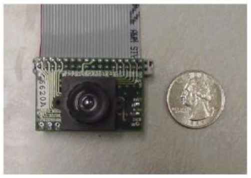
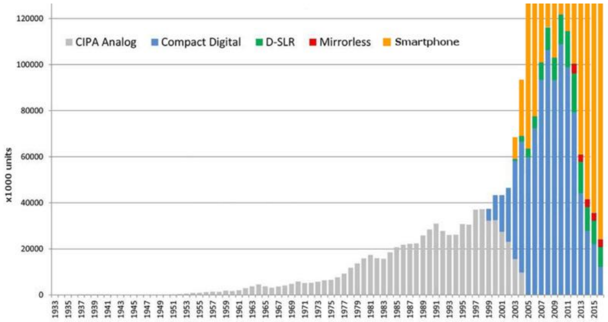
Cameras, Images and Image Processing

Cameras

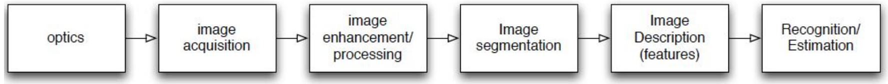
- Most flexible sensory modality
- Complex sensory processing
- Not discussed in any detail
- Offers wide range
- Diverse tasking of sensor
- Relatively inexpensive
- Computationally demanding



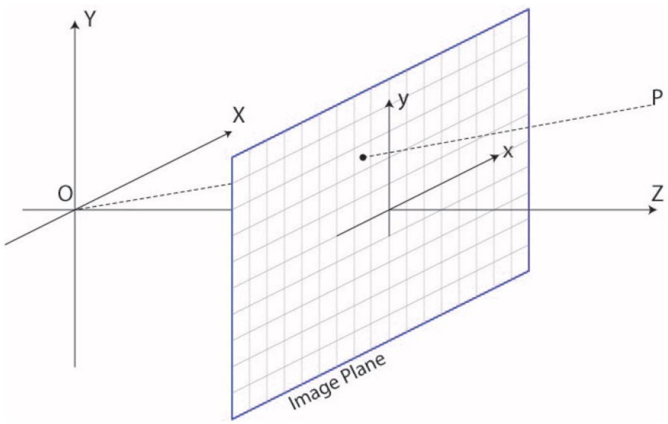
Cameras



The processing chain



The pinhole camera model



The Pin-Hole Model

- The relations are then:

$$\frac{x}{\lambda} = \frac{X}{Z}$$
$$\frac{y}{\lambda} = \frac{Y}{Z}$$
$$\Rightarrow$$
$$x = \frac{\lambda X}{Z}$$
$$y = \frac{\lambda Y}{Z}$$

In homogenous coordinates

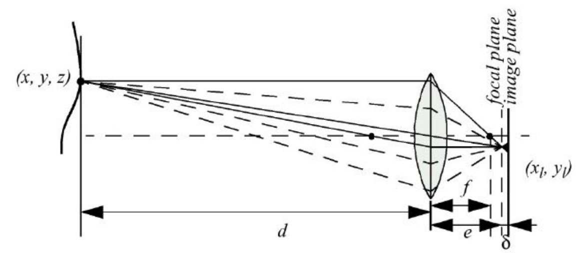
- Remember Homogeneous Coordinates?

$$\vec{P} = \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

- Define the Perspective transform as

$$\mathbf{T} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & \frac{1}{\lambda} & 0 \end{bmatrix}$$

Depth from defocus



- Basic geometry: $\frac{1}{f} = \frac{1}{d} + \frac{1}{e}$
- The smear is proportional to distance

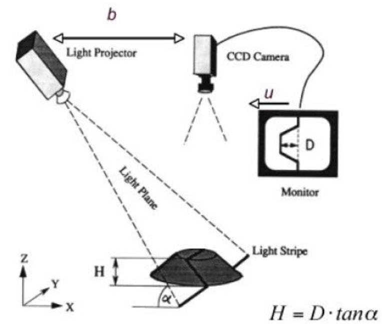
$$R = \frac{L\delta}{2e}$$

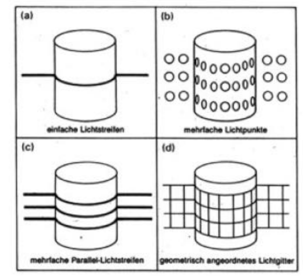
Defocus example

- Local sharpness: $LS = \sum_{x,y} |I(x,y) - I(x-1,y)|$

Structured light

- Segmentation of images is a “hard” problem
- Active illumination simplifies the problem
- In particular in industrial inspection





Images

- Images are basically a 2D array of intensity/color values
- Image types

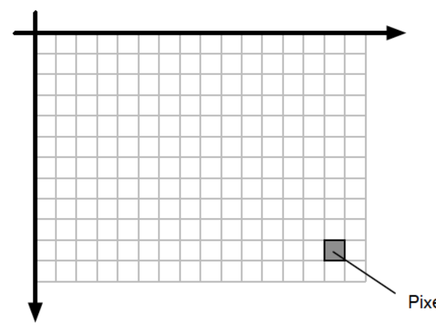
Color

Grayscale

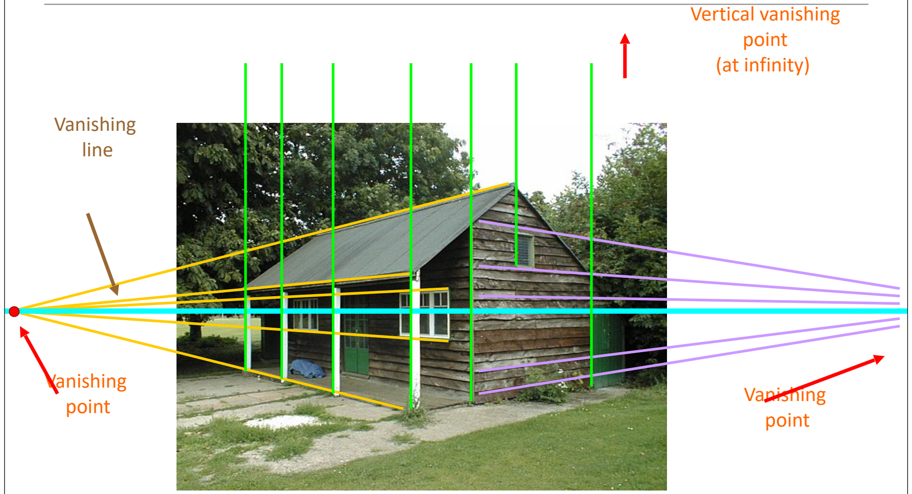
B-W

Images

- Matrix of values
- The picture element is named a pixel



Photographs are Projections



Slide Courtesy Antonio Criminisi

Intrinsic Calibration

3 × 3 Calibration Matrix K

$$m = \begin{bmatrix} u \\ v \\ w \end{bmatrix} = K[I \quad 0]M = \begin{bmatrix} \alpha & s & u_0 \\ & \beta & v_0 \\ & & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ T \end{bmatrix}$$

Recover image (Euclidean) coordinates by normalizing :

$$\hat{u} = \frac{u}{w} = \frac{\alpha X + sY + u_0}{Z}$$

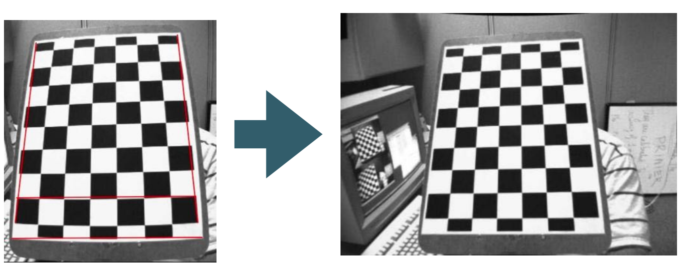
$$\hat{v} = \frac{v}{w} = \frac{\beta Y + v_0}{Z}$$

skew

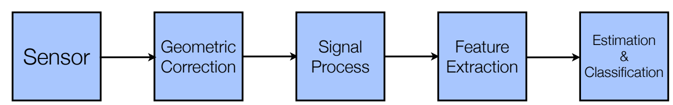
5 Degrees of Freedom !

Camera Calibration

- You have serious distortion on the RB5
- OpenCV (opencv-python) has tools for calibration
- https://opencv-python-tutroals.readthedocs.io/en/latest/py_tutorials/py_calib3d/py_calibration/py_calibration.html



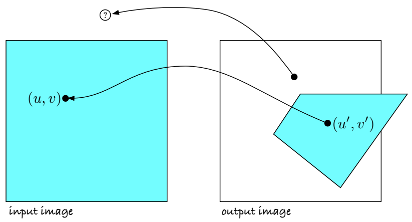
The Basic Process

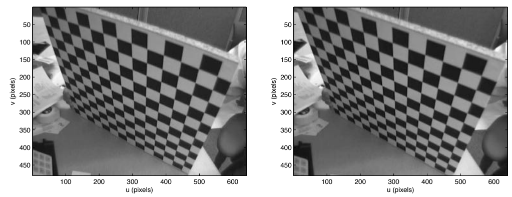


- Geometric Correction - Alignment to a calibration model
- Signal Processing - Clean up of data and signal conditioning
- Feature Extraction - Data compression and signal separation
- Estimation - Model, Space and Time Integration for estimation of key parameters
- Classification/Categorization - Assignment of one of N classes to data

Geometric Correction

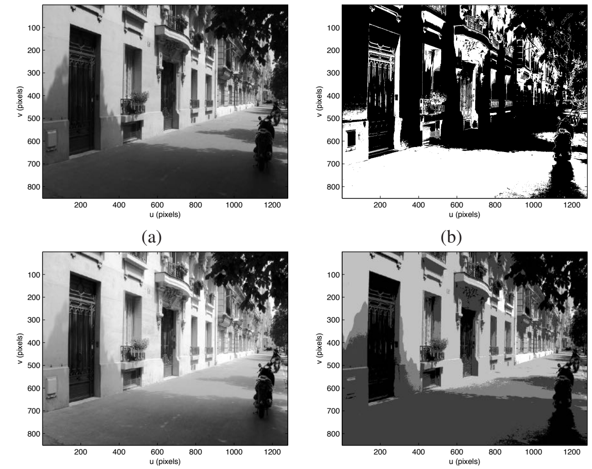
- Typically warping of signal to remove distortions





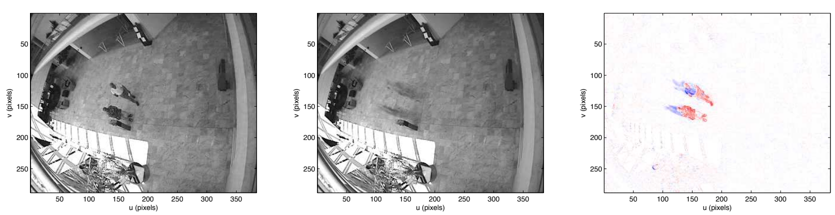
(a) (b)

Signal enhancement



(a) (b) (c) (d)

Spatial operations

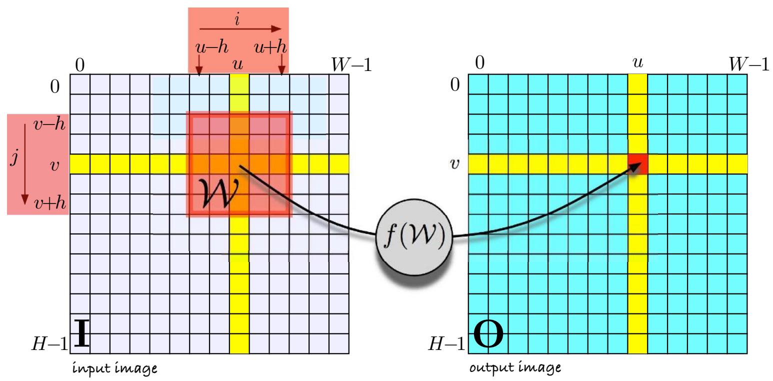


(a) (b) (c)

Filtering

- Noise removal
- Edge detection
- Texture description
- Multi-scale algorithms
- Feature detection
- Matched filters

Spatial convolutions



(c) Henrik I Christensen

What is Image Filtering?

Modify the pixels in an image based on some function of a local neighborhood of the pixels

10	5	3
4	5	1
1	1	7

Some function →

	7	

Linear Filtering

- Linear case is simplest and most useful
  - Replace each pixel with a linear combination of its neighbors.
- Prescription for linear combination is called the convolution kernel.

10	5	3
4	5	1
1	1	7

*

0	0	0
0	0.5	0
0	1.0	0.5

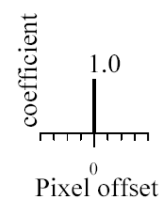
kernel

=

	7	

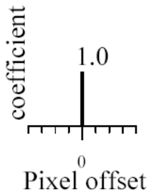
Filtering Examples

original



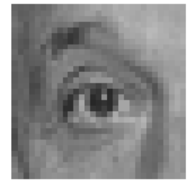
Filtering Examples: Identity

original

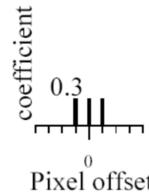


Filtered
(no change)

Filtering Examples



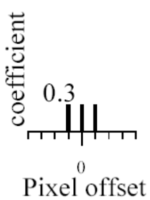
original



?

Filtering Examples: Blur

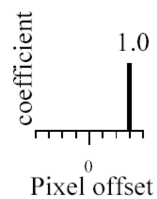
original



Blurred (filter applied in both dimensions).

Filtering Examples

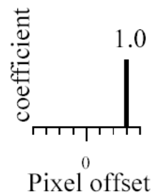
original

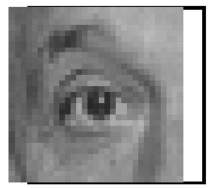


?

Filtering Examples: Shift

original





shifted

Convolutions to enhance images

Properties of convolution. Convolution obeys the familiar rules of algebra, it is commutative

$$A \otimes B = B \otimes A$$

associative

$$A \otimes B \otimes C = (A \otimes B) \otimes C = A \otimes (B \otimes C)$$

distributive (superposition applies)

$$A \otimes (B + C) = A \otimes B + A \otimes C$$

linear

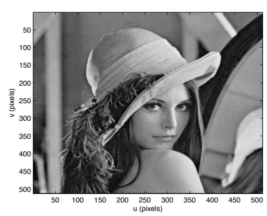
$$A \otimes (\alpha B) = \alpha(A \otimes B)$$

and shift invariant — if $S(\cdot)$ is a spatial shift then

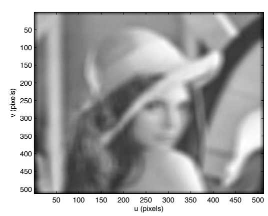
$$A \otimes S(B) = S(A \otimes B)$$

that is, convolution with a shifted image is the same as shifting the result of the convolution with the unshifted image.

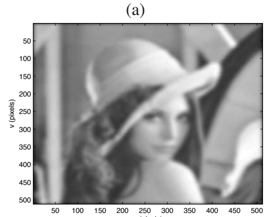
Smoothing Example



(a)

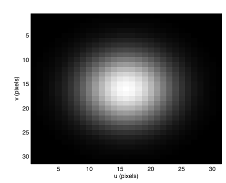


(b)

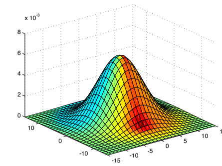


(c)

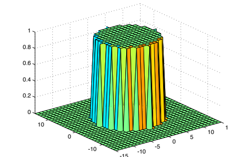
A few typical kernels



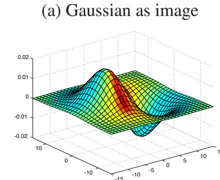
(a) Gaussian as image



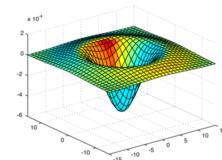
(b) Gaussian



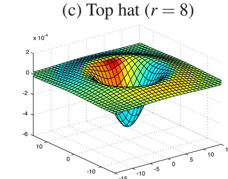
(c) Top hat ($r = 8$)



(d) Derivative of Gaussian



(e) Laplacian of Gaussian



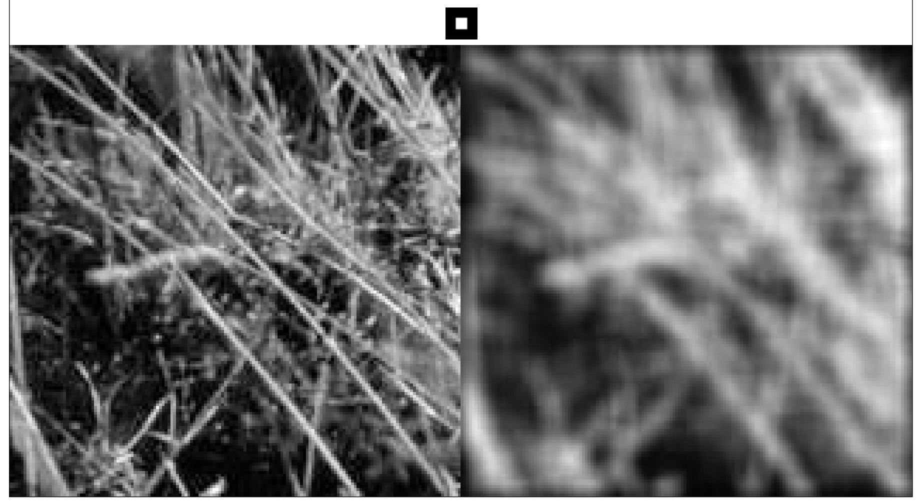
(f) Difference of Gaussian

Convolution

- Represent these weights as an image, H
- H is usually called the **kernel**
- Operation is called **convolution**

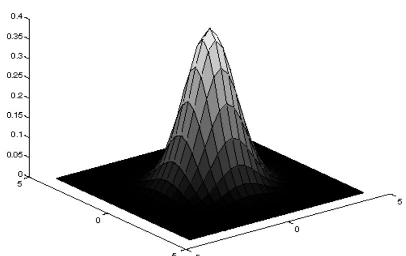
$$R_{ij} = \sum_{u,v} H_{i-u,j-v} F_{uv}$$

Example: Smoothing by Averaging



Smoothing with a Gaussian

- Averaging does not model defocussed lens well
- impulse response should be fuzzy blob



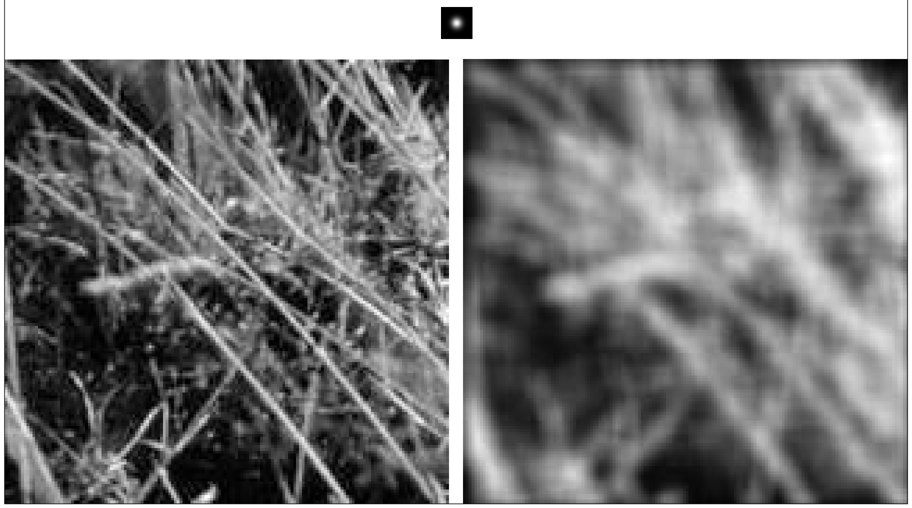
An Isotropic Gaussian

- The picture shows a smoothing kernel proportional to

$$\exp\left(-\left(\frac{x^2+y^2}{2\sigma^2}\right)\right)$$

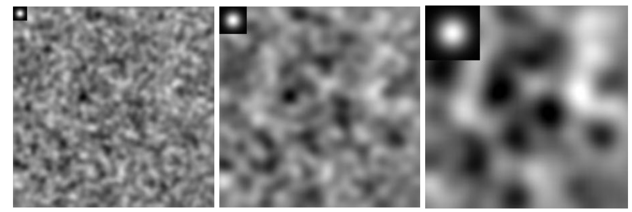
- reasonable model of a circularly symmetric blob

Smoothing with a Gaussian



Filter responses are correlated

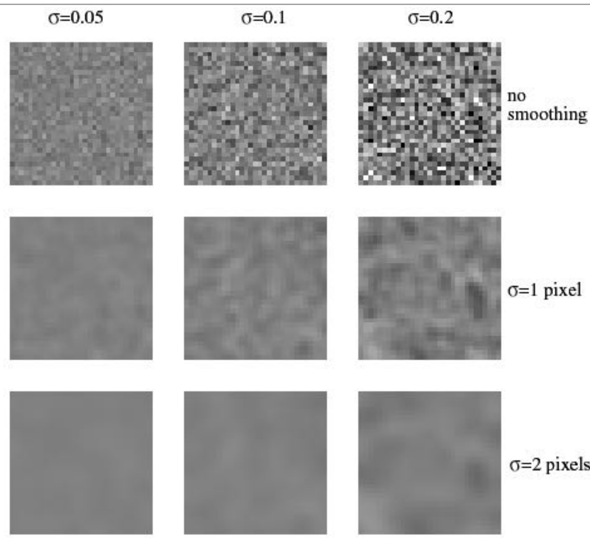
- Correlated over scales similar to scale of filter
- Filtered noise is sometimes useful
  - looks like some natural textures, can be used to simulate fire, etc.



sigma=1

sigma=16

The effects of smoothing

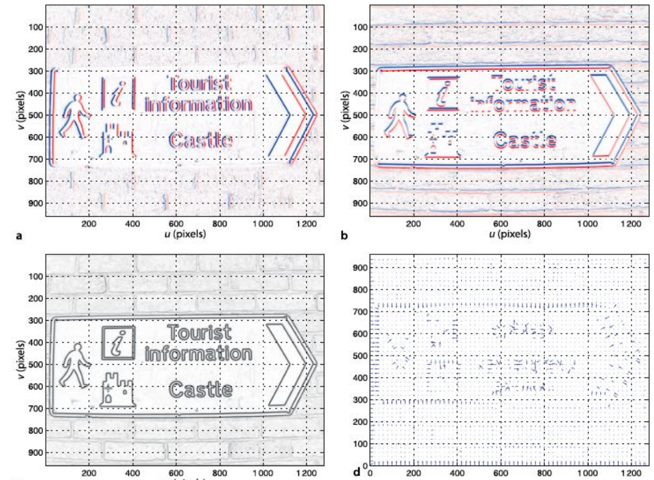


Edge Detection

- Sobel Kernel (Corke Chapter 12)

```
>> Du = ksobel
Du =
    -1     0     1
    -2     0     2
    -1     0     1
```

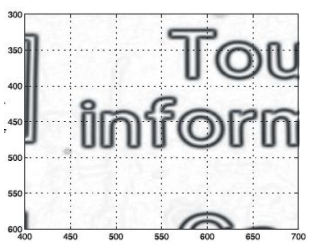
Magnitude & direction

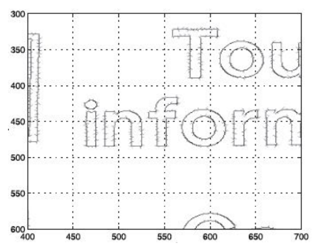


|DoG| vs. Canny

- Canny: smart post-processing of edge

- Derivative of Gaussian operator, then take magnitude





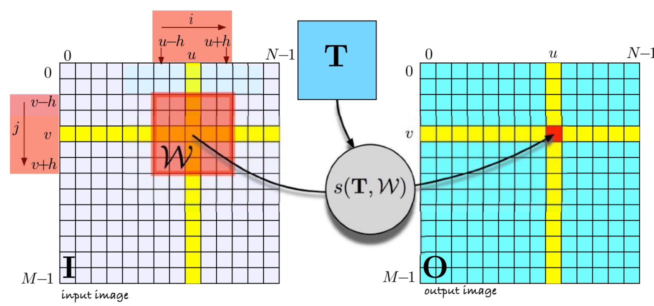
$$\nabla \mathbf{I} = \mathbf{D} \otimes (\mathbf{G}(\sigma) \otimes I) = \underbrace{(\mathbf{D} \otimes \mathbf{G}(\sigma))}_{\text{DoG}} \otimes \mathbf{I}$$

$$\mathbf{G}_u(u,v) = -\frac{u}{2\pi\sigma^2} e^{-\frac{u^2+v^2}{2\sigma^2}}$$

```
>> Iu = iconv( castle, -kdgauss(2) );
>> Iv = iconv( castle, -kdgauss(2)' );
>> m = sqrt( Iu.^2 + Iv.^2 );
```

Template matching

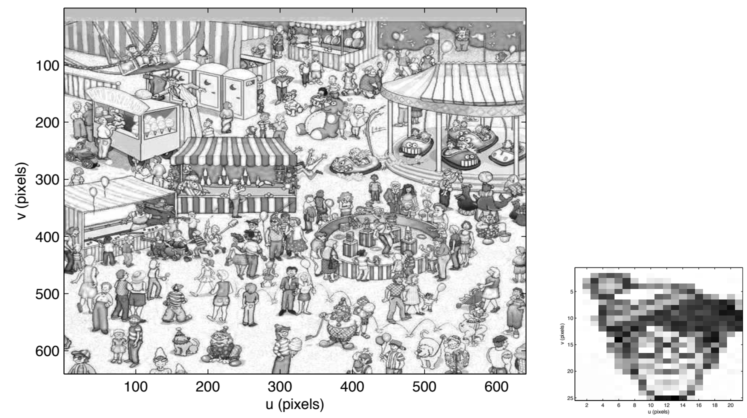
- In some cases it is entirely possible to match signals to templates
- The template could be sub-images, or processed versions of an arbitrary signal



Typical performance metrics

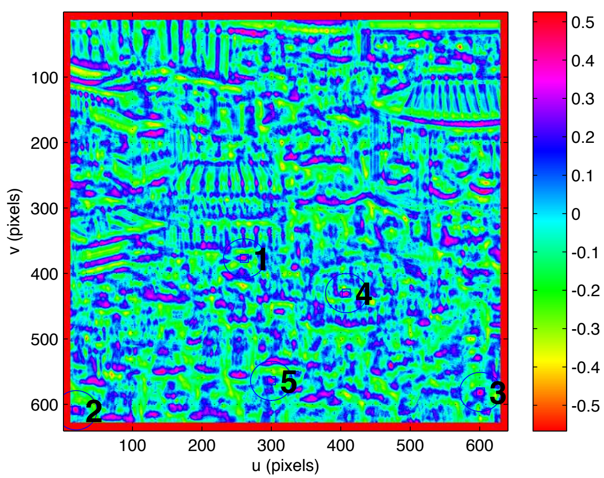
Sum of absolute differences		
SAD	$s = \sum_{(u,v)\in \mathbf{I}} \left\| \mathbf{I}_1[u,v] - \mathbf{I}_2[u,v] \right\|$	`sad`
ZSAD	$s = \sum_{(u,v)\in \mathbf{I}} \left\| (\mathbf{I}_1[u,v] - \bar{\mathbf{I}}_1) - (\mathbf{I}_2[u,v] - \bar{\mathbf{I}}_2) \right\|$	`zsad`
Sum of squared differences		
SSD	$s = \sum_{(u,v)\in \mathbf{I}} \left(\mathbf{I}_1[u,v] - \mathbf{I}_2[u,v] \right)^2$	`ssd`
ZSSD	$s = \sum_{(u,v)\in \mathbf{I}} \left((\mathbf{I}_1[u,v] - \bar{\mathbf{I}}_1) - (\mathbf{I}_2[u,v] - \bar{\mathbf{I}}_2) \right)^2$	`zssd`
Cross correlation		
NCC	$s = \frac{\sum_{(u,v)\in \mathbf{I}} \mathbf{I}_1[u,v] \cdot \mathbf{I}_2[u,v]}{\sqrt{\sum_{(u,v)\in \mathbf{I}} \mathbf{I}_1^2[u,v] \cdot \sum_{(u,v)\in \mathbf{I}} \mathbf{I}_2^2[u,v]}}$	`ncc`
ZNCC	$s = \frac{\sum_{(u,v)\in \mathbf{I}} (\mathbf{I}_1[u,v] - \bar{\mathbf{I}}_1) \cdot (\mathbf{I}_2[u,v] - \bar{\mathbf{I}}_2)}{\sqrt{\sum_{(u,v)\in \mathbf{I}} (\mathbf{I}_1[u,v] - \bar{\mathbf{I}}_1)^2 \cdot \sum_{(u,v)\in \mathbf{I}} (\mathbf{I}_2[u,v] - \bar{\mathbf{I}}_2)^2}}$	`zncc`

Where is waldo?

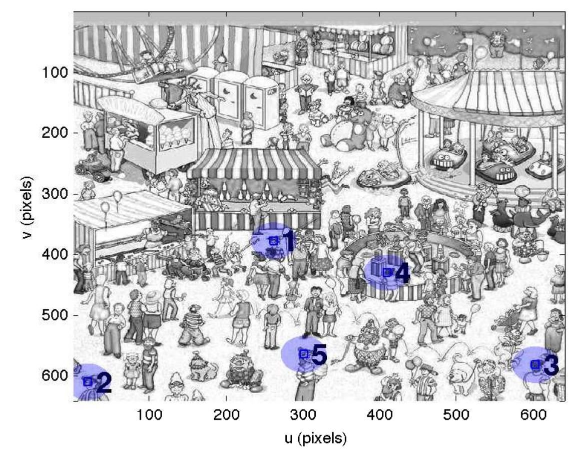


(a) (b)

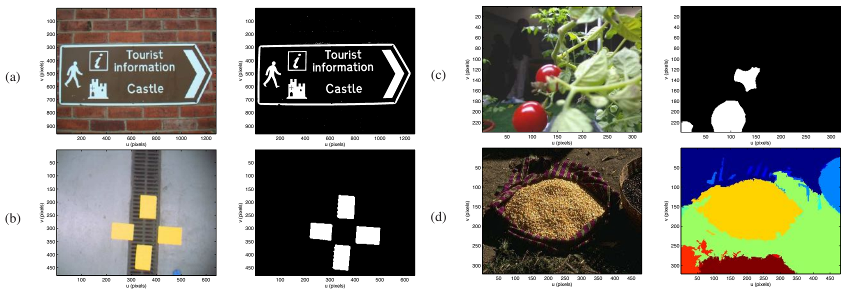
Where is Waldo?



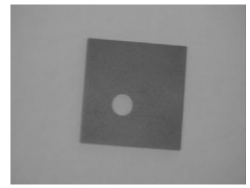
Where is Waldo?

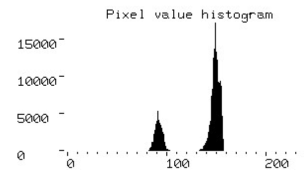


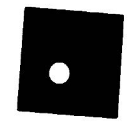
Feature Extraction



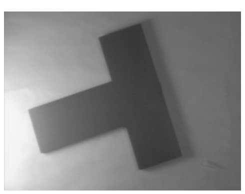
Simple object detection

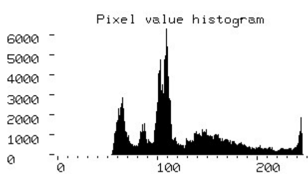


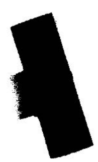




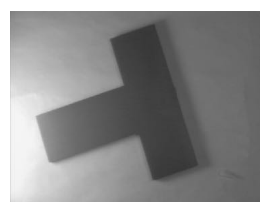
If only life was that simple

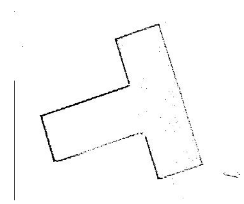


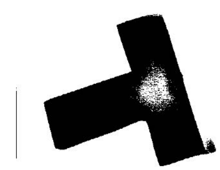




Adaptive thresholding


Original


7x7 mask


140x140 mask

Feature extraction

- Broad set of possible features depending on sensor modality
  - Point Estimation
  - Line Estimation (mathematical vs finite lines)
  - Place Estimation
  - Geometric features (#holes, shape descriptors)
  - Statistical Features (typical moments, central moments, ...)
  - Basic geometry

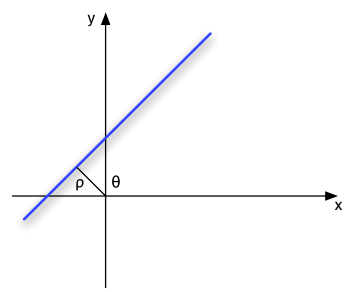
Line Estimation

- Lines are a predominant feature in engineered environments
- There is an abundance of methods for line estimation
- LSQ, Split-Merge, Hough, EM-estimation,
- RANSAC is frequently used (Fischler & Bolles, 1981)

Voting based methods

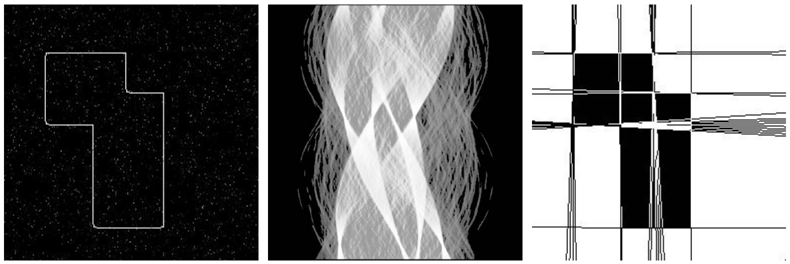
- Voting provides a simple estimator for detection
- Voting requires:
  1. A Voting Space
  2. A voting function (structure function)
  3. A decision function (often local extrema)
- Hough (1962) is one of the most widely used. Can also be used for lines and other shapes (Ballard, 1981)

Hough based estimator

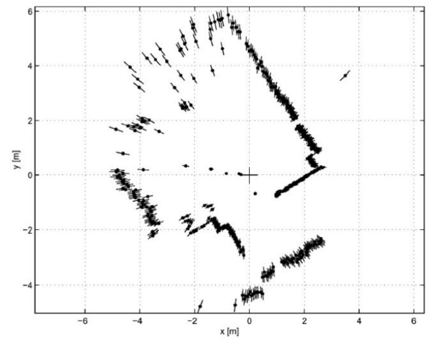


- Line model:
  $\rho = x * \cos(\theta) + y * \sin(\theta)$
- Voting space: $[\theta, \rho]$
- Voter: traverse θ space
- Local maximum w. NMS
- for all points in (x,y)
  for each $\theta : 0 \rightarrow \pi$
  calc ρ and increment (θ, ρ)
- Generates infinite lines.

Basic Hough Example

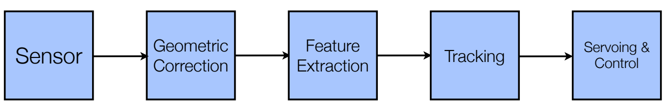


Hough on polar / range - bearing data



- Scanning is in polar coordinates.
- The density of points is varying.
- Close structure will accumulate more points.
- Range weighting can compensate. Proposed by Forsberg *et al.* (1993). Weight by $\frac{1}{\cos(\psi_i - \theta)}$

The approach for visual navigation



- Need to detect robust features for objects (we will discuss more next two sessions)
- Tracking of features over time to "keep" features in view
- Control vehicle to achieve the task objective

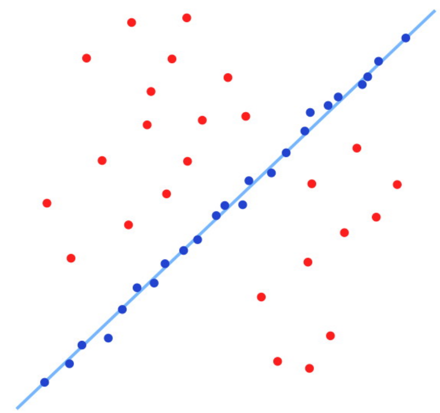
RANSAC - Random Sampling Consensus

- Estimation of parameters from N data items
- There are M data point in total
- How do we find the best parameters when there are many outliers?

RANSAC - Algorithm

1. selects N data items at random
2. estimates parameter $\vec{x}$
3. finds how many data items (of M) fit the model with parameter vector $\vec{x}$ within a user given tolerance. Call this K.
4. if K is big enough, accept fit and exit with success.
5. repeat 1..4 L times
6. fail if you get here

RANSAC Example Result



LSQ line fitting

- Least square minimization:
  - Line equation: $y = ax + b$
  - Error in fit: $\sum_i (y_i - ax_i - b)^2$
  - Solution:

$$\begin{pmatrix} \overline{y^2} \\ \bar{y} \end{pmatrix} = \begin{pmatrix} \overline{x^2} & \bar{x} \\ \bar{x} & 1 \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix}$$

  - Minimizes vertical errors. Non-robust!

TLS line fitting

- Line equation: $ax + by + c = 0$
- Error in fit: $\sum_i (ax_i + by_i + c)^2$ where $a^2 + b^2 = 1$.
- Solution:

$$\begin{pmatrix} \overline{x^2} - \bar{x}\bar{x} & \overline{xy} - \bar{x}\bar{y} \\ \overline{xy} - \bar{x}\bar{y} & \overline{y^2} - \bar{y}\bar{y} \end{pmatrix} \begin{pmatrix} a \\ b \end{pmatrix} = \mu \begin{pmatrix} a \\ b \end{pmatrix}$$

  where μ is a scale factor.
- $c = -a\bar{x} - b\bar{y}$

Summary

- Starting to think about images as a primary modality for feedback
- The main sensor for CSE276A homework
- There are much more to image processing than we can cover. The book (Corke, 2023) covers much more material
- Most of the processing covered by the OpenCV library - https://opencv.org/