

# Video

Grundlagen Videokompression

Videokompression: Motion Compensation

# Video

## 1. Grundlagen der Videokompression

- Damit im Film Bewegung flüssig dargestellt werden kann, benötigt man ca. 15-25 Bilder pro Sekunde.
- Wie bei Bildformaten sind für die Farbinformationen in der Regel 24 Bit pro Bildpunkt vonnöten.
- Das bedeutet bei einer Auflösung von 720x576 Bildpunkten (DVD):
  - $720 \times 576 \text{ Pixel} =$   
 $(414.720 \text{ Pixel} \times 24 \text{ Bit}) / 8 = 1.244.160 \text{ Byte} =$   
 $1.215 \text{ MByte} = 1.215 \text{ MByte} \times 25 \text{ Bilder/Sek.} =$   
**30.375 MByte/s**  $\approx$  **30 GByte** (29,67 GByte)  
 $30 \text{ GByte} \times 60 = 1.800 \text{ GByte/min} =$   
**108.000 GByte pro Std. Film** (völlig unkomprimiert!)
- Für eine praktikable Nutzung digitalen Videos muss also eine effiziente Kompressionsform her.

## 1.1 Video-Kompressionstechniken

- Die Kompression von **Bildern** basiert im Wesentlichen auf der **Korrelation zwischen Pixeln**.
- Die Pixel eines Videobildes sind **nicht unabhängig voneinander**, vielmehr gibt es Korrelationen zwischen benachbarten Pixeln.
- In einem gewissen Rahmen ist also der Wert eines bestimmten Pixels anhand der Werte benachbarter Pixel **vorhersagbar**.
- Die Kompression von **Video-Sequenzen** beruht neben der **Korrelation von Pixeln** in einem einzelnen Bild auch auf der **Korrelation zwischen aufeinander folgenden Bildern (Frames)**.
- Grundsätzlich werden zwei Arten von Redundanz beseitigt:
  - **räumliche Redundanz**, die in jedem einzelnen Bild (Frame) existiert; Dies entspricht der Redundanz bei unbewegten Bildern.
  - **temporale Redundanz**, d.h. Bereiche eines Bildes, die sich in Vorgängern und Nachfolgern wiederholen.

### 1.1.1 Intuitive Kompressionsansätze

- **Subsampling:**  
nur jedes zweite Frame wird ausgewählt und in den komprimierten Datenstrom geschrieben. Der Dekodierer dupliziert die Frames.
- **Differencing:**  
nur Unterschiede werden in den Datenstrom geschrieben (Pixel-Koordinaten, Unterschiede zwischen den Pixeln); sind die Unterschiede zu groß, wird das Frame komplett geschrieben.
  - **"Lossy" Differencing:**  
liegt der Unterschied zwischen zwei Pixeln in zwei Frames unterhalb einer wahrnehmbaren Schwelle, werden die Pixel als gleich interpretiert.
- **Block Differencing:**  
blockweiser Vergleich zwischen zwei Frames.
- **Motion Compensation (Bewegungskompensation):**  
Die Unterschiede zwischen aufeinanderfolgenden Bildern sind relativ gering. Sie resultieren aus einer **Veränderung der Kameraposition**, oder einer **Positionsänderung der dargestellten Objekte** oder beides zusammen. Dieser Umstand wird ausgenutzt, um eine bessere Kompression zu erreichen. (s. → „Kompression durch Bewegungskompensation“)

## 1.2 Dekodierung von komprimierten Videos

- Die Enkodierung von Videodateien kann im Prinzip relativ lange dauern (und tut es auch in der Regel...):  
→ **asynchrone Übertragung.**
- Da ein Dekodierer aber bei der Wiedergabe verwendet wird, muss er sehr schnell sein, um die korrekte Bildrate zu liefern.
- Deshalb arbeitet der Dekodierer normalerweise parallel, d.h., mehrere Dekodierzyklen arbeiten gleichzeitig an mehreren Frames.
- In einer solchen Situation kann es sinnvoll sein, nicht nur die Vorgänger von Bildern bei der Kodierung zu berücksichtigen, sondern auch Nachfolger.
- Wenn z.B. durch die Bewegung eines Objektes ein Hintergrund sichtbar wird, kann es für ein Bild nützlich sein, zu wissen, in welche Richtung sich das Objekt bewegt, d.h. also welcher Hintergrund im nächsten Bild zu sehen sein wird.

### 1.3 Arbeitsschritte zur Beseitigung temporaler Redundanz

- Das erste Bild einer Sequenz wird mithilfe der Kompressionstechniken für unbewegte Bilder (→ JPEG) komprimiert. (→ Beseitigung **räumlicher Redundanz** oder **Bildkompression**)
- Bei den folgenden Bildern werden nur die **Unterschiede zum jeweiligen Vorgänger** festgestellt und kodiert.
- Ist ein Bild sehr unterschiedlich von seinem Vorgänger, wird es als **erstes Bild einer neuen Sequenz** aufgefasst und unabhängig komprimiert.
- Ein Bild, welches unter Bezug auf den Vorgänger komprimiert wird, heisst **(P)redicted frame**, ein unabhängig kodiertes Bild wird **(I)ntra frame** genannt.
- Da bei Bildern verlustbehaftete Kompressionsmethoden angewendet werden, kann sich der damit verbundene Fehler bei aufeinanderfolgenden Bildern **akkumulieren**.  
Auch deshalb sollten von Zeit zu Zeit **Intra frames** eingeschaltet werden.

### 1.3.1 (I)ntra, (P)redicted-Frames

- Das **Intra**-Frame wird mit **I** bezeichnet; es wird nur unter Berücksichtigung der eigenen Bildinformation kodiert. Das Kompressionsverfahren ist meist stark an JPEG angelehnt. (→ **YCbCr**)
- Das **Inter**-Frame wird mit **P** (predictive) bezeichnet. Es berücksichtigt auch Bildinformationen des am nächsten liegenden I -oder P-Vorgängers.  
**Nachteil:** Zunahme von Kodierfehlern.  
**Vorteil:** höhere Kompression (P-Frames benutzen dafür *motion compensation*).

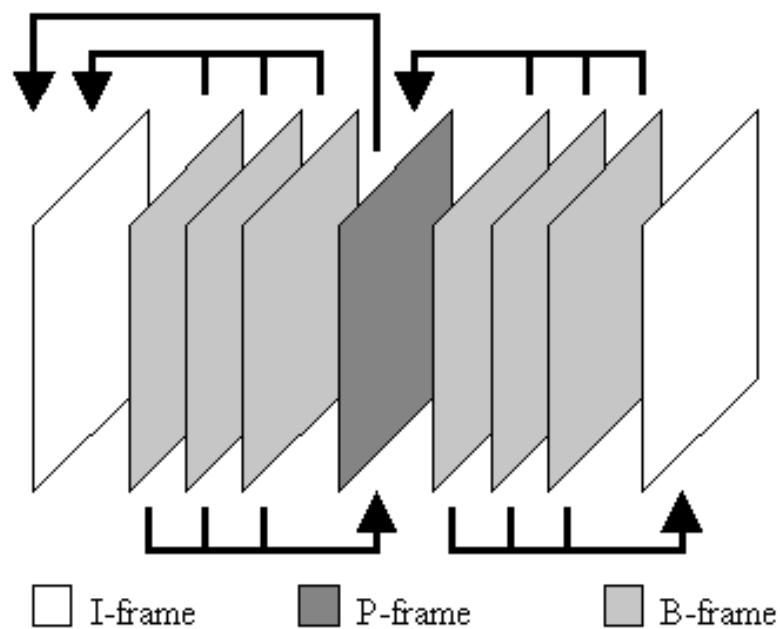


Abb.: Abhängigkeiten der frame-Typen eines MPEG-1 Videostroms

### 1.3.2 (B)idirectional Frames

- Ein Frame, welches sowohl Vorgänger als auch **Nachfolger** zum Kodieren verwendet, wird als **B-Frame** (bidirectional) bezeichnet.
- Um dies zu ermöglichen, werden die einzelnen Bilder vom Encoder nicht in **display order**, sondern in **bitstream order** abgespeichert:

B1 B2 **I3** B4 B5 P6 B7 B8 P9 B10 B11 P12 (display order)

**I3** B1 B2 P6 B4 B5 P9 B7 B8 P12 B10 B11 (bitstream order)

- B-Frames dürfen nur I- oder P-Frames als Referenz nutzen.
- Der Decoder kann in **bitstream order** die B-frames anhand des vorangestellten I-frames rekonstruieren.
- **Vorteil:** noch höhere Kompression

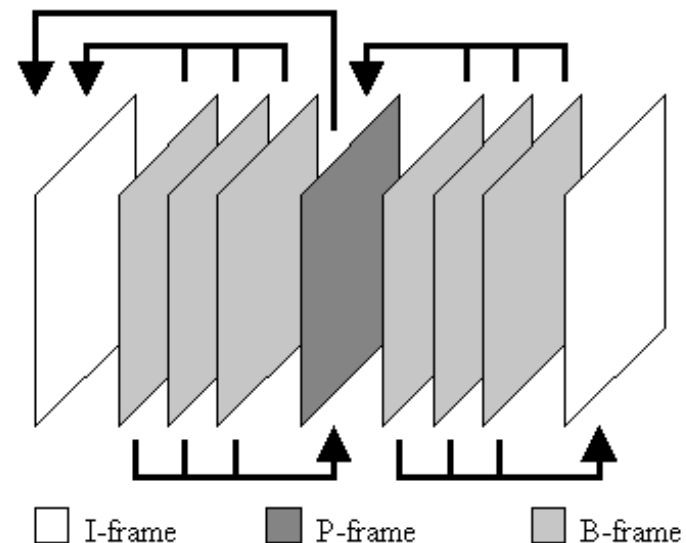


Abb.: Abhängigkeiten der frame-Typen eines MPEG-1 Videostroms

### 1.3.2 Kompression durch Bewegungskompensation (Motion Compensation)

- **Bewegungskompensation** = Technik zur Beseitigung temporaler Redundanz in Video-Sequenzen.
- Grundgedanke: Zwei aufeinander folgende Frames einer Sequenz sind üblicherweise sehr ähnlich – enthalten somit **viel (temporal) redundante Informationen**. Redundante Informationen müssen aber nicht mit jedem Frame neu abgespeichert werden.
- **Motion Compensation** berücksichtigt daher für die Redundanzreduktion die *Veränderungen in den Bewegungen* einzelner, gleicher Bildkomponenten.
- Die Veränderungen werden als **Bewegungsvektoren** (*motion vector*) gespeichert.
- Somit können die Bildinformationen eines aktuellen Frames durch einen Vergleich mit einem vorherigen Referenzframe über die Bestimmung der Bewegungsvektoren festgehalten werden (Resultat = P-Frame).
- Werden auch Informationen aus einem folgenden Frame zur Kodierung der Bildinformationen des aktuellen Frames hinzugezogen, entsteht ein **bidirektional kodiertes Frame** (B-Frame).

### 1.3.3 Motion Compensation

- bewegt sich nur ein Teil des Bildes, brauchen nur die vorherige Position des bewegten Teilbildes (Blocks), seine jetzige Position und Informationen zur Abgrenzung des Teilbildes kodiert zu werden.
- In der Regel sind diese Teilbereiche Blöcke einer festen Größe.
- Man speichert nur die Unterschiede der Positionen, nicht den Bildinhalt selbst! (Man sagt dem Decoder also nur, wo er sich die Bildinformation holen kann, d.h. letztlich in welchem I- oder P-frame.)
- Diese Differenz zwischen den Bildern heißt **Bewegungsvektor** (engl. *motion vector*).



## 1.4 Blockbasierte Bewegungskompensation (Block Motion Compensation)

### *Framefragmentierung / Blockvergleich*

- Bei der **blockbasierten Motion Compensation** wird das Frame zunächst in **Blöcke** zerlegt (*Frame Segmentation*).
- MPEG-Standard: Blöcke à 16x16 Pixel.
- Jeder Block des aktuellen Bildes wird mit dem entsprechenden Block des Vorgängers verglichen.

### *Search Threshold*

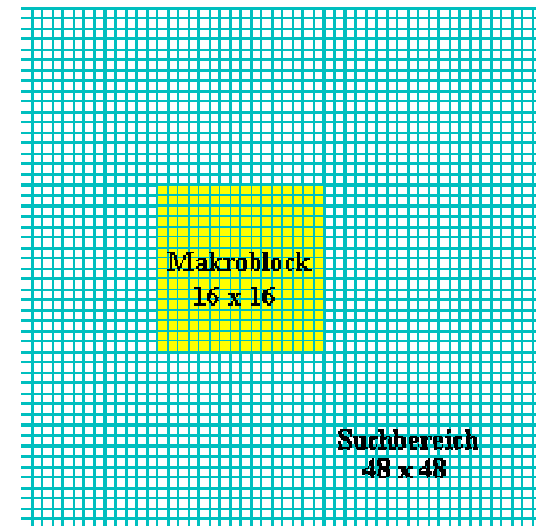
- Abhängig von einem **Schwellenwert** (*Search Threshold*) der die Ähnlichkeit der Blöcke repräsentiert, wird bei nahezu identischen Blöcken (Unterschied unterhalb des Schwellenwerts) vom Kodierer angenommen, dass keine Bewegung vorliegt.

### *Block search*

- Wird der Schwellenwert überschritten, so startet ein **Blocksuchalgorithmus**.
- Man speichert die Positionsänderungen der einzelnen Blöcke, nicht die der einzelnen Pixel (→ Datenreduktion).
- Die Blockgröße hat also wesentlichen Einfluss auf die Wirksamkeit der Kompression:
  - Zu große Blöcke vermindern die Chance identische Blöcke zu finden
  - zu kleine Blöcke können zur Generierung sehr vieler Bewegungsvektoren führen.

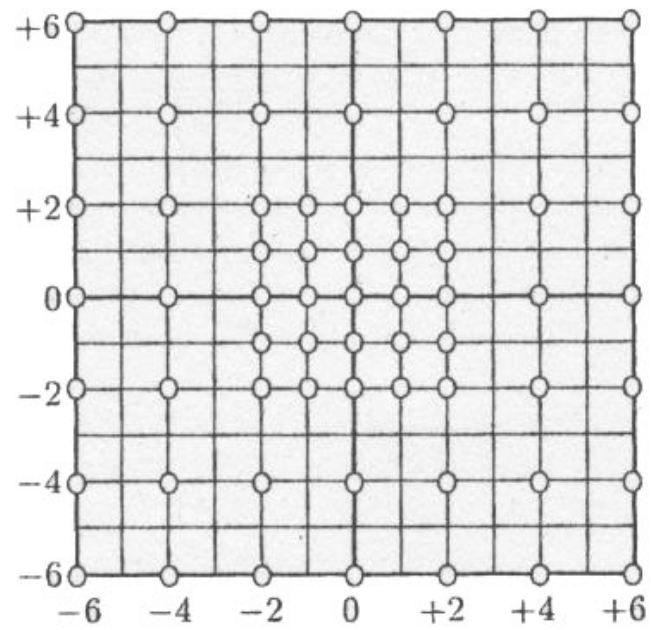
## Blocksuche (Block Search)

- Ein Algorithmus sucht nach der **bestmöglichen Übereinstimmung** eines Blocks des momentanen Frames mit einem des Referenzframes.
- Als mögliche **Vergleichsgröße für den Grad der Übereinstimmung** (distortion measure) verwendet man z.B.:
  - Den **Mittelwert der absoluten oder quadrierten Differenz** der Pixel innerhalb zweier Blöcke.
  - Der Block mit dem niedrigsten Wert ist der beste Treffer.
- Die Suche wird in der Regel auf einen eng begrenzten Bereich (search area) im vorherigen Frame begrenzt.
- Aufgrund der Annahme, dass von Frame zu Frame die Bewegung eines Objektes (also die räumliche Verschiebung) gering ist, kann der Suchbereich relativ stark eingegrenzt werden.
- Dies muss auch so sein, denn **Block search** ist **zeitintensiv**: Schon ein Suchbereich von 48x48 Bildpunkten ergibt max. 2304 Möglichkeiten.

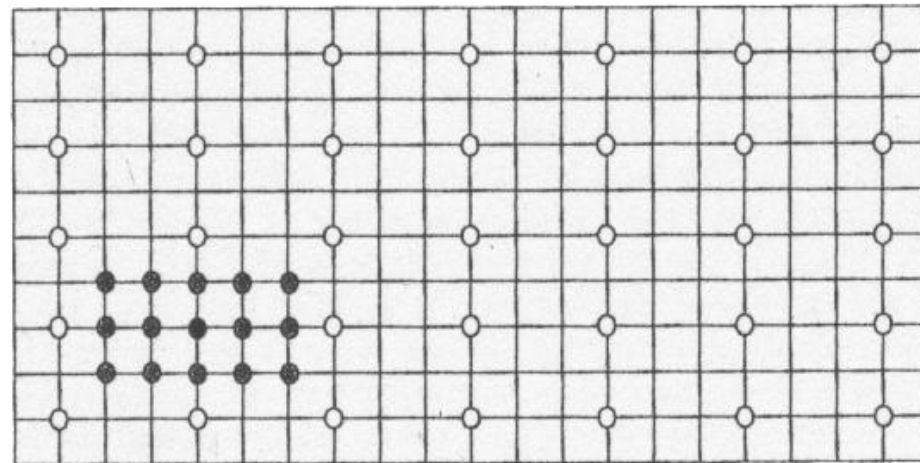


## *Suboptimale Suchstrategien für Block Search*

- Die Suche nach identischen Blocks (Block Search) ist sehr **zeitaufwendig**. Um die Rechenzeit bei der Blocksuche zu verringern, werden in der Regel „ungenauere“ Suchmethoden eingesetzt, wie z.B.:
  - 1. *Signature Based Methods:***  
Im ersten Schritt werden alle in Frage kommenden Blocks mit *einfachen Blocksuch-Methoden* (z.B. Mittelwert der absoluten Differenz) auf die Ähnlichkeit überprüft. In den nächsten Schritten werden genauere Methoden nur auf die *Blöcke mit den besten Ergebnissen* aus Schritt 1 angewendet.
  - 2. *Distance-Diluted Search:***  
Sich schnell bewegende Objekte wirken verschwommen. Deshalb können sie mit einer geringeren Blockübereinstimmungsgenauigkeit kodiert werden. Um solche Objekte zu erfassen, werden alle nahe beim Ursprungsbloc liegenden Blöcke genau verglichen, immer weniger und immer ungenauer aber mit zunehmender Distanz.
  - 3. *Locality-Based Search:***  
Zunächst wird über *räumlich relativ weit verteilte Blocks* gesucht; der „*beste Treffer*“ ist Ausgangsbasis für eine *intensivierte Suche um diesen Treffer herum*: Hier besteht die hohe Wahrscheinlichkeit, einen besseren/ den besten Block zu finden.



(a)



○ — first wave. ● — best match of first wave. ● — second wave.

(b)

**Figure 5.4:** (a) Distance-Diluted Search for  $dx = dy = 6$ . (b) A Locality Search.

## ***Abhängigkeitsbestimmte Algorithmen für Block Search***

- Da sich bewegende Objekte innerhalb eines Frames in der Regel größer sind als einzelne Blocks, kann man annehmen, dass benachbarte Blocks korrelierende Bewegungsvektoren aufweisen.
- Diese Annahme wird von Algorithmen, die Abhängigkeitsverhältnisse zu Grunde legen (*dependent algorithms*) bei der Suche nach übereinstimmenden Blocks genutzt:

### *Temporale Abhängigkeit*

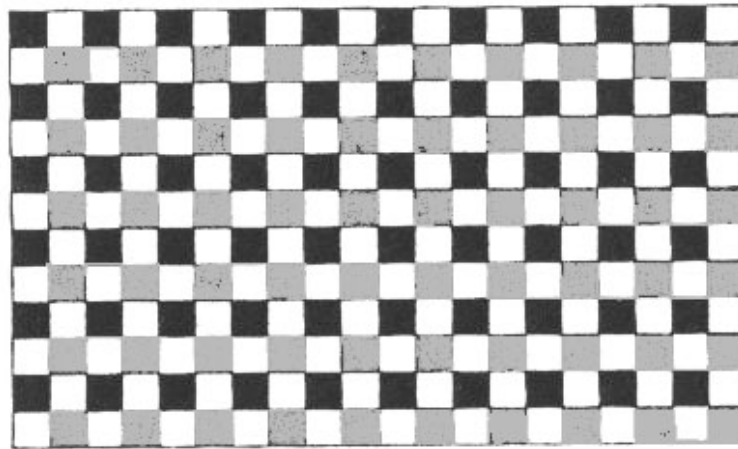
- Es wird angenommen, dass der Bewegungsvektor des aktuellen Blocks gleich dem Bewegungsvektor desselben Blocks im vorherigen Bild ist.
- Dies macht Sinn, wenn man von einer gleichförmigen Bewegung ausgeht.

### *Räumliche Abhängigkeit*

- In einem Algorithmus, der *räumliche Abhängigkeit* zu Grunde legt, werden benachbarte Blocks von Block X benutzt, um den Bewegungsvektor von X vorherzusagen.
- Natürlich kann man für die Berechnung nur Nachbarblocks benutzen, deren Vektoren bereits errechnet wurden.
- Die meisten Blocks haben acht Nachbarn, aber alle acht zu vergleichen ist ineffektiv. Die beste Variante ist vier symmetrische Nachbarblöcke zu nehmen.

## Räumliche Abhängigkeit

- Drei Schritte sind erforderlich für die Überprüfung von jeweils vier symmetrisch angeordneten Nachbarn auf Übereinstimmung:
  1. Zuerst wird in der ersten und danach in jeder zweiten Reihe für jeden zweiten Block der Vektor berechnet (**schwarze Kästchen**).
  2. Im zweiten Schritt wird in den bisher unberücksichtigten Zeilen versetzt für jeden zweiten Block der Vektor berechnet (**graue Kästchen**). Hier können bereits vier symmetrisch angeordnete Blöcke zum Vergleich herangezogen werden.
  3. Im dritten Schritt werden alle restlichen Vektoren (**weiße Kästchen**) berechnet.



(f)

Figure 5.6: Strategies for Spatial Dependent Search Algorithms.

## ***Bewegungsvektorberechnung und -Korrektur***

- Nachdem ein „passender“ Block im Referenzframe gefunden wurde, wird der Bewegungsvektor als Differenz der beiden Positionen der Blöcke berechnet (obere linke Ecke des Blocks).
- Verschiedene Faktoren können eine Optimierung des berechneten Bewegungsvektors notwendig machen (z.B. suboptimaler Suchalgorithmus).

## ***Schätzfehler-Kodierung***

- Motion compensation ist **verlustbehaftet**, da üblicherweise die zwei als korrespondierend behandelten Blöcke nicht absolut identisch sind.
- Durch Speicherung der Differenz zwischen dem unkomprimierten (original) und komprimierten Block kann der Decoder eine Verbesserung der Bildqualität erreichen. Dies wird Block für Block durchgeführt und nur für Blöcke, die stark voneinander abweichen.

## *Kodierung von Bewegungsvektoren*

- Da ein großer Teil (nahezu die Hälfte) des jeweils aktuellen Bildes in Bewegungsvektoren konvertiert werden kann (und somit komprimiert!), ist die **Kodierung der Ergebnisse**, also der Vektoren, von hoher Bedeutung.
- Bewegungsvektoren sollten **verlustfrei** kodiert werden.

## *Kodierung von Bewegungsvektoren*

- Da ein großer Teil (nahezu die Hälfte) des jeweils aktuellen Bildes in Bewegungsvektoren konvertiert werden kann (und somit komprimiert!), ist die **Kodierung der Ergebnisse**, also der Vektoren, von hoher Bedeutung.
- Bewegungsvektoren sollten **verlustfrei** kodiert werden.
- Zwei Eigenschaften von Bewegungsvektoren helfen dabei:
  1. sie **korrelieren** miteinander:  
Wenn ein Bild Block für Block verglichen wird, sind die Bewegungsvektoren nebeneinander liegender Blöcke meist nicht sehr unterschiedlich, d.h. sie korrelieren miteinander.

## *Kodierung von Bewegungsvektoren*

- Da ein großer Teil (nahezu die Hälfte) des jeweils aktuellen Bildes in Bewegungsvektoren konvertiert werden kann (und somit komprimiert!), ist die **Kodierung der Ergebnisse**, also der Vektoren, von hoher Bedeutung.
- Bewegungsvektoren sollten **verlustfrei** kodiert werden.
- Zwei Eigenschaften von Bewegungsvektoren helfen dabei:
  1. sie **korrelieren** miteinander:  
Wenn ein Bild Block für Block verglichen wird, sind die Bewegungsvektoren nebeneinander liegender Blöcke meist nicht sehr unterschiedlich, d.h. sie **korrelieren** miteinander.
  2. ihre **Verteilung ist ungleichmäßig**:  
Die Bewegungsvektoren weisen außerdem nicht in alle möglichen Richtungen, sondern in eine oder zwei bevorzugte. Sie sind also **ungleichmäßig** verteilt.

## *Kodierung von Bewegungsvektoren*

- Bisher hat sich keine Kodierungsmethode als die ideale herausgestellt, aber einige als sehr wirksam:
- Besonders gut haben diese Methoden funktioniert:
  - **Schätzung des Bewegungsvektors aufgrund der vorhergehenden Blöcke in derselben Zeile und Spalte eines Bildes.** Berechnung der Differenz zwischen der Schätzung und dem tatsächlichen Vektor.  
Anschließende Huffman-Kodierung → (MPEG-1)
  - **Gruppierung der Bewegungsvektoren zu Blocks.**  
Sind alle Vektoren im Block identisch, wird der Vektor einmal für den gesamten Block kodiert. Andere Blocks werden nach Methode 1 kodiert.

## *Teilschritte der Motion Compensation*

Blockzerlegung
Blockvergleich
Blocksuche
Bewegungsvektorermittlung
Bewegungsvektorkorrektur
Schätzfehlerkodierung
Bewegungsvektorkodierung

## Der Videoteil in MPEG-1 / MPEG-2: Syntax

- Die interne Darstellung und Interpretation des kodierten Datenstroms nennt man **Syntax**. Der Datenstrom läßt sich dabei in sechs Schichten (die MPEG-Video Layer) unterteilen, die in einer Hierarchie angeordnet sind:
  1. **Sequence-Layer:**  
Eine Sequenz fasst eine oder mehrere GOP zusammen; der Header-Teil der Sequenz beinhaltet allgemeine Informationen wie z.B. Bildformat, Übertragungsrate u.a.
  2. **Group of Pictures (GOP)- Layer:**  
Jede Sequenz enthält eine Serie von Bilder; GOP beginnen ebenfalls mit einem Header
  3. **Picture-Layer:**  
Primäre Kodierungseinheit; Farbmodell: Luminanz/Chrominanz (YCbCr)
  4. **Slice-Layer:**  
Aufeinanderfolge von Makroblöcken; Fehlerkorrekturverbesserung
  5. **Macro Block-Layer:**  
Aggregation mehrerer Blöcke (4x)
  6. **Block-Layer:**  
kleinste Organisationseinheit, 8x8 Pixel.

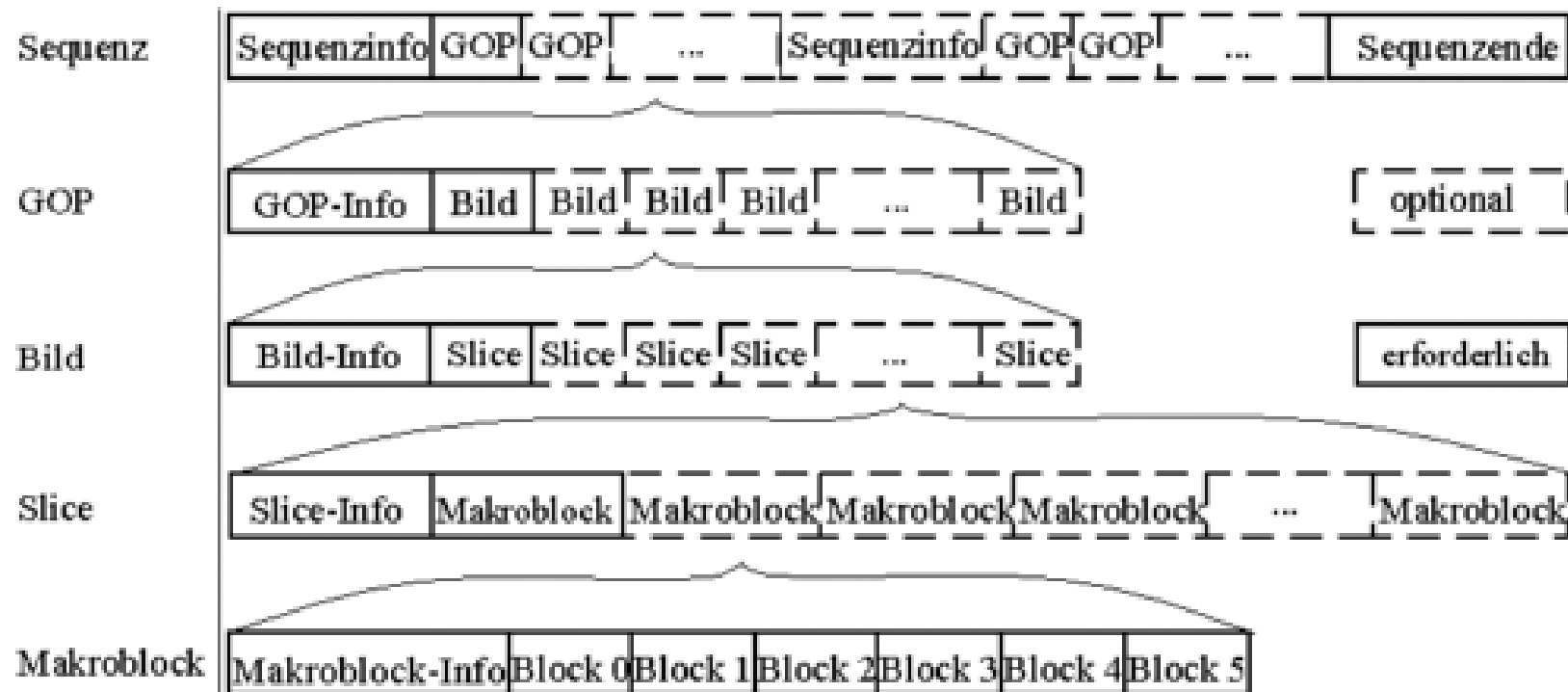


Abb.: Die Schichten der MPEG Syntax

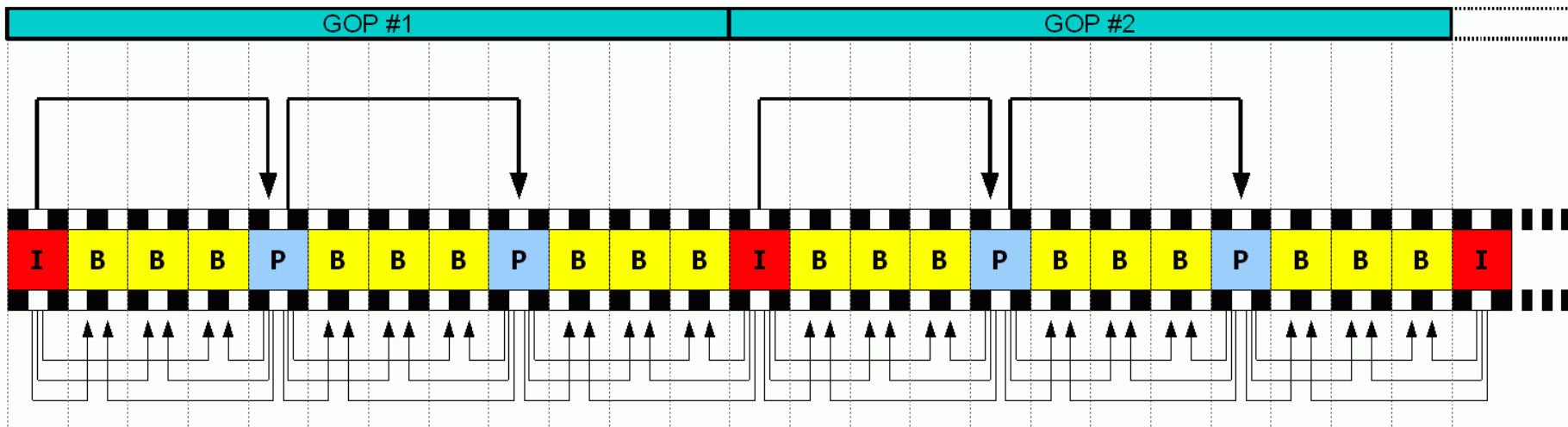
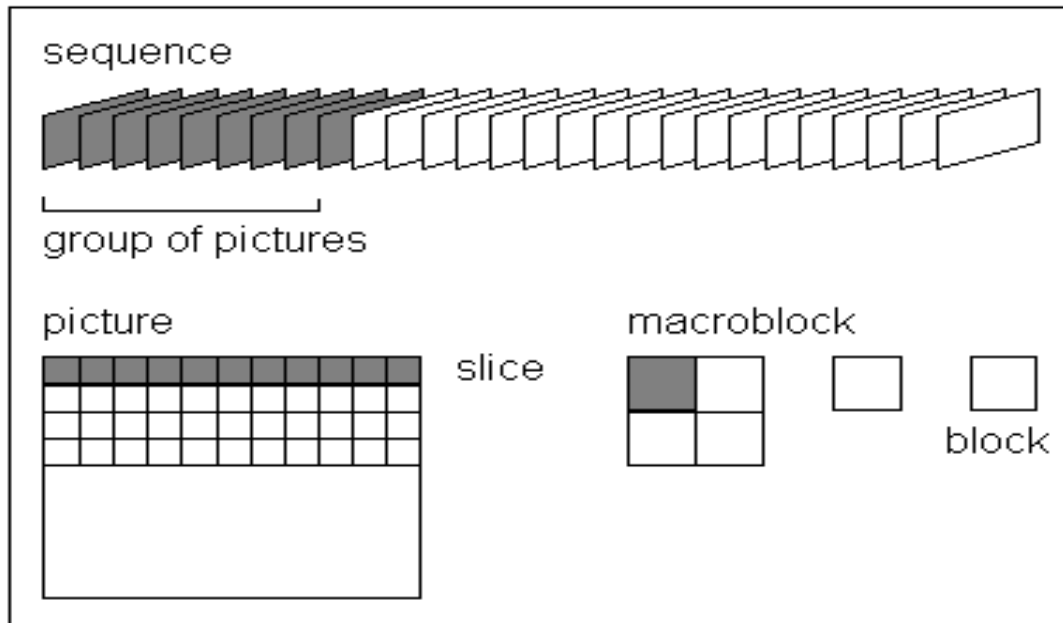
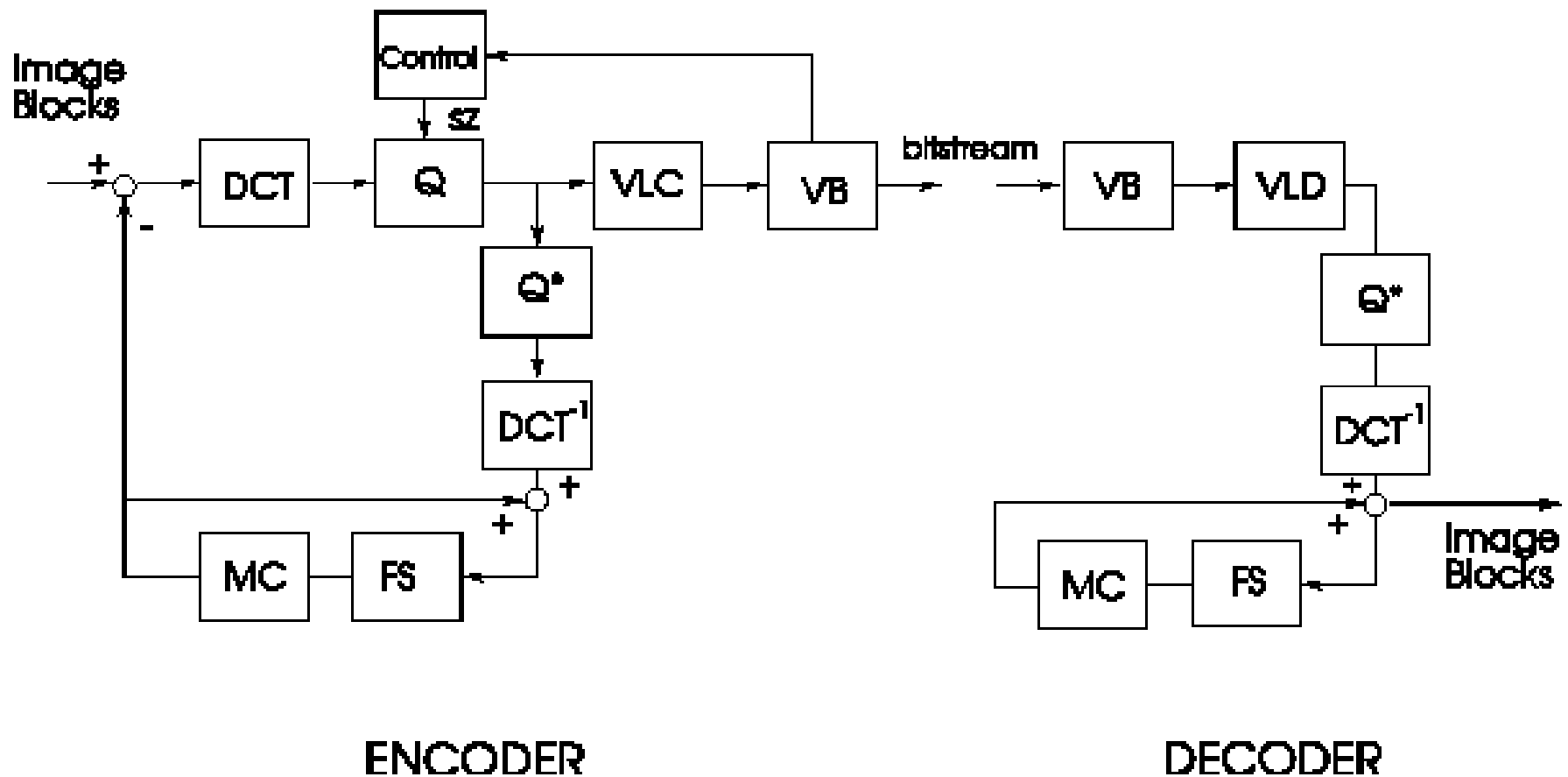


Abb.: GOPs in MPEG-Dateien

## Kompression in MPEG

- Es werden eine Reihe von Kompressionstechniken kombiniert angewandt, um eine hohe Kompressionsrate zu erzielen, insbes. DCT, Motion Compensation, Huffman, RLE.

### HYBRID DCT/DPCM CODING SCHEME



## Aufgaben

- 1) **Wiederholen** Sie den Stoff dieser Sitzung **bis zur nächsten Sitzung** (siehe dazu den Link zur Sitzung auf der HKI-Homepage). Informieren Sie sich zusätzlich durch eigene Literaturrecherche!
- 2) Beantworten Sie die Fragen aus der Sammlung „**beispielhafte Klausurfragen**“ zum Bereich **Video** (soweit in dieser Sitzung behandelt).