# M30242 Graphics and Computer Vision



## Lecture 9 Motion Detection and Feature Tracking

# Introduction

- Motion detection and feature tracking deal with the issues of motion calculation.
- Normally, we want to know the direction and speed of objects. Sometimes we just want to know if a scene is indeed static and but have no interest in the nature of the motion.
- We normally need to work with video instead of still images.
- Applications
  - Surveillance, security
  - Robotics, traffic control, military, …
  - Reconstruction of 3D structures

# Existing Methods

- Many methods have been developed and new methods are emerging.
  - Many are application specific.
  - Many are modifications of existing methods.
- But we don't see ground-breaking advances often.
- In this lecture, we introduce the principles of popular, easy-to-implement methods. A thorough treatment of the topic is beyond the scope of this unit.

# Categories of Methods

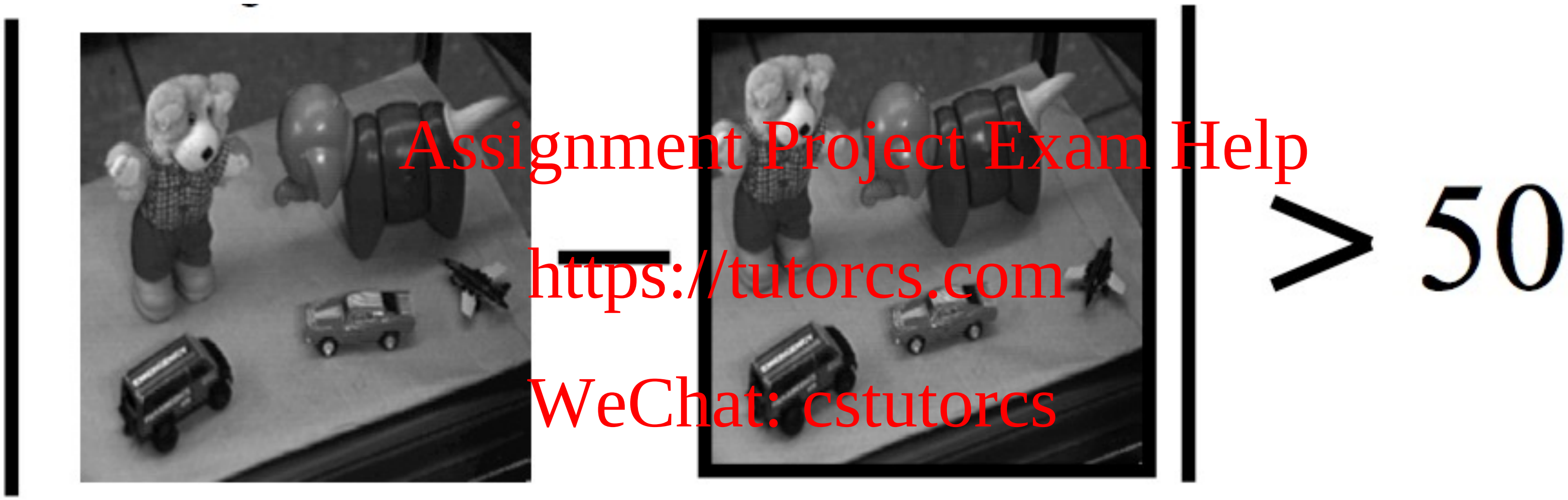
- Image subtraction (image difference)
  - Computing the differences between the consecutive frames.
  - Not suitable for applications where quantitative direction, speed or geometric information (size, distance, etc) are essential.
  - Stationary camera.
- Optical flow (next lecture)
  - Detect relative motions (camera v.s. scene or vice versa).
  - Stationary or moving camera, where relative motions are important, e.g., in robotics.
- Feature tracking: tracking a small set of salient points/features
  - Camera and/or scene objects are moving.
  - Most useful but the hardest (depending upon the features being tracked).

# Motion Detection by Image Difference

- Image subtraction is a very simple strategy for motion detection:
  - perform pixel-wise subtraction between the current frame with the previous frame ( or a pre-selected reference image),
  - threshold the resulted difference values.



# Cont’d





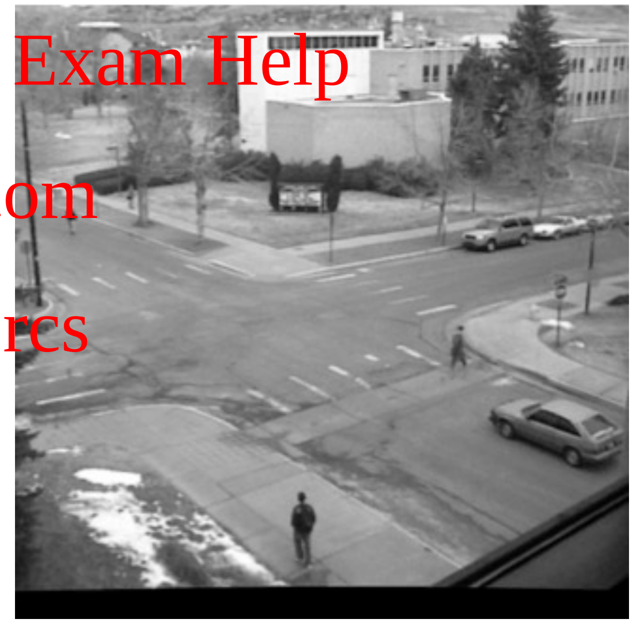

Candidate areas for motion

# Application

Traffic monitoring at a junction



Reference image (a frame
at an earlier time/instant)



A video frame at a different
time

# Result



Obviously, for the method to produce reliable results, the background must be very stable.

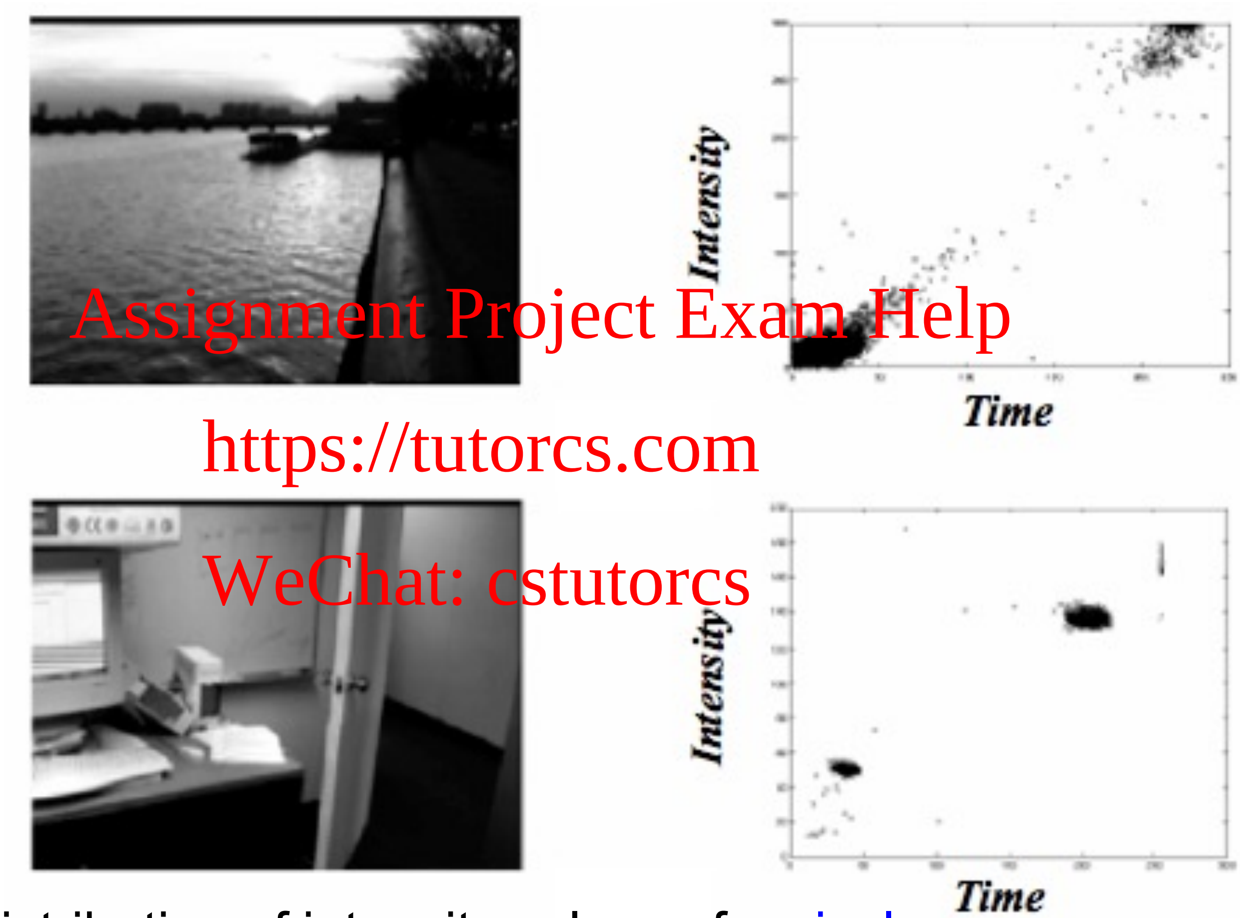
# Difficulties with Reference Images

- Some background pixels are not described by constant values:
  - motion: background is usually not static, e.g., trees in the wind.
  - lighting (clouds, shadows) changes intensity values and highlights.

# Difficulties with Reference Images

- Some background pixels are not described by constant values:
  - motion: background is usually not static, e.g., trees in the wind.
  - lighting (clouds, shadows) changes intensity values and highlights.

# Non-static Reference Images






A bi-modal distribution of intensity values of a pixel:

Top: image of water surface from a stationary camera.

Bottom: images of monitor flicker.

What intensity value should be used as a reference?

# Modelling Reference Images

- In such scenes, the background pixels are not described by fixed intensity values. The values vary from frame to frame.
- To get a good estimate of the values, the statistics of pixel values of the frames *over a period* could be used:
  - the average of the pixel intensities of a set images, or
  - more sophisticated statistics.

# Pixel Average

- Taking the average over a set of N images

$$I_{ref}(x,y) = (1/N) \sum I_i(x,y)$$



- This can be easily done in Matlab by calling function

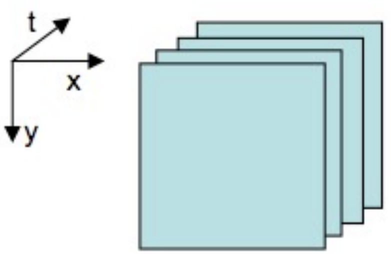


```
mean(M, dim)
```



where M(x,y,time) are the "stack" of images/frames. The x-and y are dimensions of the images/frames, the 3rd dimension is time. Setting dim=3 means that average is taken along the 3rd dimension (time).

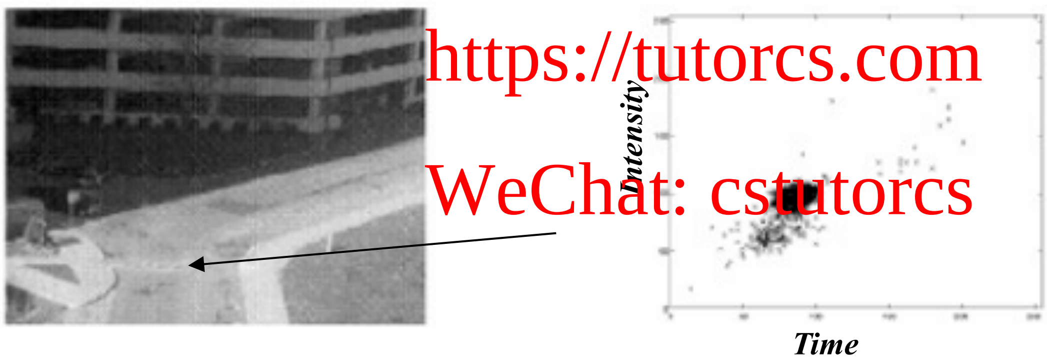
M(x,y,time)



# Intensity Distribution

- A more sophisticated method is to decide the reference intensities according to the probabilities of the intensity distributions of the pixel values.






Intensity distribution of a single pixel during different times of a day

- Statistics can predict an expected intensity value for that pixel.
  - More info see Stauffer and Grimson, "Learning Patterns of Activities Using Real-time Tracking," *IEEE Trans on Pattern Analysis and Machine Vision*, v.22 no.8, 2000)

# Statistics-Based Approaches

- Statistics-based methods can better handle the problem of unstable scene/images.



- Simple statistics such as the average difference of two frames can be used to detect the significant changes in video streams, e.g., change of scenes.

# Average Difference

- The *average of the differences* between two frames at instant $I_t$ and $I_{t+\Delta}$ is defined as:



$$d_{pixel}(I_t, I_{t+\Delta}) = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N} \left| I_t(i,j) - I_{t+\Delta}(i,j) \right|}{MN}$$

  Where $i, j$ are the row and column indices of a pixel, $M$x$N$ are the image size (total number of pixels).

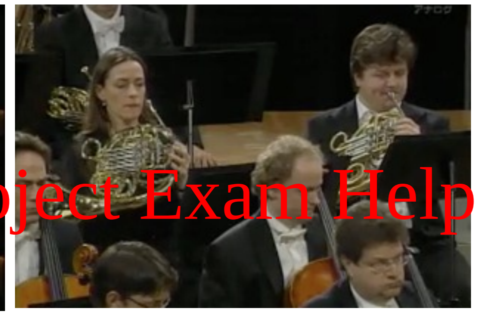
- That is, the average difference is the sum of the differences between the corresponding pixels divided by the total number of pixels.

# Applications

- Average difference can be used to detect shot boundaries of videos.
- Shot boundaries of videos are time points where significant changes happens. It can be caused by
  - Actual scene changes such as intrusion of objects
  - Camera actions: pan: sweeping a horizontal view of the scene, zoom
  - Video editing effects: fading, dissolving, and wiping
- The detected shot boundaries can be time-stamped to support random/quick access of videos, for example, in digital libraries. It becomes increasingly important for analysis of surveillance videos

# An Example











# Problems with Average Difference

- Average difference tends to produce a large difference when there is even a small amount of camera pan/zoom.



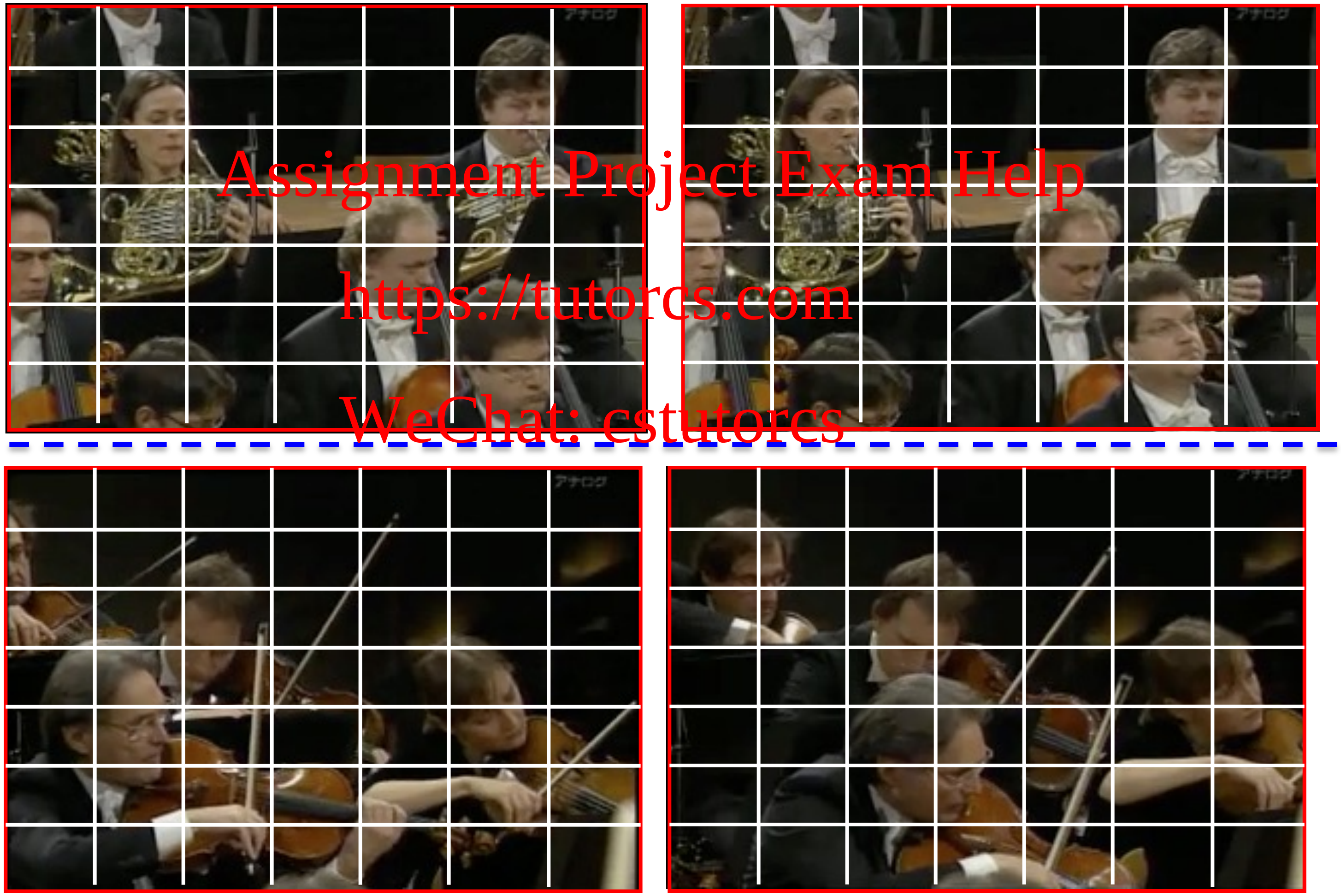
 

# Block-Based Statistics

- A more robust variation of the method is to break the images into large blocks and test to see if most of the blocks are essentially the same in both images.
- If enough blocks have very small differences, two images are considered to belong to the same shot.



# Cont’d



# Block Statistics

- When comparing two blocks, the statistics of the blocks, means and variances, are used.
- The similarity (or distance) between two blocks is calculated as



$$p_{block}(B_1, B_2) = \frac{\left[ \frac{\sigma_1^2 + \sigma_2^2}{2} + \left( \frac{\mu_1 - \mu_2}{2} \right)^2 \right]^2}{\sigma_1^2 \sigma_2^2}$$

  – Where $\mu_1$, $\sigma_1^2$ and $\mu_2$, $\sigma_2^2$ are the means and variances of the pixel intensities of block $B_1$ and $B_2$

# Variance of N Samples

- The variance, $\sigma^2$, of $N$ samples is defined as the average of the squared difference between the samples and their mean, i.e.,



$$\sigma^2 = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2 = \frac{1}{N}\sum_{i=1}^{N}x_i^2 - \mu^2$$

  - where $x_i$ is the value of a sample (e.g., intensity value of a pixel), and
  - $\mu$ is the mean:

$$\mu = \frac{1}{N}\sum_{i=1}^{N}x_i$$

$$\begin{aligned}
\sigma^2 &= \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2 \\
&= \frac{1}{N}\sum_{i=1}^{N}\left(x_i^2 - 2\mu x_i + \mu^2\right) \\
&= \frac{1}{N}\sum_{i=1}^{N}x_i^2 - \frac{2\mu}{N}\sum_{i=1}^{N}x_i + \frac{\mu^2}{N}\sum_{i=1}^{N}1 \\
&= \frac{1}{N}\sum_{i=1}^{N}x_i^2 - 2\mu^2 + \mu^2 \\
&= \frac{1}{N}\sum_{i=1}^{N}x_i^2 - \mu^2
\end{aligned}$$

# Block Distance & Image Difference

- Normally, a threshold is set for the distance value between the blocks. As a result, the block distance is binary: 



$$d_{block}(B_1,B_2) = \begin{cases} 1 & if\ p > t \\ 0 & if\ p \leq t \end{cases}$$



  - Where $t$ is some threshold value
- The difference between two images (frames) is defined as the sum of the block distances of the entire image.

$$d(I_1,I_2) = \sum_{i=1}^{all\_block} d_{block}(B_{iI_1}, B_{iI_2})$$

# Other Statistics for Block Comparison

- Other image statistics, e.g., block histograms, could also be used for image comparison.
  - *Intersection* and *match* of two histograms (see lecture on colour image analysis).

# Summary

- This category of methods are simple in principle and easy to implement.
  - Simple configuration and calculation.
- Limitations
  - No speed or directional information of motions.

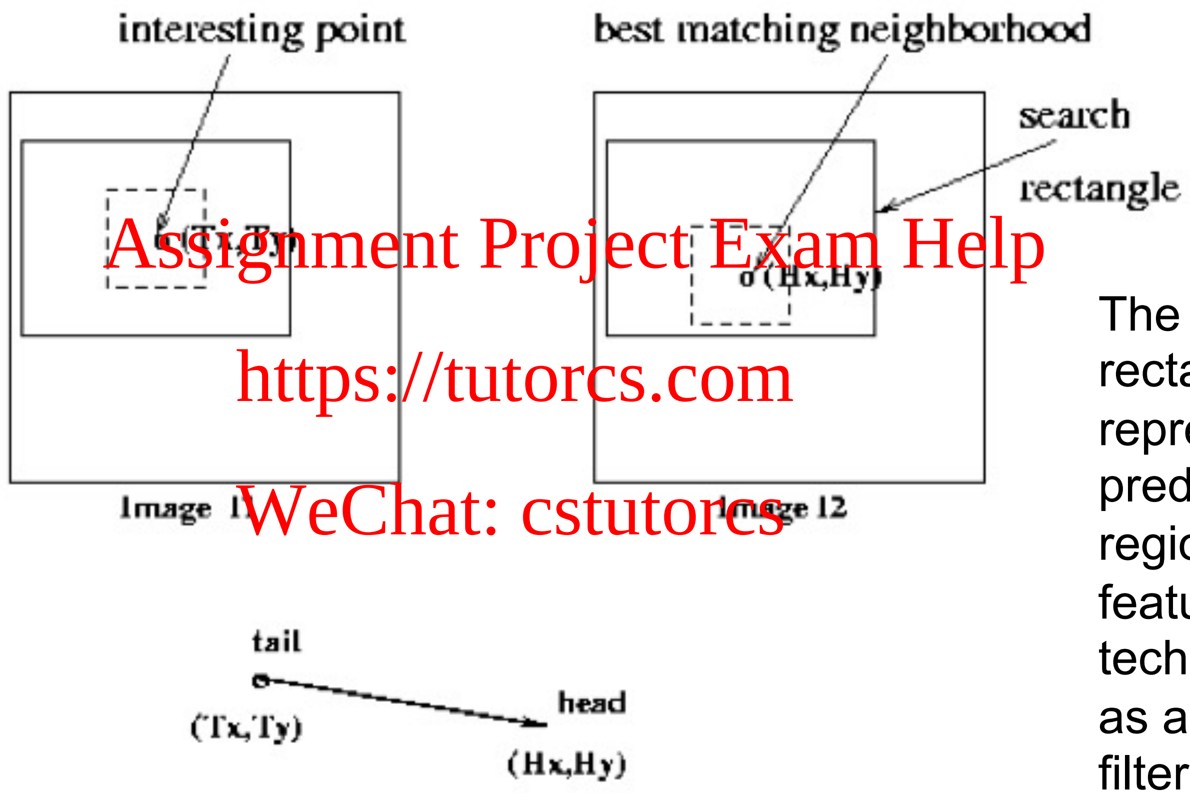
# Motion Detection By Feature Tracking

- This category involves many methods. The methods usually work by
  - Tracking a small set of salient features on the objects across frames.
  - Extracting motions of the objects from the motions of the features.
- Different feature tracking methods exist, but their effectiveness and reliability are very much application dependent.
- Can be difficult to implement, especially in 3D
- Lots of research had been done, lots of problems remain.

# Tracking by Template Matching

- The approach works by first selecting a small image region/patch that contains some unique features, e.g., a region unique in shape, intensity or colours. The selected region is called a template.
- The template is then used in searching other images/frames for finding the location of the same feature by matching.
- The commonly used matching criteria is the cross-correlation between the pixel values of the template and the image region being compared. Strong correlation is expected when a match is found.
- Methods in this category are called template matching.

# Cont’d






The search rectangle represents the predicted target region of the feature using techniques such as a Kalman filter.

The motion vector obtained by tracking the feature using template matching

# Cross Correlation

- One of the criteria for cross-correlation is to minimse the sum of squared differences (SSD)

$$SSD = \sum_{x,y} \left( f(x,y) - h(x,y) \right)^2$$



  – Here $h(x,y)$ and $f(x,y)$ are intensities of the corresponding pixels of the template and the corresponding image region.
- Note that, the calculation assumes that only translation has occurred!
- This criterion may produce good result when the conditions are met (i.e., no rotation, scaling, or other shape transformations).

# Tracking Salient Points

- This is a variation of template matching.
- Salient points: points that are locally unique.

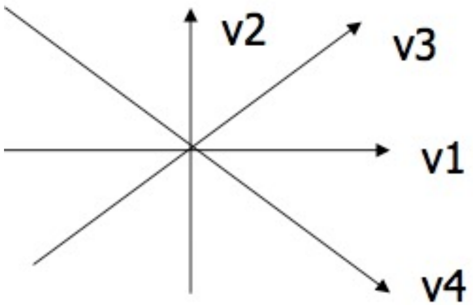
- They can be different things, e.g., corners, local maxima of gradient, etc.

- Therefore, the detection methods are different, depending upon what you choose as salient points.

- Your own observation and understanding of the nature of the application problems are important.

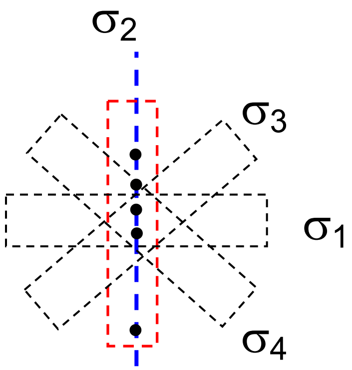
# Detecting Salient Points By Computing Variances

- A point can be considered as salient if the variances (changes in intensity values) around it are high.
- The variance "around it" are calculated in pre-defined orientations:
  - Vertical, horizontal, and diagonal directions are typical.
- If these variances are high enough (above some threshold), then the point is considered to a salient point.



# Variance Calculation

- Moravec Interest Operator (MIO)
  - Computing variances along 4 directions at each image pixel.
  - If the minimum of the 4 variances exceeds some threshold, then the pixel qualifies for a salient point.
- Variance computation along a direction:
  - choose a window along a direction, e.g., a horizontal window of 1xN pixels.
  - compute the mean and the sum of squares of differences of the pixels within the local windows (as defined before).




$$\sigma^2 = \frac{1}{N}\sum_{i=1}^{N}(x_i - \mu)^2 = \frac{1}{N}\sum_{i=1}^{N}x_i^2 - \mu^2$$

# Further Reading

- Shapiro, L.G., Stockman, G.C., Computer Vision, Prentice-Hall, 2001, ISBN 0-13-030796-3
  - Chapter 9