Module IN 2018

Introduction to Augmented Reality

Prof. Gudrun Klinker

Basics of Computer Vision

SS 2018

Exam dates

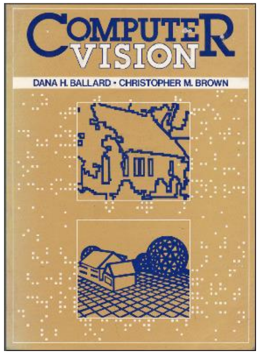
Planned (not yet final)

- 24.7.2018 (Tuesday), 8:00-9:30
  MW 2001, Rudolf-Diesel-Hörsaal (5510.02.001)

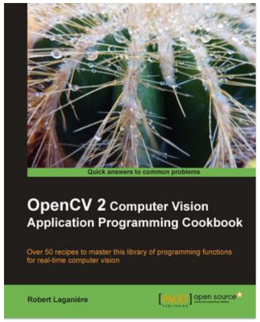
- 9.10.2018 (Tuesday), 8:00-9:30
  MW 1801, Ernst-Schmidt-Hörsaal (5508.01.801)

Literature

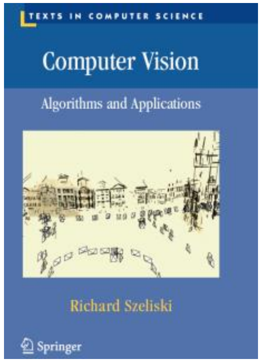
- D.Ballard and C.Brown: Computer Vision, 1982.
  Online version:
  http://homepages.inf.ed.ac.uk/rbf/BOOKS/BANDB/bandb.htm



- R. Laganiere: OpenCV 2 - Computer Vision Applications Programming Cookbook, Packt Publishing open source, 2011.



- R. Szeliski: Computer Vision - Algorithms and Applications, Springer Verlag, 2011.
  Online version: http://szeliski.org/Book/
  (You are welcome to download the PDF from this Web site for personal use, but **not** to repost it on any other Web site. Please post a link to this URL (http://szeliski.org/Book) instead. An electronic version of this manuscript will continue to be available even after the book is published. Note, however, that while the content of the electronic and hardcopy versions are the same, the page layout (pagination) is different, since the electronic version is optimized for online reading.)

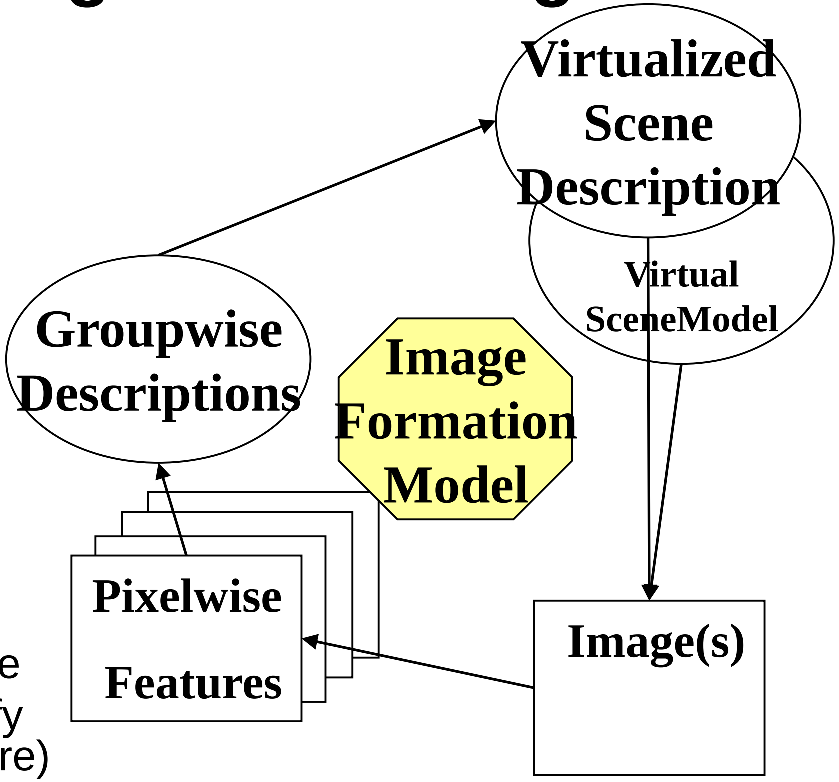


Agenda

→ 1. Image Formation (Geometry, Radiometry)
2. Feature Detection
3. Feature Matching and Tracking

1. Image Formation

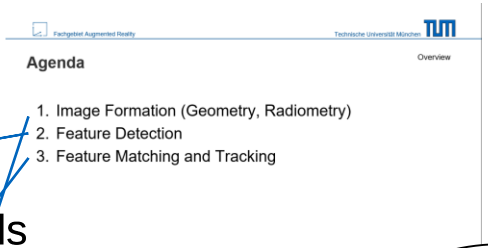
1.1 Image Formation vs. Image Processing



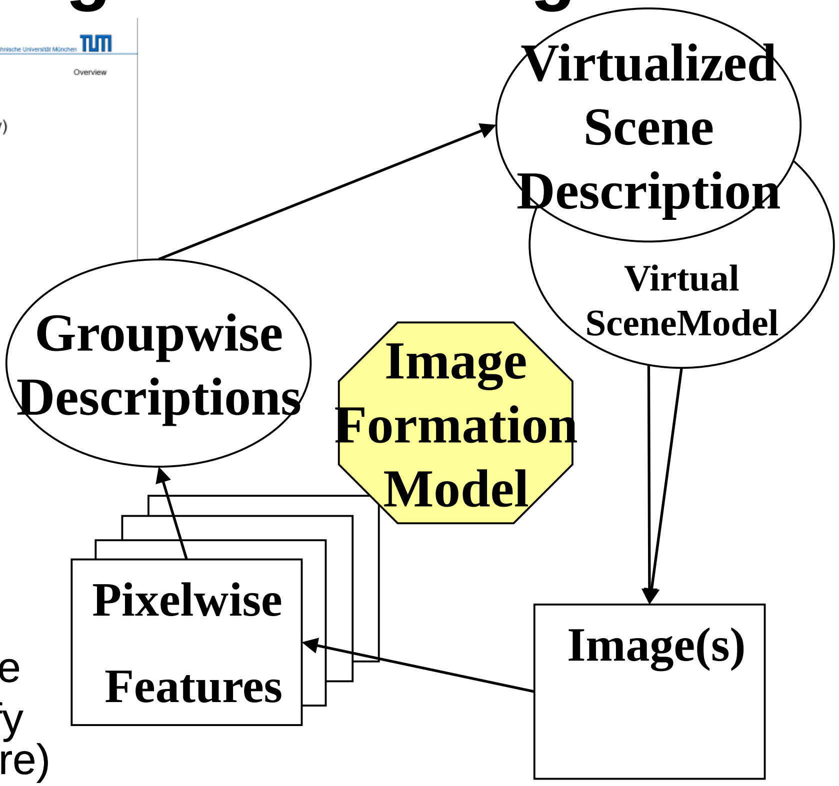
- Image processing
  - Bottom up
    - Analyze / group pixels
    - Formulate hypotheses / use heuristics
    - Generate / group features
    - Formulate more hypotheses
    - Derive scene description
- Back projection
  - Top down
    - Use scene description
    - Project modeled features into the image
    - Find (closest) features in image to verify that model is visible in image (and where)
- Image formation
  - Process of defining a formal (physics-based) model for top-down & bottom-up scene and image analysis

Computer vision is generally a combination of all! (Interpretation cycle)

1.1 Image Formation vs. Image Processing



- Image processing
  - Bottom up
    - Analyze / group pixels
    - Formulate hypotheses / use heuristics
    - Generate / group features
    - Formulate more hypotheses
    - Derive scene description
- Back projection
  - Top down
    - Use scene description
    - Project modeled features into the image
    - Find (closest) features in image to verify that model is visible in image (and where)
- Image formation
  - Process of defining a formal (physics-based) model for top-down & bottom-up scene and image analysis



Computer vision is generally a combination of all! (Interpretation cycle)

Overview

1. Image Formation

1.1 Image formation vs. Image processing

→ 1.2 Virtual scene models vs. Physical world

1.3 Sensor models

- 1.3.1 Image geometry
- 1.3.2 Image radiometry

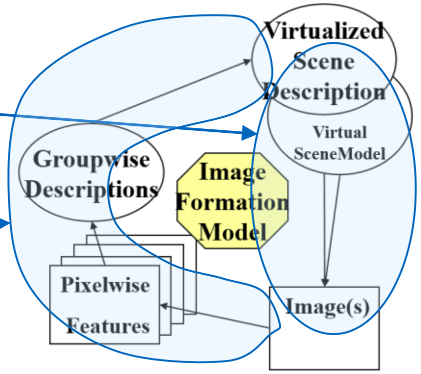
1.2 Virtual Scene Models vs. Physical World

Simplifying assumptions

- The physical environment is extremely complex
- **Physics** tries to model the physical world
  (determining ever more complex relationships with ever increasing numbers of formulas and parameters)
- **Computer graphics** considers and simplifies physics models and develops algorithms

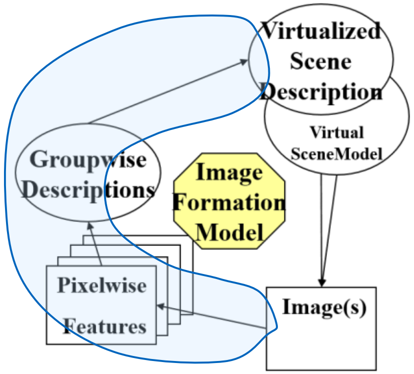
  to render synthensized images as realistically but also as fast as possible (top-down)
- **Computer vision** simplifies the models even more

  to estimate the parameters (bottom-up)
  rather than „just plugging them in“ (top-down)



1.2 Virtual Scene Models vs. Physical World

Typical assumptions
(special areas of computer vision focus on relaxing these assumptions)

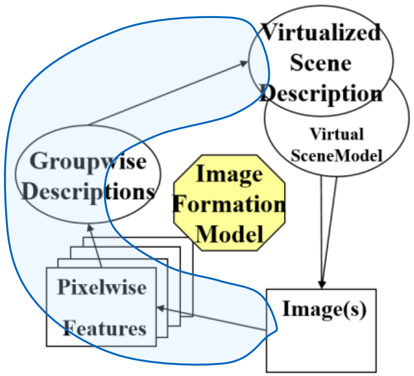


- The scene is „suitably“ illuminated
  - known positions of light sources
  - not too bright, not too dark
  - no highlights, no inter-reflections, no shadows
- The objects are in „suitable“ positions
  - reasonably large to be recognizable
  - not occluded, …
- The objects have appearance properties (3D) that are projected into easily detectable image features (2D)
  - specific shape (silhouette or also internal edges)
  - specific material properties (colored textures)

1.2 Virtual Scene Models vs. Physical World

These might be some of your assumptions (marker tracker)

- The scene is „suitably“ illuminated
  - not too bright, not too dark
    - white objects should have pixel values > „some threshold“
    - black objects should have pixel values < „some threshold“
- The objects are in „suitable“ positions
  - reasonably large and not occluded
    - image region showing the marker contains „enough“ pixels
    - hands/fingers are not covering the marker
    - the marker is not partially outside the image
- The objects have easily detectable image features (2D)
  - specific shape and specific material properties
    - 3D squares that are projected into a 2D quadrangles (with four straight lines)
    - well-defined patterns of small black squares inside each marker



1.2 Virtual Scene Models vs. Physical World

NOTE:

Using these simplifying assumptions, you will NOT be able to produce a very robust marker tracker.

- It will fail frequently.
- You are encouraged to experiment with this:
  - When does you tracker fail?
    (further limiting assumptions beyond what‘s mentioned above?)
  - How inconvenient is this? (→ user studies)
  - Which are the most important issues that should be improved?
  - Any idea, HOW?

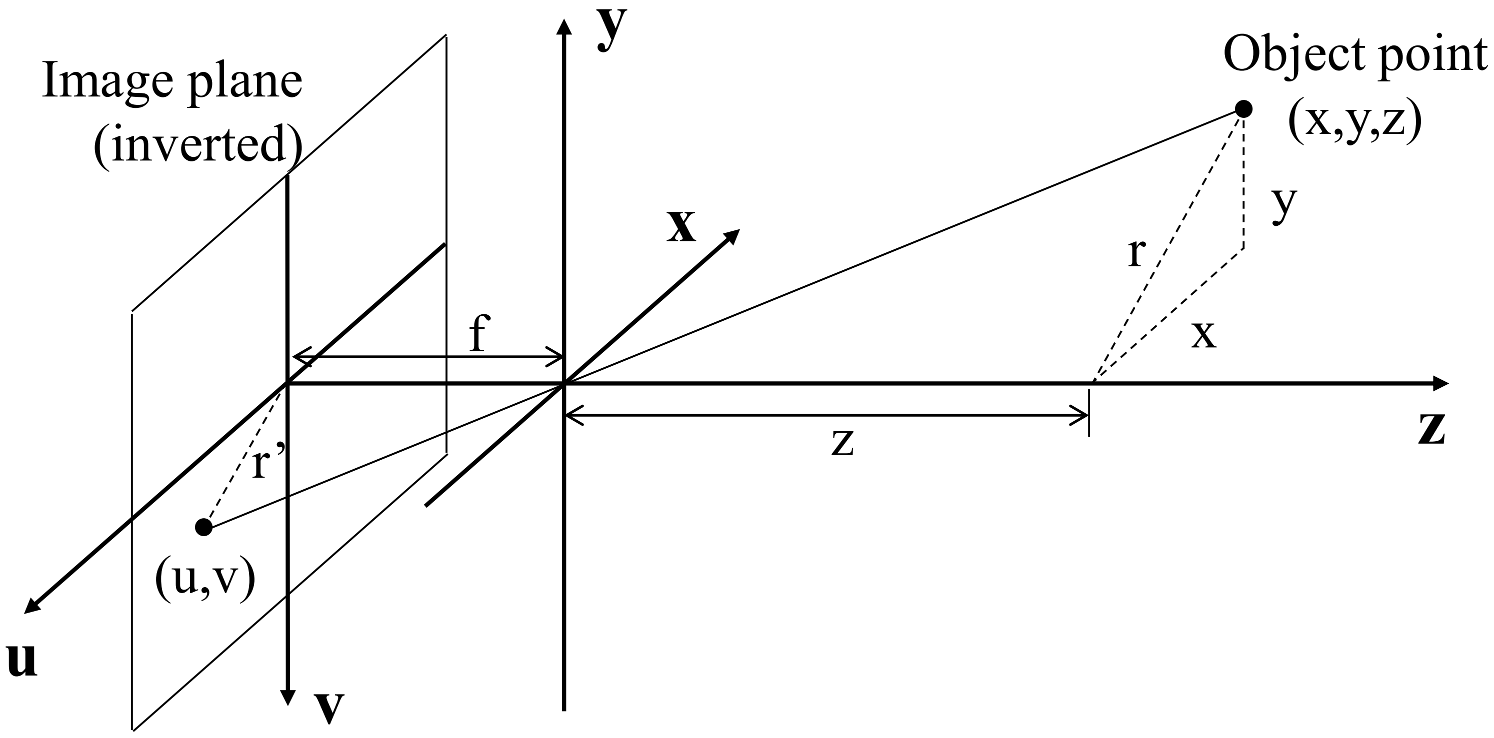
Publically available marker trackers have gone through many improvement cycles to address the most critical issues.

1. Image Formation

1.3 Sensor Models

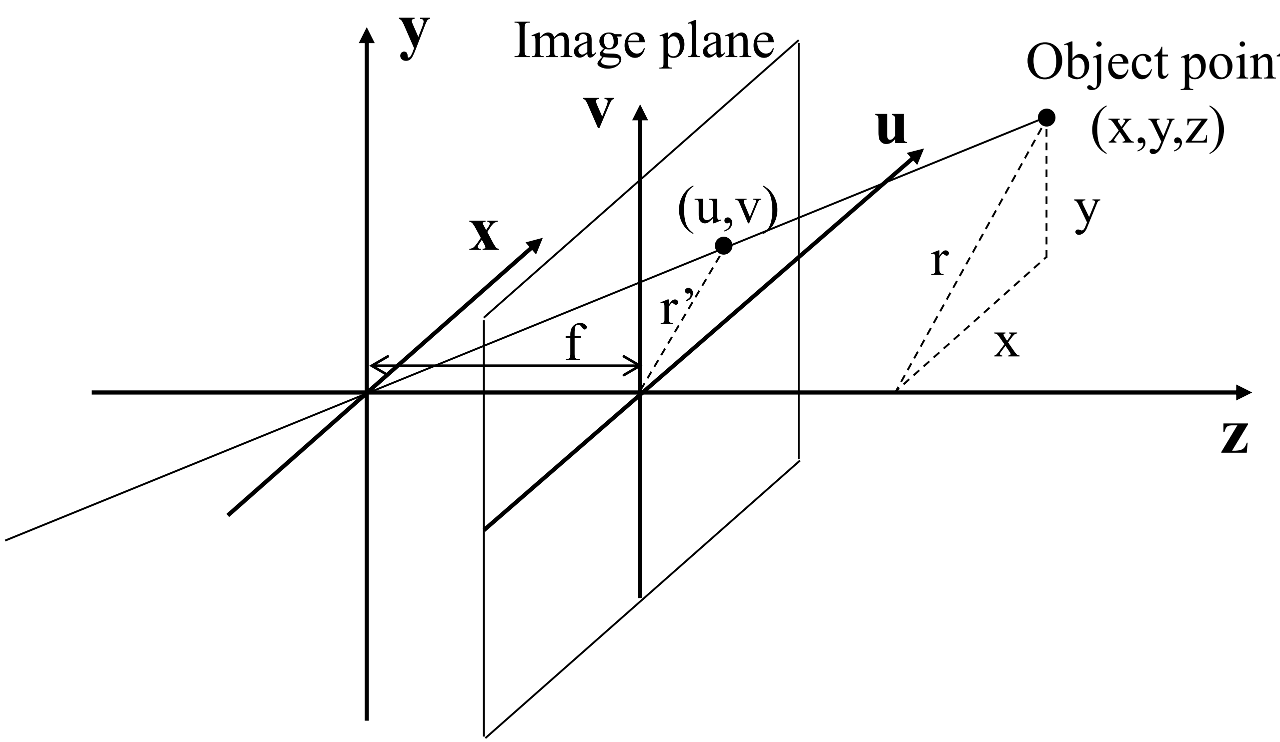
1.3.1 Image Geometry

Pinhole Projection

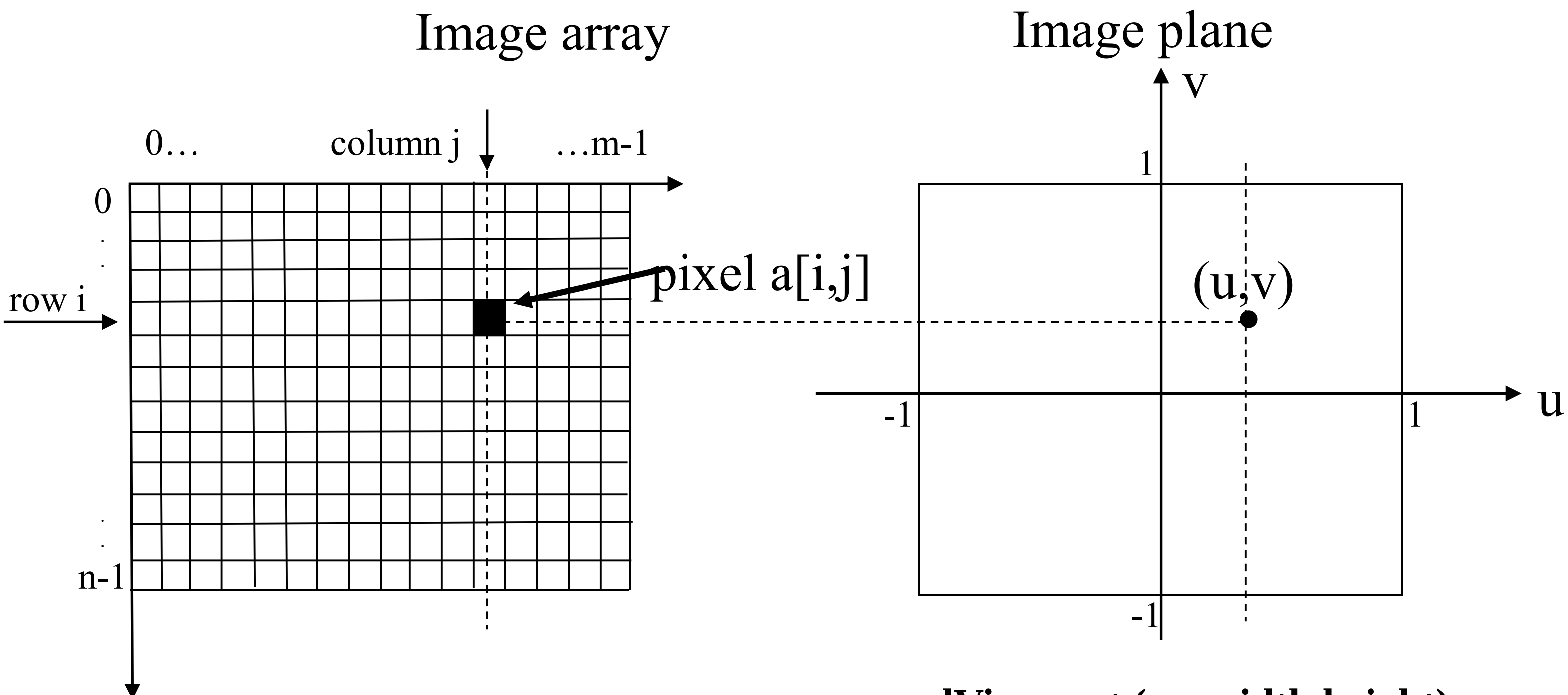


1.3.1 Image Geometry

Pinhole Projection



1.3.1 Image Geometry



glViewport (x,y,width,height)

$$image\ processing: (img\ array \rightarrow img\ plane)$$

$$u = j - \frac{width-1}{2} \qquad v = -\left(i - \frac{height-1}{2}\right)$$

$$\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & -\frac{width-1}{2} \\ 0 & -1 & \frac{height-1}{2} \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} j \\ i \\ 1 \end{pmatrix}$$

$$image\ formation: (img\ plane \rightarrow img\ array)$$

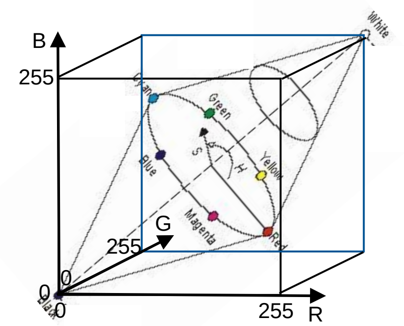
$$j = u + \frac{width-1}{2} \qquad i = -v + \frac{height-1}{2}$$

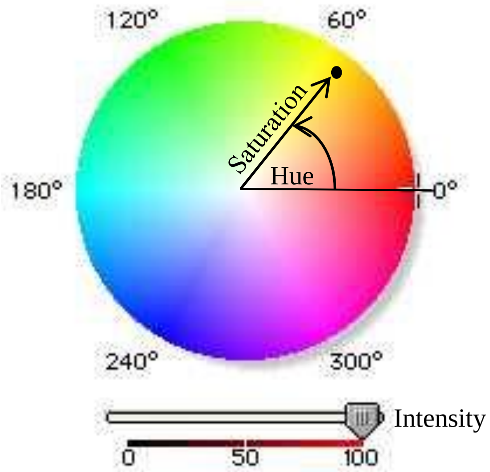
$$\begin{pmatrix} j \\ i \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & \frac{width-1}{2} \\ 0 & -1 & \frac{height-1}{2} \\ 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} u \\ v \\ 1 \end{pmatrix}$$

1.3.2 Image Radiometry

Basics

- Color pixel [i,j] = {R,G,B} or {R,G,B,a}
- Graypixel [i,j] = {I}
- HSI color representation
  - Intensity: $$I = \frac{R+G+B}{3}$$
  - Hue: $$H = a\cos\left(\frac{2R-G-B}{2\sqrt{(R-G)^2+(R-G)(G-B)}}\right)$$
  - Saturation $$S = 1 - \frac{\min(R,G,B)}{I}$$





- Common practice (for fast processing):
  I = R or I = G

Summary

1. Image Formation

Take home messages (things you should know):

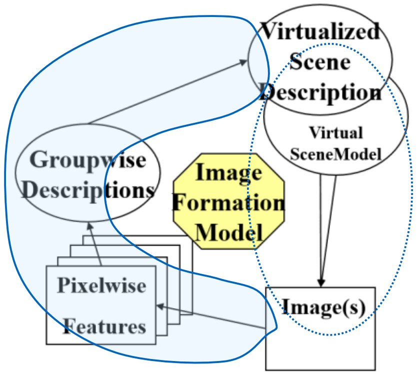
- Top down versus bottom up computer vision
  - Top down: virtualized scene description (virtual model) → image
  - Bottom up: image → virtualized scene description
- An image is a large 2D array of „pixels“
  Each pixel is an intensity or a color vector
- Pinhole projection (camera) model
  - 3D scene point (x,y,z), image point (u,v) and camera center form a line
- Color models
  - RGB vs IHS
  - Typical in computer vision: only grayscale images $I = \frac{R+G+B}{3}$ or $I = R$ or $I = G$

Agenda

1. Image Formation (Geometry, Radiometry)
2. Feature Detection
3. Feature Matching and Tracking

Overview

2. Feature Detection

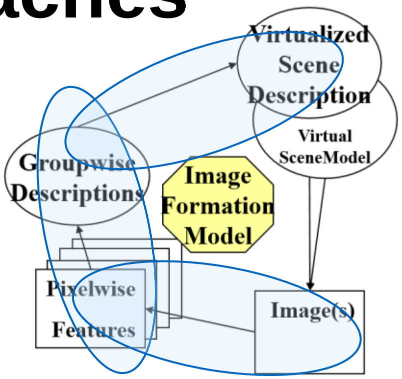


→ 2.1 Region- (similarity)- based approaches

2.2 Edge- (difference)- based approaches

Overview

2.1 Region- (Similarity)- Based Approaches



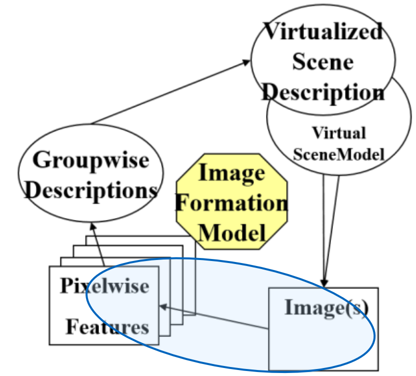
1. Thresholding, Histograms
2. Connected Components
3. Region Boundaries
4. Region Properties
5. Morphological Operations

0	0	2	6	7	2	1	5	0
1	1	2	7	8	3	5	6	4
5	5	5	6	7	7	8	8	3
0	5	6	7	7	8	2	3	3
0	1	2	6	6	1	6	5	2

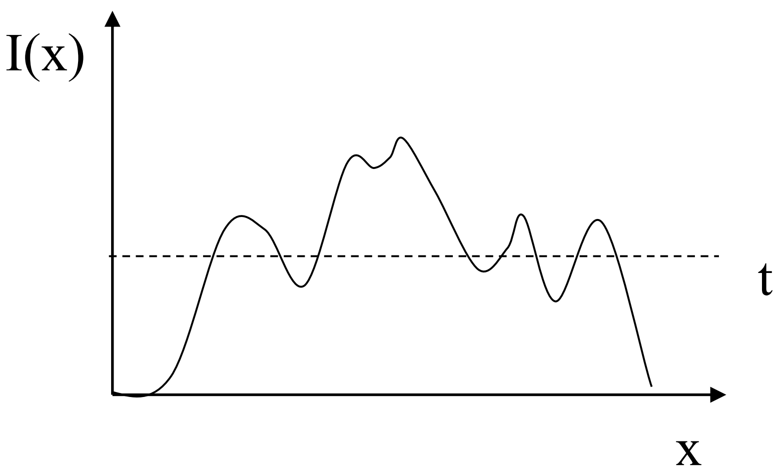
2.1.1 Thresholding, Histograms

- Thresholding

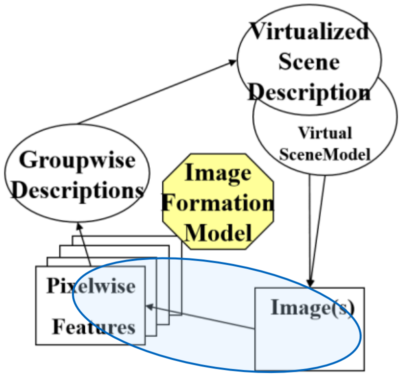
$$f_B[i,j] = \begin{cases} 0, \text{ if } f_A[i,j] < t \\ 1, \text{ if } f_A[i,j] > t \end{cases}$$



0	0	2	**6**	**7**	2	1	**5**	0
1	1	2	**7**	**8**	3	**5**	**6**	4
5	**5**	**5**	**6**	**7**	**7**	**8**	**8**	3
0	**5**	**6**	**7**	**7**	**8**	2	3	3
0	1	2	**6**	**6**	1	**6**	**5**	2



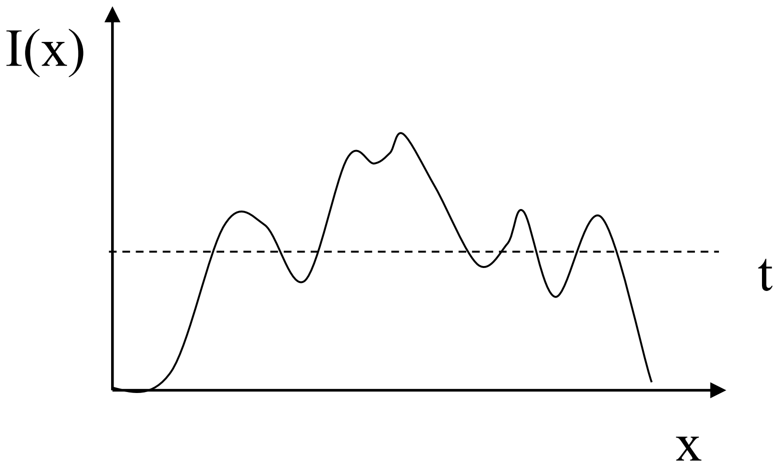
2.1.1 Thresholding, Histograms



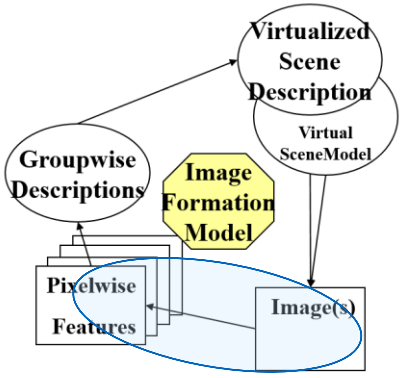
- Thresholding

$$f_B[i,j] = \begin{cases} 0, \text{ if } f_A[i,j] < t \\ 1, \text{ if } f_A[i,j] > t \end{cases}$$

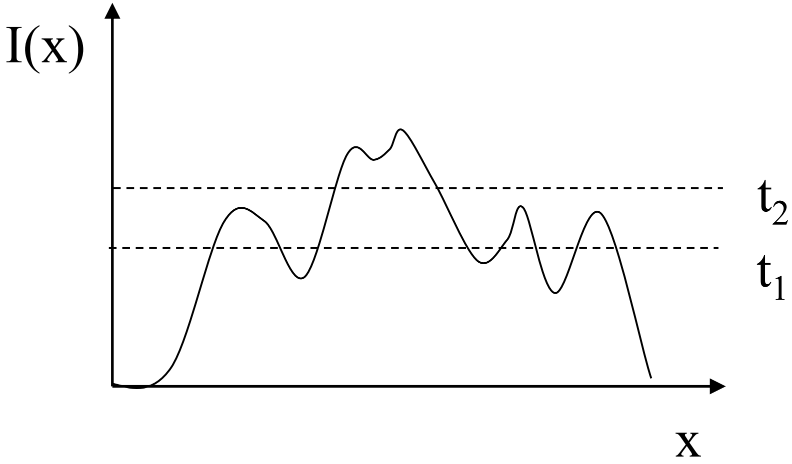
0	0	0	1	1	0	0	1	0
0	0	0	1	1	0	1	1	0
1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	0	0	0
0	0	0	1	1	0	1	1	0

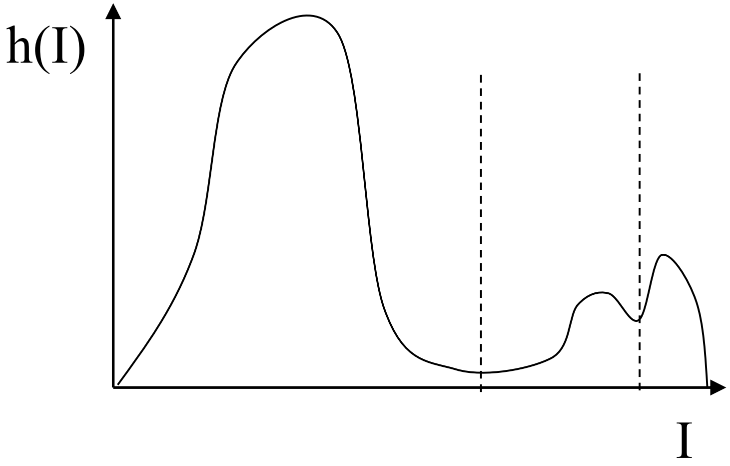


2.1.1 Thresholding, Histograms

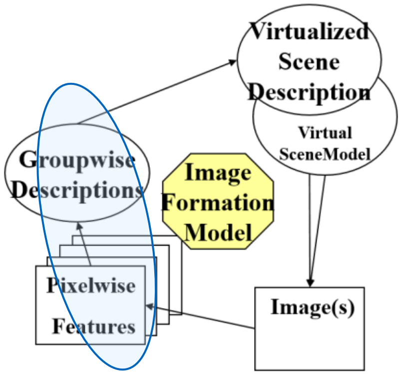


- Lookup tables $f_B[i,j] = \text{lookup}\ [f_A[i,j]]$
- P-tile method
- Mode method
- Iterative threshold selection
- Adaptive, variable thresholding
- Double thresholding





2.1.2 Connected Components



Pixel Neighborhoods

- 4-neighbors

$$\begin{matrix} 0 & 1 & 0 \\ 1 & 1 & 1 \\ 0 & 1 & 0 \end{matrix}$$

- 8-neighbors

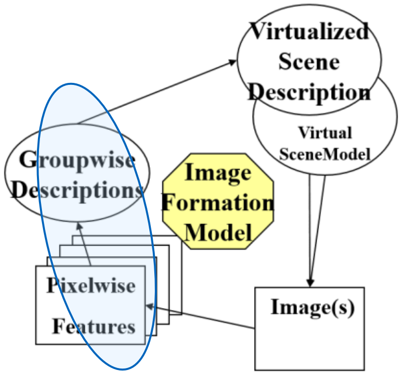
$$\begin{matrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{matrix}$$

- 4-path, 8-path
- Connected components
- Background and holes
- Boundary

```
0 0 0 1 1 0 0 1 0
0 0 0 1 1 0 1 1 0
1 1 1 1 1 1 1 1 0
0 1 1 1 1 1 0 0 0
0 0 0 1 1 0 1 1 0
```

Foreground		Background	
8-connected	$\begin{matrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{matrix}$	4-connected	$\begin{matrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{matrix}$
4-connected	$\begin{matrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{matrix}$	8-connected	$\begin{matrix} 0 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \end{matrix}$

2.1.2 Connected Components

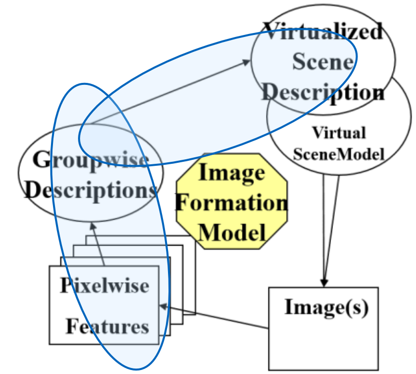


Algorithms

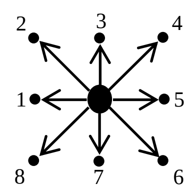
- Two-pass approach progressing in row-major order
- Recursive region growing
  - find seed
  - check each neighbor that hasn't been visited yet
  - if neighbor is inside the region,
    - label it
    - recurse

0	0	0	1	1	0	0	1	0
0	0	0	1	1	0	1	1	0
1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	0	0	0
0	0	0	1	1	0	2	2	0

2.1.3 Region Boundaries



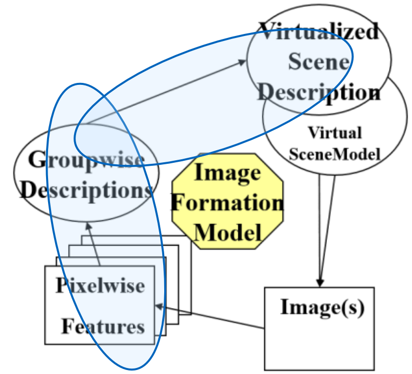
- Boundary pixel:
  A pixel that has 4-neighbors that don't belong to the region.
- Crab algorithm to follow a region boundary:
  - Find starting boundary pixel
  - Set starting search direction
  - Find next boundary pixel
  - Reset starting search direction



0	0	0	1	1	0	0	1	0
0	0	0	1	1	0	1	1	0
1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	0	0	0
0	0	0	1	1	0	2	2	0

- Shape fitting (ellipsoid, polygon, …)

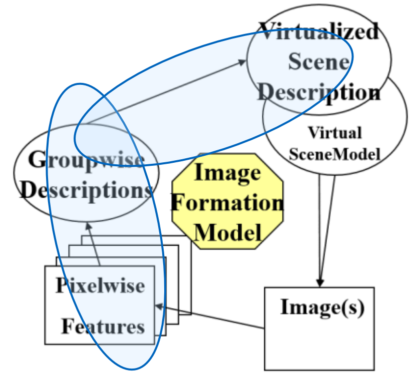
2.1.4 Region Properties



- Area A and perimeter P
- Compactness
  Relationship between A and P
- Boundary (Chain Code)

2	3	4
1	x	5
8	7	6

2.1.4 Region Properties



0	0	0	**1**	**1**	0	0	**1**	0
0	0	0	**1**	**1**	0	**1**	**1**	0
1	**1**	**1**	**1**	**1**	**1**	**1**	**1**	0
0	**1**	**1**	**1**	**1**	0	0	0	0
0	0	0	**1**	**1**	0	**1**	**1**	0

Geometric (Moments)

- **Size** (zeroth-order moment)

$$A = \sum\sum B[i,j]$$

- **Position** (first-order moments)

$$m_i = \sum\sum \frac{i\,B[i,j]}{A}$$

$$m_j = \sum\sum \frac{j\,B[i,j]}{A}$$

- **Orientation** (second-order moments)

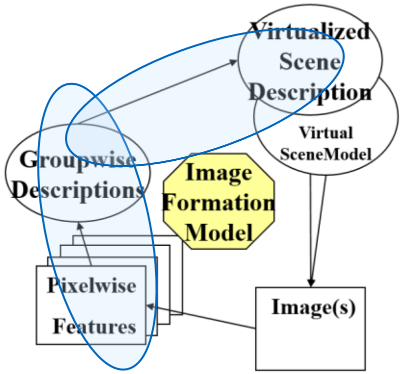
$$a = \sum\sum (i - m_i)^2\, B[i,j]$$

$$b = 2\sum\sum (i - m_i)\,(j - m_j)\, B[i,j]$$

$$c = \sum\sum (j - m_j)^2\, B[i,j]$$

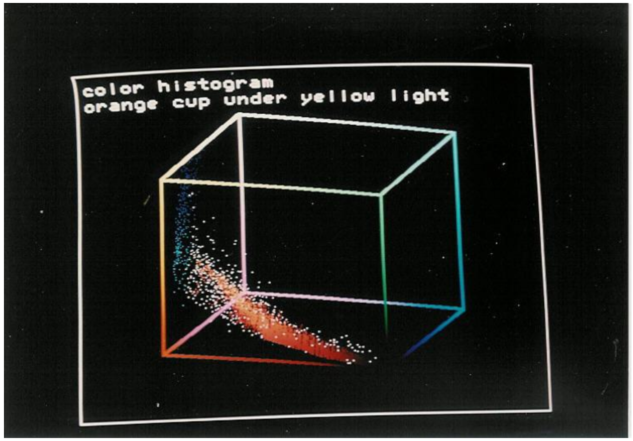
$$\tan 2\theta = \frac{b}{a - c}$$

2.1.4 Region Properties



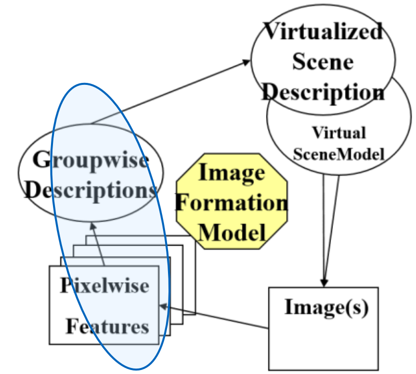
Photometric

- Mean intensity (color)
- Intensity variation
  (color variation: 3 eigenvectors)



- Repetitive patterns (textures)

2.1.5 Morphological Operations



- Intersection of two masks
- Union of two masks
- Complement of a mask

0 0 0 1 1 0 0 1 0
0 0 0 1 1 0 1 1 0
1 1 1 1 1 1 1 1 0
0 1 1 1 1 1 0 0 0
0 0 0 1 1 0 1 1 0

- **Dilation** of a mask by a structuring element (simple case: region expansion)

1 1 1
1 X 1
1 1 1

- **Erosion** (simple case: region shrinking) (morphological dual of dilation; dilation of the complement)

0 0 0
0 X 0
0 0 0

2.1.5 Morphological Operations

- **Opening**: erosion + dilation

Original

0	0	0	1	1	0	0	1	0
0	0	0	1	1	0	1	1	0
1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	0	0	0
0	0	0	1	1	0	1	1	0

Erode

1	0	1
0	X	0
1	0	1

0	0	0	1	1	0	0	1	0
0	0	0	1	1	0	1	1	0
1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	0	0	0
0	0	0	1	1	0	1	1	0

Dilate

0	1	0
1	X	1
0	1	0

0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0
0	0	0	1	1	0	0	0	0
0	0	0	1	1	0	0	0	0
0	0	0	0	0	0	0	0	0

0	0	0	0	0	0	0	0	0
0	0	0	1	1	0	0	0	0
0	0	1	1	1	1	0	0	0
0	0	1	1	1	1	0	0	0
0	0	0	1	1	0	0	0	0

→ Remove thin areas (spurious branches and points)

0	0	0	1	1	0	0	1	0
0	0	0	1	1	0	1	1	0
1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	0	0	0
0	0	0	1	1	0	1	1	0

2.1.5 Morphological Operations

- **Closing**: dilation + erosion

Original

0	0	0	1	1	0	0	1	0
0	0	0	1	1	0	1	1	0
1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	0	0	0
0	0	0	1	1	0	1	1	0

Dilate

0	1	0
1	X	1
0	1	0

0	0	0	1	1	0	0	1	0
0	0	0	1	1	0	1	1	0
1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	0	0	0
0	0	0	1	1	1	1	1	0

Erode

1	0	1
0	X	0
1	0	1

0	0	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	1	1

0	0	0	1	1	1	1	1	1
0	0	1	1	1	1	1	1	1
1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	1	0	0
0	0	1	1	1	1	1	1	0

→ Fill holes

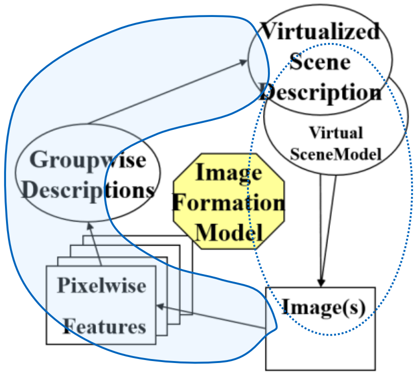
0	0	0	1	1	0	0	1	0
0	0	0	1	1	0	1	1	0
1	1	1	1	1	1	1	1	0
0	1	1	1	1	1	0	0	0
0	0	0	1	1	0	1	1	0

2.1 Region- (Similarity)- Based Approaches

Take home messages (things you should know):

- Region-based algorithms search for areas of similar pixels
- They often turn greyscale images into binary masks
- In these masks, neighboring pixels are grouped into „connected components“ to form „regions“
  - 4-connected vs. 8-connected neighbors
- Regions can be used to analyse higher-level feature/ object/ scene descriptions
- They can be post-processed to reduce the impact low-level pixel errors
  - Dilation vs. erosion

Overview



2. Feature Detection

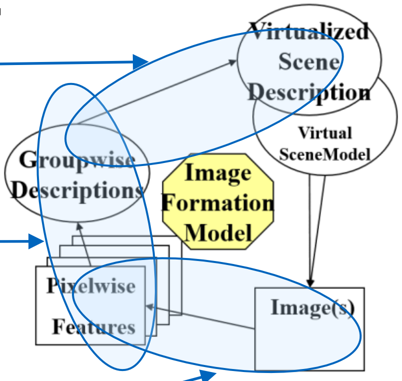
2.1 Region- (similarity)- based approaches

→ 2.2 Edge- (difference)- based approaches

Overview

2.2 Edge- (Difference)- Based Approaches

- 2.2.1 Basics
- 2.2.2 Robust edge detection
- 2.2.3 Shape fitting

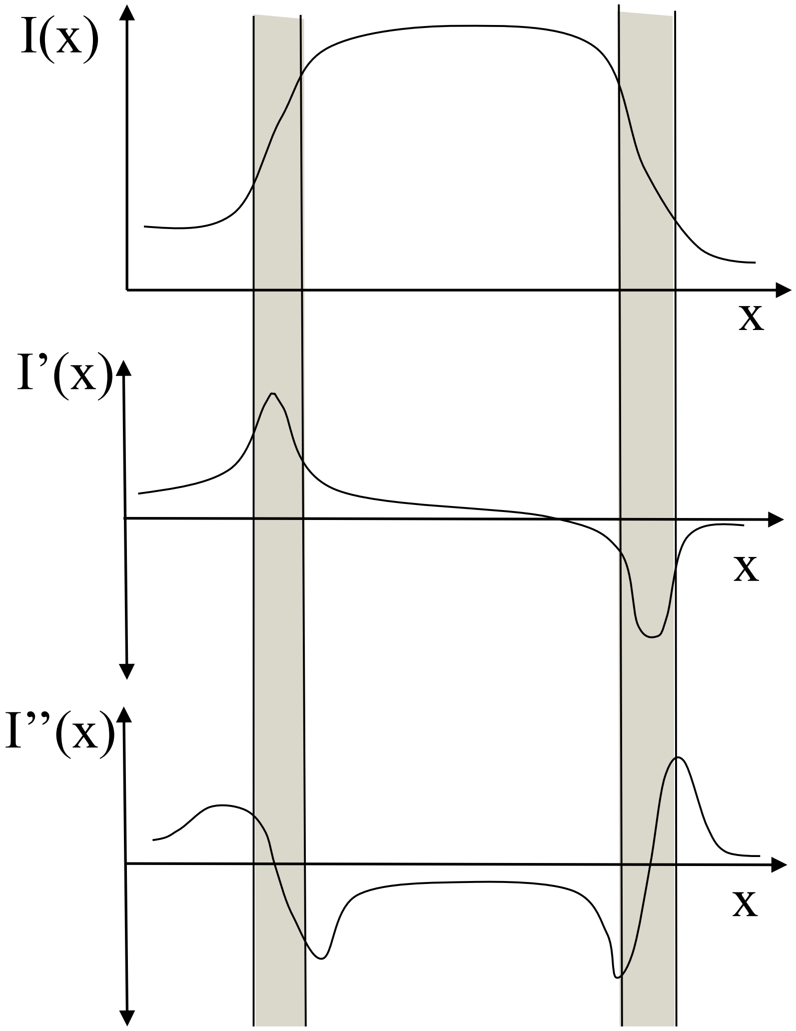


2.2.1 Basics

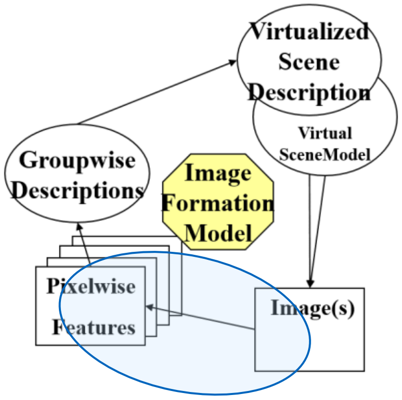
Maximal/Minimal
Image Gradient

Optima in the
1rst derivative

Zero-crossings in the
2nd derivative



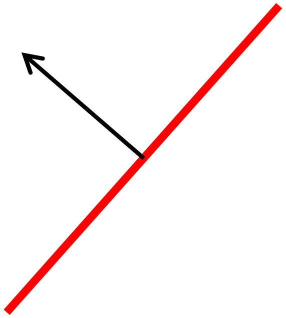
2.2.1 Basics



Continuous Case

Image gradient $\mathbf{G} = (G_x, G_y)$

First derivative of f(x,y)



- Orientation (direction of steepest ascent)

$$\alpha(x, y) = \tan^{-1}\left(\frac{G_y}{G_x}\right)$$

- Magnitude

$$\|\mathbf{G}\| = \sqrt{G_x{}^2 + G_y{}^2}$$
$$\approx \|G_x\| + \|G_y\|$$

2.2.1 Basics

Discrete Approximations

- Robert's Cross

$$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \qquad \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$$

- Sobel

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

- Prewitt

$$\begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix} \qquad \begin{bmatrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{bmatrix}$$

2.2.1 Basics

The Laplacian (2nd Derivative)

- Use zero-crossings of 2. derivative of f(x,y) (representing optima of the 1. derivative)

$$\nabla^2 \mathbf{f} = \frac{\delta^2 f}{\delta x^2} + \frac{\delta^2 f}{\delta y^2}$$

$$\frac{\delta^2 f}{\delta x^2} = f[i, j+1] - 2f[i, j] + f[i, j-1]$$

$$\frac{\delta^2 f}{\delta y^2} = f[i+1, j] - 2f[i, j] + f[i-1, j]$$

$$\boxed{\begin{matrix} 2 & -1 & 2 \\ -1 & -4 & -1 \\ 2 & -1 & 2 \end{matrix}} \quad \boxed{\begin{matrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{matrix}} \quad \boxed{\begin{matrix} 1 & 4 & 1 \\ 4 & -20 & 4 \\ 1 & 4 & 1 \end{matrix}}$$

- Problem: very sensitive to noise!

2.2.1 Basics

Image Convolution

$$h(x,y) = f(x,y) * g(x,y)$$

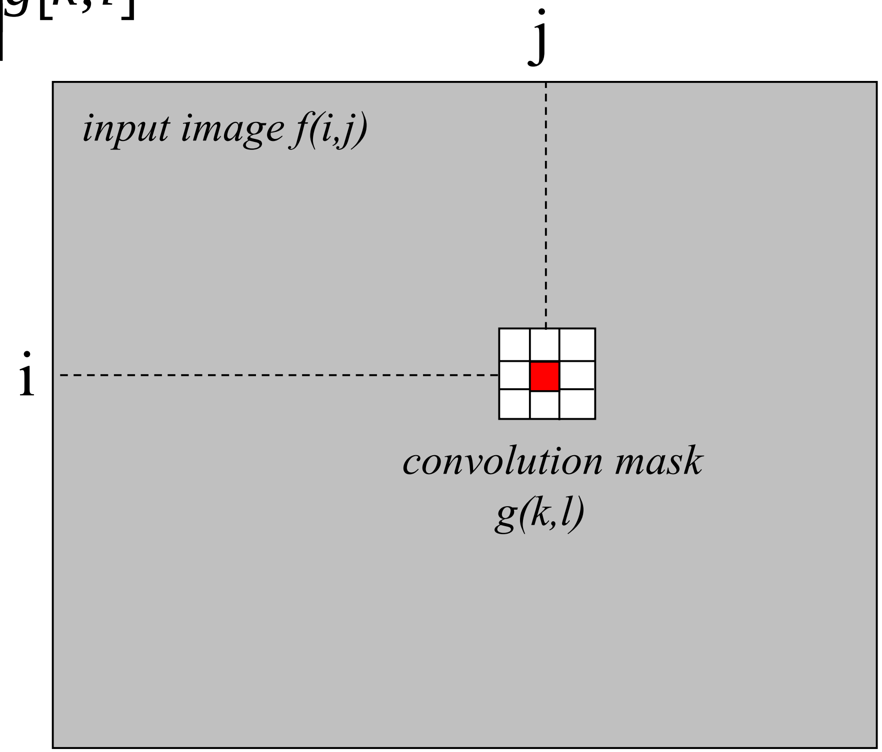
$$= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f(x+x', y+y')g(x', y')dx'dy'$$

$$h[i,j] = \sum_{k=0}^{n-1}\sum_{l=0}^{m-1} f\left[i-\frac{n}{2}+k, j-\frac{n}{2}+l\right]g[k,l]$$

2.2.1 Basics

Discrete Image Convolution

$$h[i,j]=\sum_{k=0}^{n-1}\sum_{l=0}^{m-1} f\left[i-\frac{n}{2}+k,\ j-\frac{n}{2}+l\right]g[k,l]$$

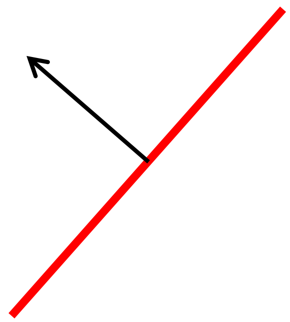


2.2.1 Basics

Result = Edgels

Pixel-wise indicator of local

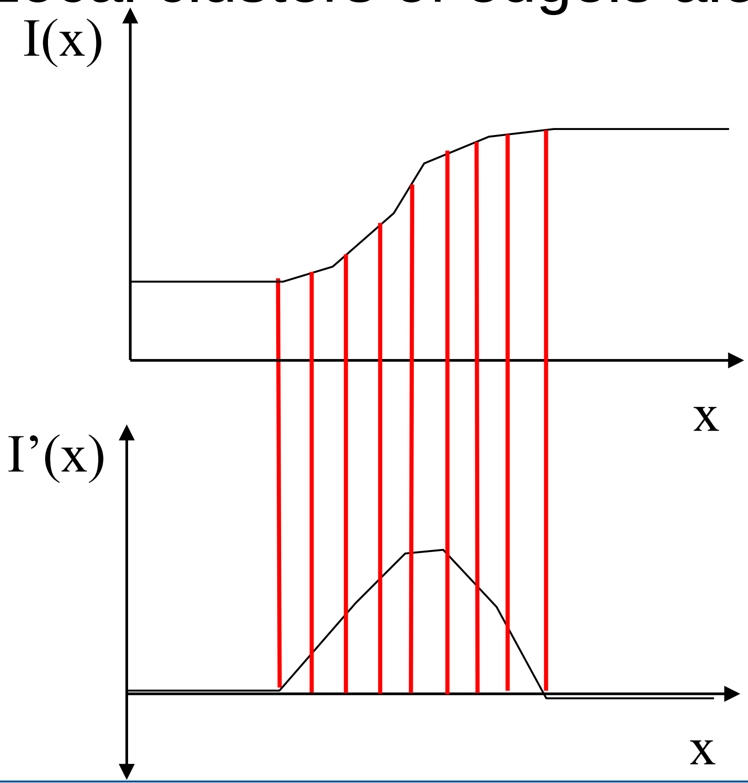
- edge strength
- dominant edge direction

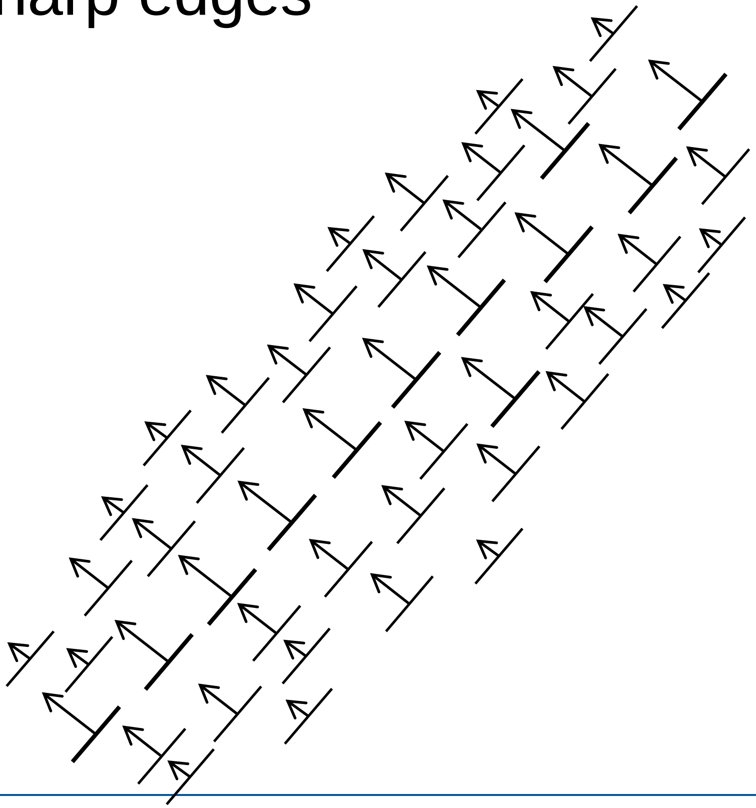


2.2.1 Basics

Edgels

Local clusters of edgels along unsharp edges

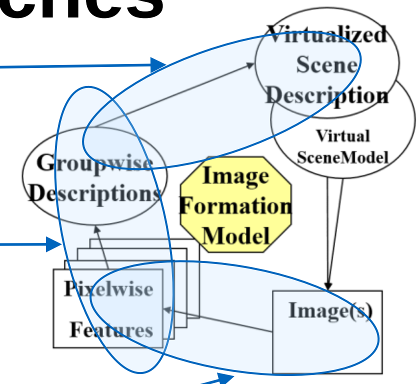




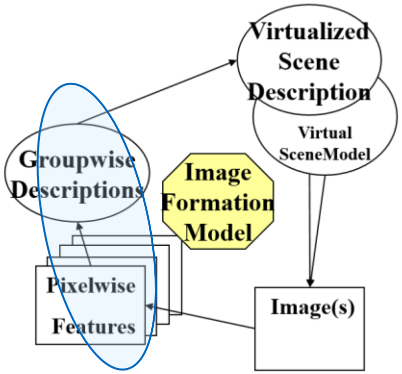
Overview

2.2 Edge- (Difference)- Based Approaches

- 2.2.1 Basics
- → 2.2.2 Robust edge detection
- 2.2.3 Shape fitting



2.2.2 Robust Edge Detection



Principle

- Detection problems (classification errors)
  - Lack of accuracy (position, orientation)
  - False edges (“false positives”)
  - Missing edges (“false negatives”)
- Combined filters for
  - Smoothing (noise reduction)
  - Enhancement (edge detection)
  - Detection (magnitude thresholding)
  - [Localization (sub-pixel precision)]

2.2.2 Robust Edge Detection

Combined filters for
- Smoothing (noise reduction)
- Enhancement (edge detection)
- Detection (magnitude thresholding)
- [Localization (sub-pixel precision)]

Smoothing Filters

- Local pixel averages

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix} * 1/9 \qquad \begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix} * 1/16$$

- Gaussian low-pass filter

$$g[k,l] = e^{\frac{-(k^2+l^2)}{2\sigma^2}}$$

$$\text{with} \quad k: \frac{-n}{2}..\frac{n}{2}, \quad l: \frac{-m}{2}..\frac{m}{2}$$

$$\begin{bmatrix} 1 & 1 & 2 & 2 & 2 & 1 & 1 \\ 1 & 2 & 2 & 4 & 2 & 2 & 1 \\ 2 & 2 & 4 & 8 & 4 & 2 & 2 \\ 2 & 4 & 8 & 16 & 8 & 4 & 2 \\ 2 & 2 & 4 & 8 & 4 & 2 & 2 \\ 1 & 2 & 2 & 4 & 2 & 2 & 1 \\ 1 & 1 & 2 & 2 & 2 & 1 & 1 \end{bmatrix} * 1/144$$

2.2.2 Robust Edge Detection

Combined filters for
- Smoothing (noise reduction)
- Enhancement (edge detection)
- Detection (magnitude thresholding)
- [Localization (sub-pixel precision)]

Enhancement

- Gradient-based edge detection
- Second derivative

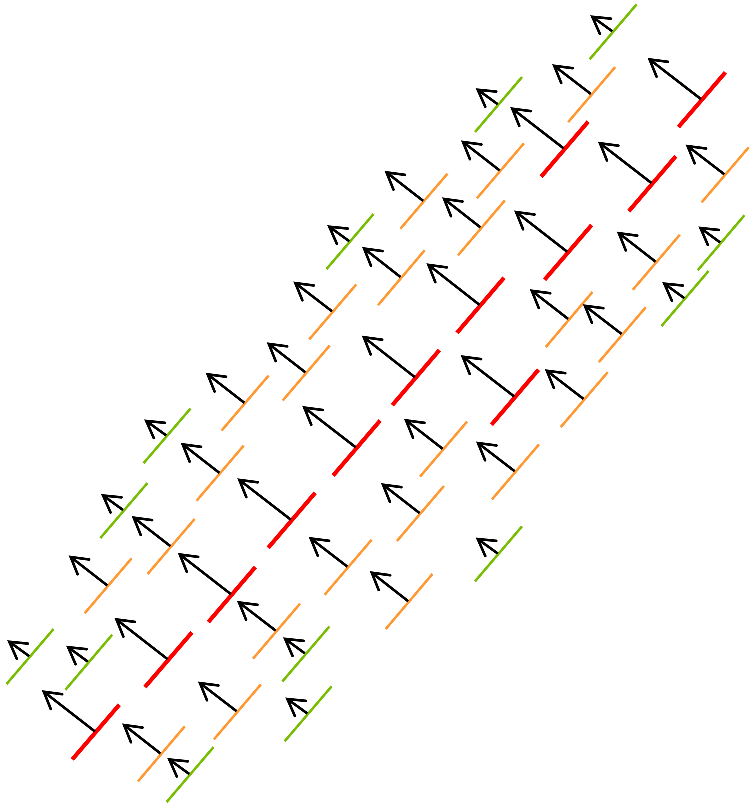
2.2.2 Robust Edge Detection

Combined filters for
- Smoothing (noise reduction)
- Enhancement (edge detection)
- Detection (magnitude thresholding)
- [Localization (sub-pixel precision)]

Detection

Thinning of edgel clusters:

- Determination of local maxima along the dominant edge direction
- Computation of zero-crossings in the 2nd derivative



2.2.2 Robust Edge Detection

Combined filters for
- Smoothing (noise reduction)
- Enhancement (edge detection)
- Detection (magnitude thresholding)
- [Localization (sub-pixel precision)]

Examples Laplacian of Gaussian (LoG) (“Mexican Hat Operator”)

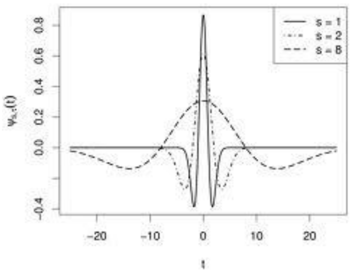
- **Smoothing** with a Gaussian filter
- **Enhancement** by 2. derivative edge detection
- **Detection** of zero crossings in 2. derivative (possibly in combination with large peak in 1. derivative)
- **Localization** with sub-pixel precision using linear interpolation

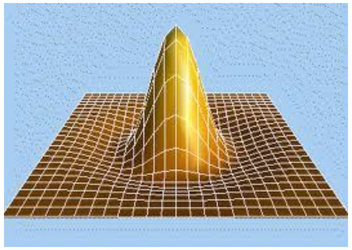
2.2.2 Robust Edge Detection

Combined filters for
- Smoothing (noise reduction)
- Enhancement (edge detection)
- Detection (magnitude thresholding)
- [Localization (sub-pixel precision)]

Examples: Laplacian of Gaussian (LoG) ("Mexican Hat Operator")

$$h(x,y) = \Delta^2\left[g(x,y) * f(x,y)\right]$$
$$= \left[\Delta g^2(x,y)\right] * f(x,y)$$





0	0	-1	0	0
0	-1	-2	-1	0
-1	-2	16	-2	-1
0	-1	-2	-1	0
0	0	-1	0	0

0	0	0	0	0	0	-1	-1	-1	-1	-1	0	0	0	0	0	0
0	0	0	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	0	0
0	0	-1	-1	-1	-2	-3	-3	-3	-3	-3	-2	-1	-1	-1	0	0
0	0	-1	-1	-2	-3	-3	-3	-3	-3	-3	-3	-2	-1	-1	0	0
0	-1	-1	-2	-3	-3	-3	-2	-3	-2	-3	-3	-3	-2	-1	-1	0
0	-1	-2	-3	-3	-3	0	2	4	2	0	-3	-3	-3	-2	-1	0
-1	-1	-3	-3	-3	0	4	10	12	10	4	0	-3	-3	-3	-1	-1
-1	-1	-3	-3	-2	2	10	18	21	18	10	2	-2	-3	-3	-1	-1
-1	-1	-3	-3	-3	4	12	21	24	21	12	4	-3	-3	-3	-1	-1
-1	-1	-3	-3	-2	2	10	18	21	18	10	2	-2	-3	-3	-1	-1
-1	-1	-3	-3	-3	0	4	10	12	10	4	0	-3	-3	-3	-1	-1
0	-1	-2	-3	-3	-3	0	2	4	2	0	-3	-3	-3	-2	-1	0
0	-1	-1	-2	-3	-3	-3	-2	-3	-2	-3	-3	-3	-2	-1	-1	0
0	0	-1	-1	-2	-3	-3	-3	-3	-3	-3	-3	-2	-1	-1	0	0
0	0	-1	-1	-1	-2	-3	-3	-3	-3	-3	-2	-1	-1	-1	0	0
0	0	0	0	-1	-1	-1	-1	-1	-1	-1	-1	-1	0	0	0	0
0	0	0	0	0	0	-1	-1	-1	-1	-1	0	0	0	0	0	0

2.2.2 Robust Edge Detection

Combined filters for
- Smoothing (noise reduction)
- Enhancement (edge detection)
- Detection (magnitude thresholding)
- [Localization (sub-pixel precision)]

Examples: Canny Operator

- First derivative of a Gaussian
  (sensitive in the direction of steepest change, insensitive along edge)

$$S[i,j] = G[i,j;\sigma] * I[i,j]$$

$$P[i,j] = -S[i,j] + S[i,j+1] - S[i+1,j] + S[i+1,j+1]$$
$$Q[i,j] = S[i,j] + S[i,j+1] - S[i+1,j] - S[i+1,j+1]$$

$$\boxed{\begin{matrix} -1 & 1 \\ -1 & 1 \end{matrix}}$$

$$\boxed{\begin{matrix} 1 & 1 \\ -1 & -1 \end{matrix}}$$

- Nonmaxima suppression (ridge thinning)
- Double thresholding to detect and link edges

2.2.2 Robust Edge Detection

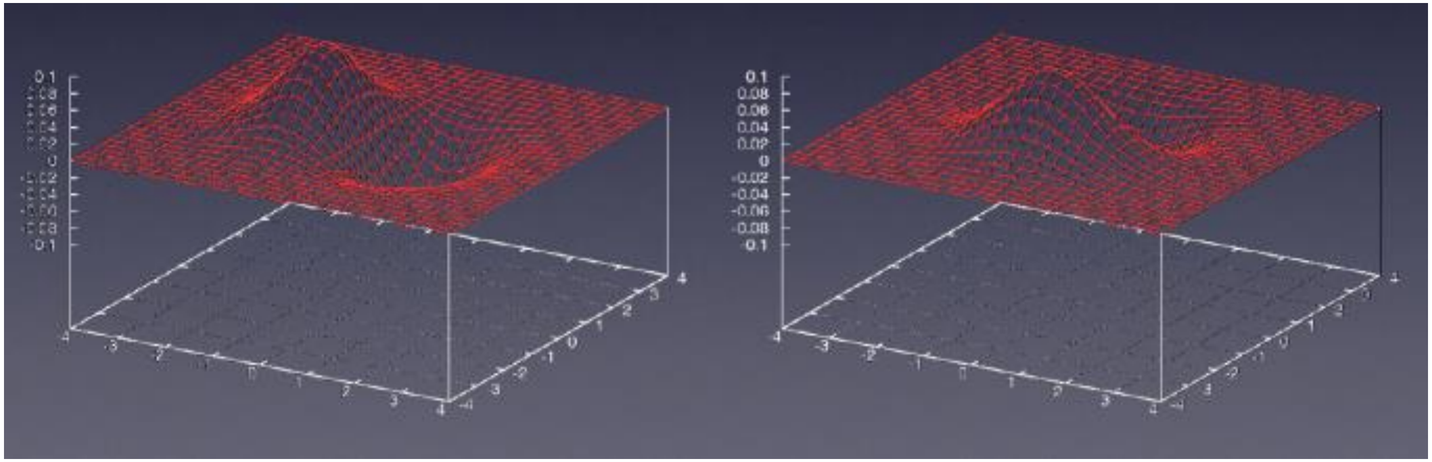
Combined filters for
- Smoothing (noise reduction)
- Enhancement (edge detection)
- Detection (magnitude thresholding)
- [Localization (sub-pixel precision)]

Examples: Canny Operator

- First derivative of a Gaussian
(sensitive in the direction of steepest change, insensitive along edge)

$$\begin{array}{|cc|}\hline -1 & 1 \\ -1 & 1 \\ \hline\end{array}$$

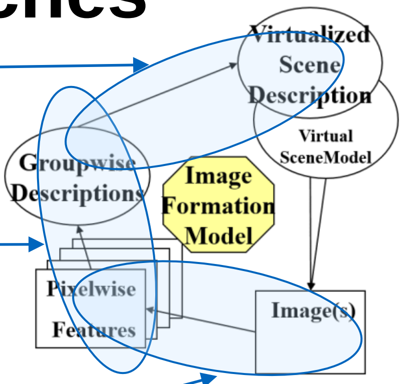
$$\begin{array}{|cc|}\hline 1 & 1 \\ -1 & -1 \\ \hline\end{array}$$



- Nonmaxima suppression (ridge thinning)
- Double thresholding to detect and link edges

Overview

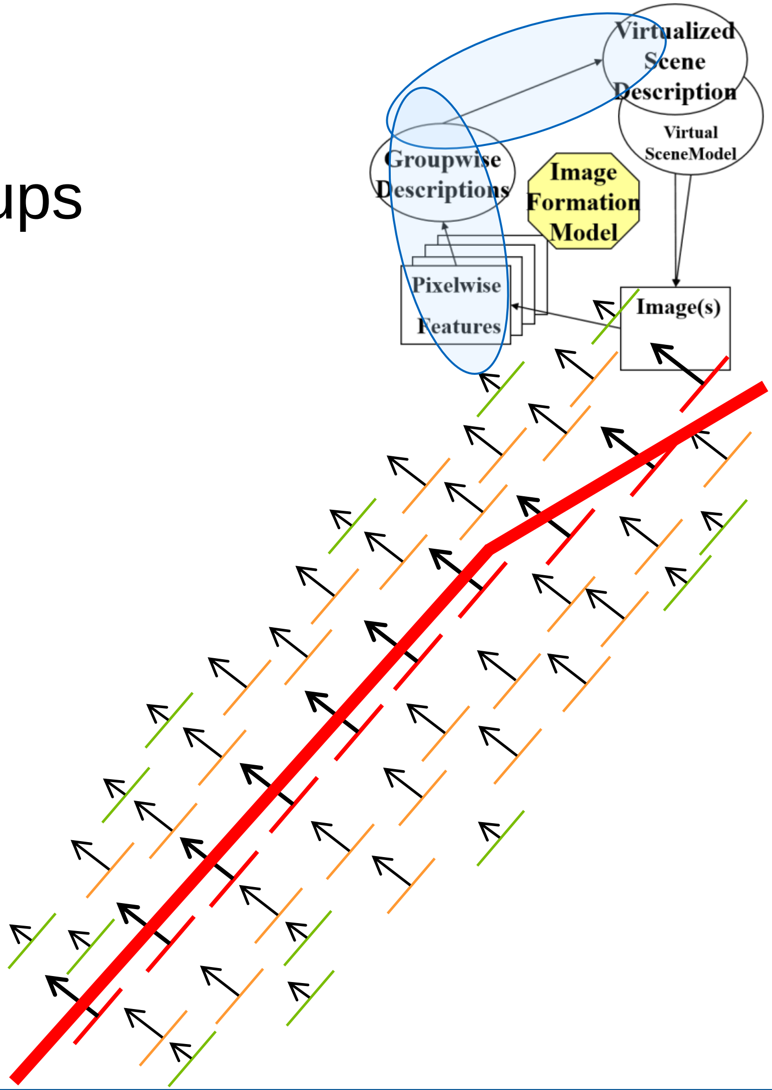
2.2 Edge- (Difference)- Based Approaches



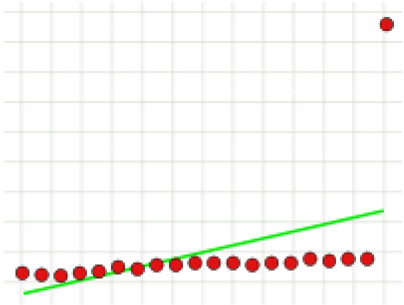
- 2.2.1 Basics
- 2.2.2 Robust edge detection
- → 2.2.3 Shape fitting

2.2.3 Shape Fitting

- Determination of consistent groups of edgels (clustering)
- Robust estimation of shape parameters (least squares, disregarding outliers)

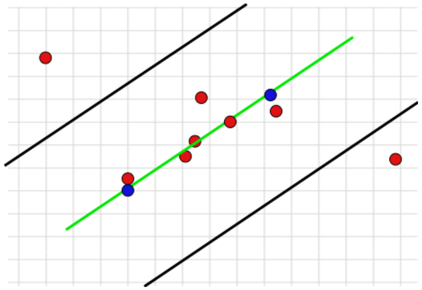


2.2.3 Shape Fitting



Robust Fitting

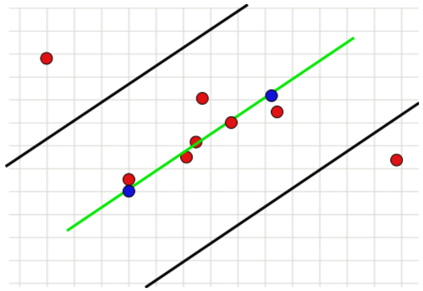
- RANSAC (Random Sample Consensus [Fischler and Bolles 1981]
  - Iterate:
    - Select a small random subset of points (or edgels)
    - Fit a line (or other geometric curve) to them (Exact computation or least squares)
      - ➠ form a hypothesis (generate a hypothesized line model)
    - Check for all existing points (edgels) how well they conform with the hypothesized line
      - ➠ Determine the "consensus set"
  - Select line model with best consensus set.
  - Find and analyze outliers.



2.2.3 Shape Fitting

Quality Criteria

- Goodness of fit
  - Maximum absolute error
  - Mean squared error
  - Normalized maximum error
  - Number of sign changes
  - Ratio of curve length to end point distance



2.2.3 Shape Fitting

Models

- Line segments (Polylines)
- Circular arcs
- Conic sections
- Cubic splines

2.2.3 Shape Fitting

Models

Polylines

- Sequence of line segments
  - Implicit line representation f(x,y) = 0
  - Distance from line -> f(x,y) = d

- Polyline splitting (recursive subdivision)
- Segment merging (bottom-up approach)
- Split and merge

2.2 Edge- (Difference)- Based Approaches

Take home messages (things you should know):

- Edge detection works by finding large differences between neighboring pixels → „edgels“
(first or second derivative of image function)
- Edge detectors can be described by masks; convolution is a general scheme to apply masks to every pixel
  - Masks for simple edge detectors

2.2 Edge- (Difference)- Based Approaches

Take home messages (things you should know), cont.:

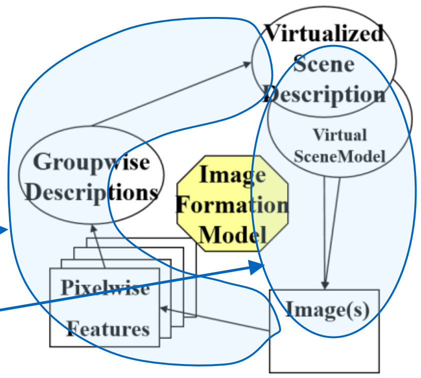
- Edgels alone are not enough; to determine higher level image structures (edges), edgels need to be smoothed and combined → robust edge detection
  - Integration (smoothing) vs. differentiation (sharpening)
- Higher level shape descriptors (e.g. lines) can be fitted to edges

Agenda

1. Image Formation (Geometry, Radiometry)
2. Feature Detection
3. Feature Matching and Tracking

3. Feature Matching and Tracking

Overview

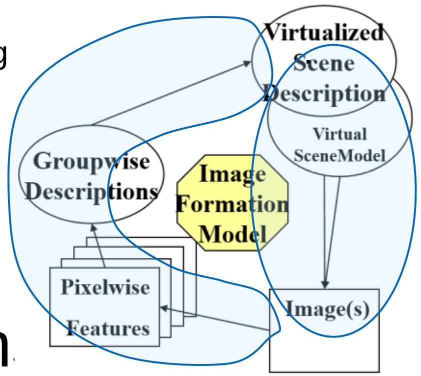


- 3.1 Basic algorithm
- 3.2 Initialization:
  Feature detection and identification
- 3.3 Tracking
- 3.4 Feature redetection

3.1 Basic Algorithm

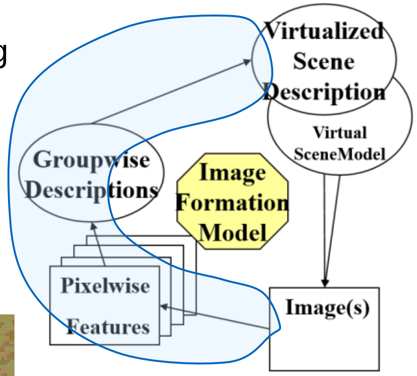
- Initialization: find known feature positions in image$_{1..i-1}$
- Tracking loop: (real-time)
  - Predict approximate position of each feature in the image$_i$
  - Search local image areas for features
  - Handle exceptions:
    - Disappearing feature
    - Partially visible feature
    - Reappearing feature
  - …Update feature motion models…
  - …Compute 3D interpretation…
  - If tracking is lost: re-initialize
- Continue loop

3.2 Init: Feature Detection/Identification



Top-Down and/or Bottom-Up
(Using a scene model or heuristic assumption)

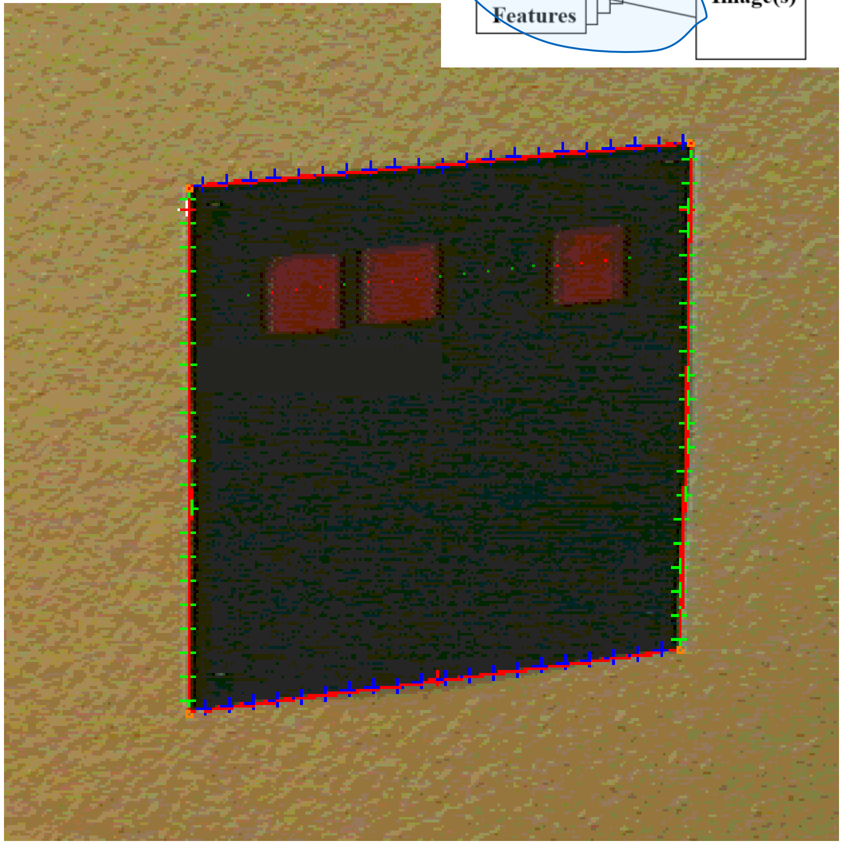
- Photometric (Colors)
  - Color constancy problems: changing illumination, shadowing, …
  - Invariant to changing image resolution / object size
- Geometric (Shapes)
  - One object at a time (unique recognition): Geometric invariants under 3D transformations and projections
  - Groups of objects in combination
    (Hough transforms, graph matching)



3.2 Init: Feature Detection/Identification

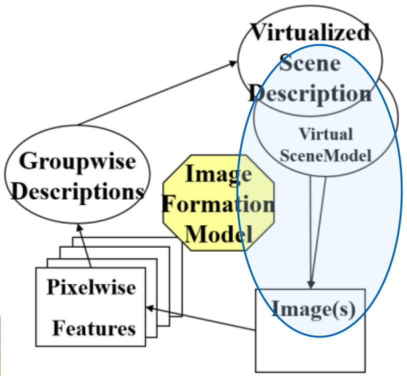
Example: Target Detection and Identification

- Find dark blobs on bright background
- Fit quadrilateral polygons
- Find corners
- Read ID label



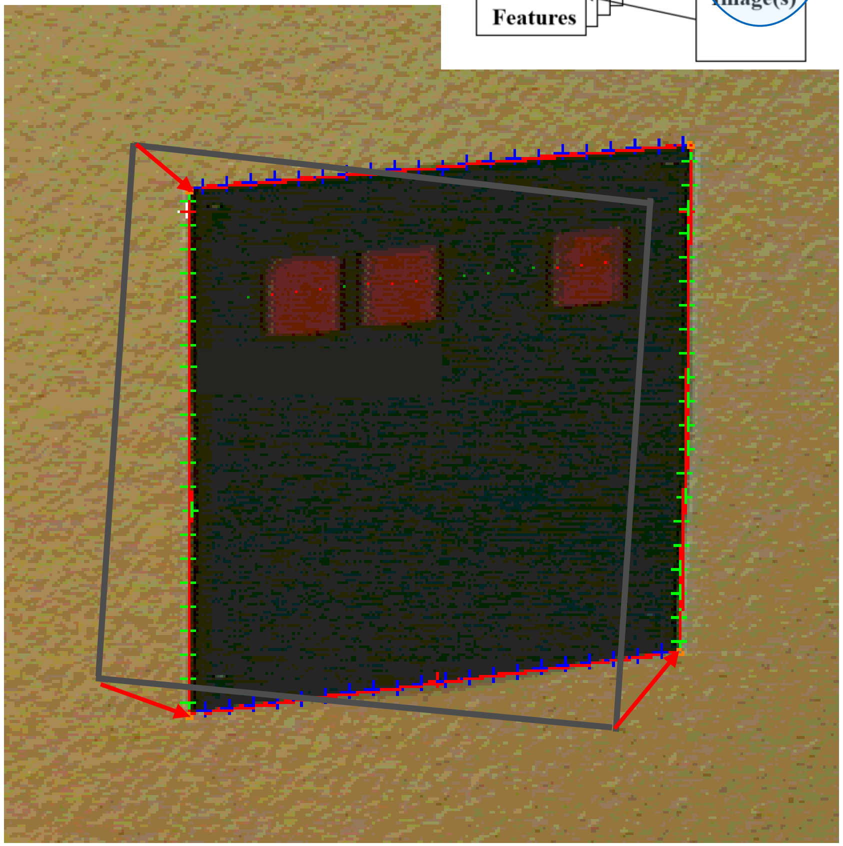
3.3 Tracking (2D)

3. Feature Matching and Tracking



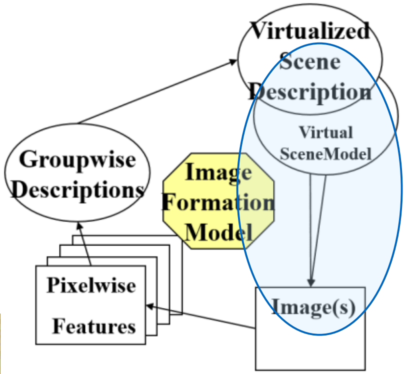
2D Motion Estimation

- Compute local motion vectors of every feature (images n-1, n-2)
- (more images to estimate higher-order motion models)



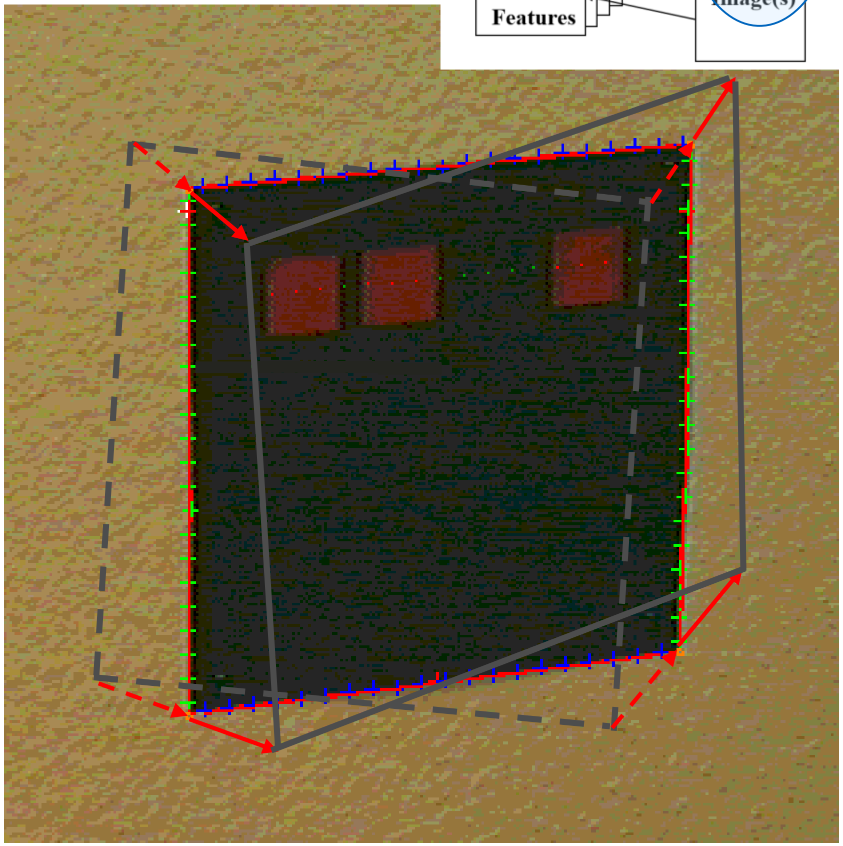
3.3 Tracking (2D)

3. Feature Matching and Tracking

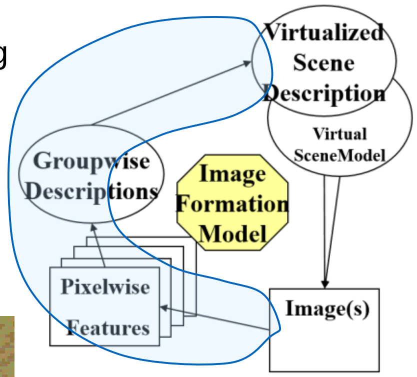


2D Motion Prediction

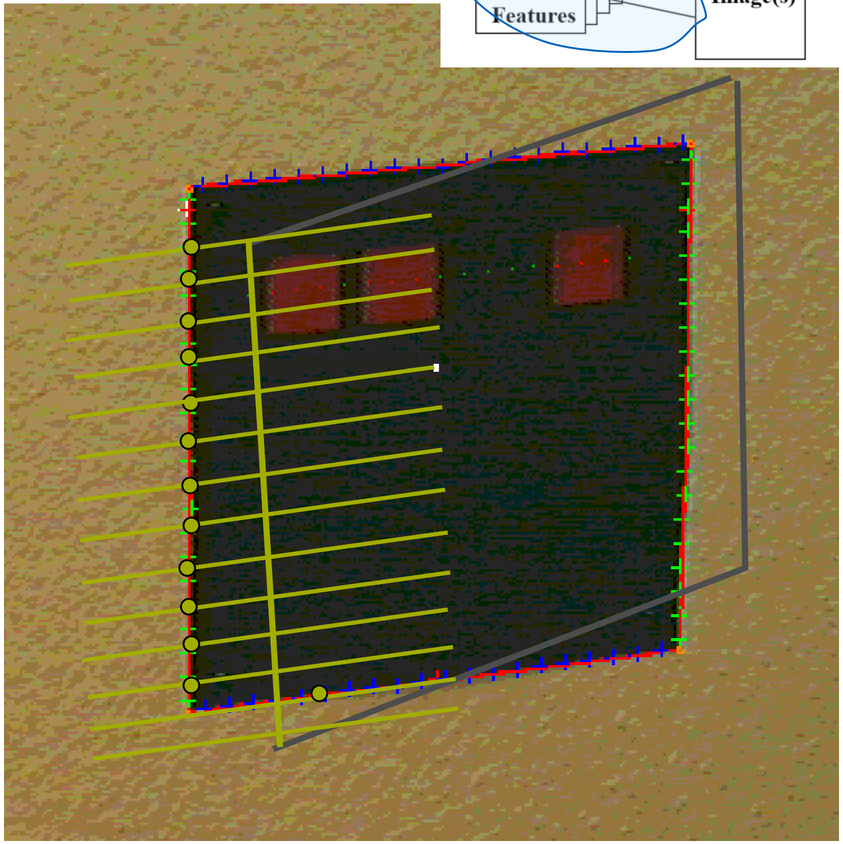
- Prediction of local feature motion for image n

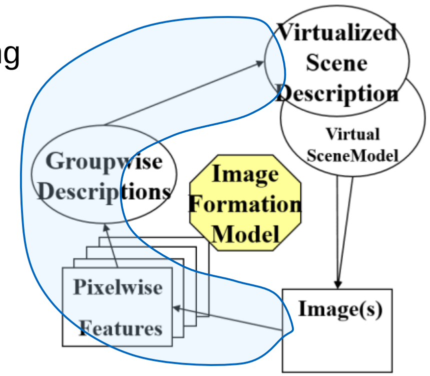


3.4 Feature (Re)detection



- Prediction of local feature motion for image n





3.4 Feature (Re)detection

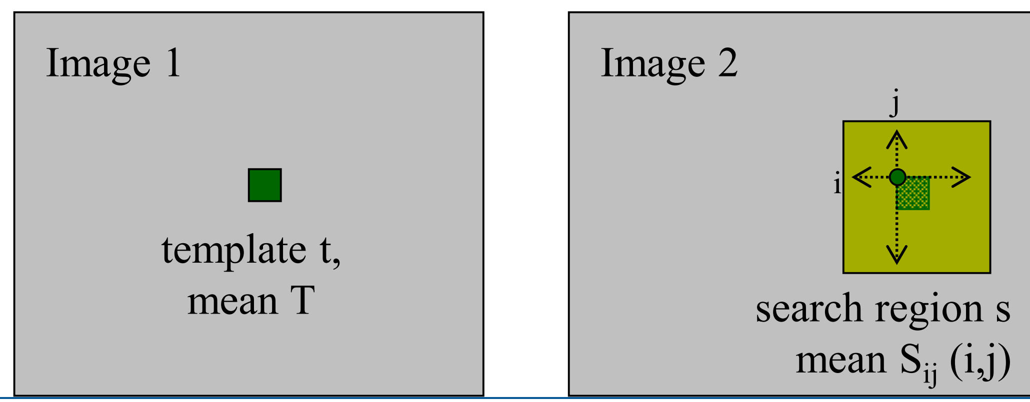
Cross Correlation

- Unnormalized:

$$r(i,j) = \sum t(x,y) * s(i+x, j+y)$$

- Normalized:

$$r(i,j) = \frac{\sum \left(t(x,y) - T\right) * \left(s(i+x, j+y) - S_{ij}\right)}{\sqrt{\sum \left(t(x,y) - T\right)^2 * \sum \left(s(i+x, j+y) - S_{ij}\right)^2}}$$



3. Feature Matching and Tracking

Take home messages (things you should know):

- Initialization vs. tracking loop
- Which problems can occur in the tracking loop?
- Motion models
- Feature re-dection concepts

Thank you!