Digital Video Processing

Outline

- Video Compression
- Segmentation in Video
- Foreground and Background Segmentation Using Motion and Color
- Layered Video
- Digital Watermarking
- Video Streaming

Video Compression

- Problem:
  - Raw video contains an immense amount of data.
  - Communication and storage capabilities are limited and expensive.

- Example HDTV video signal:

– 720x1280 pixels/frame, progressive scanning at 60 frames/s:

$$\left(\frac{720\times1280\,pixels}{frame}\right)\left(\frac{60\,frames}{\sec}\right)\left(\frac{3colors}{pixel}\right)\left(\frac{8bits}{color}\right)=1.3Gb/s$$

– 20 Mb/s HDTV channel bandwidth

→ Requires compression by a factor of 70 (equivalent to .35 bits/pixel)

Video Compression (Contd..)

Bandwidth Reduction

Application	**Data Rate**	
	Uncompressed	**Compressed**
Video Conference *352 x 240 @ 15 fps*	30.4 Mbps	64 - 768 kbps
CD-ROM Digital Video *352 x 240 @ 30 fps*	60.8 Mbps	1.5 - 4 Mbps
Broadcast Video *720 x 480 @ 30 fps*	248.8 Mbps	3 - 8 Mbps
HDTV *1280 x 720 @ 60 fps*	1.33 Gbps	20 Mbps

Video Compression (Contd..)

Standards

STANDARD	APPLICATION	BIT RATE
JPEG	Continuous-tone still-image compression	Variable
H.261	Video telephony and teleconferencing over ISDN	p x 64 kb/s
MPEG-1	Video on digital storage media (CD-ROM)	1.5 Mb/s
MPEG-2	Digital Television	> 2 Mb/s
H.263	Video telephony over PSTN	< 33.6 kb/s
MPEG-4	Object-based coding, synthetic content, interactivity	Variable
H.264	From Low bitrate coding to HD encoding, HD-DVD, Surveillance, Video conferencing.	Variable

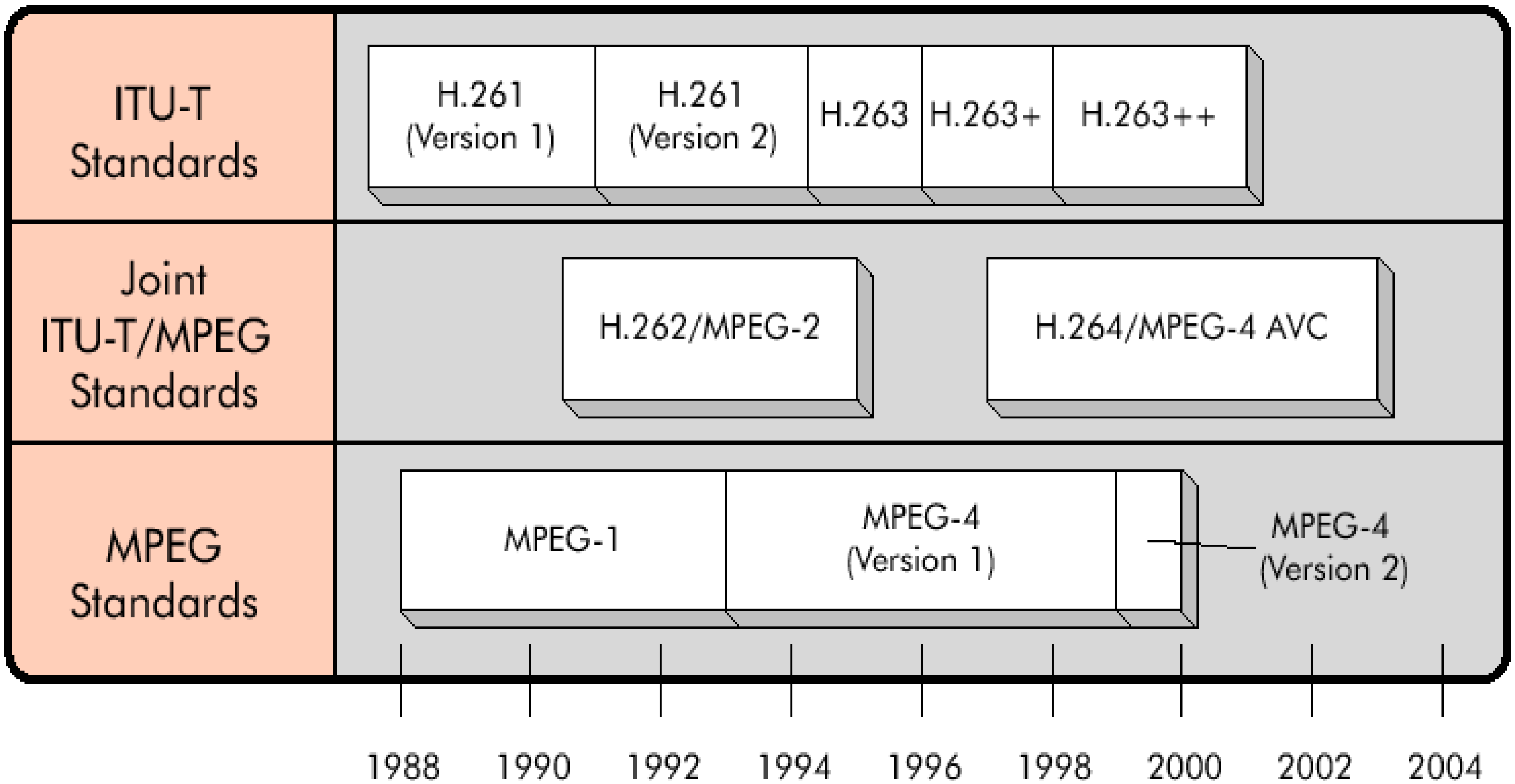
Video Compression (Contd..)

Goal of standards:

- Ensuring interoperability – Enabling communication between devices made by different manufacturers.
- Promoting a technology or industry.
- Reducing costs.

Video Compression (Contd..)

History of Video Standards



Video Compression (Contd..)

What Do the Standards Specify?

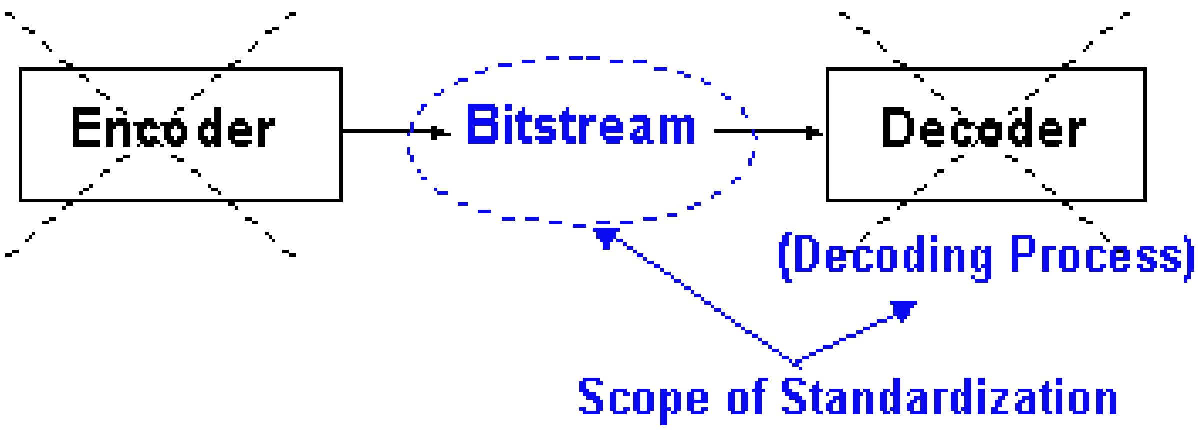
- A video compression system consists of the following:
  - An encoder
  - Compressed bit-streams
  - A decoder

- What parts of the system do the standards specify?

Video Compression (Contd..)

What Do the Standards Specify? (Contd..)

- Not the encoder, not the decoder.



Video Compression (Contd..)

What Do the Standards Specify? (Contd..)

- Just the bit-stream syntax and the decoding process, for example it tells to use IDCT, but not how to implement the IDCT.
- Enables improved encoding and decoding strategies to be employed in a standard-compatible manner.

Video Compression (Contd..)

Achieving Compression

- Reduce redundancy and irrelevancy.
- Sources of redundancy:
  - Temporal – Adjacent frames highly correlated.
  - Spatial – Nearby pixels are often correlated with each other.
  - Color space – RGB components are correlated among themselves.
- Irrelevancy – Perceptually unimportant information.

Video Compression (Contd..)

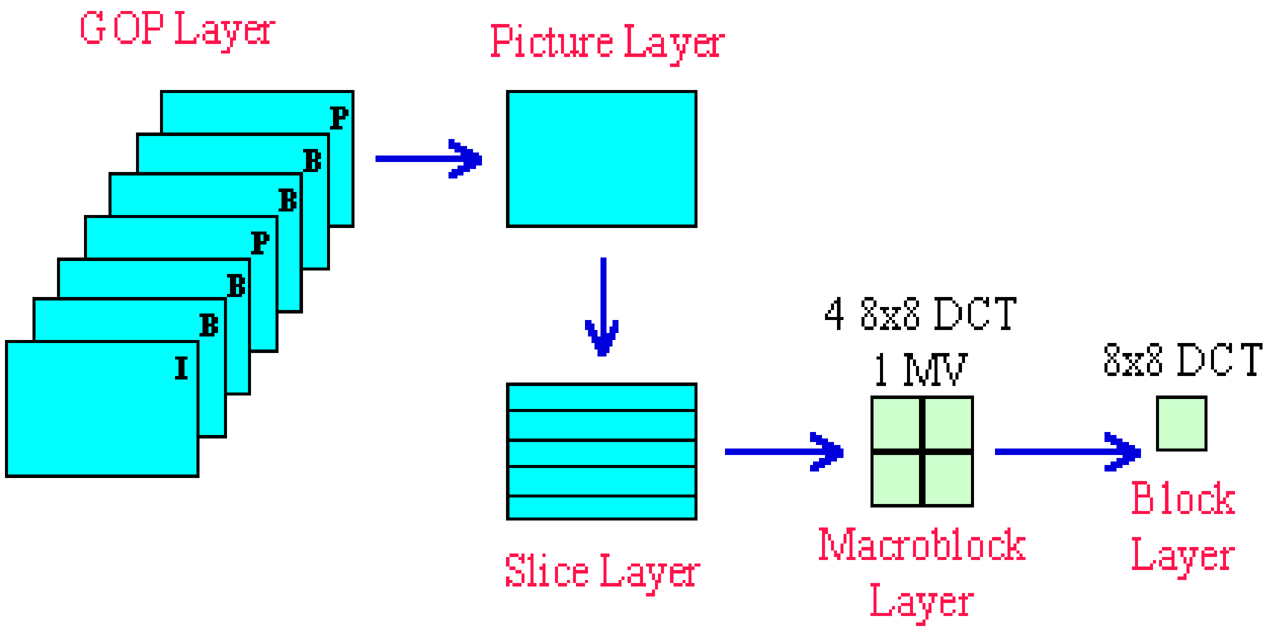
Basic Video Compression Architecture

- Exploiting the redundancies
  - Temporal – MC-prediction and MC-interpolation
  - Spatial – Block DCT
  - Color – Color space conversion
- Scalar quantization of DCT coefficients
- Run-length and Huffman coding of the non-zero quantized DCT coefficients

Video Compression (Contd..)

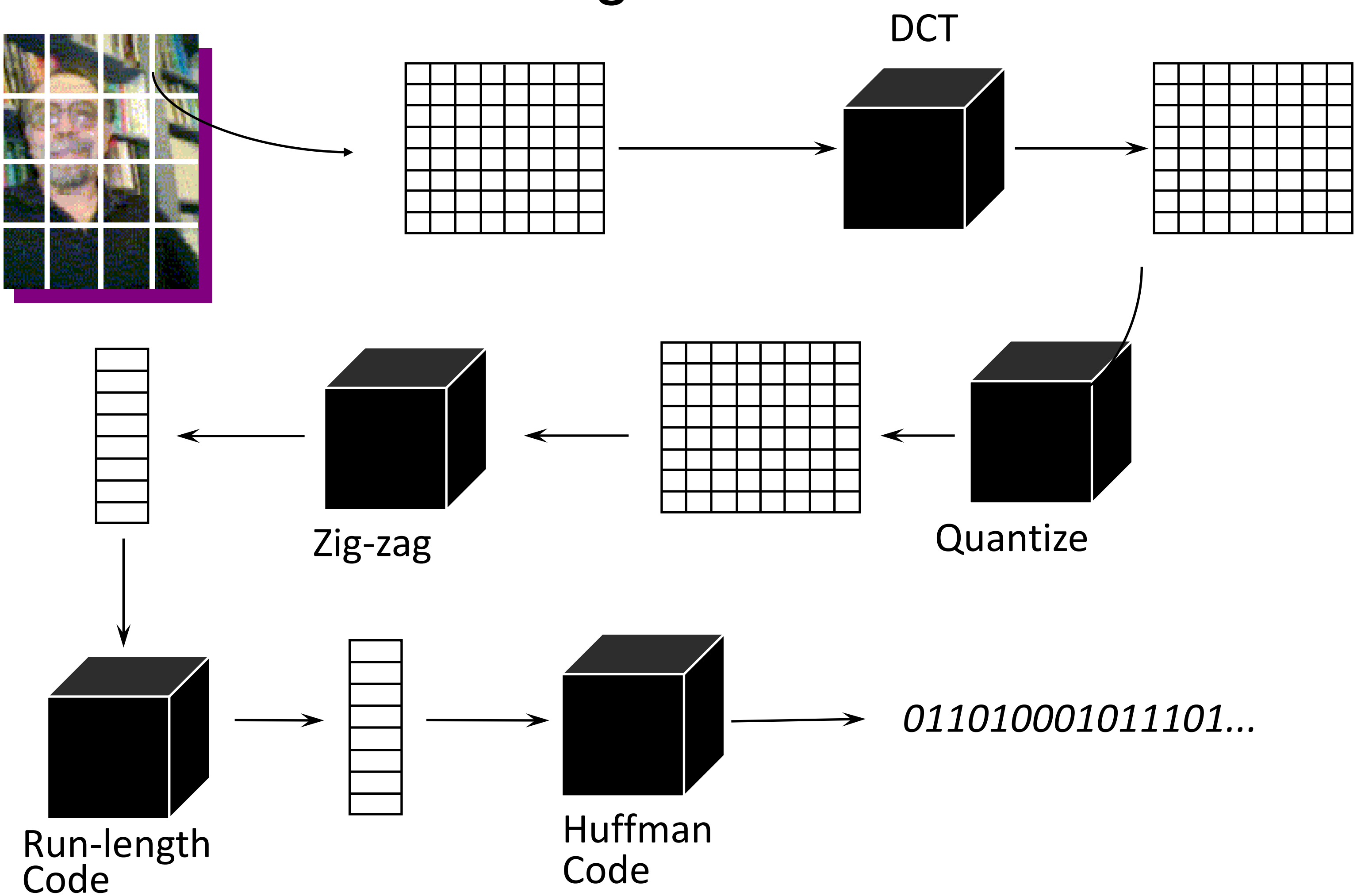
Video Structure

- MPEG Structure



Video Compression (Contd..)

Block Transform Encoding



Video Compression (Contd..)

Block Encoding

$$\begin{bmatrix} 139 & 144 & 149 & 153 \\ 144 & 151 & 153 & 156 \\ 150 & 155 & 160 & 163 \\ 159 & 161 & 162 & 160 \end{bmatrix}$$

original image

DCT →

$$\begin{bmatrix} 1260 & -1 & -12 & -5 \\ -23 & -17 & -6 & -3 \\ -11 & -9 & -2 & 2 \\ -7 & -2 & 0 & 1 \end{bmatrix}$$

DC component (1260)

AC components

Quantize →

$$\begin{bmatrix} 79 & 0 & -1 & 0 \\ -2 & -1 & 0 & 0 \\ -1 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

zigzag →

$$\begin{bmatrix} 79 & 0 & -2 & -1 & -1 & -1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

run-length code →

$$\begin{bmatrix} 0 & 79 \\ 1 & -2 \\ 0 & -1 \\ 0 & -1 \\ 0 & -1 \\ 2 & -1 \\ 0 & 0 \end{bmatrix}$$

Huffman code → *10011011100011...*

coded bitstream < 10 bits (0.55 bits/pixel)

Video Compression (Contd..)

Result of Coding/decoding

$$\begin{bmatrix} 139 & 144 & 149 & 153 \\ 144 & 151 & 153 & 156 \\ 150 & 155 & 160 & 163 \\ 159 & 161 & 162 & 160 \end{bmatrix}$$

original block

$$\begin{bmatrix} 144 & 146 & 149 & 152 \\ 156 & 150 & 152 & 154 \\ 155 & 156 & 157 & 158 \\ 160 & 161 & 161 & 162 \end{bmatrix}$$

reconstructed block

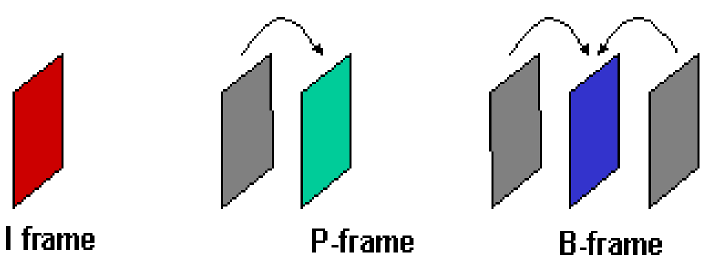
$$\begin{bmatrix} -5 & -2 & 0 & 1 \\ -4 & 1 & 1 & 2 \\ -5 & -1 & 3 & 5 \\ -1 & 0 & 1 & -2 \end{bmatrix}$$

errors

Video Compression (Contd..)

Video Compression

- Main addition over image compression
  - Exploit the temporal redundancy
- Predict current frame based on previously coded frames
- Types of coded frames:
  - I-frame – Intra-coded frame, coded independently of all other frames
  - P-frame – Predictively coded frame, coded based on previously coded frame
  - B-frame – Bi-directionally predicted frame, coded based on both previous and future coded frames

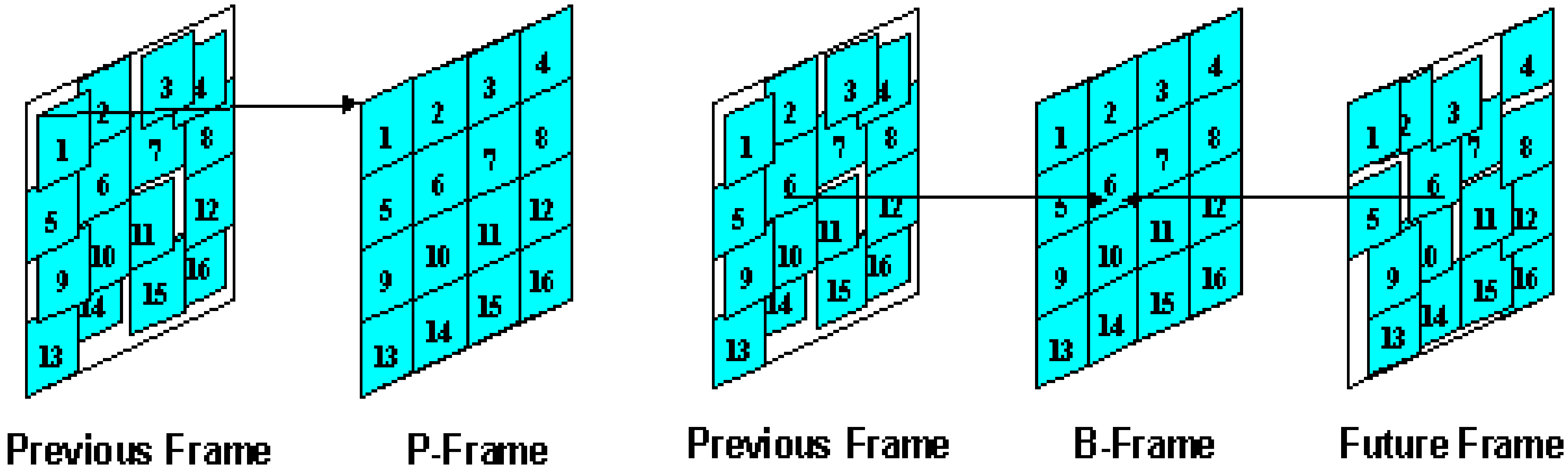


Video Compression (Contd..)

Motion Compensated Prediction (P and B Frames)

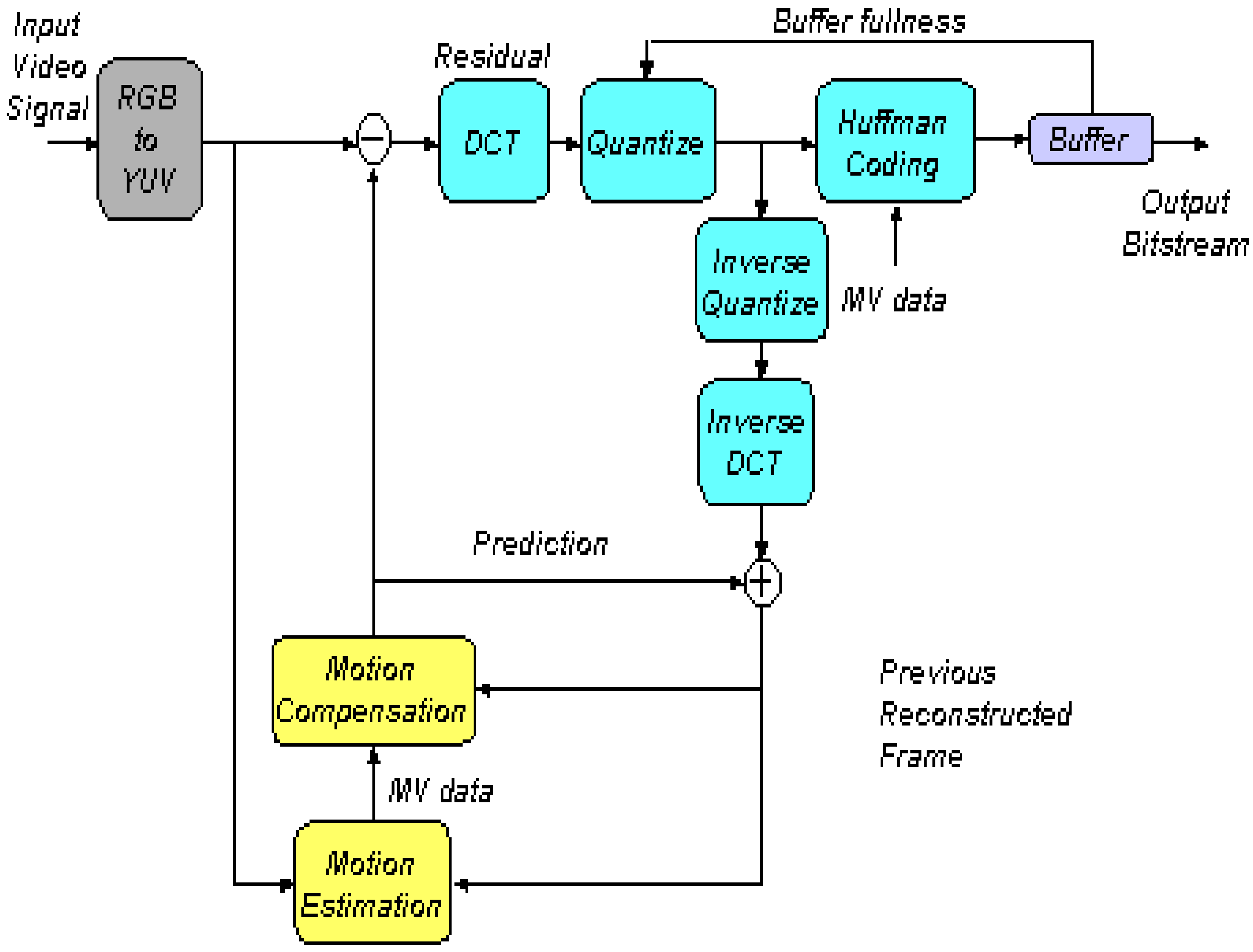
- Motion compensated prediction – predict the current frame based on a reference frame while compensating for the motion

- Examples of block-based motion-compensated prediction



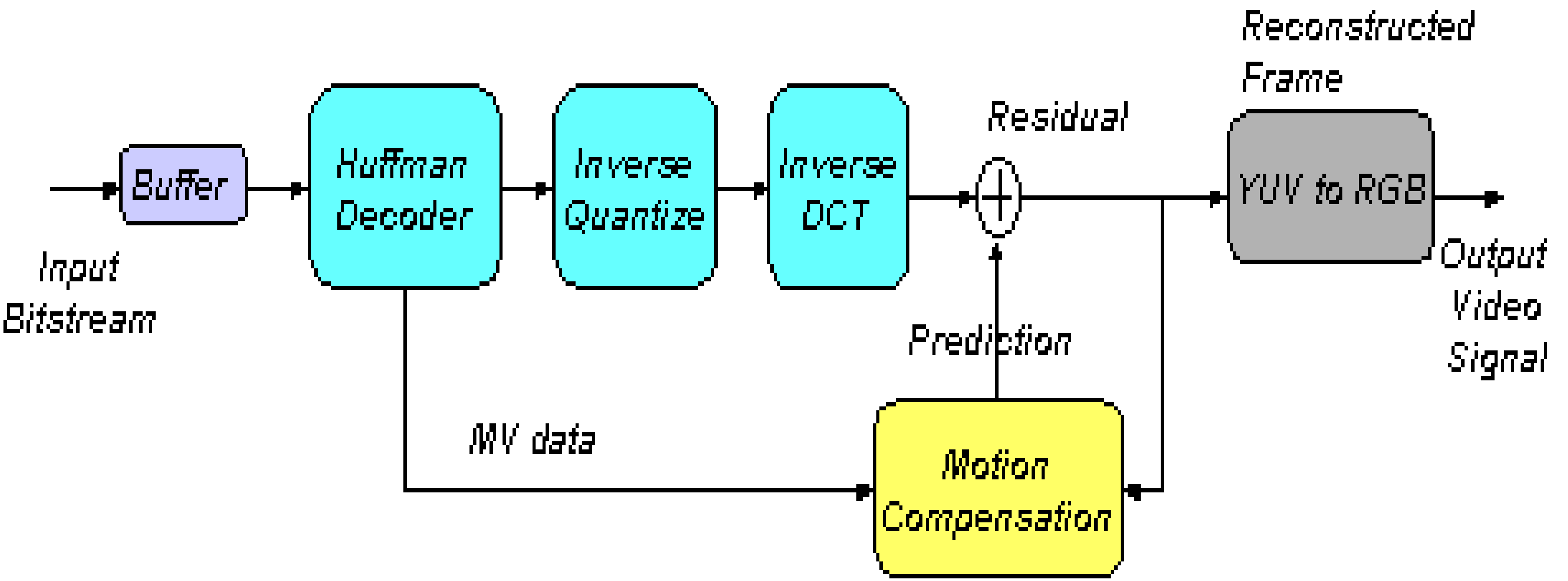
Video Compression (Contd..)

Example Video Encoder



Video Compression (Contd..)

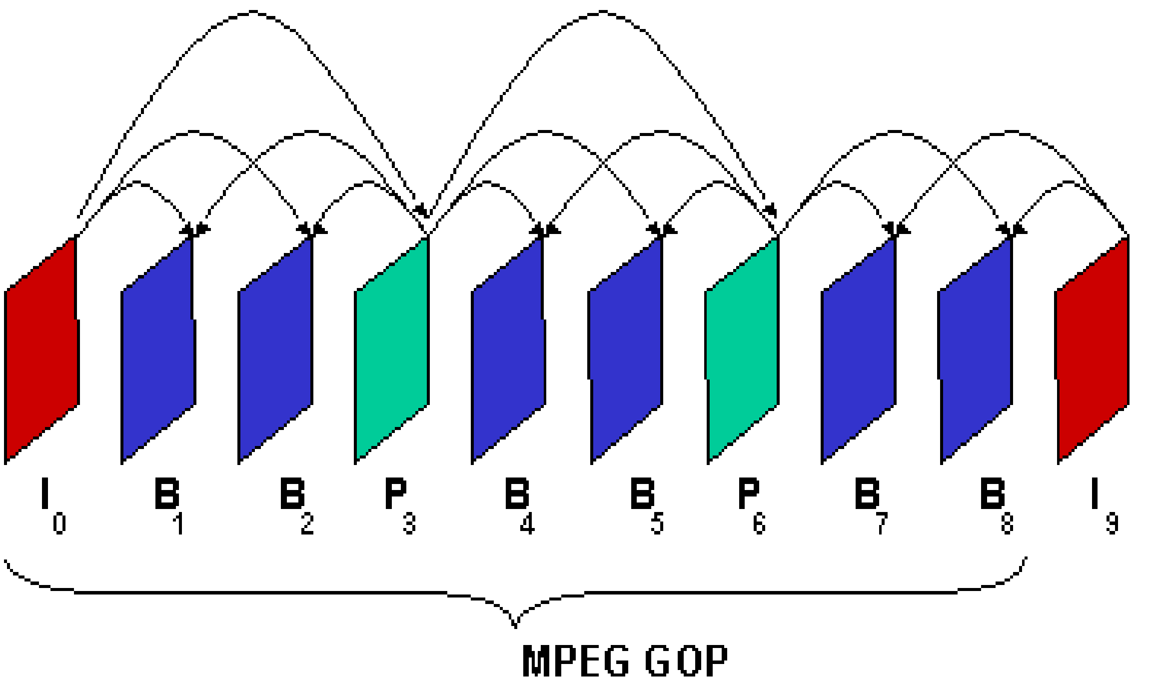
Example Video Decoder



Video Compression (Contd..)

Group of Pictures (GOP) Structure

- Enables random access into the coded bit-stream.
- Number of B frames and impact on search range.



Video Compression (Contd..)

Current Video Compression Standards

- Classification & Characterization of different standards
  - Based on the same fundamental building blocks
    - Motion-compensated prediction and interpolation
    - 2-D Discrete Cosine Transform (DCT)
    - Color space conversion
    - Scalar quantization, run-length, and Huffman coding
  - Other tools added for different applications
    - Progressive or interlaced video
    - Improved compression, error resilience, scalability, etc

Video Compression (Contd..)

Compressed Video

- To manage storage and transmission
- Several standards but MPEG is widely used
- Different MPEG versions
  - MPEG-1
  - MPEG-2
  - MPEG-4 "Coded representation of media objects"
  - MPEG-7 "Multimedia Content Description Interface"

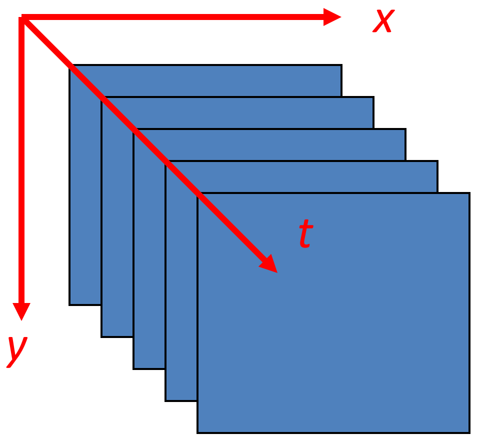
Outline

- Video Compression
- Segmentation in Video
- Foreground and Background Segmentation Using Motion and Color
- Layered Video
- Digital Watermarking
- Video Streaming

Segmentation in Video

Videos

- Videos are **Image Sequences over Time**



- An image is a function over *x* and *y*:

$$f(x, y)$$

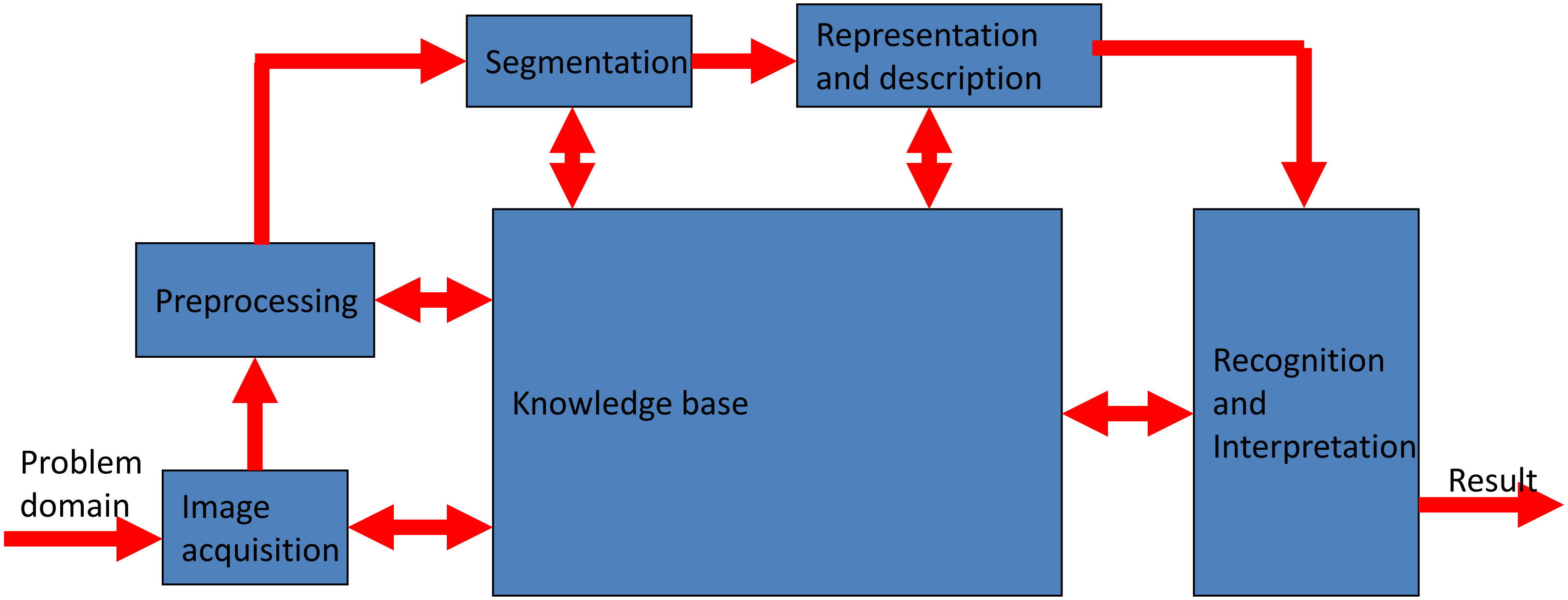
- A video is a function of images over time *t*:

$$f(x, y, t) = f_t(x, y)$$

- At each time step *t* we have an image
- Framerate = the number of images per second, e.g., 25 Images/s

Segmentation in Video (Contd..)

- General Segmentation is ***application-dependent***!
  Knowledge base!!
- Finding the object(s)
  - Preprocessing, Segmentation
- Tracking = follow the object(s) over time:
  - being able at each time *t* to give, e.g, position, color, orientation, etc.
  - Representation, Description



Segmentation in Video (Contd..)

Lots of applications!! One is:

- Separation of **Foreground** (object) and **Background** (everything else = noise)
- Result could be a
  - Binary image, containing foreground only
  - Probability image, containing the likelihood of each pixel being foreground
- Useful for further processing, such as using silhouettes, etc.
- Approaches
  - Motion-based
  - Color-based
  - Some approaches can learn!

Foreground-Background Segmentation using Motion and Color

- Motion-based
  - Image differencing
    - Model-free
    - No learning
- Color-based
  - Background subtraction
    - Background used as a model
    - No learning
  - Advanced background subtraction
    - Background is learned
  - Very advanced background subtraction
    - Background is learned

Foreground-Background Segmentation using Motion and Color (Contd..)

Image Differencing

- The motion in an image can be found by subtracting the current image from the previous image

- Algorithm
  1. Save image in last frame
  2. Capture current camera image
  3. Subtract image (= difference = motion)
  4. Threshold
  5. Delete noise

Foreground-Background Segmentation using Motion and Color (Contd..)

3. Subtract Image

- Compute pixel-wise
- Subtract previous image from input image:

$$F(x,y) = I(x,y) - B(x,y)$$

- Usually the absolute distance is applied

$$|F(x,y)|$$

1. Save image in last frame
2. Capture camera image
3. **Subtract image**
4. Threshold
5. Delete noise

Foreground-Background Segmentation using Motion and Color (Contd..)

4. Threshold

- Decide, when a pixel is supposed to be considered as a background pixel, or when it is to be considered as a foreground pixel:
- Pixel (x, y) is foreground pixel, if

$$|F(x, y)| > TH$$

- Pixel (x, y) is background pixel, if

$$|F(x, y)| \leq TH$$

- Problem: What TH??

1. Save image in last frame
2. Capture camera image
3. Subtract image
4. **Threshold**
5. Delete noise

Foreground-Background Segmentation using Motion and Color (Contd..)

5. Deleting Noise

- Singular pixels are likely to appear:
  - Pixel-noise!!
- Apply Median filter:
  - Depending on filter size, bigger spots can be erased
- Alternative: Morphologic

1. Save image in last frame
2. Capture camera image
3. Subtract image
4. Threshold
5. **Delete noise**

Foreground-Background Segmentation using Motion and Color (Contd..)

Background Subtraction

- Foreground is moving, background is stable

- Algorithm

1. **Capture image containing background**
2. Capture camera image
3. Subtract image (= difference = motion)
4. Threshold
5. Delete noise

Foreground-Background Segmentation using Motion and Color (Contd..)

Advanced Background Subtraction

- What if we have small motion in the background?
  - Bushed, leaves, etc. and noise in the camera/lighting
  - (show histo patch)
- **Learn** the background
- Capture *N* images and calculate the average background image (no object present)

1. **Calculate average background image**
2. Capture camera image
3. Subtract image (= motion)
4. Threshold
5. Delete noise

Foreground-Background Segmentation using Motion and Color (Contd..)

Very Advanced Background Subtraction

- **Use Neighborhood relation!!**
  - Compare pixel with its neighbors!!
  - **Weight them!!**
- **Learn the background and its variations!!**
  - E.g. Gaussian models (mean,var) for each pixel!!!
  - E.g. a Histogram for each Pixel
  - The more images you train on the better!!
  - Idea:
    - Some pixel may vary more than other pixels
  - Algorithm:
    - Consider each pixel (x,y) in the input image and check, how much it varies with respect to the mean and variance of the learned Gaussian models?

1. **Calculate mean and variance for each pixel**
2. Capture camera image
3. Subtract image (= motion)
4. **Weight the distances (new)**
5. **Threshold according to variance**
6. Delete noise

Foreground-Background Segmentation using Motion and Color (Contd..)

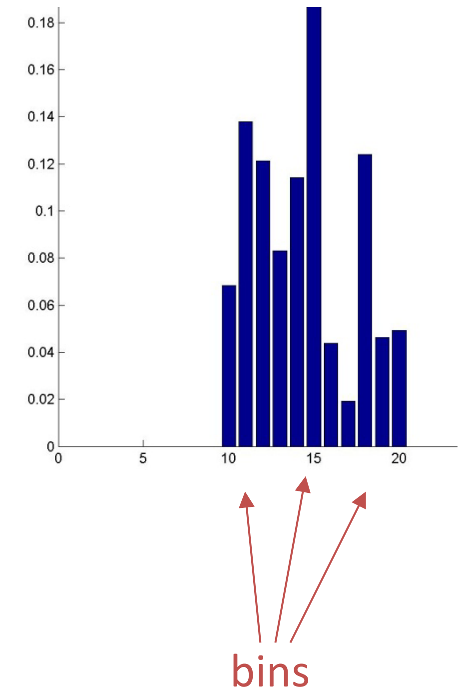
Segmentation using Color Histogram

- Given an object, defined by its color Histogram
- Algorithm:
  - Segment each pixel in the input image according to the histogram
- The result: binary or probability image.
  - White pixels: pixel in input image had color as defined by the histogram
  - Black pixel: pixel in input image did not have color defined by the histogram

Foreground-Background Segmentation using Motion and Color (Contd..)

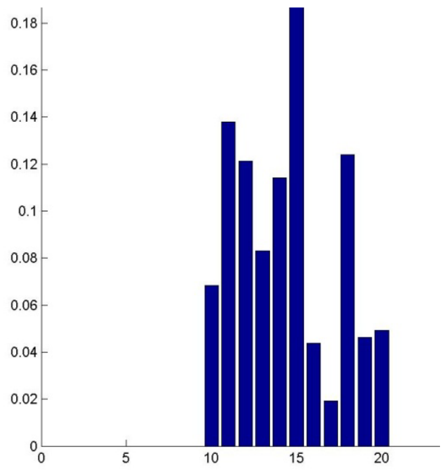
Learning the Color Histogram

- The color histogram summarizes the color distribution in an image (region)
- There is a histogram bin for each possible color!
- The columns of the histogram are high for those colors appearing often in the image (region) and low for those appearing seldom in the image.
- The more images you train on the better!!



Foreground-Background Segmentation using Motion and Color (Contd..)

Color Segmentation with Histograms



- Algorithm:
  - For each pixel in the input *I(x,y)*
    - Go to the bin having the same color as the pixel *H(I(x,y))*
    - Assign the probability value at the bin to the output image: *O(x,y)*
      - *O(x,y) = H(I(x,y))*
    - Using the above leads to a probability image
  - Run a threshold on the output image to get a binary image

Foreground-Background Segmentation using Motion and Color (Contd..)

Segmentation with Gaussian Distribution

- Given an object, calculate the mean and standard deviation of its color
  - Represent each color component (e.g., R,G,B) by a **Gaussian model**
  - That is, a Gaussian distribution: (mean,sigma)
- The same principle as in "very advanced background subtraction" !!
- Algorithm:
  - Given an input image, segment each pixel by comparing it to the Gaussian models of the object color
- For example:
  - If $Th_{min} < dist < Th_{max}$ => object pixel
  - Th_{min} = mean – 2s
  - Th_{max} = mean + 2s

Foreground-Background Segmentation using Motion and Color (Contd..)

Weight the Distances, Correlation between Pixel values

- If one pixel is considered to be a foreground pixel
  - Its neighbor is also likely to be a foreground pixel
  - If its neighbor is not considered to be a foreground pixel, one of the two might be wrong
  - Neighboring pixels are highly **correlated** (similar)

1. Calculate mean and variance for each pixel
2. Capture camera image
3. Subtract image (= motion)
4. **Weight the distances (new)**
5. Threshold according to variance
6. Delete noise

Foreground-Background Segmentation using Motion and Color (Contd..)

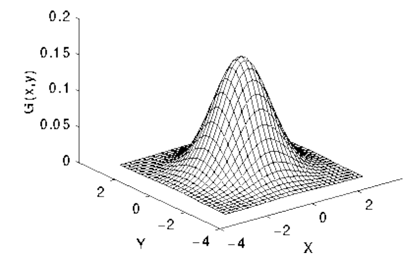
Weight the Distances

- What does a pixel say about a foreground pixel, that is further away?
  - Pixels with an increasing distance to each other are saying less about each other
  - The correlation between pixels decreases with distance

1. Calculate mean and variance for each pixel
2. Capture camera image
3. Subtract image (= motion)
4. **Weight the distances (new)**
5. Threshold according to variance
6. Delete noise

Foreground-Background Segmentation using Motion and Color (Contd..)

- Use a Gaussian for Weighting
- To test a pixel *I(x,y)*
  – Center a Gaussian on this pixel and weight the neighboring pixels accordingly => Convolution!



- 1D example:

Signal:

60	48	2	3	1	222	100

Gaussian filter:

0.25	0.5	0.25

Output:

	39.5	13.75	2.25	56.75	136.3	

Foreground-Background Segmentation using Motion and Color (Contd..)

- **Likelihood image** or **Probability image**, containing the likelihood for each pixel of being a background/foreground pixel

1. Calculate mean and variance for each pixel
2. Capture camera image
3. Subtract image (= motion)
4. **Weight the distances (new)**
5. Threshold according to variance
6. Delete noise