

Video Processing

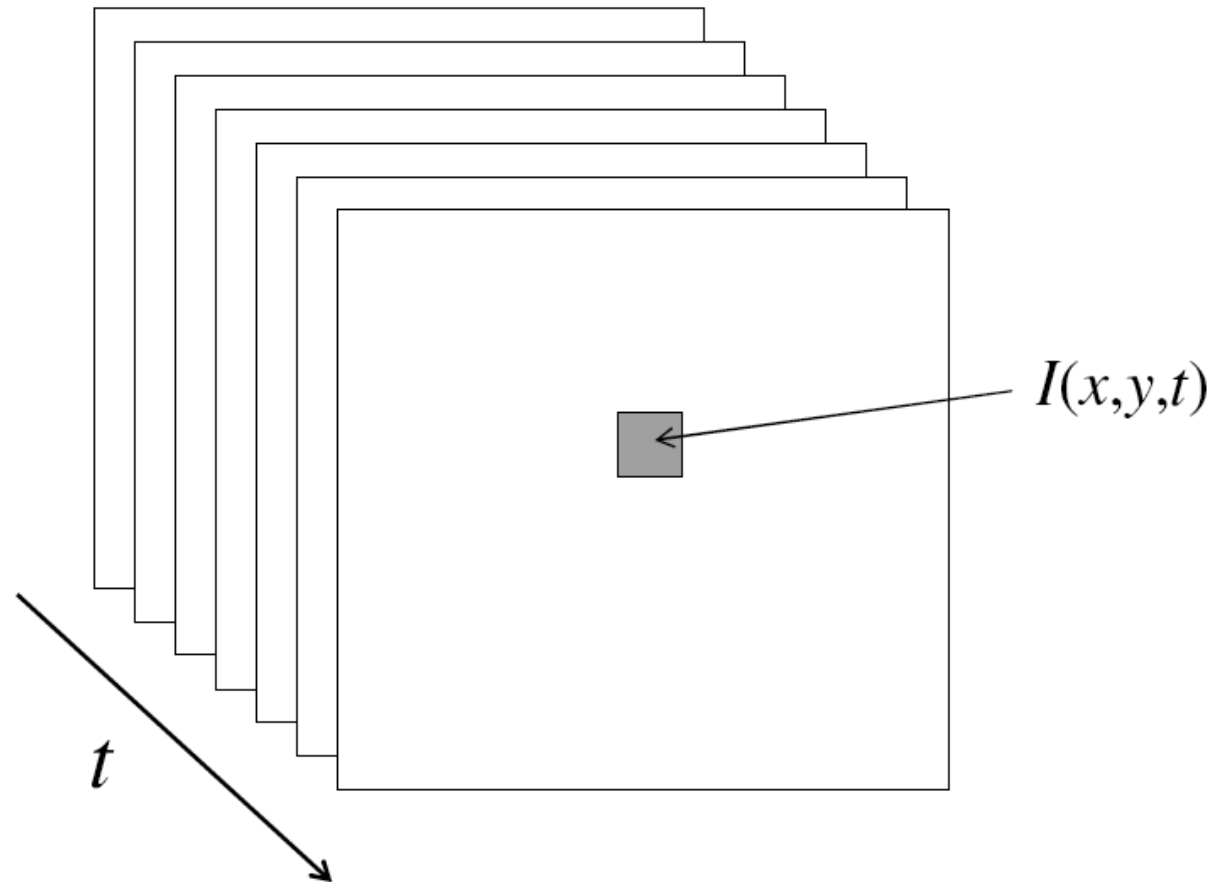
Paulo Dias





- Motion
 - Motion analysis
 - Optical Flow
- Background Subtraction
- Tracking
 - Object Tracking
 - Template Matching
 - Detection vs Tracking

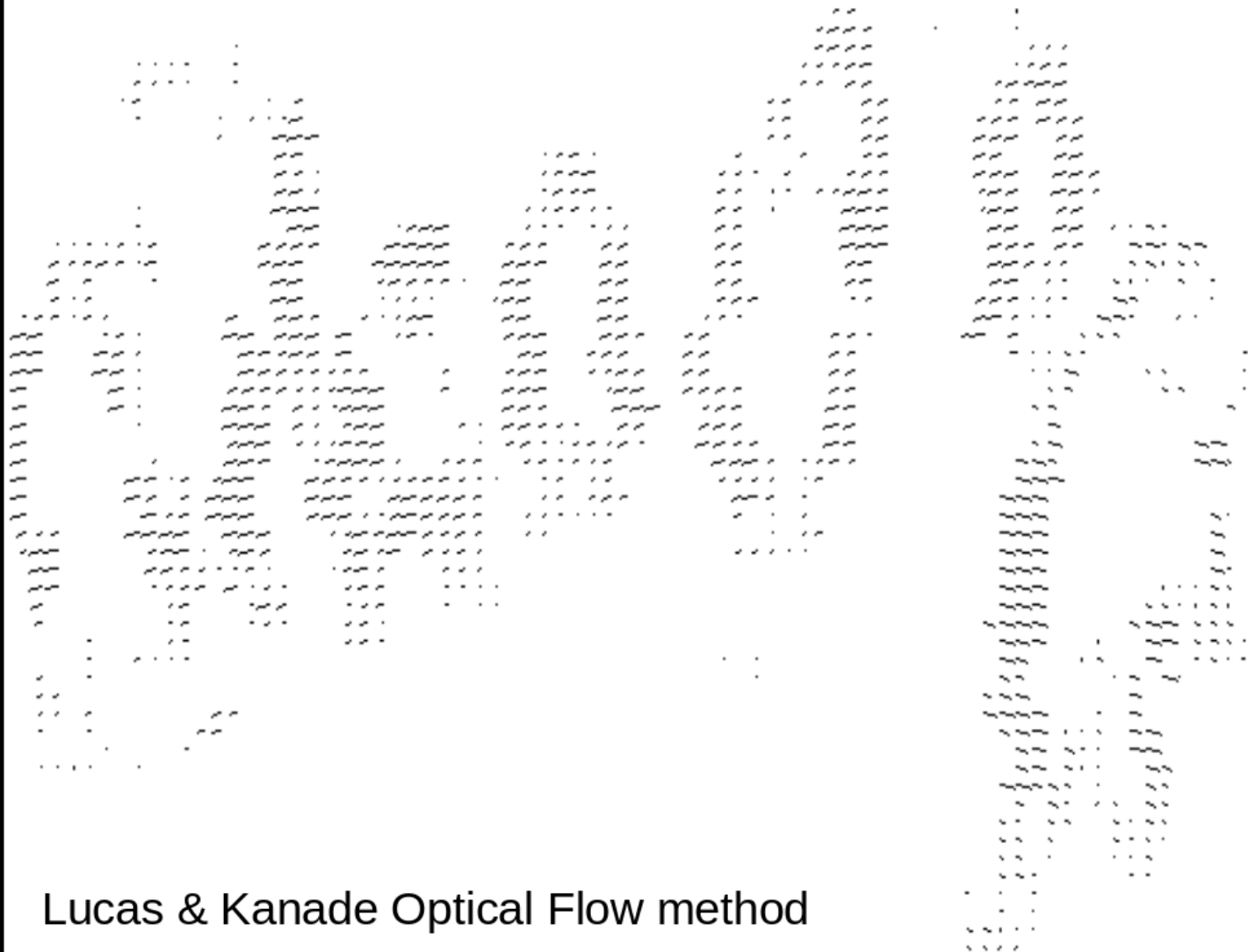
- A video is a sequence of frames captured over time
- The image data is a function of space (x, y) and time (t)





- Several information can be extracted from time varying sequences of images:
 - Camouflaged objects easily visible when moving
 - Size and position of objects are more easily determined when the objects move
 - Even simple image differencing provides edge detector for objects moving over any static background





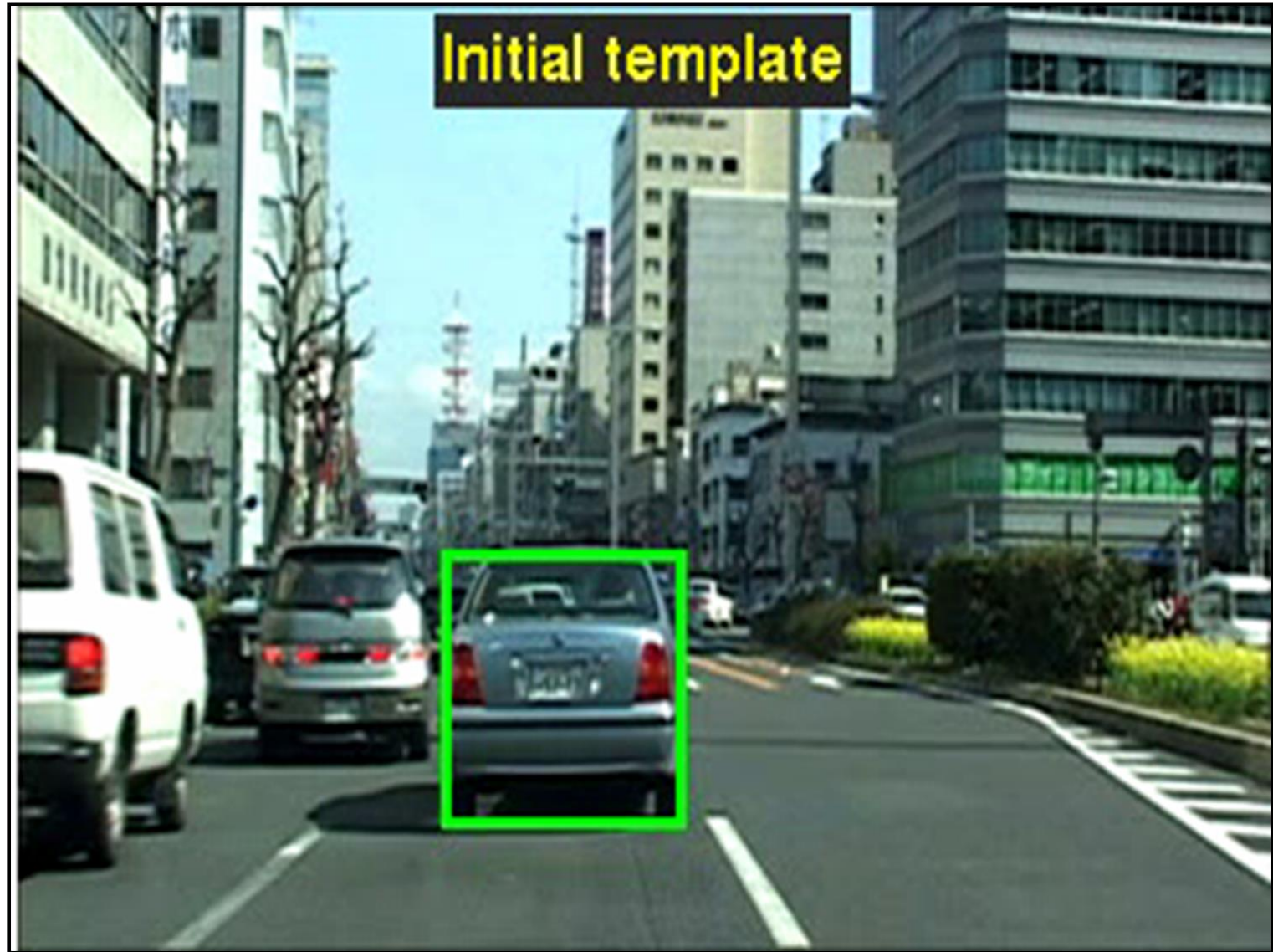


- Analysis of visual motion can be divided into two stages:
 - measurement of the motion
 - use of motion data to segment scene into objects and to extract information about shape and motion.
- There are two types of motion to consider:
 - movement in the scene = static camera,
 - movement of the camera = ego motion.
 - should be the same (motion is relative) but not always true - if scene moves illumination, shadow and specular effects need to be considered



- We are interested in finding the movement of scene objects from time-varying images (videos).
- Lots of uses
 - Track object behavior
 - Align images (mosaics)
 - 3D shape reconstruction
 - Correct for camera jitter (stabilization)
 - Special effects

Rigid Object Tracking



(Simon Baker, CMU)

Non Rigid Object Tracking

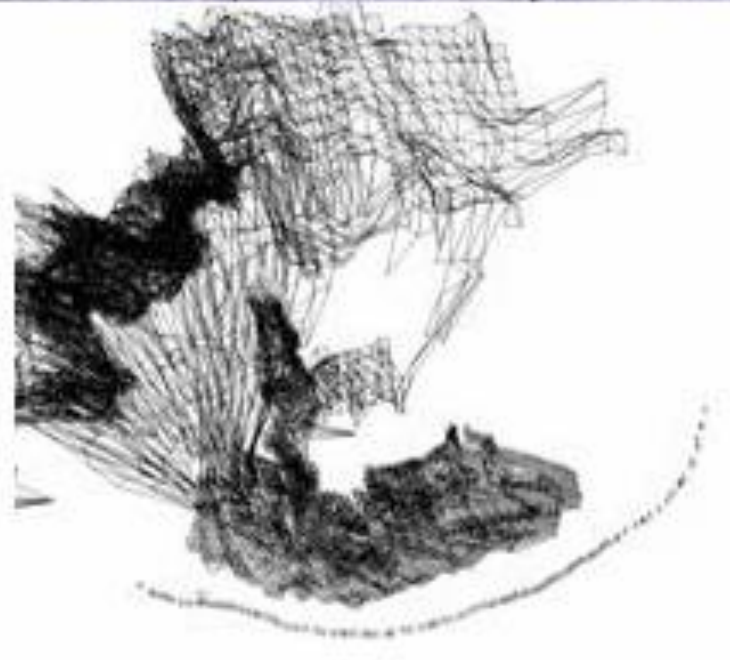
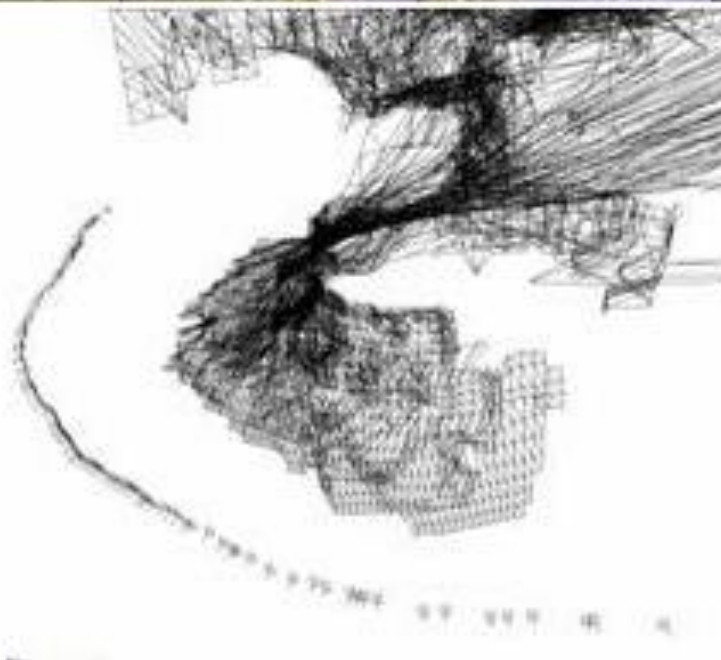


(Comaniciu et al, Siemens)



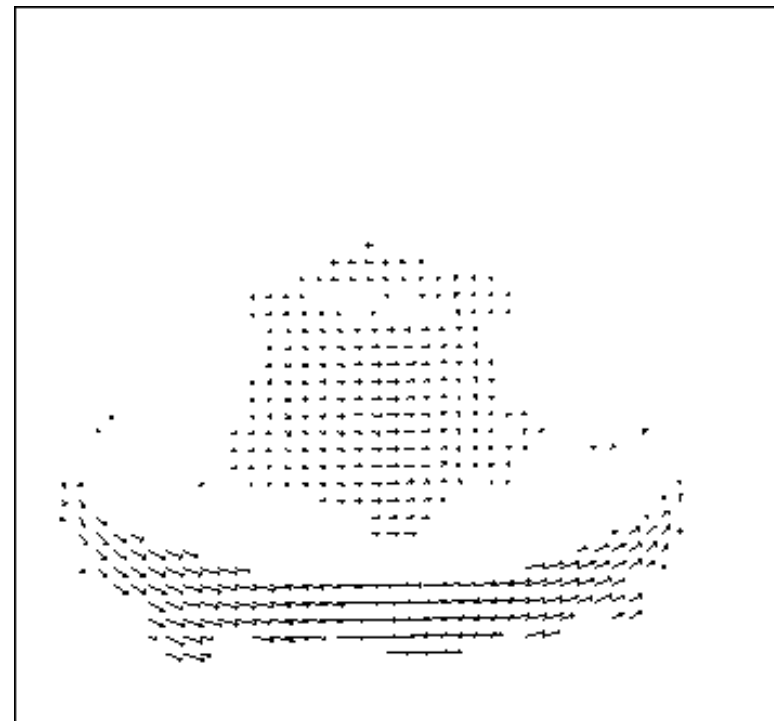
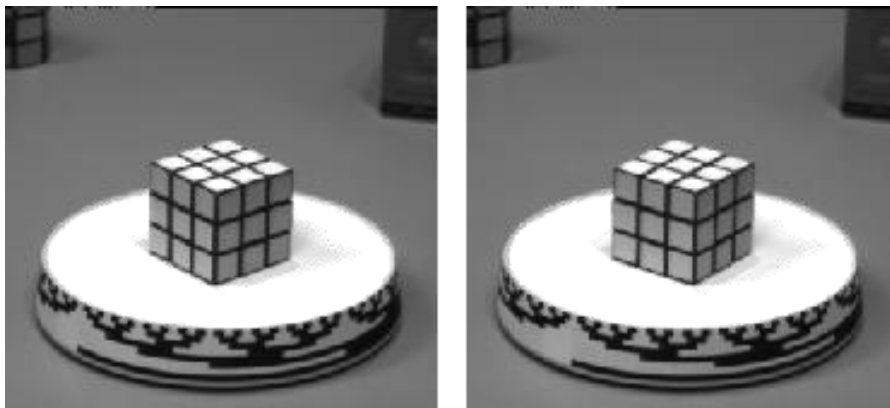
https://youtu.be/Yw_zkLaZNSQ

Structure from motion



(David Nister, Kentucky)

- motion field is the projection of the 3D scene motion into the image.





- Length of flow vectors inversely proportional to depth Z of 3D point - points closer to the camera move more quickly across the image plane.

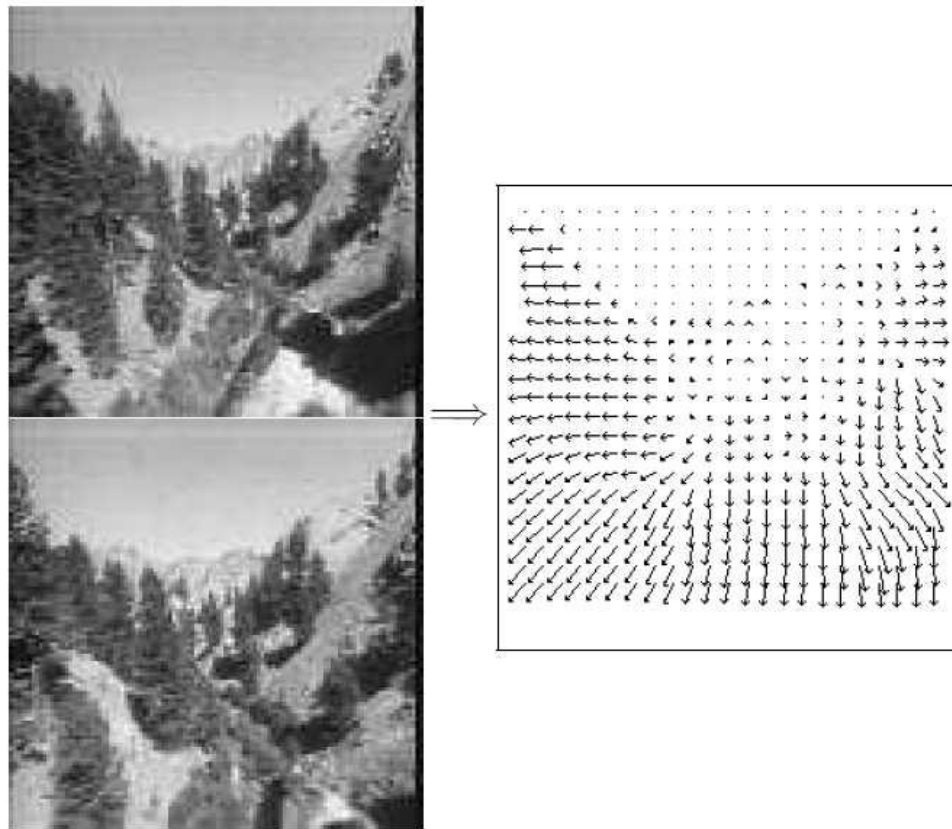


Figure 1.2: Two images taken from a helicopter flying through a canyon and the computed optical flow field.

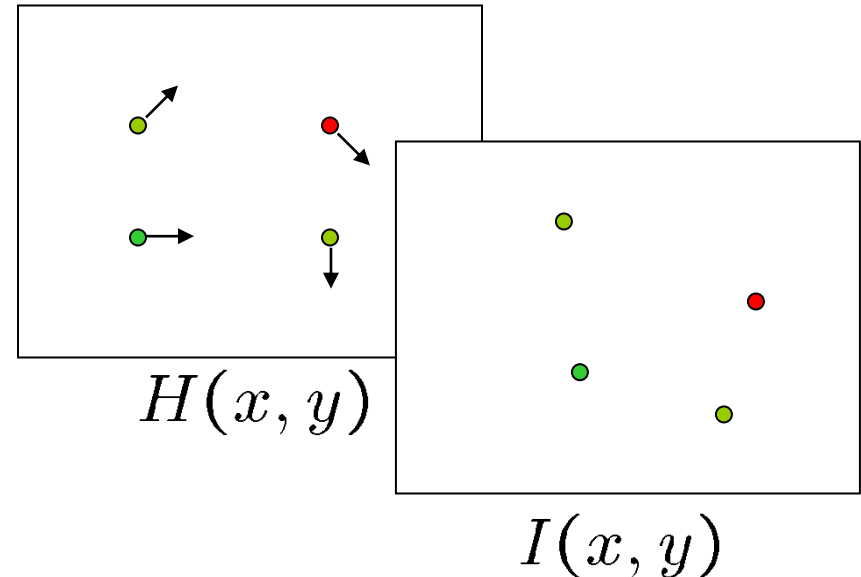


- Optical flow is the apparent motion of brightness patterns (or colours) in the image.
- Ideally, optical flow would be the same as the motion field.
- Have to be careful: apparent motion can be caused by lighting changes without any actual motion.
- To estimate pixel motion from image we have to solve the pixel correspondence problem.
- Given a pixel in frame t , look for nearby pixels with same characteristics (colour, brightness, ...) in frame $t - 1$.

Problem Definition: Optical Flow



- How to estimate pixel motion from image H to image I ?
 - Find pixel correspondences
 - Given a pixel in H , look for nearby pixels of the same color in I



- Key assumptions
 - **color constancy**: a point in H looks “the same” in image I
 - For grayscale images, this is **brightness constancy**
 - **small motion**: points do not move very far



- How to get more equations for a pixel?
 - Basic idea: impose additional constraints
 - most common is to assume that the flow field is smooth locally
 - one method: pretend the pixel's neighbors have the same (u,v)
 - If we use a 5x5 window, that gives us 25 equations per pixel!

$$0 = I_t(\mathbf{p}_i) + \nabla I(\mathbf{p}_i) \cdot [u \ v] \quad \text{color constancy}$$

$$\underbrace{\begin{bmatrix} I_x(\mathbf{p}_1) & I_y(\mathbf{p}_1) \\ I_x(\mathbf{p}_2) & I_y(\mathbf{p}_2) \\ \vdots & \vdots \\ I_x(\mathbf{p}_{25}) & I_y(\mathbf{p}_{25}) \end{bmatrix}}_{\substack{A \\ 25 \times 2}} \underbrace{\begin{bmatrix} u \\ v \end{bmatrix}}_{\substack{d \\ 2 \times 1}} = - \underbrace{\begin{bmatrix} I_t(\mathbf{p}_1) \\ I_t(\mathbf{p}_2) \\ \vdots \\ I_t(\mathbf{p}_{25}) \end{bmatrix}}_{\substack{b \\ 25 \times 1}} \quad \text{small motion constant within neighborhood}$$



- Prob: we have more equations than unknowns

$$\begin{matrix} A & d = b \\ 25 \times 2 & 2 \times 1 \quad 25 \times 1 \end{matrix} \longrightarrow \text{minimize } \|Ad - b\|^2$$

- **Solution: solve least squares problem**

- minimum least squares solution given by solution (in d) of:

$$\begin{matrix} (A^T A) & d = A^T b \\ 2 \times 2 & 2 \times 1 \quad 2 \times 1 \end{matrix}$$

$$\begin{matrix} \begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} & \begin{bmatrix} u \\ v \end{bmatrix} = - & \begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix} \\ A^T A & & A^T b \end{matrix}$$

- The summations are over all pixels in the K x K window
- This technique was first proposed by Lukas & Kanade (1981)
 - described in Trucco & Verri reading

- Optimal (u, v) satisfies Lucas-Kanade equation

$$\begin{bmatrix} \sum I_x I_x & \sum I_x I_y \\ \sum I_x I_y & \sum I_y I_y \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} = - \begin{bmatrix} \sum I_x I_t \\ \sum I_y I_t \end{bmatrix}$$

$A^T A$ $A^T b$

When is This Solvable?

- $\mathbf{A}^T \mathbf{A}$ should be invertible
- $\mathbf{A}^T \mathbf{A}$ should not be too small due to noise
 - eigenvalues λ_1 and λ_2 of $\mathbf{A}^T \mathbf{A}$ should not be too small
- $\mathbf{A}^T \mathbf{A}$ should be well-conditioned
 - λ_1 / λ_2 should not be too large ($\lambda_1 =$ larger eigenvalue)



- Optical flow:
 - Algorithms try to approximate the true motion field of the image plane.
 - The Optical Flow Constraint Equation needs an additional constraint (e.g. smoothness, constant local flow).
 - The Lucas Kanade method is the most popular Optical Flow Algorithm.
- Lucas Kanade Optical Flow in OpenCV
 - https://docs.opencv.org/4.x/d4/dee/tutorial_optical_flow.html

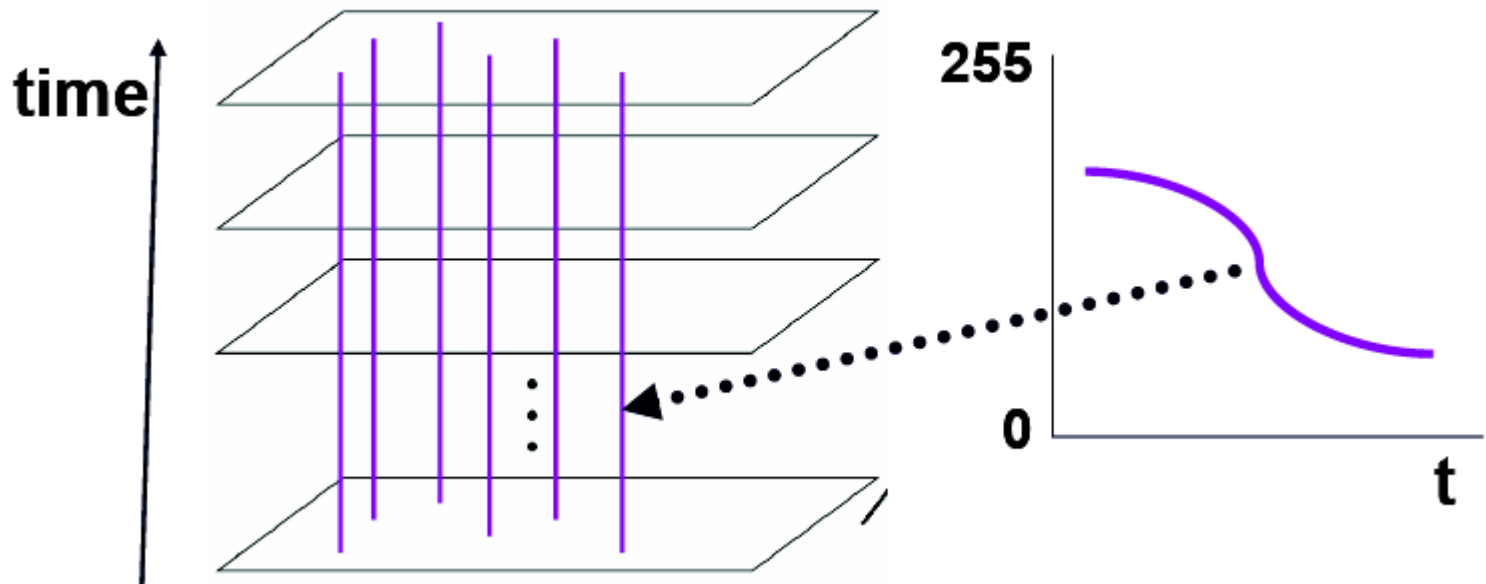


- Motion
 - Motion analysis
 - Optical Flow
- Background Subtraction
- Tracking
 - Object Tracking
 - Template Matching
 - Detection vs Tracking

Background Subtraction



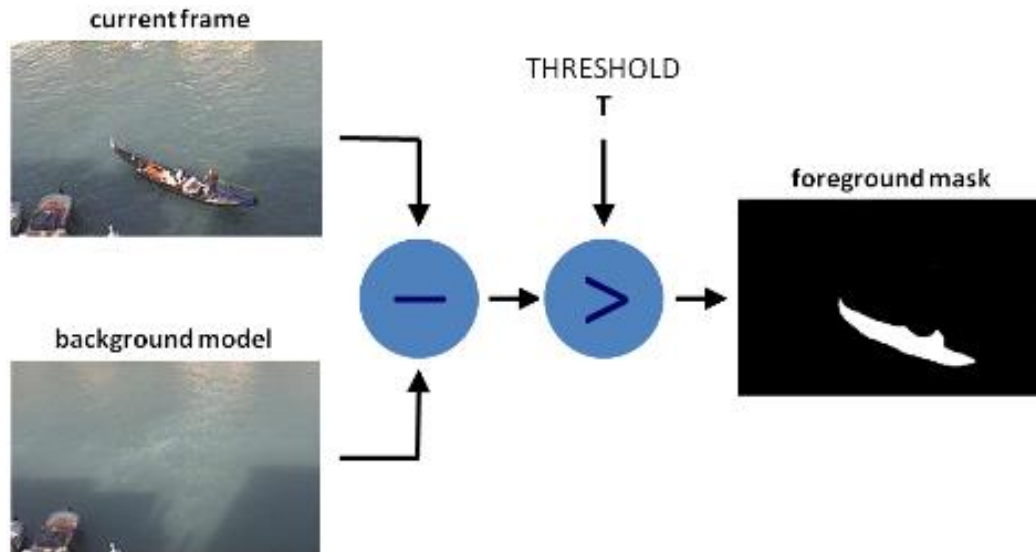
- It is possible to look at video data as a spatio-temporal volume.
- If camera is stationary, each line through time corresponds to a single ray in space.





- Background subtraction techniques are commonly used for segmenting out objects of interest in a static camera scene for applications such as:
 - Surveillance
 - Robot vision
 - Object tracking
 - Traffic applications
 - Human motion capture
 - Augmented reality

- Used for generating a foreground mask (binary image) containing moving objects in static camera setups.
- Name comes from the simple technique of subtracting the observed image from the estimated image and thresholding the result to generate the objects of interest.





- Foreground detection – how the object areas are distinguished from the background;
- Background maintenance – how the background is maintained over time;
- Post-processing – how the segmented object areas are

- Generic algorithm:
 - Create an image of the stationary background by averaging a long sequence.
 - Subtract current frame and known background frame
 - Motion detection algorithms such as these only work if the camera is stationary and objects are moving against a fixed background

Image at time t :

$I(x, y, t)$



Background at time t :

$B(x, y, t)$



—

$| > Th$





- Generic algorithm:
 - With frame differencing, background is estimated to be the previous frame. Background subtraction equation becomes $B(x, y, T) = I(x, y, t - 1)$ and $|I(x, y, t) - I(x, y, t - 1)| > Th$
 - Depending on the object structure, speed, frame rate and global threshold may or may not be useful (usually not).
 - Another approach is to model the background using a running average. A pixel is marked as foreground if

$$|I_t - B_t| > th$$

where **th** is a predefined threshold.





























The thresholding is often followed by morphological closing with a 3x3 kernel and the discarding of small regions

- The background update is

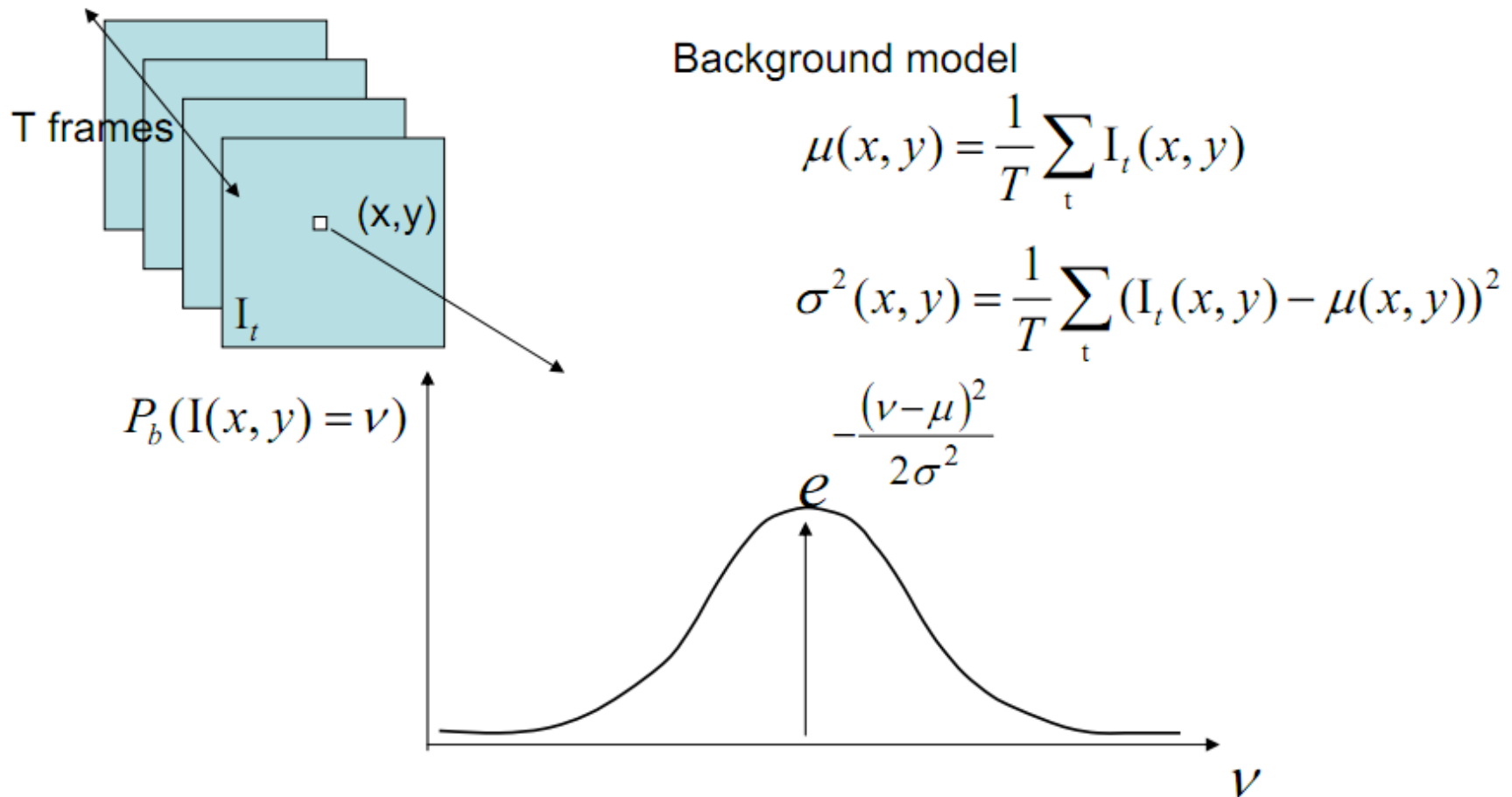
$$B_{t-1} = \alpha I_t + (1 - \alpha) B_t$$

where α is kept small to prevent artificial tails forming behind moving object

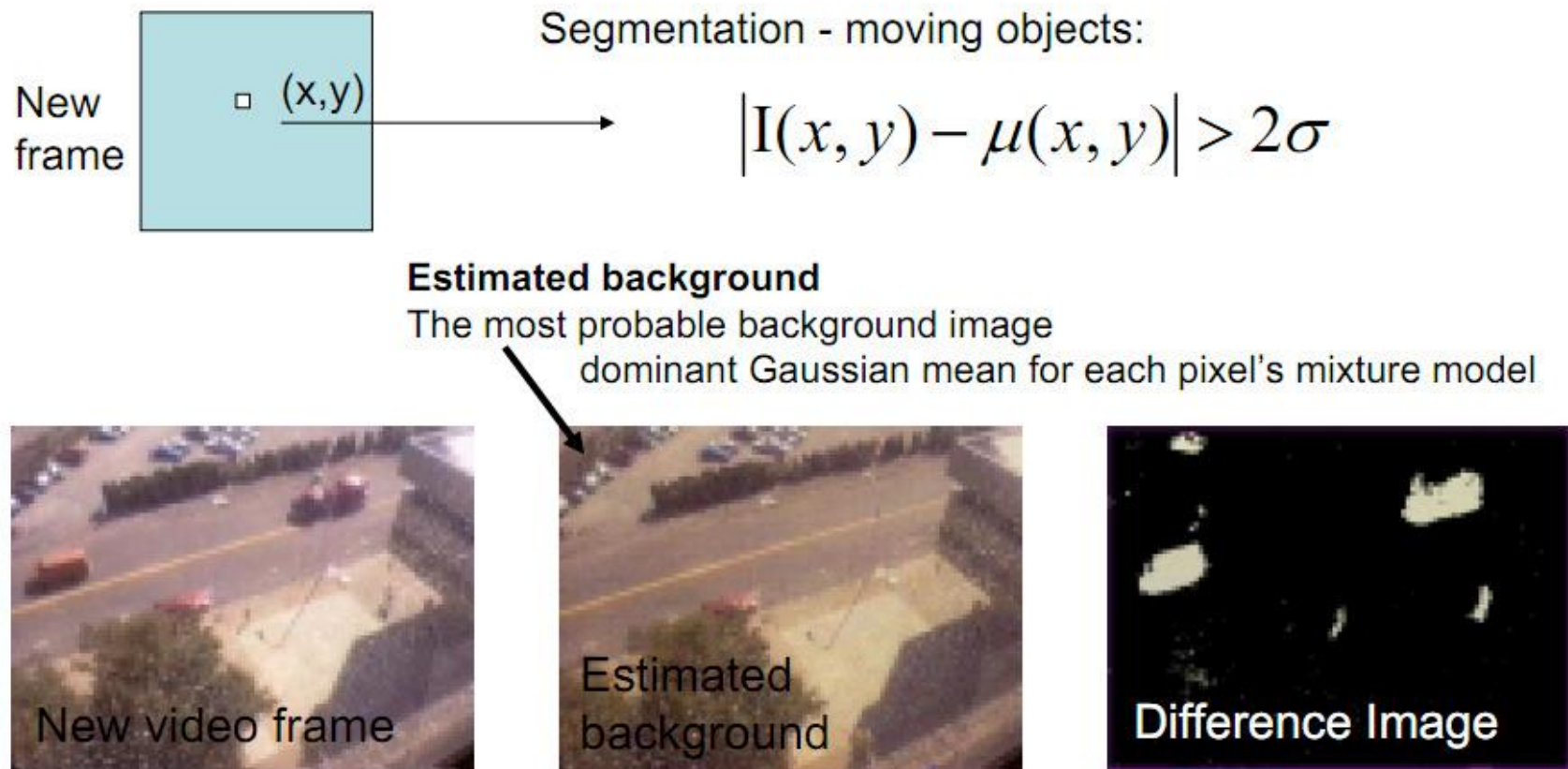
- Examples

Test Image							
	Chair moved	Light gradually brightened	Light just switched on	Tree Waving	Foreground covers monitor pattern	No clean background training	Interior motion undetectable
Ideal Foreground							
Adjacent Frame Difference							
Mean & Threshold							

- Adaptive Mixture of Gaussians



Background mixture model - example





- Motion
 - Motion analysis
 - Optical Flow
- Background Subtraction
- Tracking
 - Object Tracking
 - Template Matching
 - Detection vs Tracking



- Object tracking is a crucial issue in computer vision, especially for the applications where the environment is in continuous changing:
 - Robot Vision - mobile robot navigation, applications that must deal with unstable grasps, . . .
 - Surveillance
 - Traffic applications
 - Human motion capture

Tracking: some applications



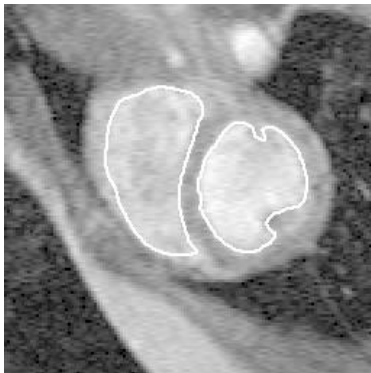
Body pose tracking,
activity recognition



Censusing a bat
population



Video-based
interfaces



Medical apps

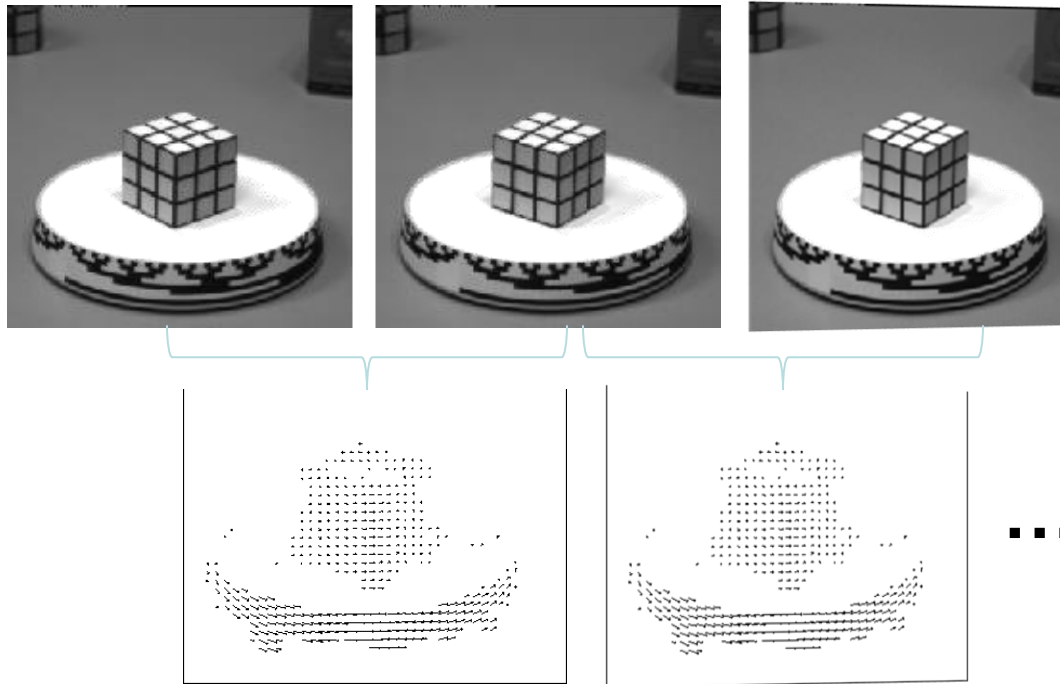


Surveillance

Optical flow for tracking?

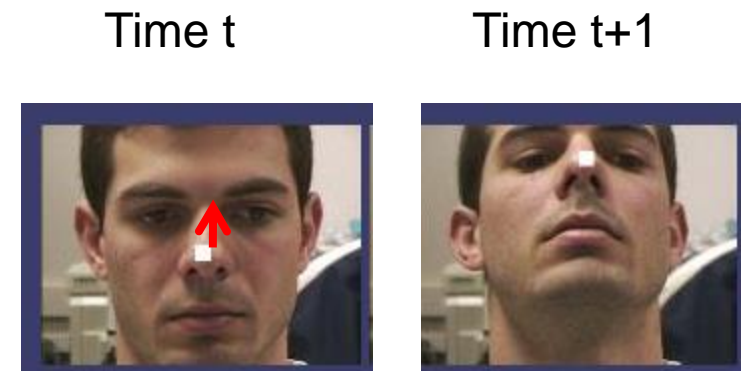
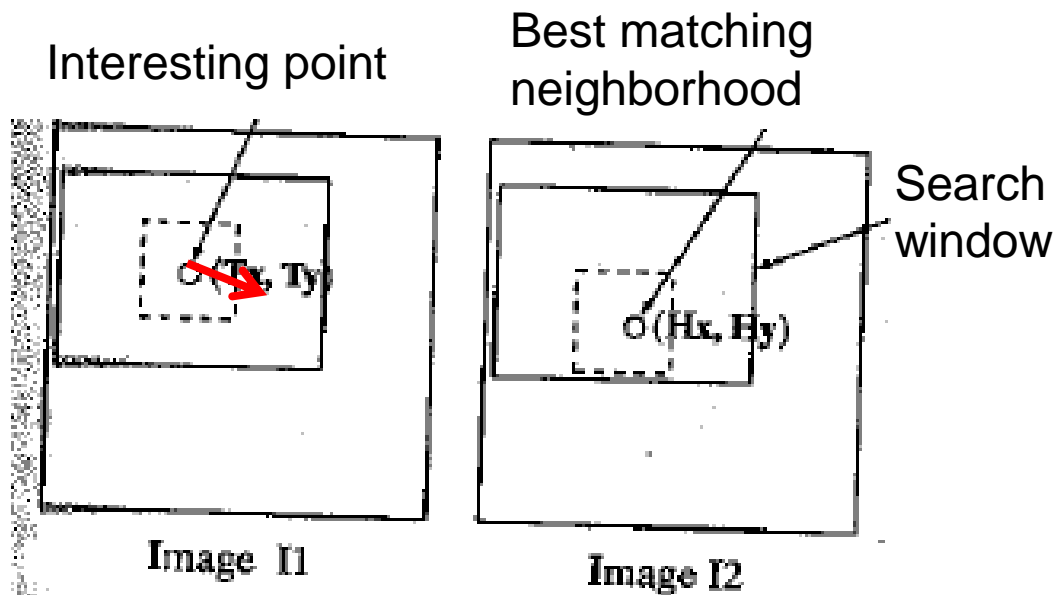


If we have more than just a pair of frames, we could compute flow from one to the other next



But flow only reliable for small motions, and we may have occlusions, texture less regions that yield bad estimates anyway...

Feature-based matching for motion



Search window is centered at the point where we last saw the feature, in image I1.

Best match = position where we have the highest normalized cross-correlation value.

Example: A Camera Mouse



- Video interface: use feature tracking as mouse replacement



- User clicks on the feature to be tracked
- Take the 15x15 pixel square of the feature
- In the next image do a search to find the 15x15 region with the highest correlation
- Move the mouse pointer accordingly
- Repeat in the background every 1/30th of a second

Detection vs. tracking



t=1



t=2

...



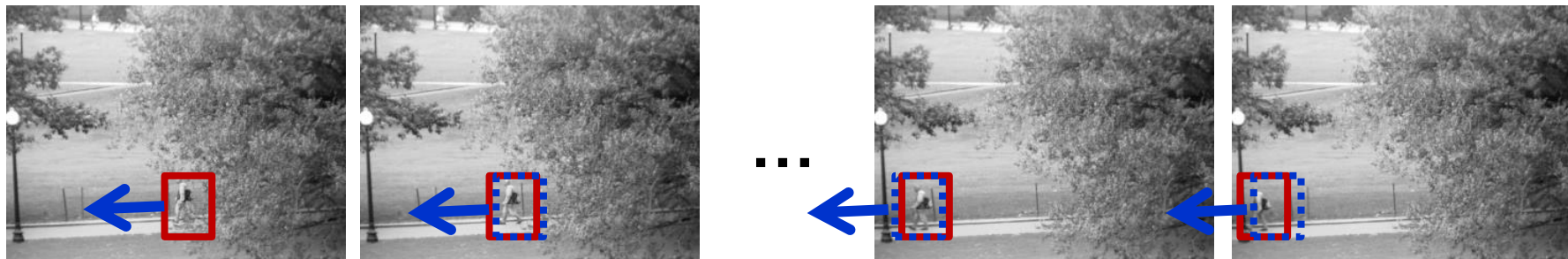
t=20



t=21



Detection: We detect the object independently in each frame and can record its position over time, e.g., based on blob's centroid or detection window coordinates



Tracking with *dynamics*: We use image measurements to estimate position of object, but also incorporate position predicted by dynamics, i.e., our expectation of object's motion pattern.

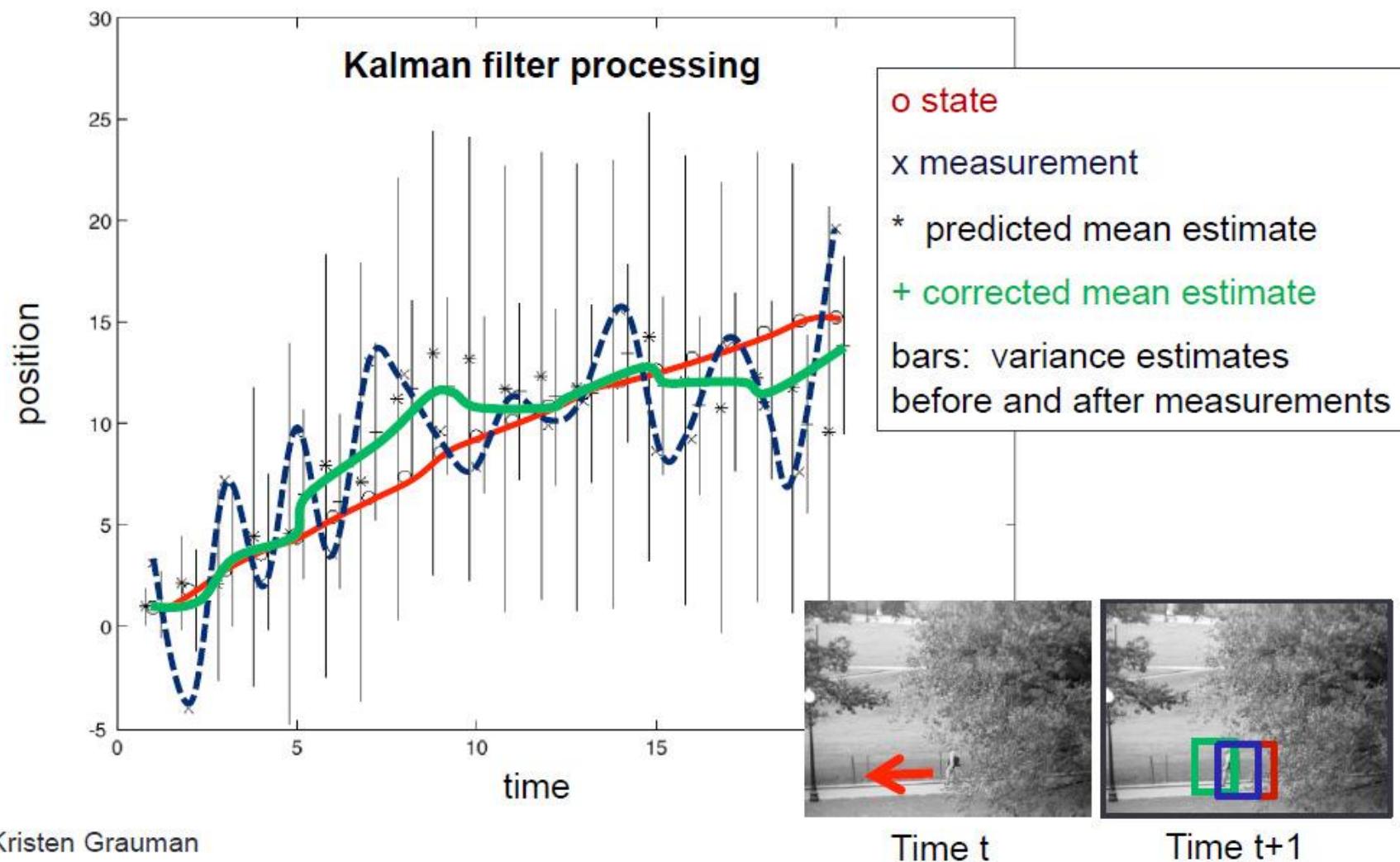


- The hidden state consists of the true parameters we care about, denoted X .
- The measurement is our noisy observation that results from the underlying state, denoted Y .
- At each time step, state changes (from X_{t-1} to X_t) and we get a new observation Y_t .



- Method for tracking linear dynamical models in Gaussian noise
- The predicted/corrected state distributions are Gaussian
- Only need to maintain the mean and covariance

Tracking: Constant velocity model

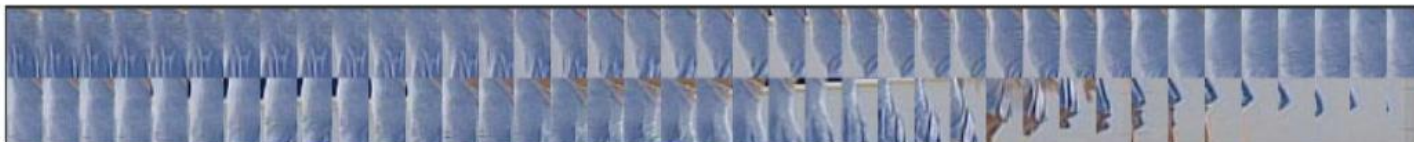


Kristen Grauman

- **Initialization**
 - Often done manually
 - Background subtraction, detection can also be used
- **Data association**, multiple tracked objects
 - Occlusions, clutter



- **Deformable** and articulated objects
- Constructing **accurate models** of dynamics - (example: Fitting parameters for a linear dynamics model)
- **Drift** - accumulation of errors over time





- Barron, “Tutorial: Computing 2D and 3D Optical Flow.”, <http://www.tina-vision.net/docs/memos/2004-012.pdf>
- CVonline: Optical Flow
- Fast Image Motion Estimation Demo
<http://extra.cmis.csiro.au/IA/changs/motion/>
- Part of this lecture is from the courses offered by Prof. Shree Nayar at Columbia University, USA and by Prof. Srinivasa Narasimhan at CMU, USA.