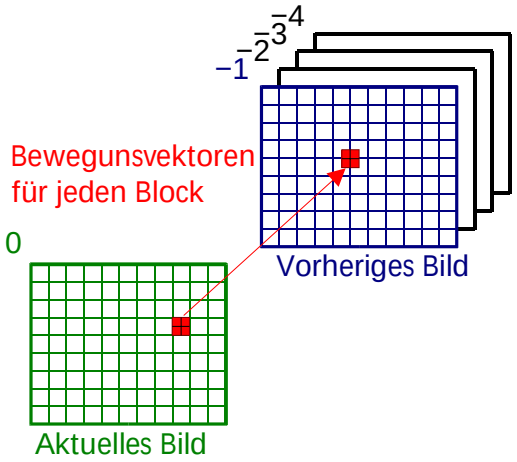
Multi–Hypothesis Coding



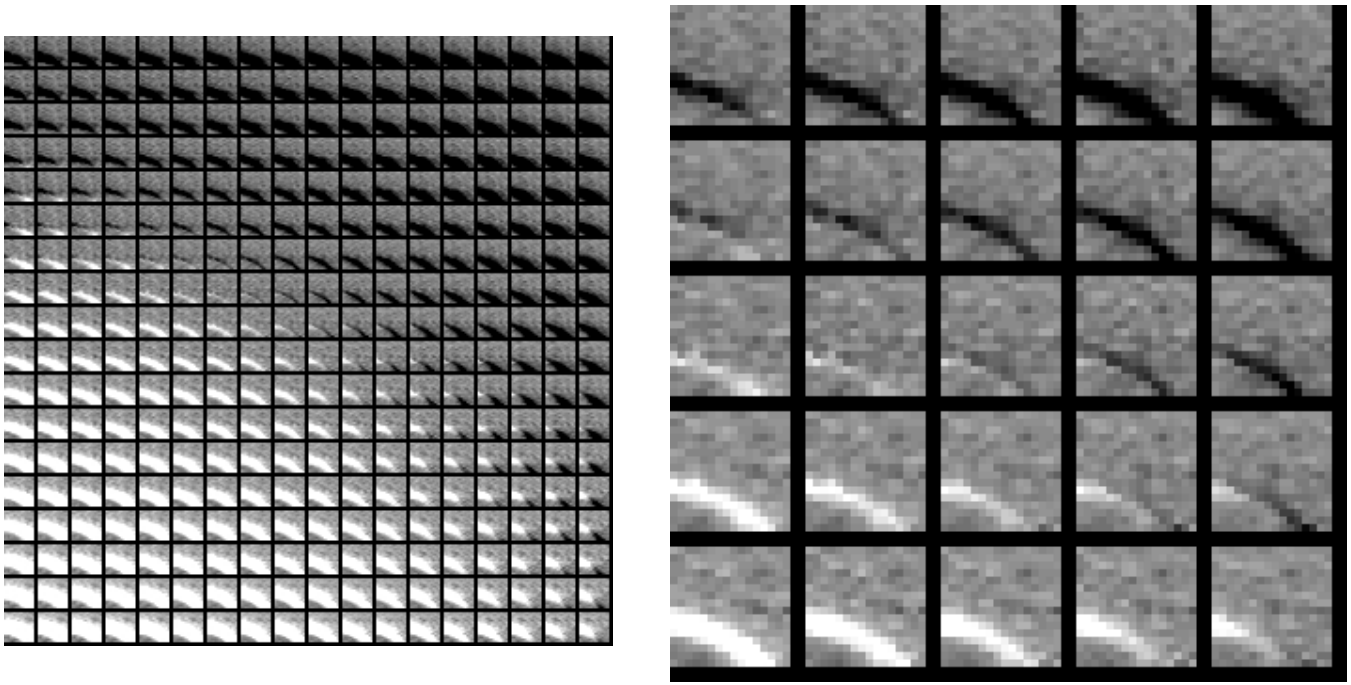
Bewegungsschätzung (Motion Estimation ME)

Keine Bewegungsdetektion im Sinne von bewegten Objekten sondern Blockmatching:

- Das aktuelle Bild wird in gleich große Blöcke aufgeteilt, üblicherweise 16x16 Pixel (Makroblock)
- Der Bewegungsvektor wird dadurch festgelegt, daß im vorhergehenden Bild der Bereich gefunden wird, der das Bewertungskriterium minimiert.
- Die Suche findet gewöhnlich nur auf der Luminanz statt.



Bewegungsschätzung (Motion Estimation ME)



Suchbereich 9x9 in Half−Pel Genauigkeit

Bewegungsschätzung

Probleme translatorischer Bewegungskompensation:

- Zoom
- Rotation
- Szenenschnitt
- Helligkeitsänderung

Bewegungsschätzung

Bei H.263 wegen Ursprung in der Videotelefonie:

- eingeschränkter Suchbereich
- nur Bezug auf das letzte Bild

Erhöhung der Auflösung durch Sub−Pel Genauigkeit, indem das Bild linear interpoliert wird:

- Half−Pel: Im H.263 verwendet
- Quarter−Pel

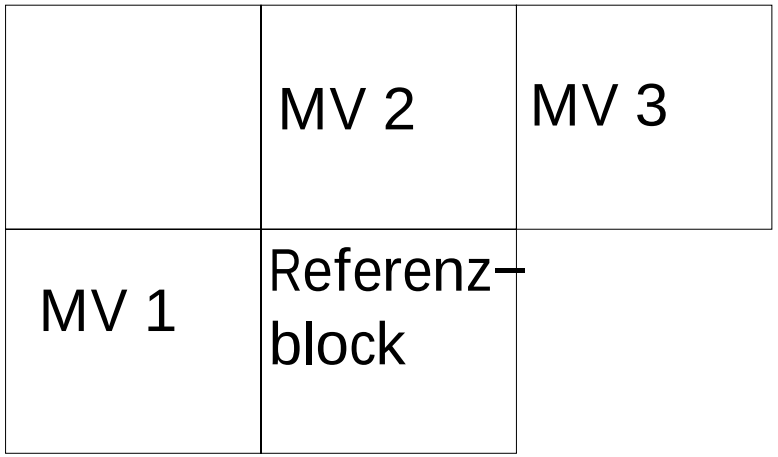
Nur die unmittelbare Umgebung des mittels Full−Pel Suche gefundenen Zielorts wird mit Sub−Pel Genauigkeit untersucht.

Codierung der Bewegungsvektoren

Je länger der Bewegungsvektor, um so unwahrscheinlicher, daher mehr Bits für lange Bewegungsvektoren nötig.

Effiziente Codierung der Bewegung ausgedehnter starrer Körper:

Aus den bisher gefundenen Bewegungsvektoren von drei Blöcken in der Nachbarschaft des Referenzblocks wird durch Medianverfahren komponentenweise (x,y) ein „predicted candidate“ Bewegungsvektor gebildet; der gefundene Bewegungsvektor wird als Differenz zu diesem predicted candidate übermittelt.



Bewertungskriterien für Bewegungsschätzung

Mean Squared Error (MSE):

$$MSE(m_x, m_y) = \sum_{j=0}^{N} \sum_{i=0}^{N} (f_{t-1}(i+m_x, j+m_y) - f_t(i,j))^2$$

$f_t(i,j)$: Luminanzwert an der Stelle i, j

Minimiert Energie des Fehlerbildes, überall verwendet, aber:

1 Subtraktion, 1Multiplikation, 1 Addition pro Pixel

Sum of Absolute Differences (SAD):

$$SAD(m_x, m_y) = \sum_{j=0}^{N} \sum_{i=0}^{N} \left| f_{t-1}(i+m_x, j+m_y) - f_t(i,j) \right|$$

1 Addition und 1 Subtraktion mit Betragsbildung pro Pixel

Full Search

Verfahren:

Berechnung der SAD für jedes Pixel im Suchbereich

Vorteile:

- einfache Berechnungsvorschrift
- konstante Ausführungszeit
- Findet mit Sicherheit das absolute Minimum

Nachteile:

- Langsam:

  $N^2 \cdot (2w+1)^2$ Operationen pro Block

  N: Blockgröße

  W: Suchbereich

  z.B. für Suchbereich ±16 und Makroblock

  je $16^2 \cdot (2 \cdot 32+1)^2 = 278784$ Additionen Subtraktionen Betragsbildungen

Full Search with Treshold (1)

Verfahren:

Wie Full−Search, allerdings wird die Berechnung des SAD abgebrochen, wenn ein Schwellwert überschritten wurde. Falls ein besseres SAD gefunden wird, als der Schwellwert, wird dieser ersetzt.
Meist Vergleich des SAD mit dem Schwellwert nach Addition jeder Zeile des aktuellen Blocks

Vorteile:

- Einfache Berechnungsvorschrift
- schneller als Full Search
- Findet mit Sicherheit das absolute Minimum

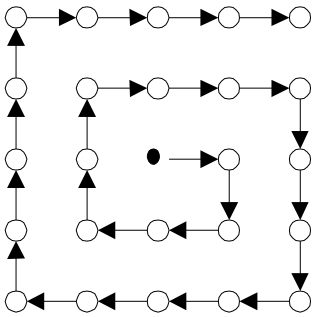
Nachteil

- Immer noch langsam
- Geschwindigkeit hängt stark vom Startwert und Gradienten ab

Full Search with Threshold (2)

Verbesserungen:

- Vor Start der Suche SAD – Bildung am Nullpunkt und am „predicted candidate“ um von Beginn an einen guten Schwellwert zu erhalten
- a–priori setzen des Schwellwerts auf einen sinnvollen Wert (z.B.: 800 – 2000)
- Spiral Search



Nachteil:

- Aufwendige Implementierung

Successive Elimination (1)

Idee:

- Schnelle Entscheidung, ob aktueller Block besseres SAD erreichen kann, als bereits gefunden wurde

Ansatz:

- Betragsungleichung: $\big||a|-|b|\big| \le |a-b|$

Für ein beliebiges Pixel an der Stelle (i,j) gilt:

$$\Big|\,|f_t(i,j)|-|f_{t-1}(i+m_x,j+m_y)|\,\Big| \;\le\; |f_t(i,j)-f_{t-1}(i+m_x,j+m_y)| \quad (1)$$

oder aufgelöst:

$$|f_t(i,j)|-|f_{t-1}(i+m_x,j+m_y)| \;\le\; |f_t(i,j)-f_{t-1}(i+m_x,j+m_y)| \quad (2a)$$

$$|f_{t-1}(i+m_x,j+m_y)|-|f_t(i,j)| \;\le\; |f_t(i,j)-f_{t-1}(i+m_x,j+m_y)| \quad (2b)$$

Successive Elimination (2)

Summation auf beiden Seiten:

$$\sum_{j=0}^{N}\sum_{i=0}^{N}\left|f_{t-1}(i+m_x,j+m_y)\right|-\sum_{j=0}^{N}\sum_{i=0}^{N}\left|f_t(i,j)\right| \;\le\; \sum_{j=0}^{N}\sum_{i=0}^{N}\left|f_t(i,j)-f_{t-1}(i+m_x,j+m_y)\right| \quad (3a)$$

$$\sum_{j=0}^{N}\sum_{i=0}^{N}\left|f_t(i,j)\right|-\sum_{j=0}^{N}\sum_{i=0}^{N}\left|f_{t-1}(i+m_x,j+m_y)\right| \;\le\; \sum_{j=0}^{N}\sum_{i=0}^{N}\left|f_t(i,j)-f_{t-1}(i+m_x,j+m_y)\right| \quad (3b)$$

Mit:

$R=\sum_{j=0}^{N}\sum_{i=0}^{N}\left|f_t(i,j)\right|$ Summennorm des Referenzblocks

$M(m_x,m_y)=\sum_{j=0}^{N}\sum_{i=0}^{N}\left|f_{t-1}(i+m_x,j+m_y)\right|$ Summennorm des Blocks an m_x,m_y

$SAD(m_x,m_y)=\sum_{j=0}^{N}\sum_{i=0}^{N}\left|f_t(i,j)-f_{t-1}(i+m_x,j+m_y)\right|$ SAD an m_x,m_y

$$R - M(m_x,m_y) \le SAD(m_x,m_y) \quad (4a)$$

$$M(m_x,m_y) - R \le SAD(m_x,m_y) \quad (4b)$$

Successive Elimination (3)

Untergrenze für SAD des aktuellen Blocks in Abhängigkeit der Summennorm des aktuellen Blocks und des Referenzblocks:

$$R - M(m_x,m_y) \leq SAD(m_x,m_y) \qquad (4a)$$

$$M(m_x,m_y) - R \leq SAD(m_x,m_y) \qquad (4b)$$

Forderung für besseres Match:

$SAD(m_x,m_y) \leq SAD_{min}$; SAD_{min} kleinstes bisher gefundenes SAD

Eine notwendige Bedingung für diese Forderung läßt sich aus (4a),(4b) bestimmen:

$$R - M(m_x,m_y) \leq SAD_{min} \qquad (5a)$$

$$M(m_x,m_y) - R \leq SAD_{min} \qquad (5b)$$

Successive Elimination (4)

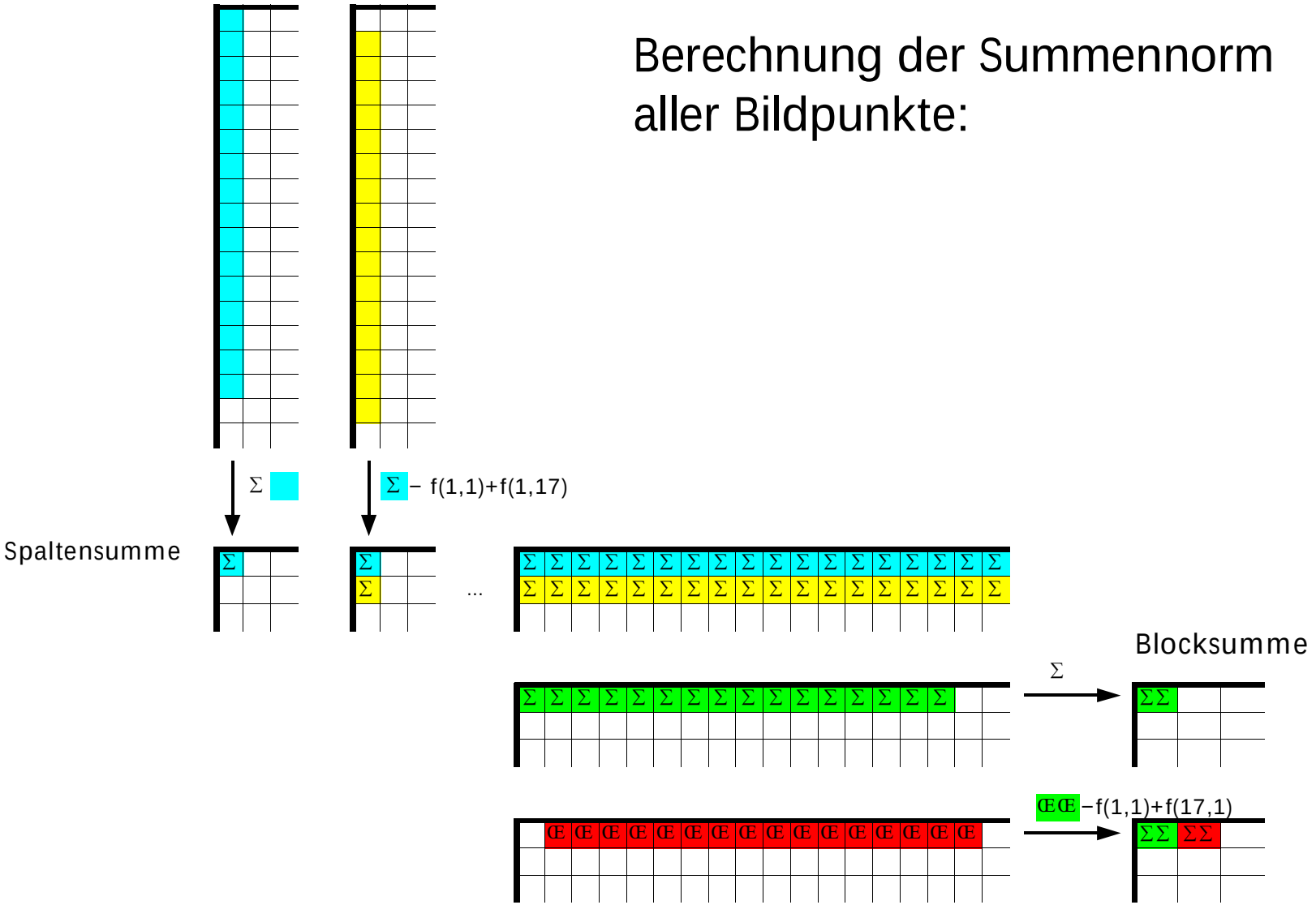
Zusammengefaßt und umgeformt erhält man eine Bedingung für die Summennormen im Suchbereich aus der Summennorm im Referenzbereich und dem bisherigen kleinsten SAD:

$$R - SAD_{min} \leq M(m_x, m_y) \leq R + SAD_{min} \qquad (6)$$

Wenn diese Ungleichung erfüllt ist, muß das SAD für die Stelle (x,y) berechnet werden, um entscheiden zu können, ob das Match besser als SAD_{min} ist

Wenn die Ungleichung nicht erfüllt ist, ist das SAD an der Stelle (x,y) in jedem Falle größer als SAD_{min}.

Successive Elimination (5)



Successive Elimination (5)

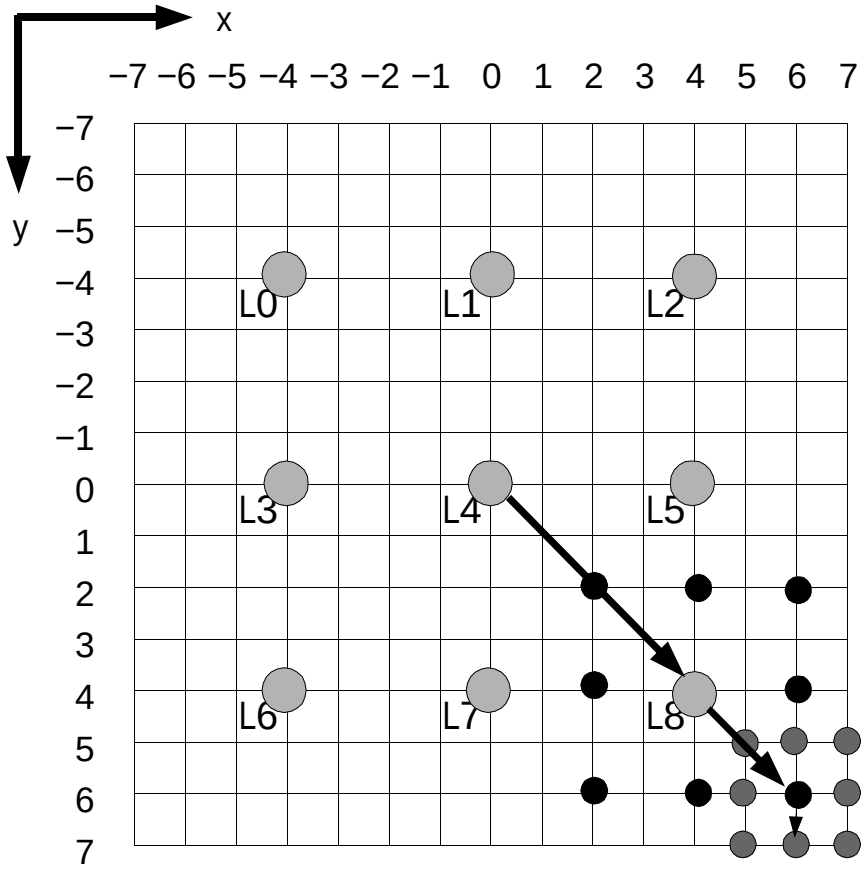
Vorteile:

- Findet mit Sicherheit das absolute Minimum
- Schnell, da SAD nur an wenigen Punkten berechnet wird

Nachteile:

- Komplexe Implemetierung
- Overhead vor der Suche zur Bildung der Summennormen:
  ~ 4*(Blockgröße)² Operationen pro Block

Three Step Search TSS (1)



Verfahren:

- Vergleichen der horizontalen, vertikalen und diagonalen Punkte im Abstand der Startschrittweite
- Am Punkt mit dem geringsten SAD: Verringerung der Schrittweite nach jedem Schritt und nächste Suche, bis Schrittweite 0 erreicht

Three Step Search TSS (2)

Vorteil:

- Sehr schnell
- Feste Berechnungsdauer (25 • SAD pro Block)

Nachteil:

- Kann in lokales Minimum laufen
- Begrenzter Suchbereich (±7 Pixel)

Exemplarischer Vergleich:

Algorithmus	PSNR [dB]	Anzahl der Suchpunkte
Full Search	31,647	1089
Sucessive Elimination	31,647	144
TSS (4 Schritte)	29,113	33

2D−Logarithmische Suche (1)

Verfahren:

- Vergleichen der horizontalen und vertikalen Punkte im Abstand der Startschrittweite
- Am Punkt mit dem geringsten SAD:
  nächste Suche, bis sich kein besseres SAD finden lässt
- Dann Verringerung der Schrittweite und erneute Suche
- Falls Schrittweite = 1:
  Abschliesende Suche an den 8 umliegenden Punkten

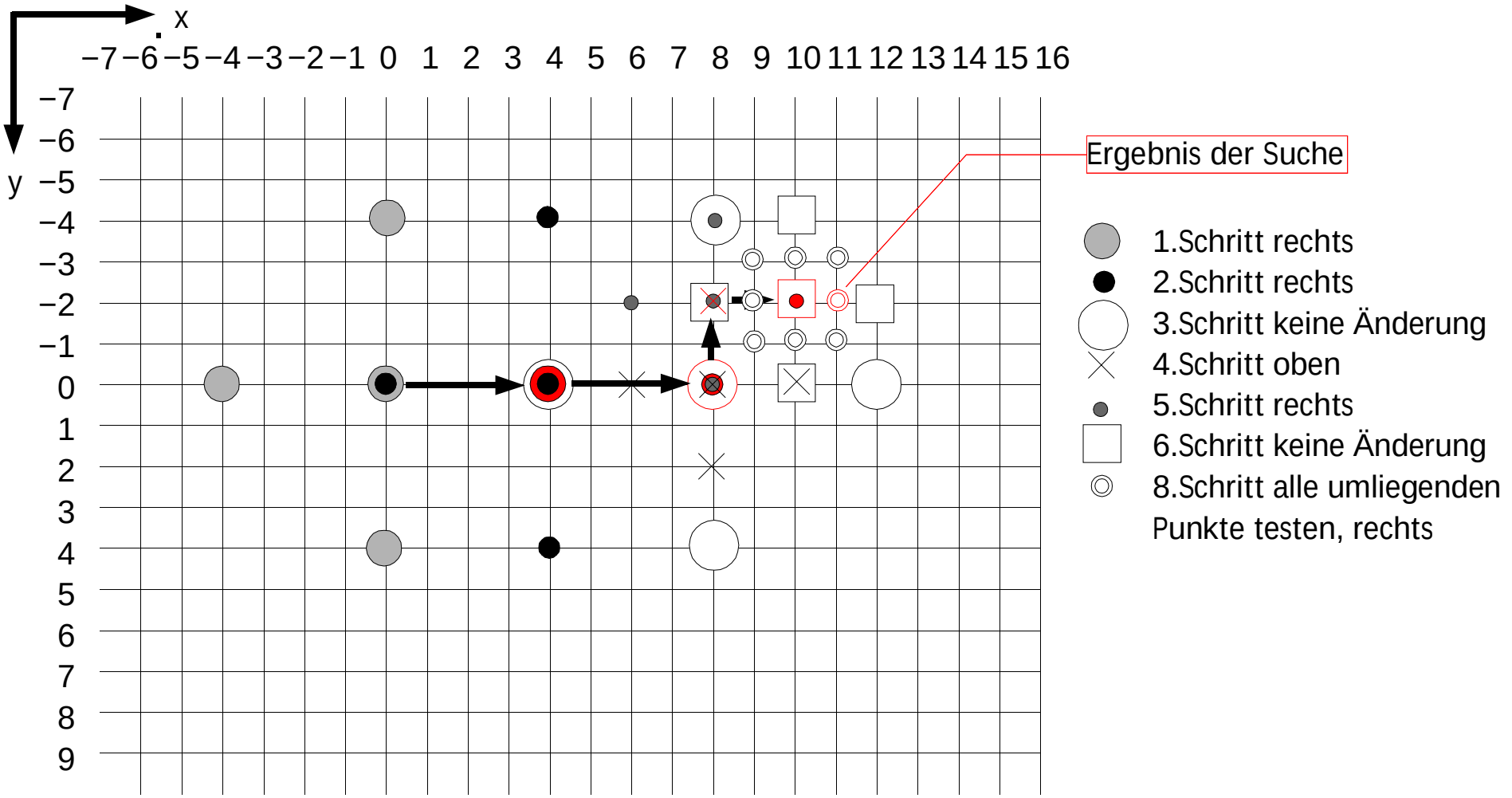
Vorteile:

- Schnell
- Findet auch große Bewegungsvektoren sehr schnell

Nachteil:

- Kann in lokales Minimum laufen

2D−Logarithmische Suche (2)



Modified Conjugate Direction Search (1)

Verfahren:

- Vom Startpunkt aus Vergleich mit den beiden Punkten links und rechts
- Am Punkt mit dem geringsten SAD:
  nächste Suche, bis sich kein besseres SAD finden lässt
- Dann Wechsel der Suchrichtung von horizontal zu vertikal bzw. umgekehrt und erneute Suche
- Solange fortfahren, bis Richtungswechsel keine Verbesserung zeigt.

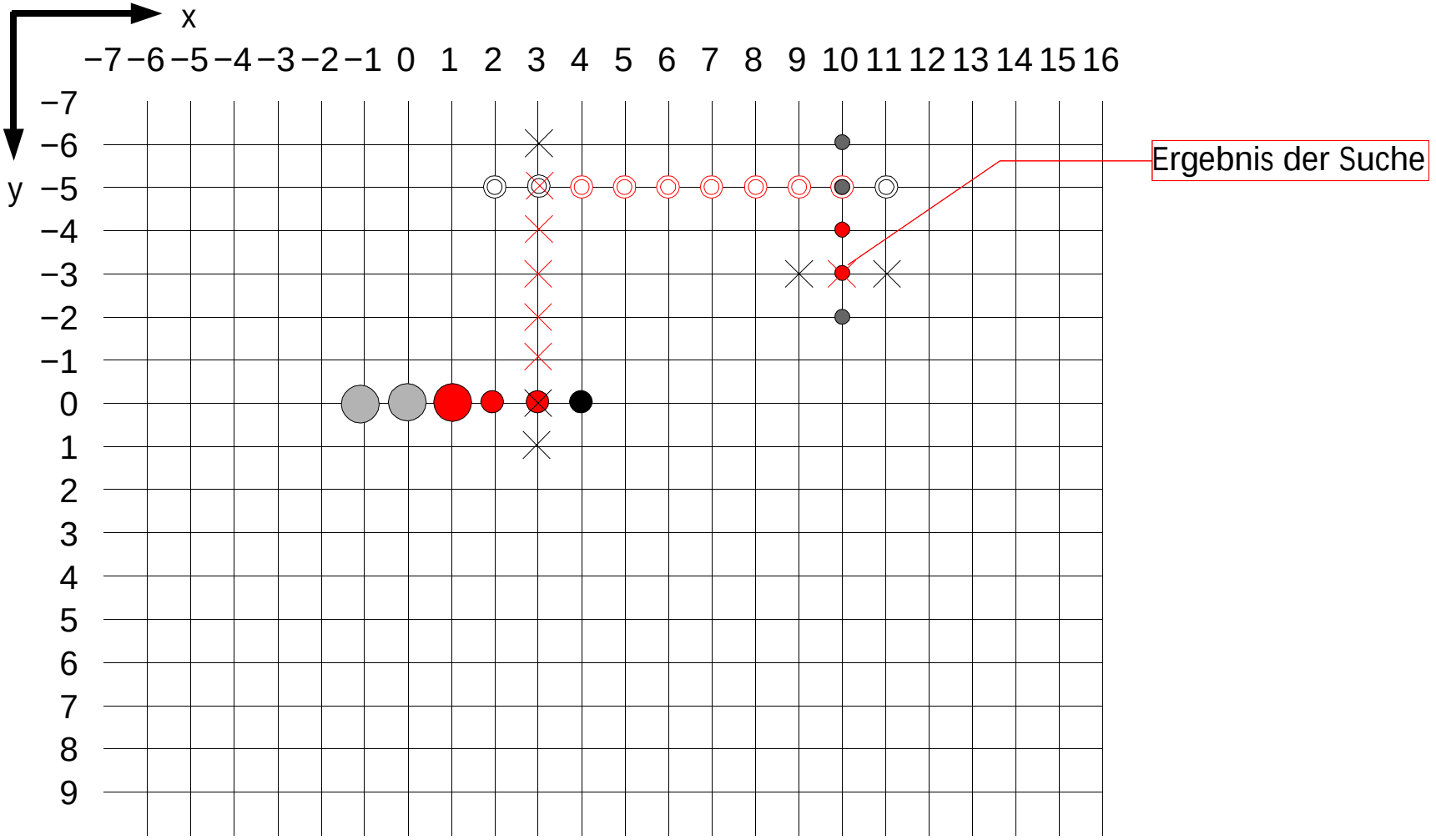
Vorteil:

- Schnell

Nachteile:

- Kann in lokales Minimum laufen
- Findet selten große Bewegungsvektoren

Modified Conjugate Direction Search (2)



Bi Area Subsampled BAS (1)

Verfahren:

- Vor Beginn der eigentlichen Suche wird erst der Startpunkt festgelegt. Dazu wird das SAD für den Bewegungsnullvektor und für den Predicted Candidate vorab bestimmt und der bessere Wert für den Start der Suche genommen.
- Vom Startpunkt aus Vergleich mit den vier Punkten links, rechts, oben, unten
- Am Punkt mit dem geringsten SAD: nächste Suche, bis sich kein besseres SAD finden lässt

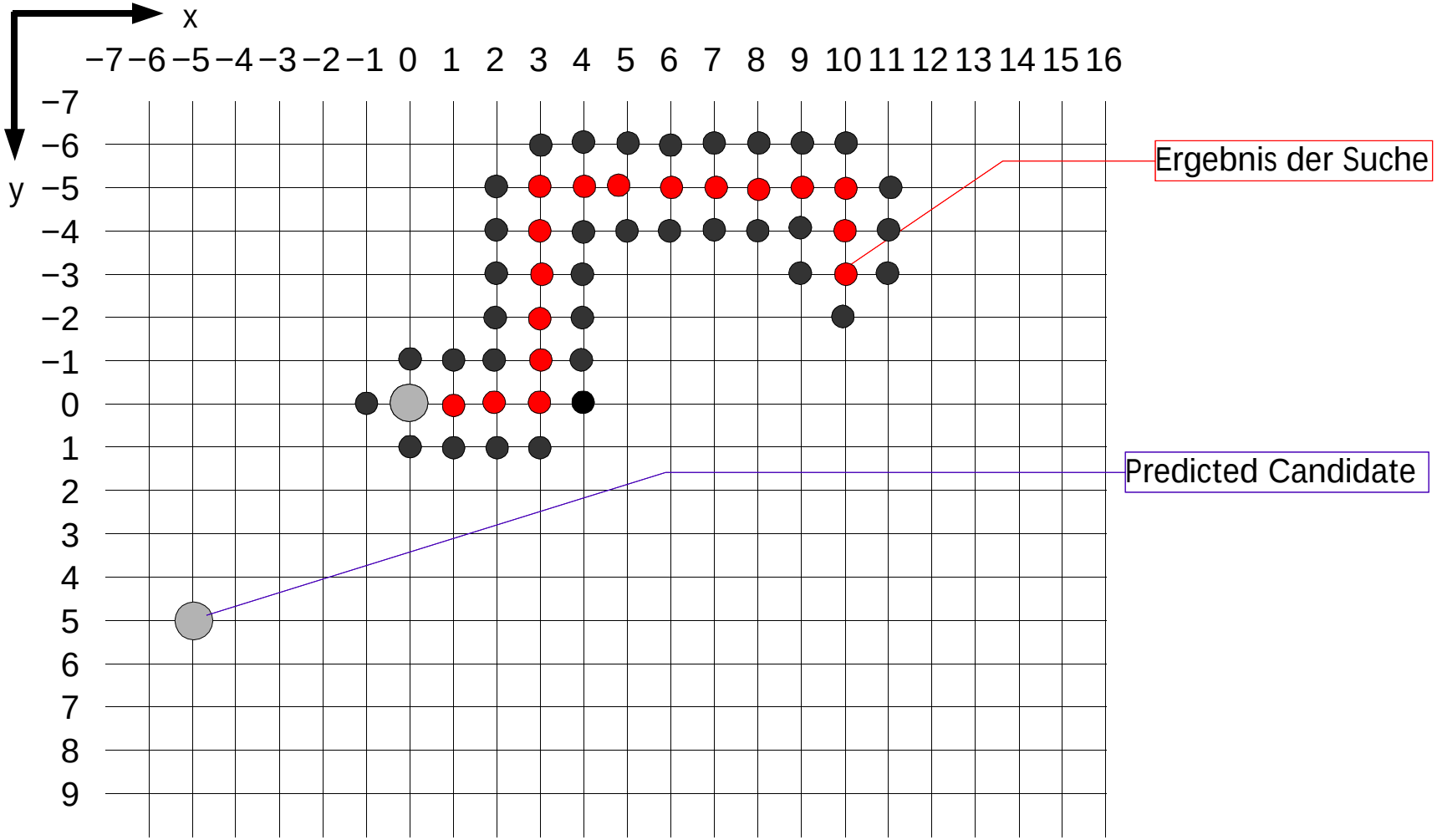
Vorteile:

- Schnell
- Einfach zu implementieren

Nachteil:

- Selten große Bewegungsvektoren
- Kann in lokales Minimum laufen

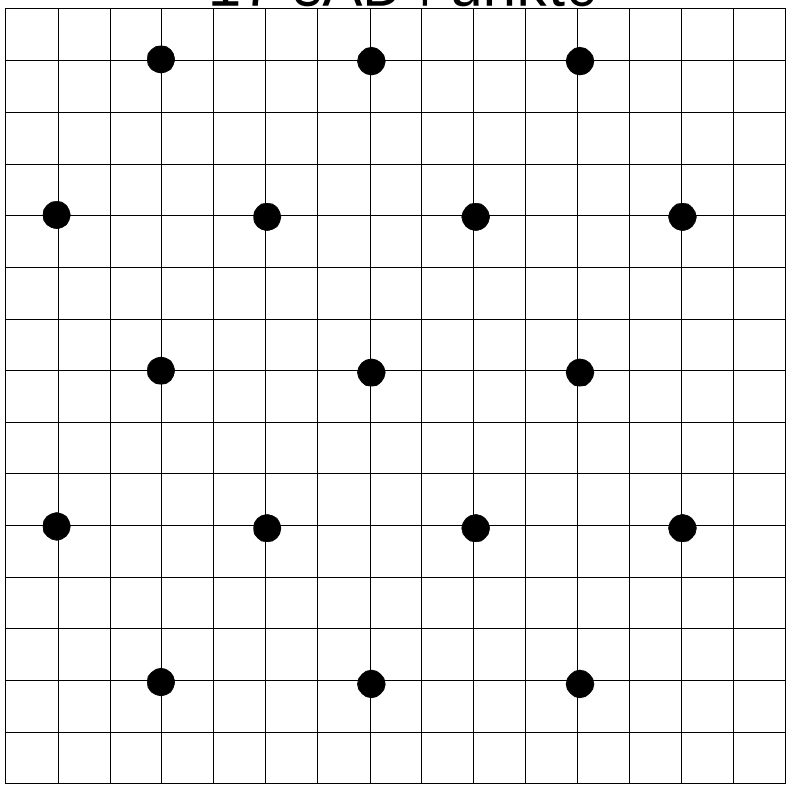
Bi Area Subsampled BAS (2)



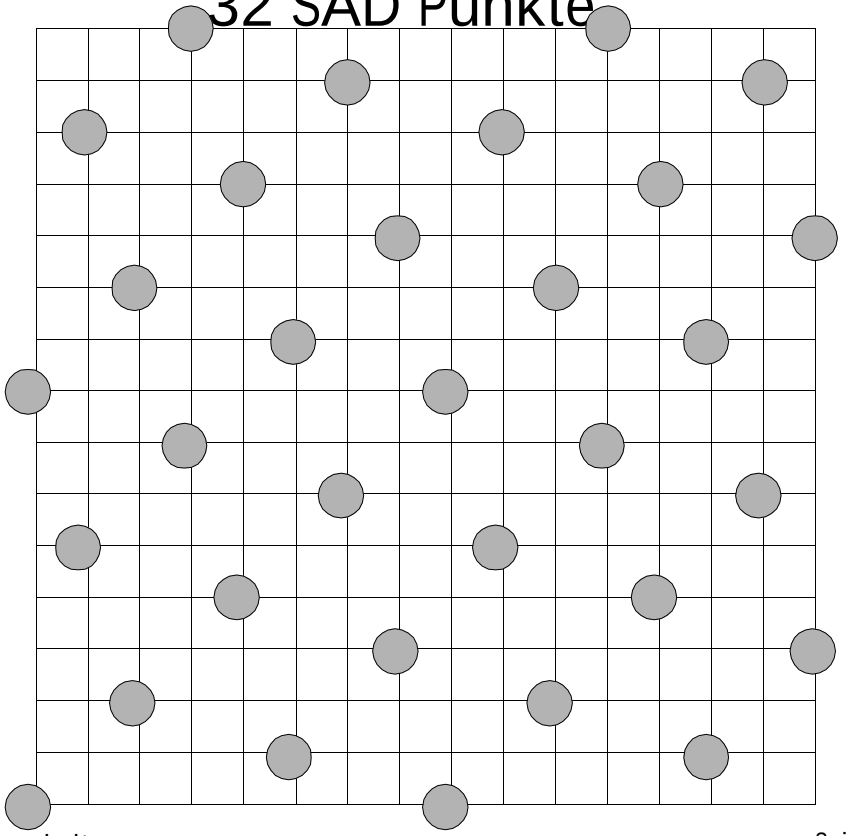
Bi Area Subsampled BAS (3)

Zusätzlich zur Reduktion der Aufsetzpunkte ist auch eine Unter–abtastung des Referenzblocks möglich: Statt 256 Differenzbildungen werden nur 17, bzw. 32 verwendet

17 SAD Punkte



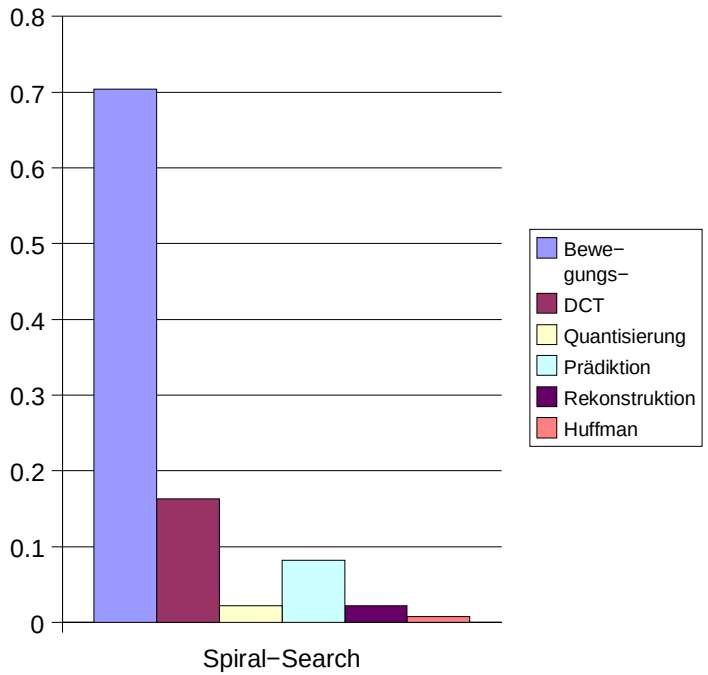
32 SAD Punkte

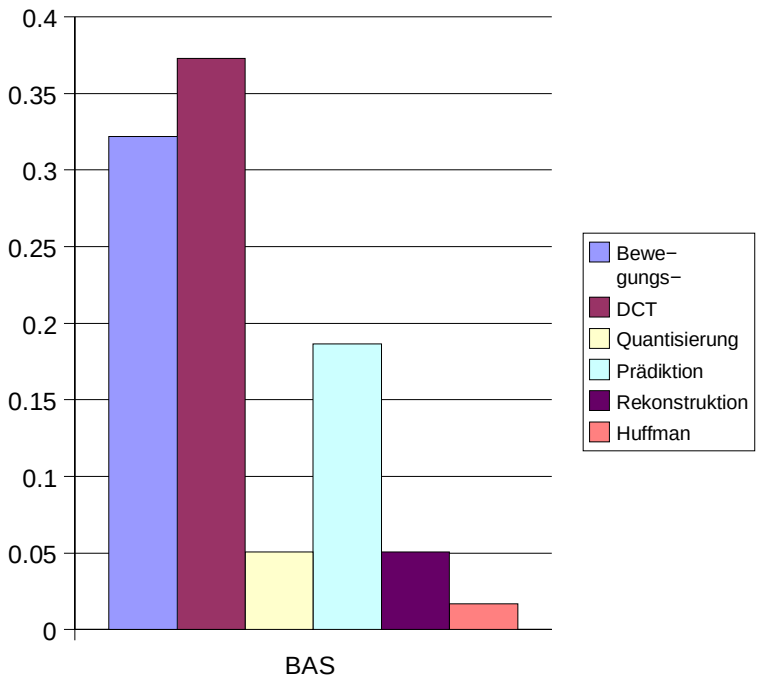


Bewegungsschätzung und Codierung (1)

Profil des Referenz–Encoders von Telenor

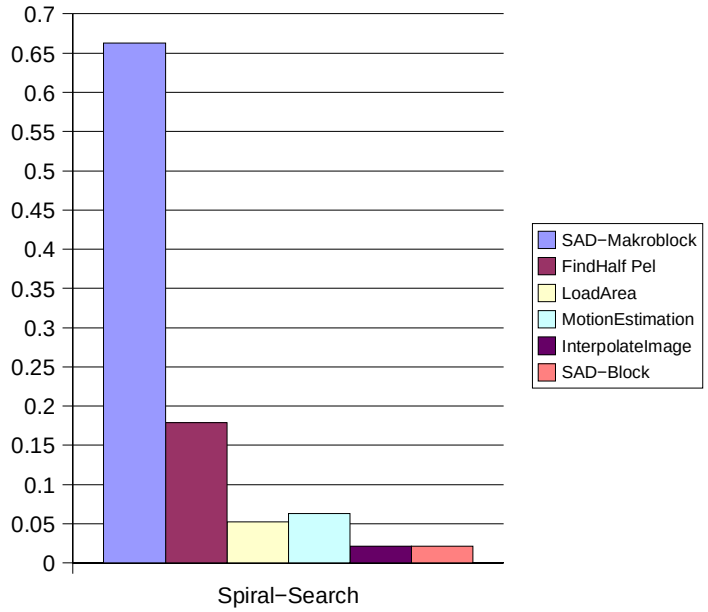
- Spiral Search with Threshold (original)
- BAS (17 –Punkte Abtastung)

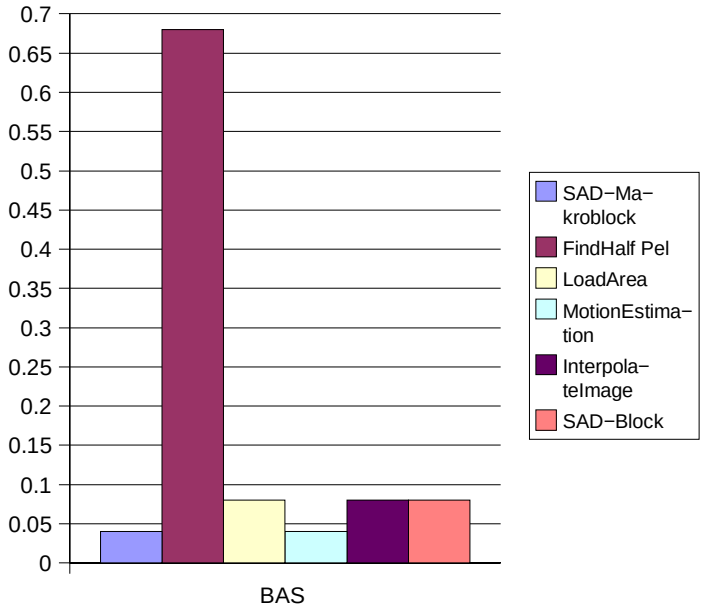




Bewegungsschätzung und Codierung (2)

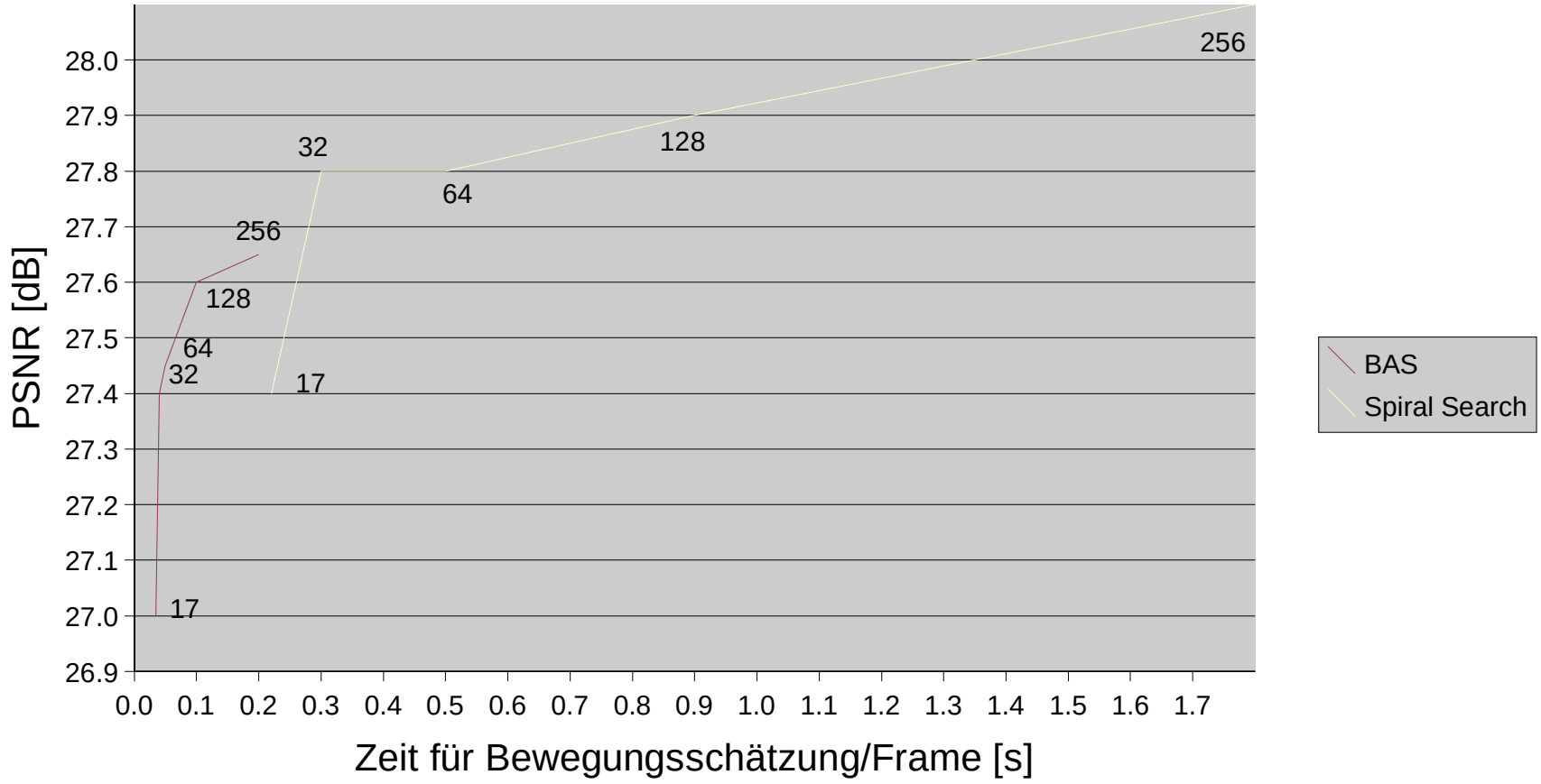
Bewegungsschätzung im Detail:





Bewegungsschätzung und Codierung (3)

Qualitätsverlust und benötigte Zeit bei
Unterabtastung des SAD mit BAS und Spiral Search



Bewegungsschätzung und Codierung (3)

Qualitätsverlust über benötigter Zeit bei Unterabtastung des SAD mit BAS