Machine Learning

Lesson 06. Intro to OpenCV

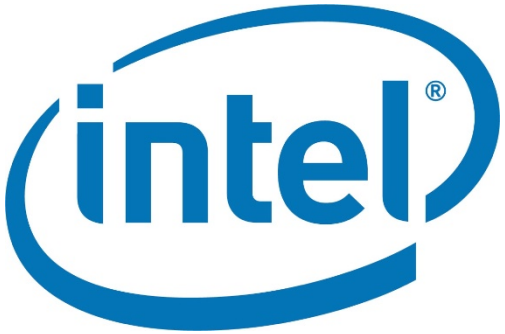
Kirill Svyatov

Ulyanovsk State Technical University,

Faculty of Information Systems and Technologies

Introduction

- OpenCV is an Image Processing library created by Intel and maintained by Willow Garage.
- Available for C, C++, and Python
- Newest update is version 3.2
- Open Source and free
- Easy to use and install



OpenCV Functionality

(More than 350 algorithms)

- Basic structures and operations
- Image Analysis
- Structural Analysis
- Object Recognition
- Motion Analysis and Object Tracking
- 3D Reconstruction

Basic structures and operations

Image TYPES

- **The TYPE is a very important aspect of OpenCV**
- Represented as CV_<Datatype> <Number of Channels>
- Example Datatypes/ Depths

OpenCV Tag	Representation	OpenCV Value
CV_8U	8 bit unsigned integer	0
CV_8S	8 bit signed integer	1
CV_16U	16 bit unsigned integer	2
CV_16S	16 bit signed integer	3
CV_32S	32 bit signed integer	4
CV_32F	32 bit floating point number	5
CV_64F	64 bit floating point number	6

CV_64F contain more information but hold more RAM!
In ML often have to use CV_8U format

Basic Structures and Operations

- Multidimensional array operations
- Dynamic structures operations
- Drawing primitives
- Utility functions

Victor Eruhimov:

Multidimensional array operations include operations on images, matrices and histograms. In the future, when I talk about image operations, keep in mind that all operations are applicable to matrices and histograms as well. Dynamic structures operations concern all vector data storages. They will be discussed in detail in the Technical Section. Drawing primitives allows not only to draw primitives but to use the algorithms for pixel access. Utility functions, in particular, contain fast implementations of useful math functions.

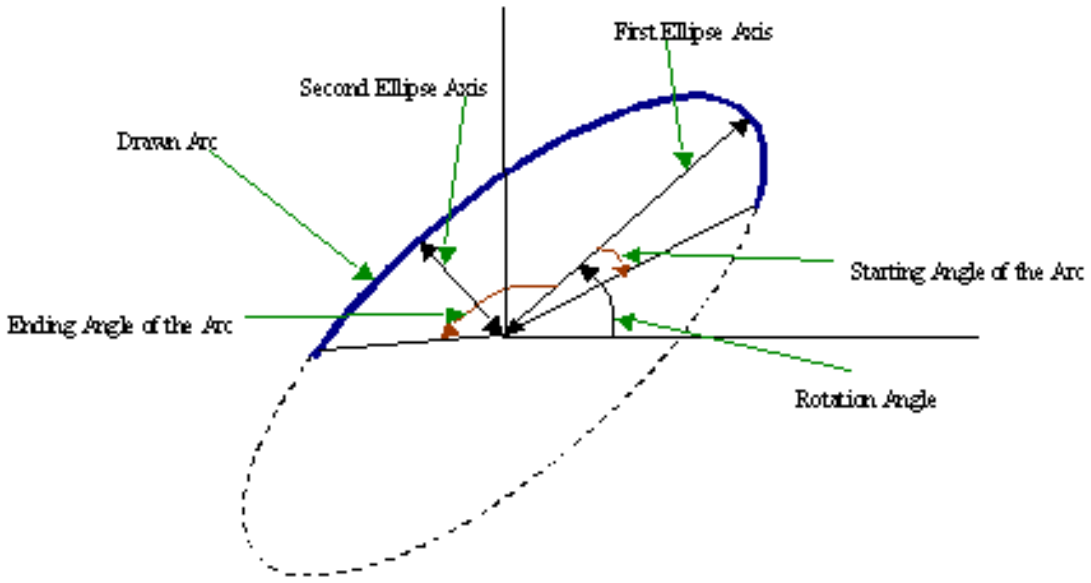
Converting colorspaces

Function cvtColor(image, image, code)

- Codes:
  - **CV_BGR2GRAY**
  - **CV_BGR2HSV**
  - **CV_BGR2LUV**

Drawing ellipse example

Draws a simple or thick elliptic arc or fills an ellipse sector.



Parameters:

- **img** – Image.
- **center** – Center of the ellipse.
- **axes** – Half of the size of the ellipse main axes.
- **angle** – Ellipse rotation angle in degrees.
- **startAngle** – Starting angle of the elliptic arc in degrees.
- **endAngle** – Ending angle of the elliptic arc in degrees.
- **color** – Ellipse color.
- **thickness** – Thickness of the ellipse arc outline, if positive. Otherwise, this indicates that a filled ellipse sector is to be drawn.
- **etc…**

Transformations: Translation

Translation is the shifting of object's location:

```
img = cv2.imread('messi5.jpg',0)
rows,cols = img.shape
M = np.float32([[1,0,100],[0,1,50]])
dst = cv2.warpAffine(img,M,(cols,rows))
```

Remember width = number of columns, and height = number of rows

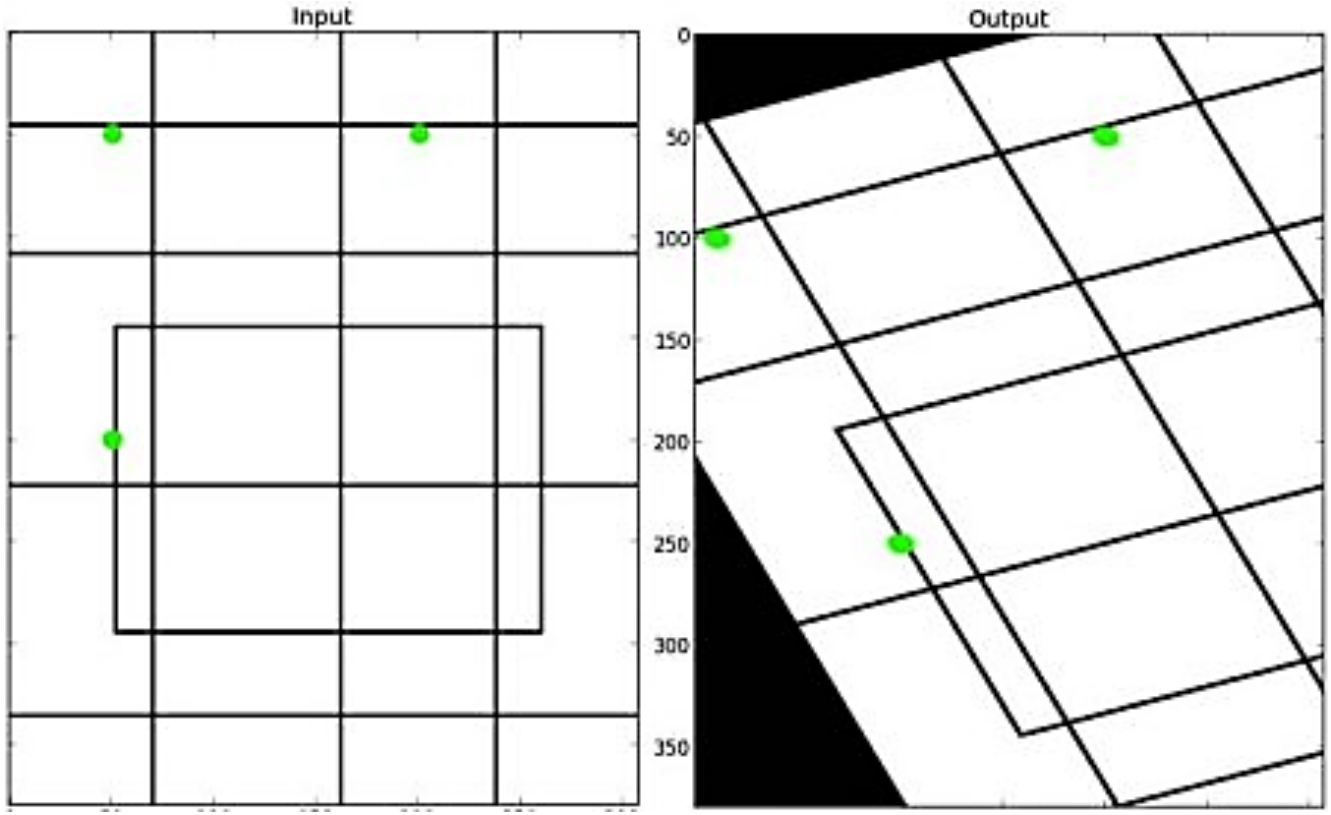
Transformations: Rotation

Rotation of an image for an angle θ is achieved by the transformation matrix of the form
To find this transformation matrix, OpenCV provides a function, **cv2.getRotationMatrix2D** :

```
img = cv2.imread('messi5.jpg',0)
rows,cols = img.shape
M = cv2.getRotationMatrix2D((cols/2,rows/2),90,1)
dst = cv2.warpAffine(img,M,(cols,rows))
```

Transformations: Affine Transformation

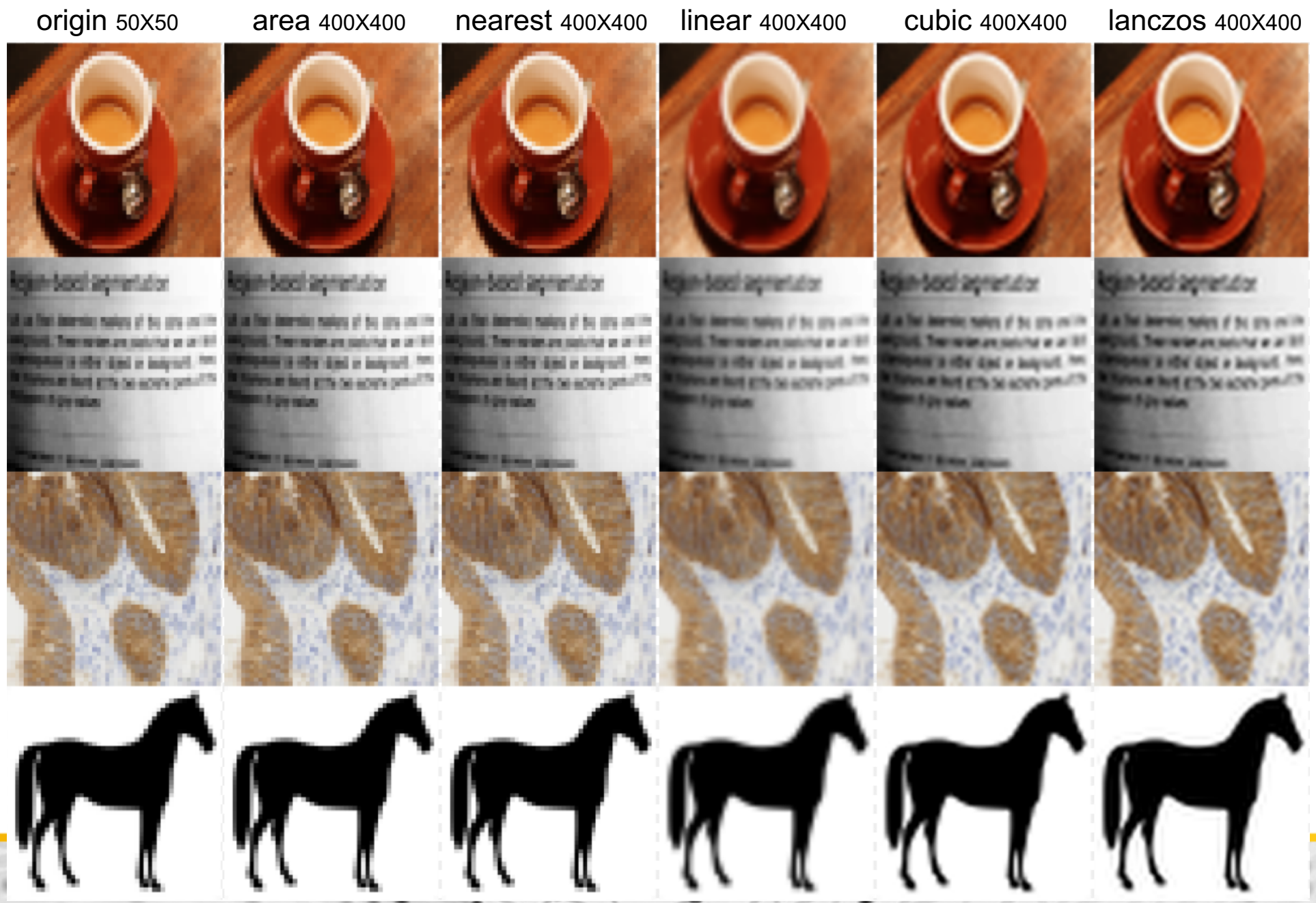
In affine transformation, all parallel lines in the original image will still be parallel in the output image. To find the transformation matrix, we need three points from input image and their corresponding locations in output image:



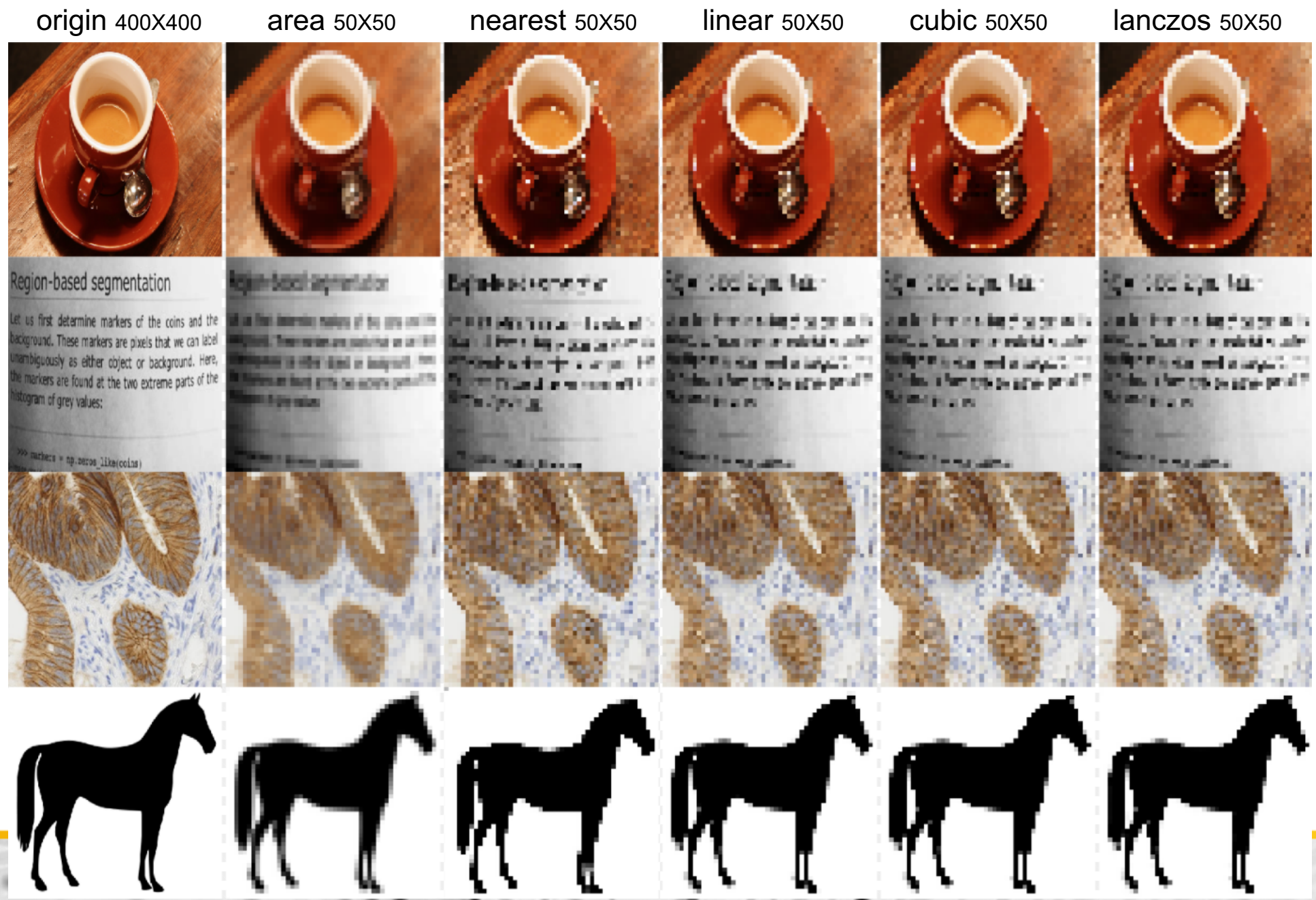
Transformations: Affine Transformation code

```
img = cv2.imread('drawing.png')
rows,cols,ch = img.shape

pts1 = np.float32([[50,50],[200,50],[50,200]])
pts2 = np.float32([[10,100],[200,50],[100,250]])
M = cv2.getAffineTransform(pts1,pts2)
dst = cv2.warpAffine(img,M,(cols,rows))

plt.subplot(121),plt.imshow(img),plt.title('Input')
plt.subplot(122),plt.imshow(dst),plt.title('Output')
plt.show()
```

Transformations: Upsampling resize



Transformations: Downsampling resize



Transformations: resize in code

Very simple:

```
W = target_width
H = , target_height

result_image = cv2.resize(source_im, (W, H), interpolation=cv2.INTER_AREA)
```

Image Analysis

Image Analysis

- Thresholds
- Statistics
- Pyramids
- Morphology
- Distance transform
- Flood fill
- Feature detection
- Contours retrieving

Image Thresholding

- Fixed threshold;
- Adaptive threshold;

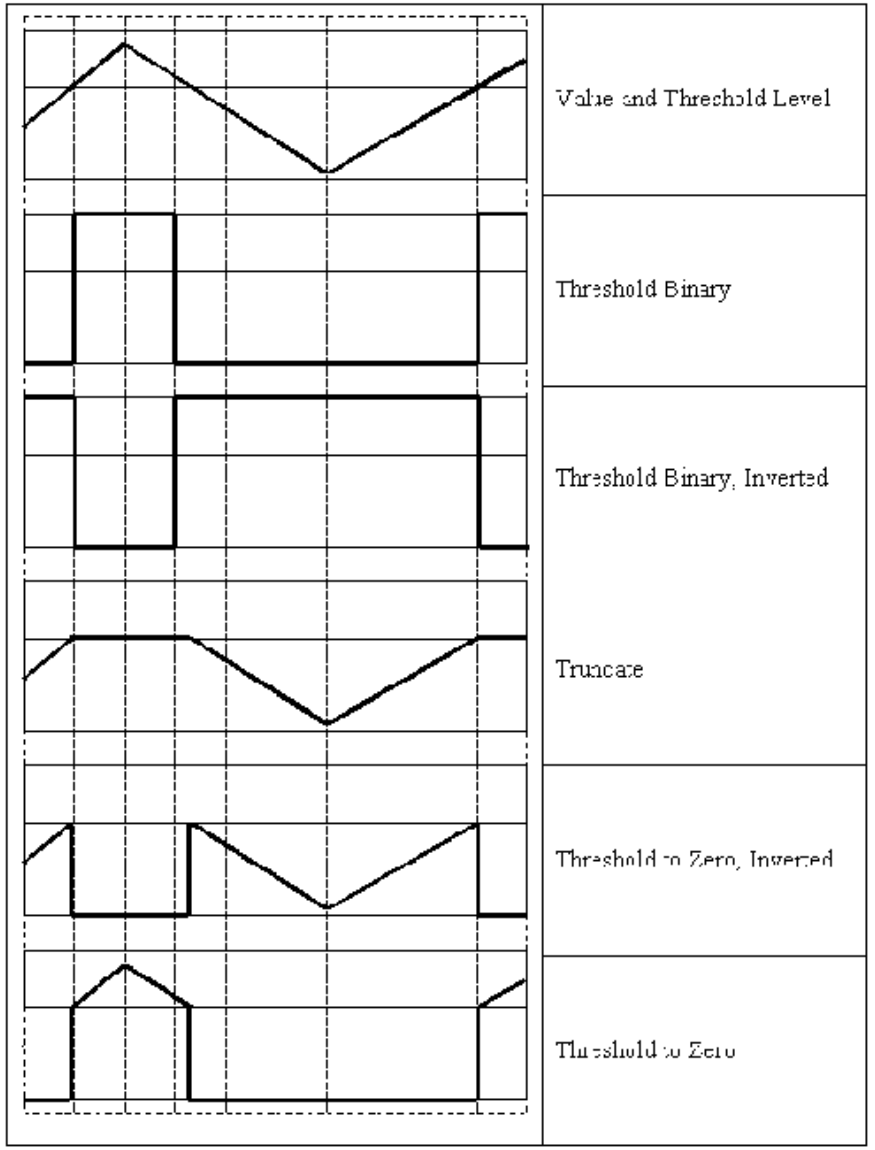


Image Thresholding Examples

Original Image

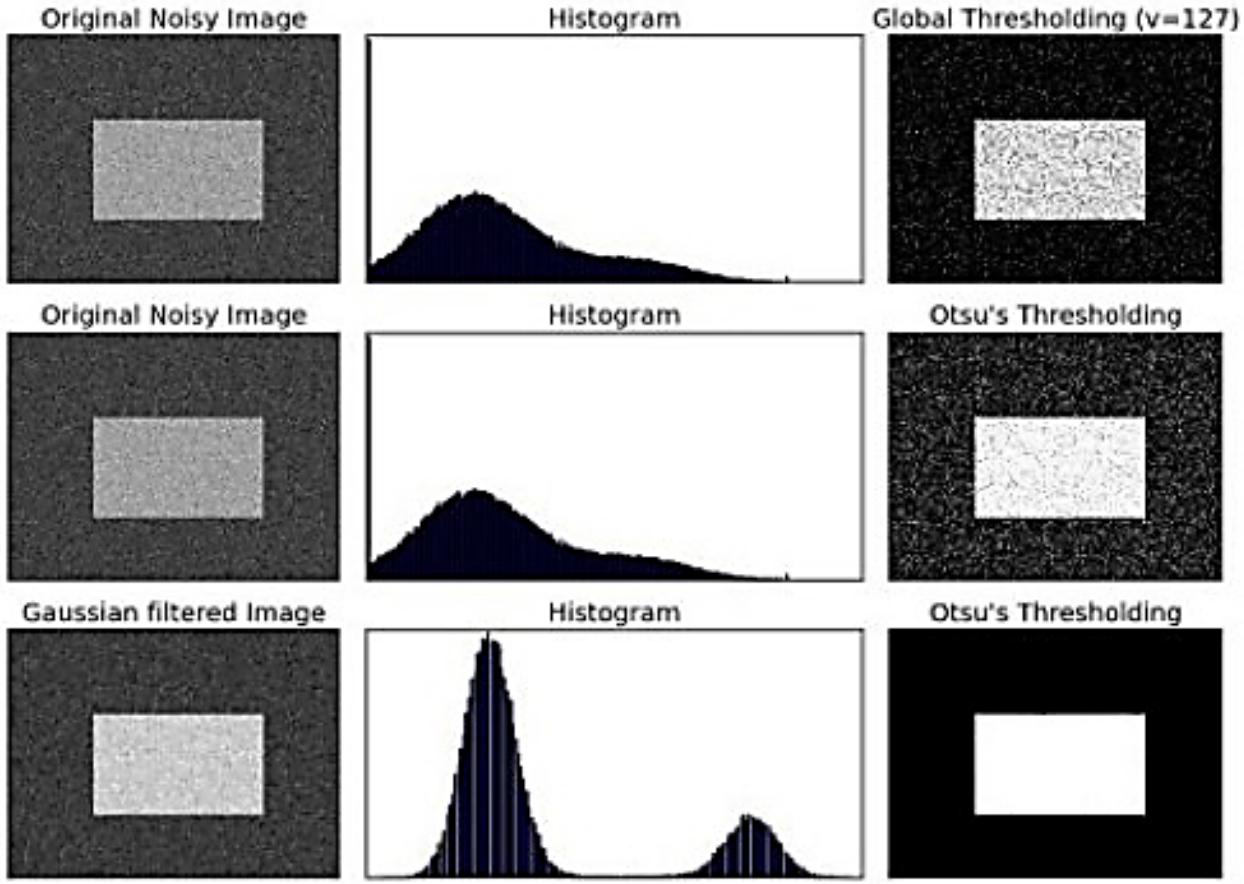
Global Thresholding (v = 127)

Adaptive Mean Thresholding

Adaptive Gaussian Thresholding

Image Thresholding for denoising Examples

(Use Otsu's Binarization)



Statistics

- min, max, mean value, standard deviation over the image
- Norms C, L1, L2
- Multidimensional histograms
- Spatial moments up to order 3 (central, normalized, Hu)

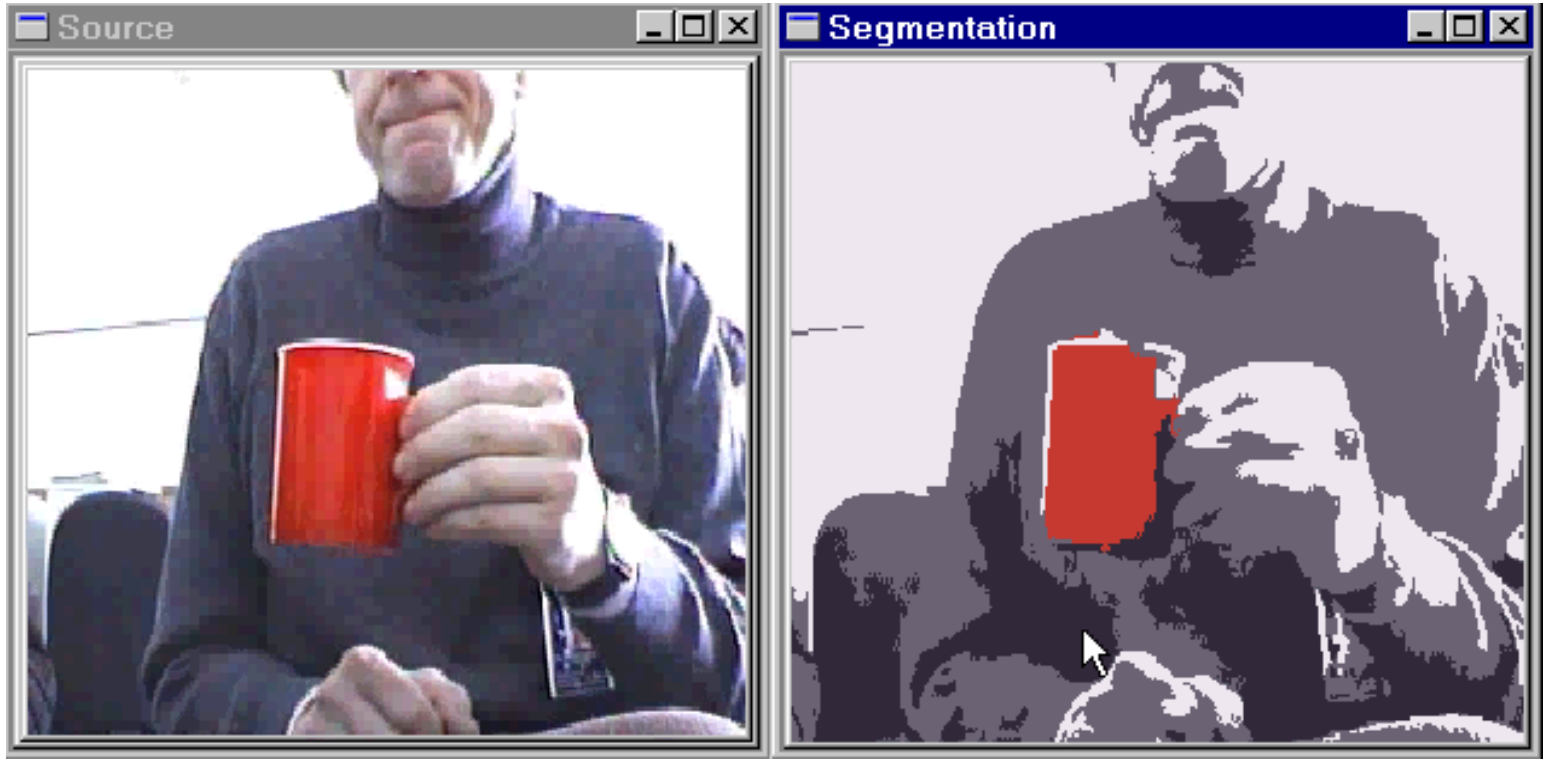
Victor Eruhimov:

In addition to simple norm calculation, there is a function that finds the norm of the difference between two images.

Pyramid-based color segmentation

On still pictures

And on movies

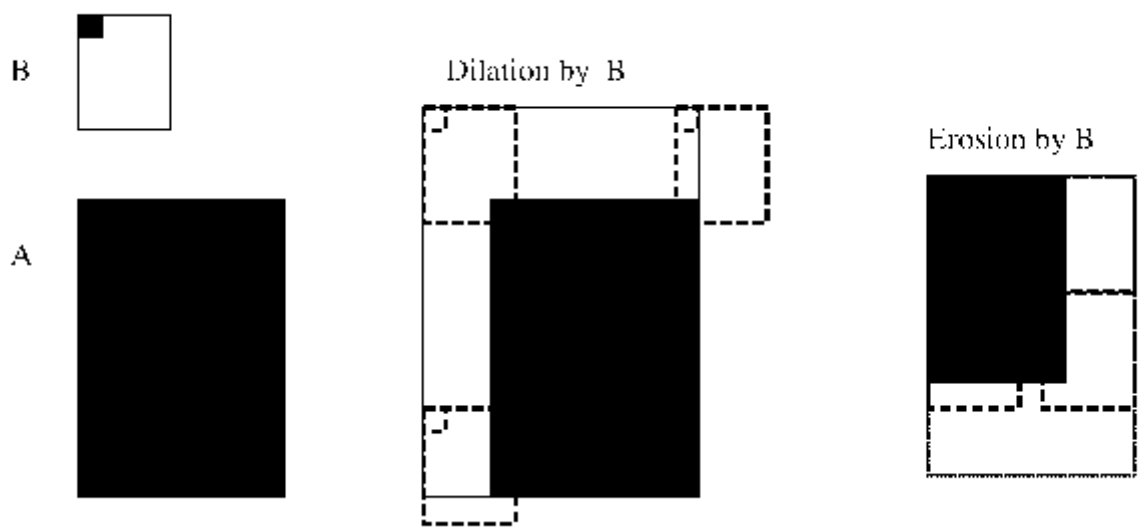


Morphological Operations

- Two basic morphology operations using structuring element:
  - erosion
  - dilation
- More complex morphology operations:
  - opening
  - closing
  - morphological gradient
  - top hat
  - black hat

What is Structuring Element?

- The rectangle mask with given position of a principal pixel:



Morphological Operations Examples

Morphology - applying Min-Max. Filters and its combinations:

Image I

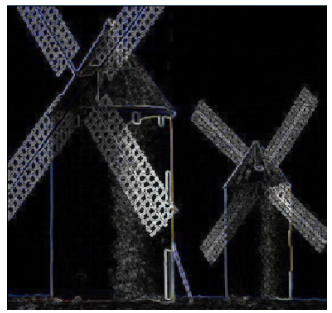
Erosion IΘB

Dilatation I⊕B

Opening IoB= (IΘB)⊕B

Closing I•B= (I⊕B)ΘB

Grad(I)= (I⊕B)-(IΘB)



TopHat(I)= I - (IΘB)

BlackHat(I)= (I⊕B) - I

Distance Transform

- Calculate the distance for all non-feature points to the closest feature point
- Two-pass algorithm, 3x3 and 5x5 masks, various metrics predefined

Flood Filling

- Simple
- Gradient

Original image

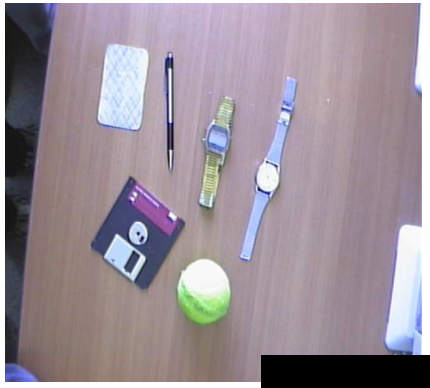
Tolerance interval ± 5

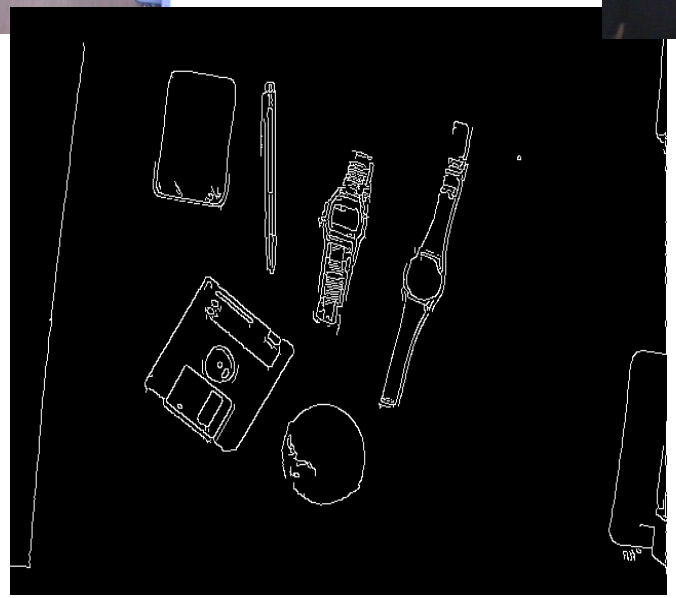
Tolerance interval ± 6

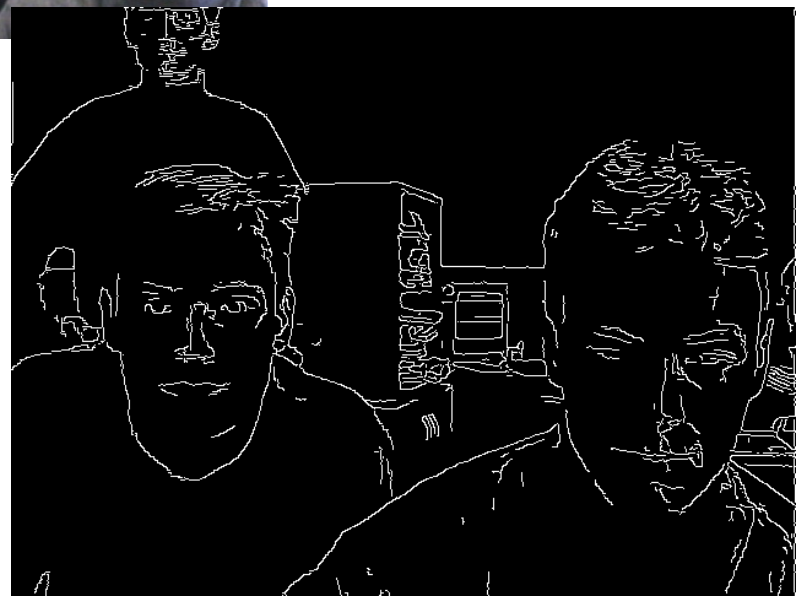
Feature Detection

- Fixed filters (Sobel operator, Laplacian)
- Optimal filter kernels with floating point coefficients (first, second derivatives, Laplacian)
- Special feature detection (corners)
- Canny operator
- Hough transform (find lines and line segments)
- Gradient runs

Canny Edge Detector





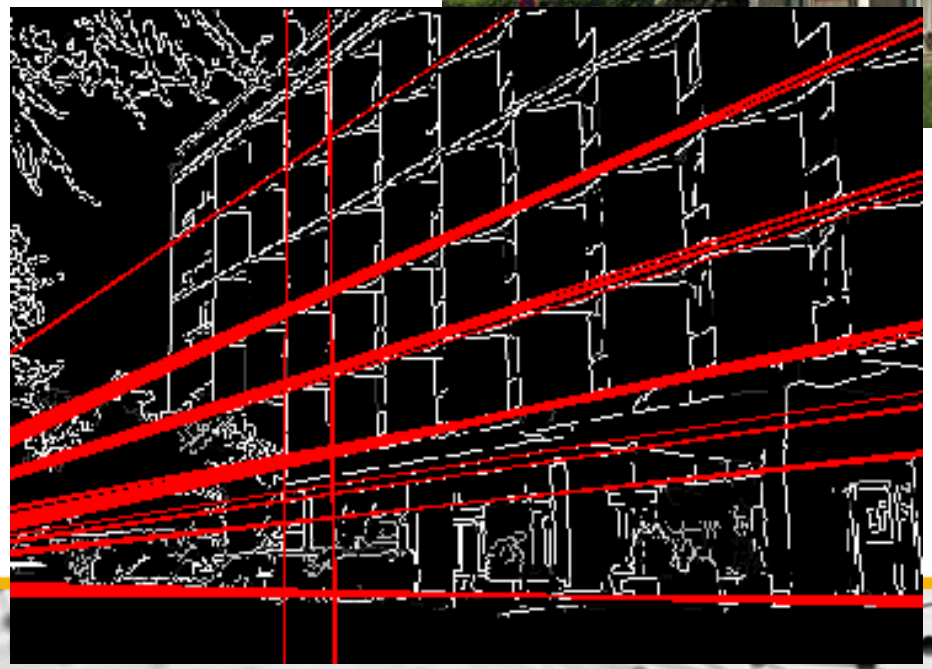


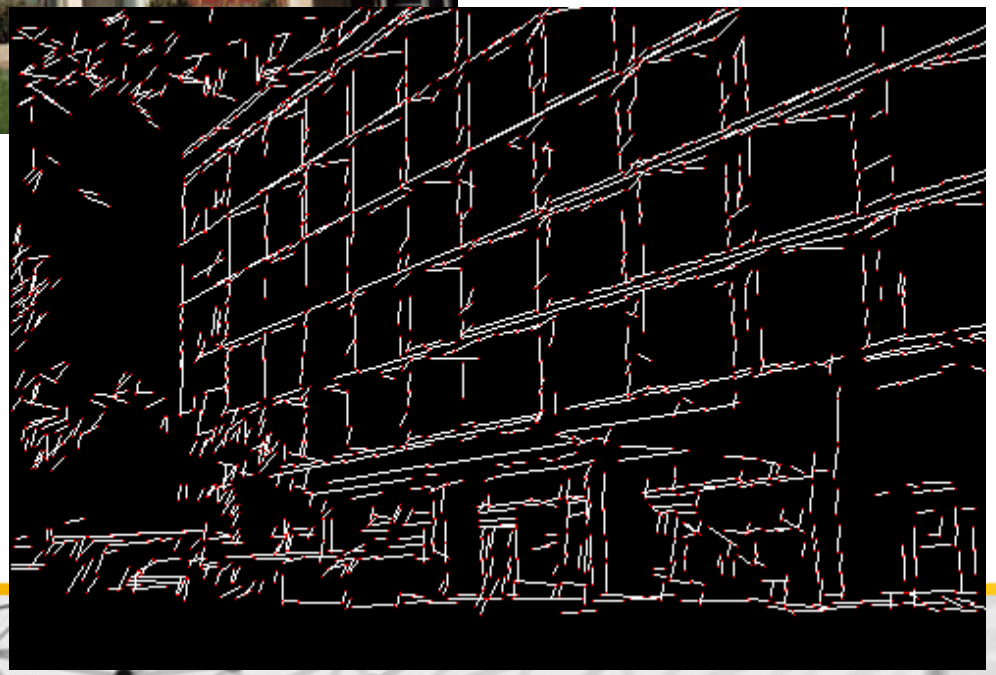
Hough Transform

Detects lines in a binary image

•Standard Hough Transform

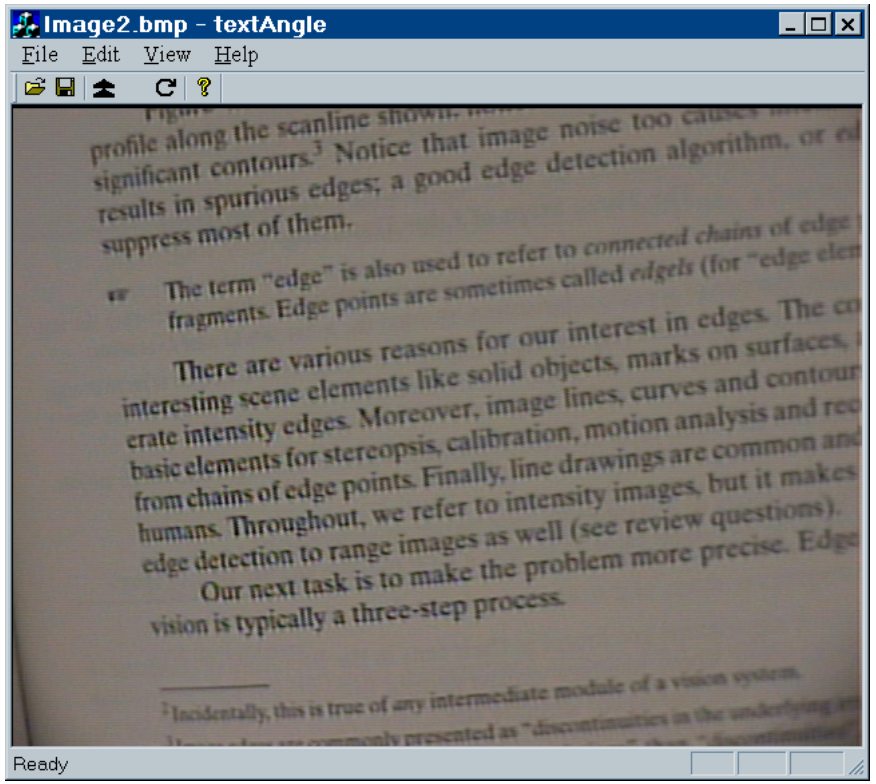
•Probabilistic Hough Transform



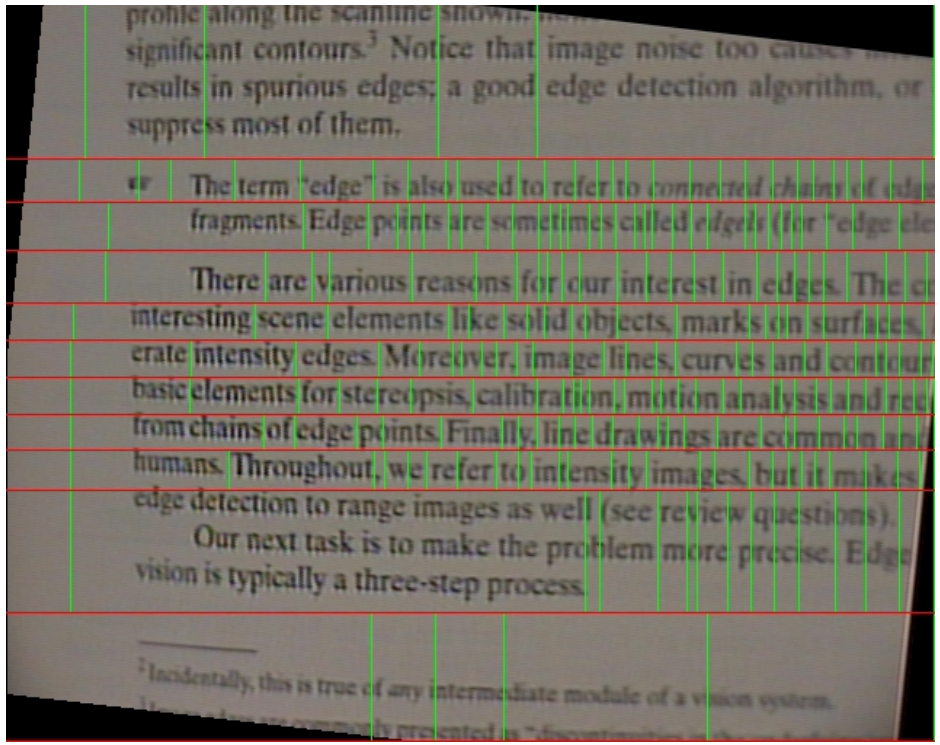


Another Sample of the Hough Transform Using

Source picture

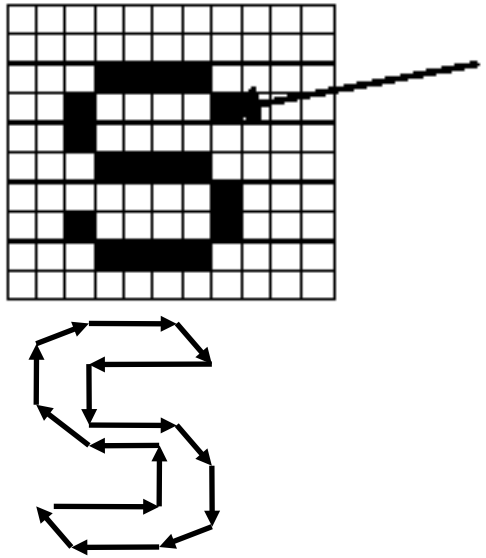


Result



Contour Retrieving

- The contour representation:
  - ✓Chain code (Freeman code)
  - ✓Polygonal representation

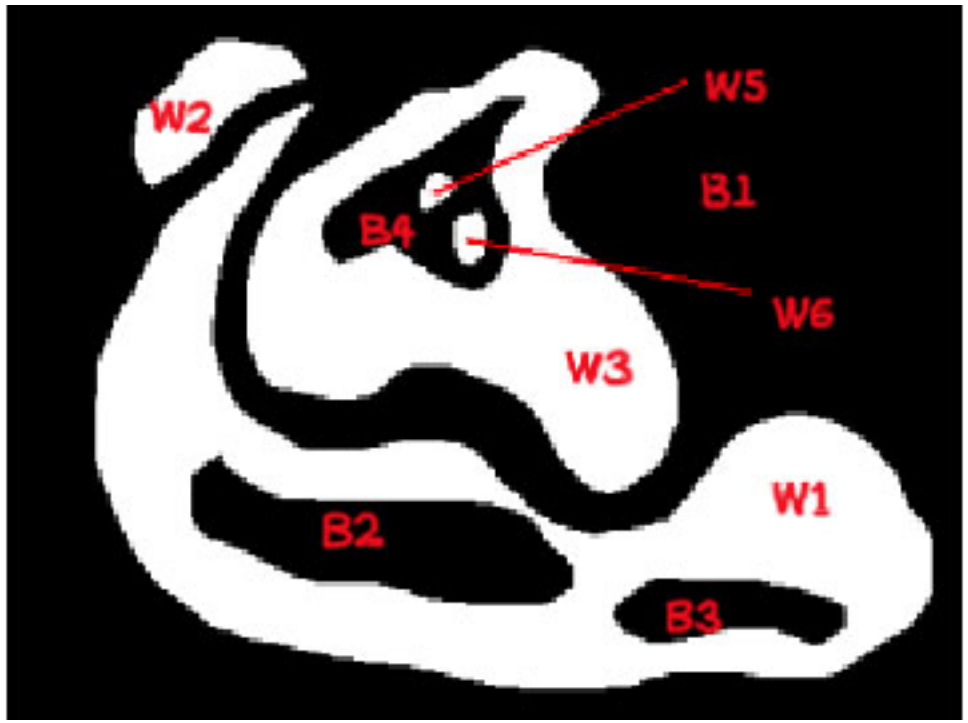


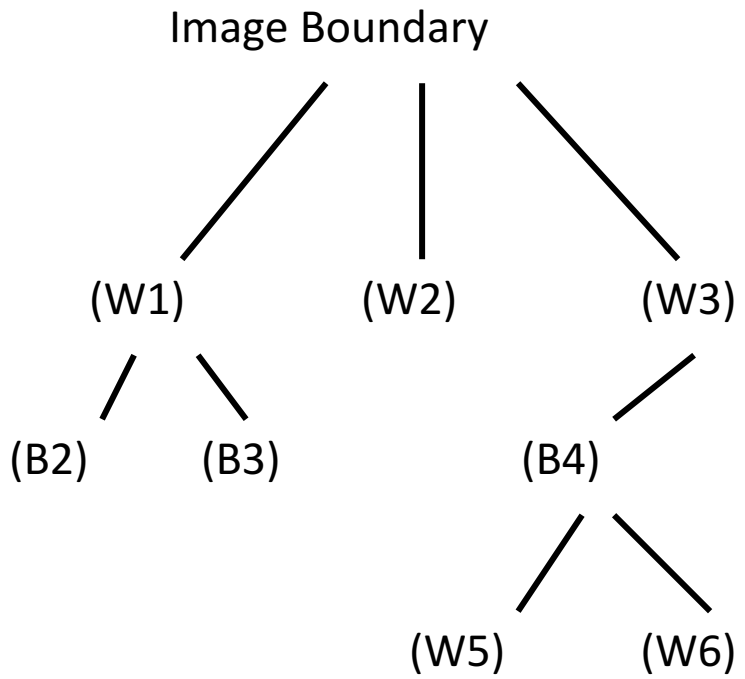
Initial Point

Chain code for the curve:
34445670007654443

Contour representation

Hierarchical representation of contours

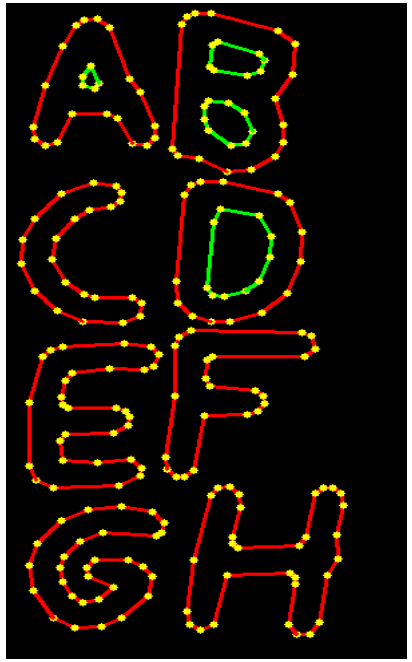




Contours Examples

Source Picture
(300x600 = 180000 pts total)

Retrieved Contours
(<1800 pts total)



After Approximation
(<180 pts total)

And it is rather fast: ~70 FPS for 640x480 on complex scenes

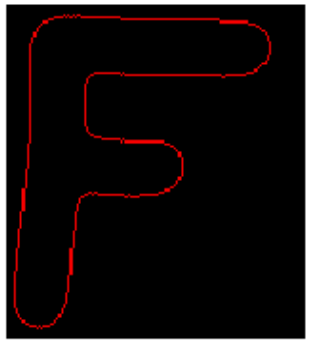
Structural Analysis

Structural Analysis

- Contours processing
  - Approximation
  - Hierarchical representation
  - Shape characteristics
  - Matching
- Geometry
  - Contour properties
  - Fitting with primitives
  - PGH: pair-wise geometrical histogram for the contour.

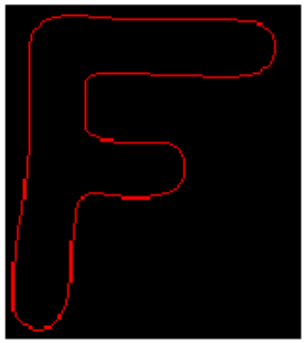
Contour Processing

- Approximation:
  - ✓RLE algorithm (chain code)
  - ✓Teh-Chin approximation (polygonal)
  - ✓Douglas-Peucker approximation (polygonal);
- Contour moments (central and normalized up to order 3)
- Hierarchical representation of contours
- Matching of contours



Source Image

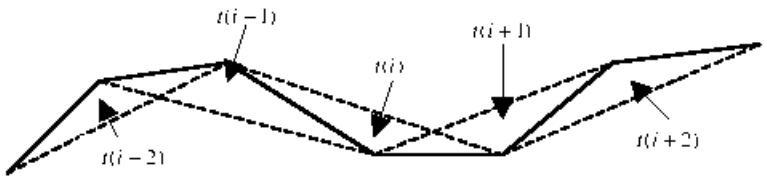
Rosenfeld-Johnston Algorithm Output

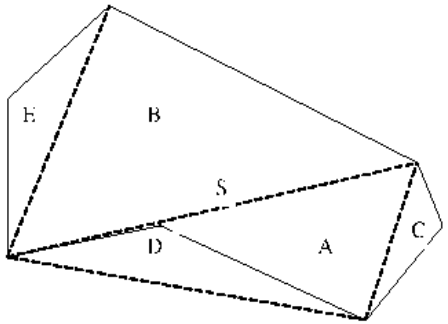


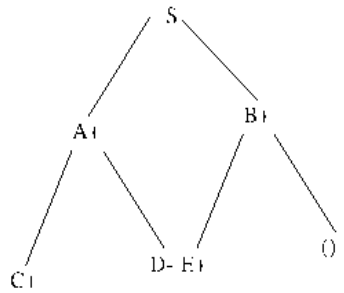
Teh-Chin Algorithm Output

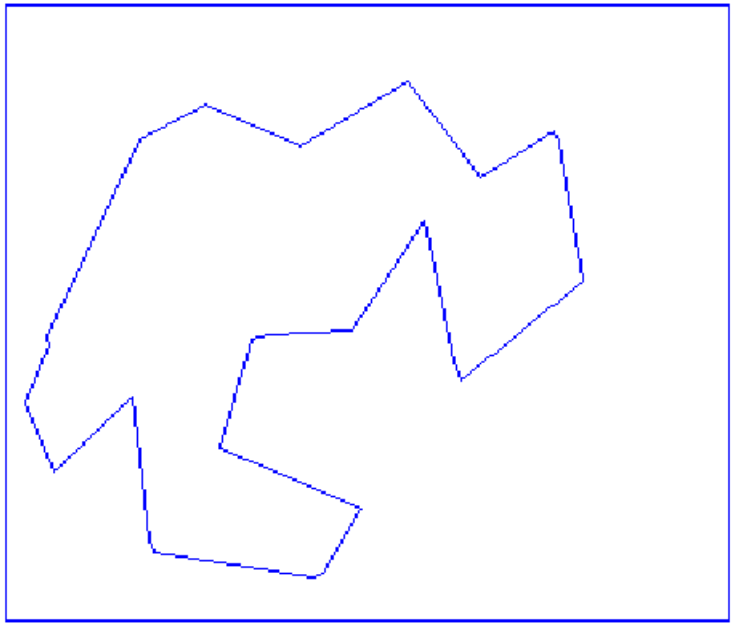
Hierarchical Representation of Contours

- A contour is represented with a binary tree
- Given the binary tree, the contour can be retrieved with arbitrary precision
- The binary tree is quasi invariant to translations, rotations and scaling





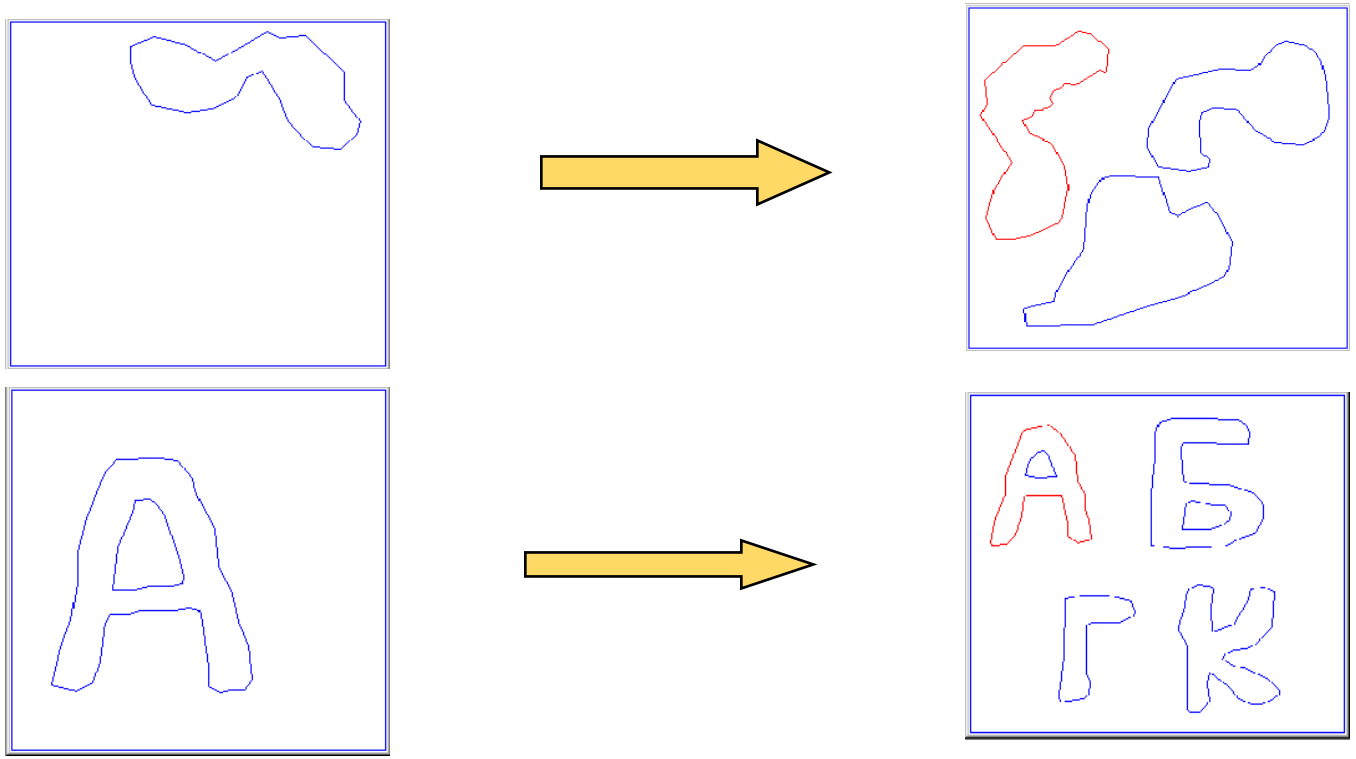




Contours matching

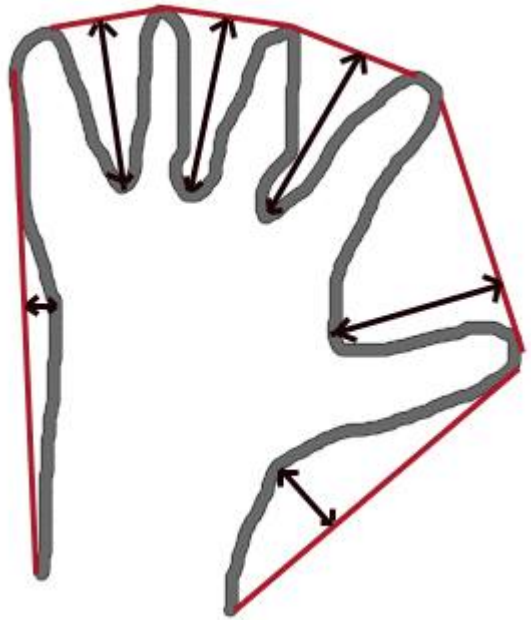
- Matching based on hierarchical representation of contours

Find all matching contours:



Geometry

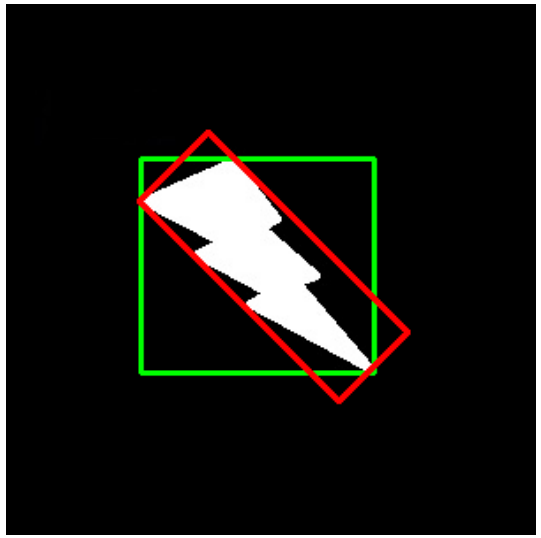
- Properties of contours: (perimeter, area, convex hull, convexity defects, rectangle of minimum area)
- Fitting: (2D line, 3D line, circle, ellipse)
- Pair-wise geometrical histogram

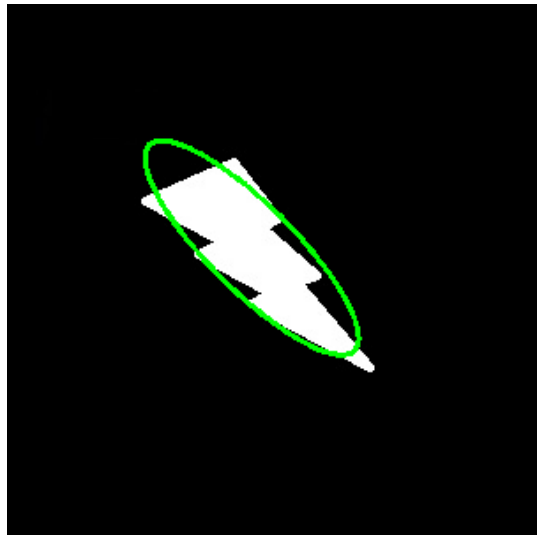


Geometry

Bounding Rectangle:

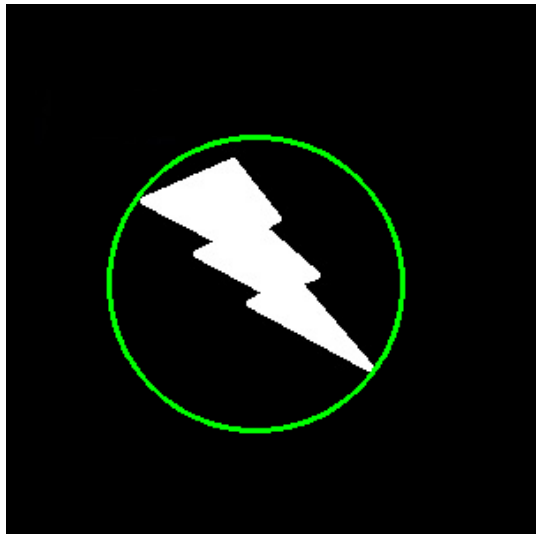
- Straight Bounding Rectangle
- Rotated Rectangle





Fitting an Ellipse

Minimum Enclosing Circle



Fitting a Line

Object Recognition

Object Recognition

- Eigen objects
- Hidden Markov Models

Eigen Objects

$$u^i = \{u_1^i, u_2^i, \ldots u_n^i\} \in R^n,\ 1 \le i \le m,\ m << n,$$

$$C = \{c_{ij}\} - \text{cov}\, ariance\ \ matrix,$$

$$c_{ij} = \sum_l (u_l^i - \bar{u}_l) \cdot (u_l^j - \bar{u}_l),\ \ \bar{u}_l = 1/m \sum_{k=1}^{m} u_l^k,$$

$$e_l^i = 1/\sqrt{\lambda_i} \sum_k v_k^i \cdot (u_l^k - \bar{u}_l),$$

$$e^i = \{e_1^i, e_1^i, \ldots, e_n^i\}, i = 1, \ldots m_1 \le m - eigen\ \ basis,$$

$$\lambda_i\ \ and\ \ v^i = \{v_1^i, v_2^i, \ldots, v_m^i\} - eigen\ values\ and\ eigen\ vectors.$$

Eigen Objects: result

Hidden Markov Model Definitions

$S = \{s_1, s_2, \ldots, s_N\}$ - The set of states

$O = \{o_1, o_2, \ldots, o_M\}$ - The set of measurements

$q_t \in O$ - The state at time *t*

$A = \{a_{ij} \setminus a_{ij} = P(q_t = s_j \mid q_{t-1} = s_i)\}$ - The transition probability matrix

$B = \{b_{ij} \setminus b_{ij} = P(o_i \mid s_j)\}$ - The conditional probability matrix

$\Pi = \{\pi_i \setminus \pi_i = P(q_0 = s_i)\}$ - The starting states distribution

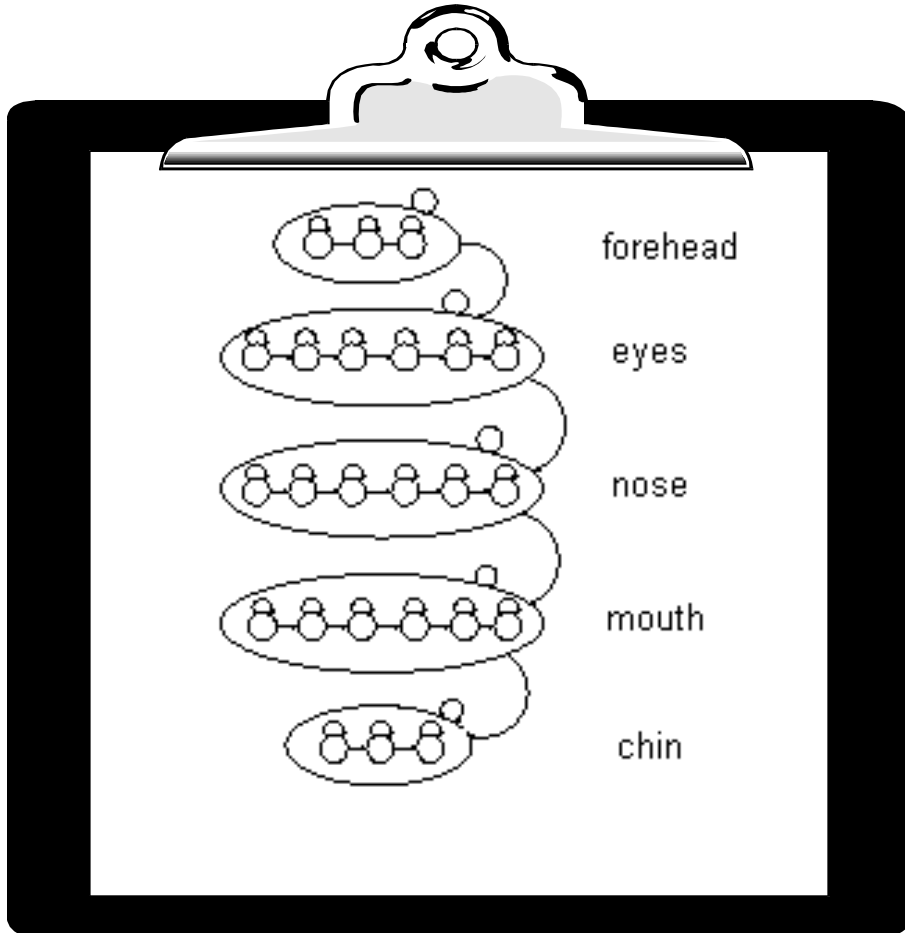
Hidden Markov Model Algorithms

- Model parameters estimation
- Recognition
- Viterbi algorithm

Embedded HMM for Face Recognition

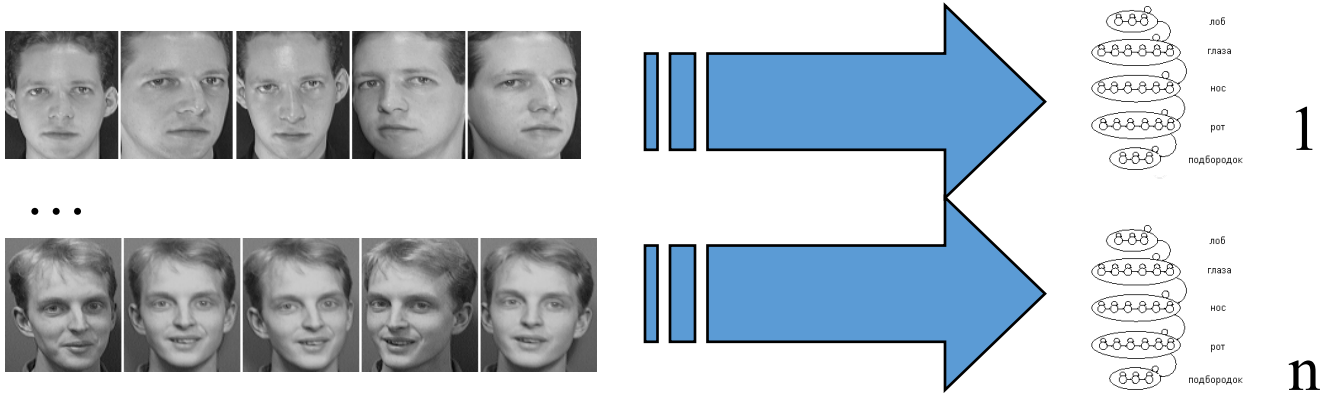
- Face ROI partition

Model-

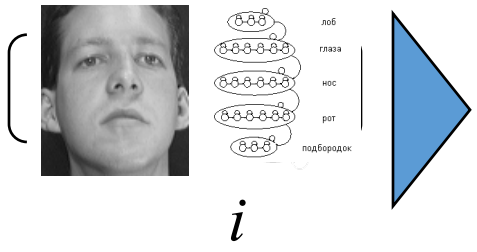


Face recognition using Hidden Markov Models

- One person – one HMM
- Stage 1 – Train every HMM



- Stage 2 – Recognition



P*i* - probability

Choose **max(P*i*)**

Motion Analysis and Object Tracking

Motion Analysis and Object Tracking

- Background subtraction
- Motion templates
- Optical flow
- Active contours
- Estimators

Background Subtraction

- Background model (normal distribution)
- Background statistics functions:
  - ✓Average
  - ✓Standard deviation
  - ✓Running average

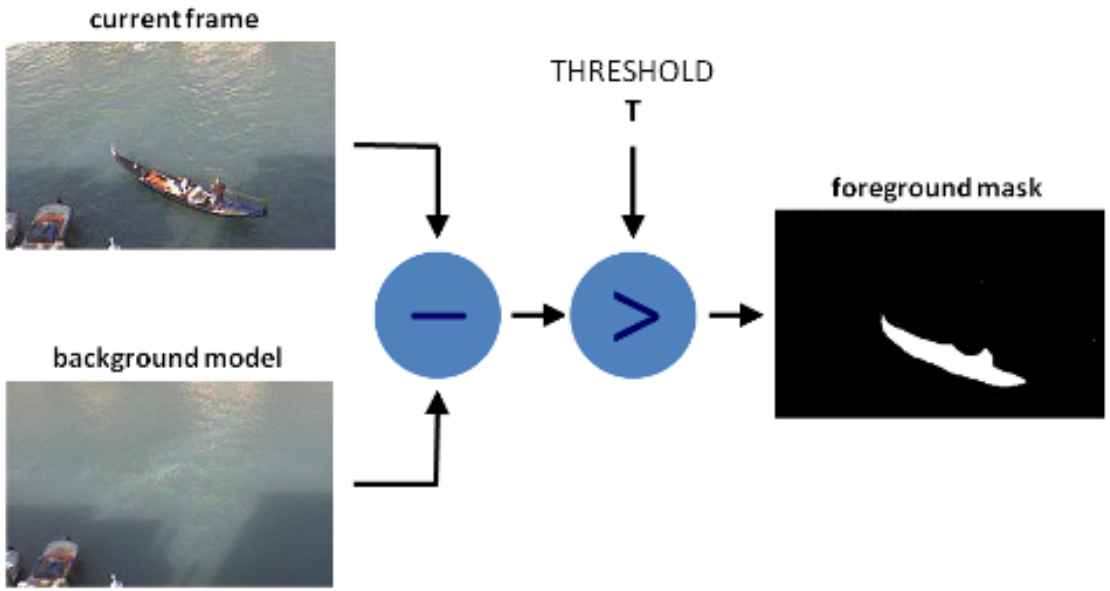
Running average

- Computes the sum of two images:

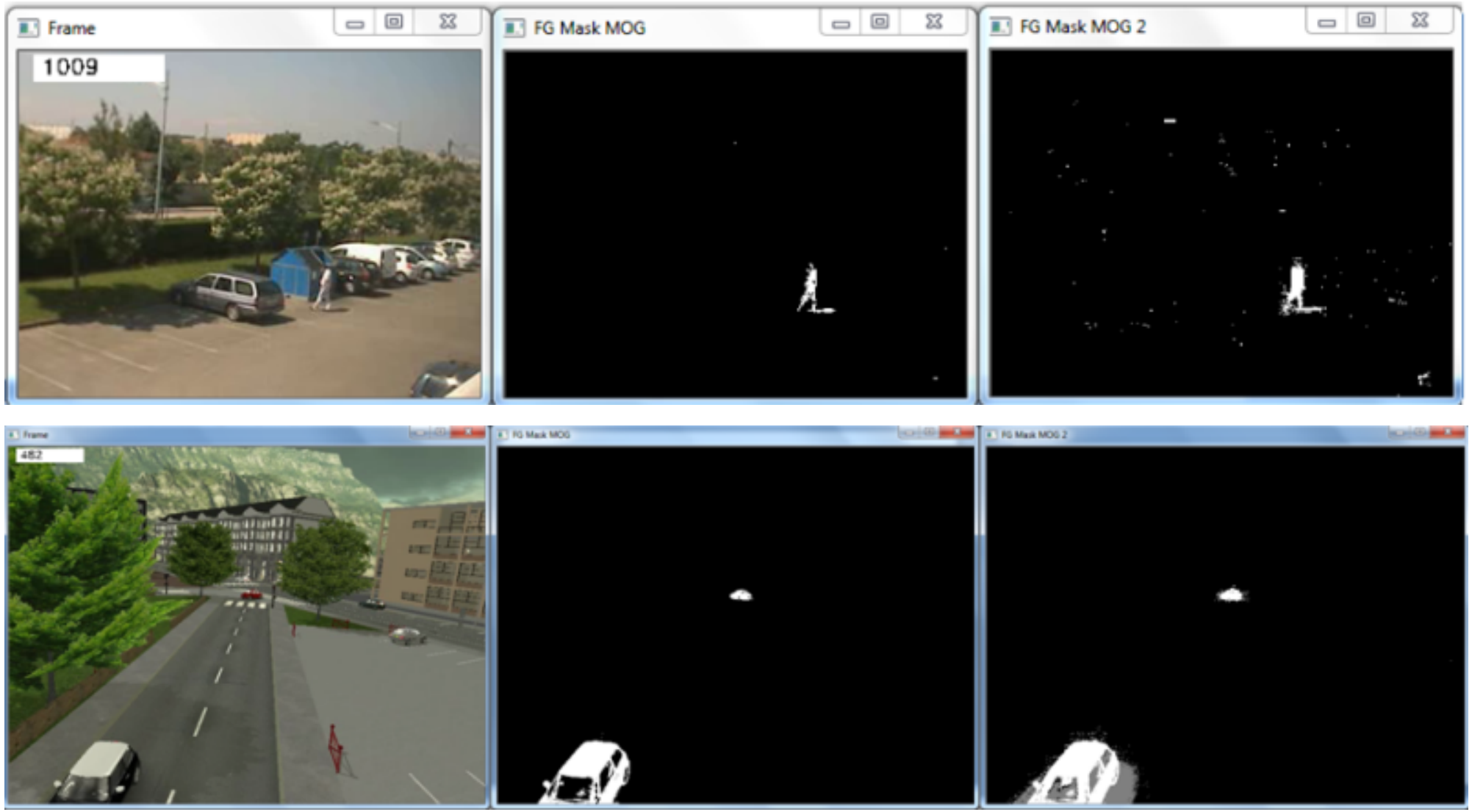
$$\mu_{ij}^{t} = \alpha \cdot I_{ij}^{t} + (1-\alpha) \cdot \mu_{ij}^{t-1},\ 0 \leq \alpha \leq 1$$

Background Subtraction Example

- Background subtraction (BS) is a common and widely used technique for generating a foreground mask (namely, a binary image containing the pixels belonging to moving objects in the scene) by using static cameras.
- As the name suggests, BS calculates the foreground mask performing a subtraction between the current frame and a background model, containing the static part of the scene or, more in general, everything that can be considered as background given the characteristics of the observed scene.



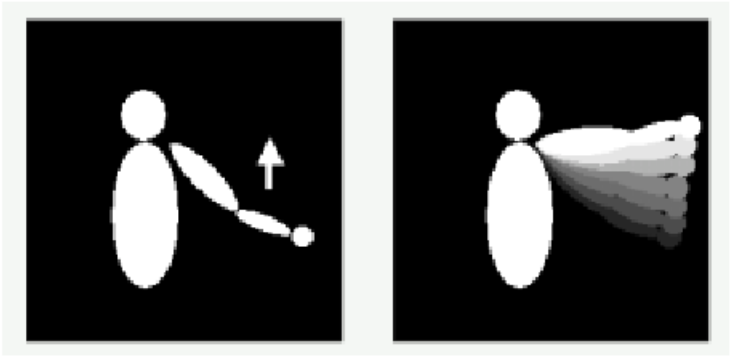
Background Subtraction Example



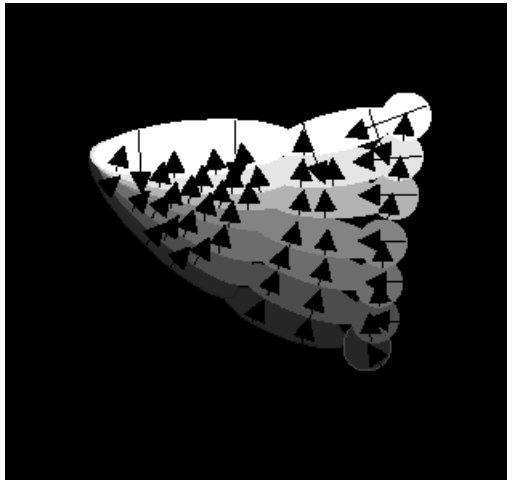
Motion Templates

- Object silhouette
- Motion history images
- Motion history gradients
- Motion segmentation algorithm

silhouette MHI

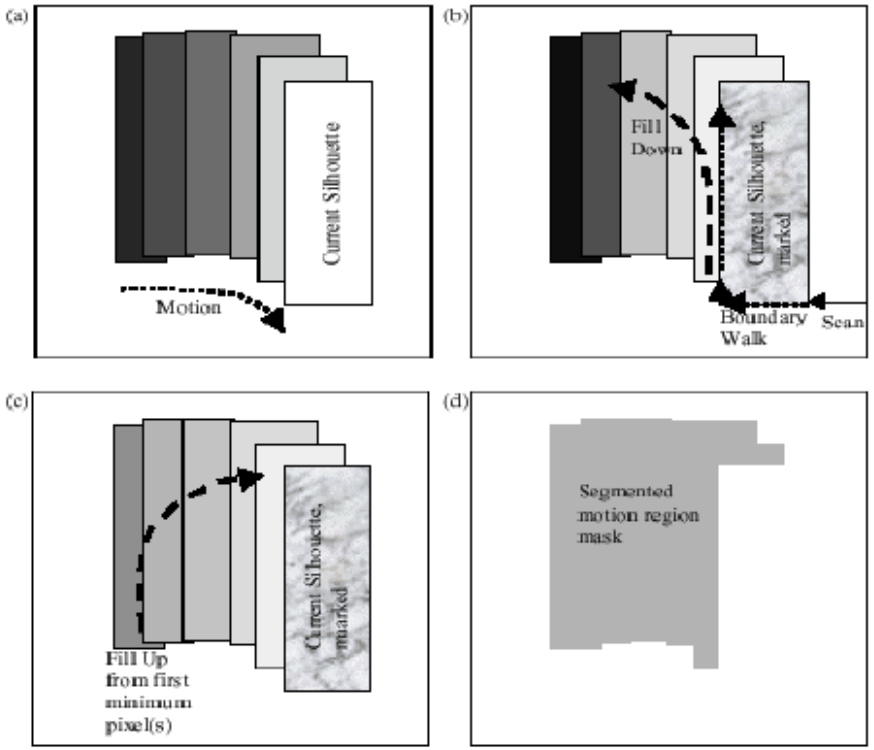


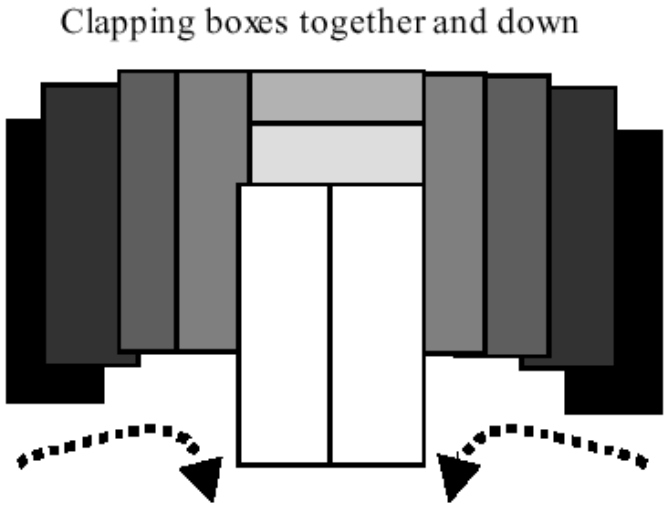
MHG



Motion Segmentation Algorithm

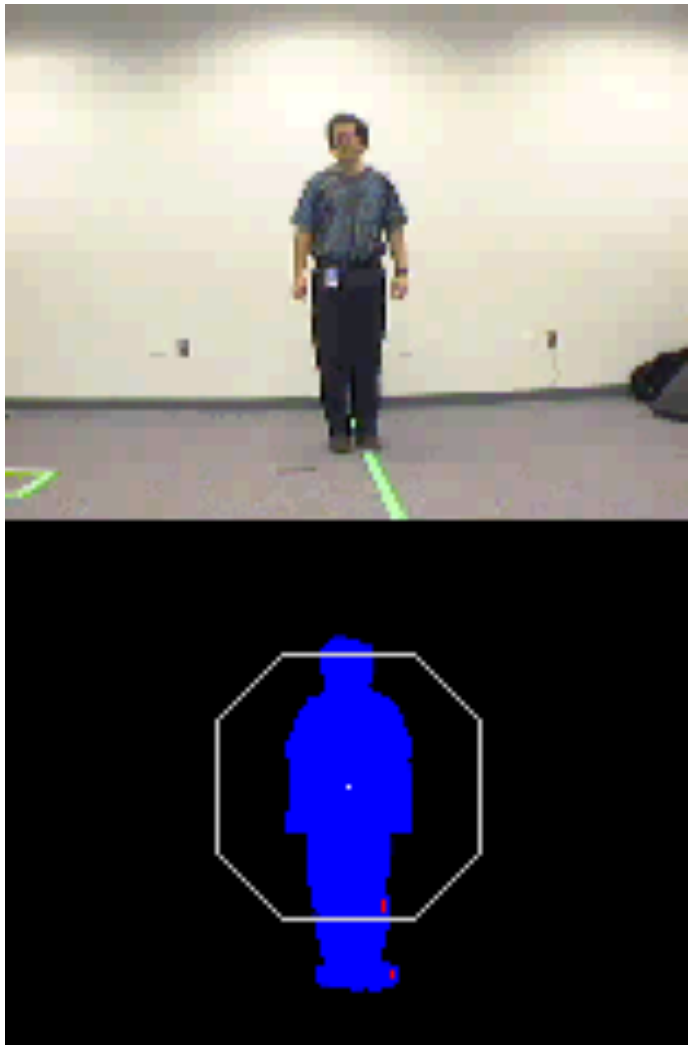
- Two-pass algorithm labeling all motion segments





Motion Templates Example

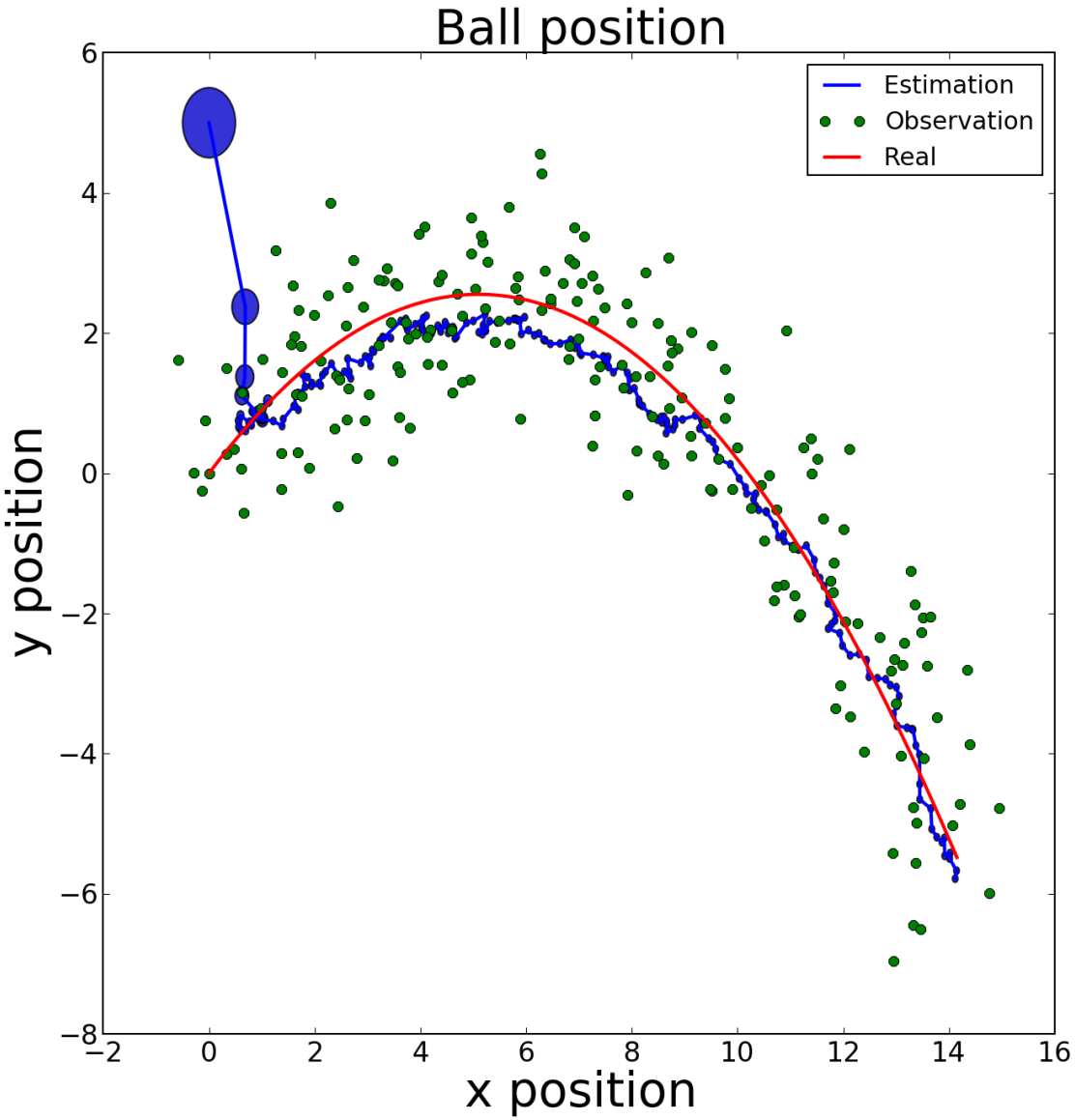
•Motion templates allow to retrieve the dynamic characteristics of the moving object



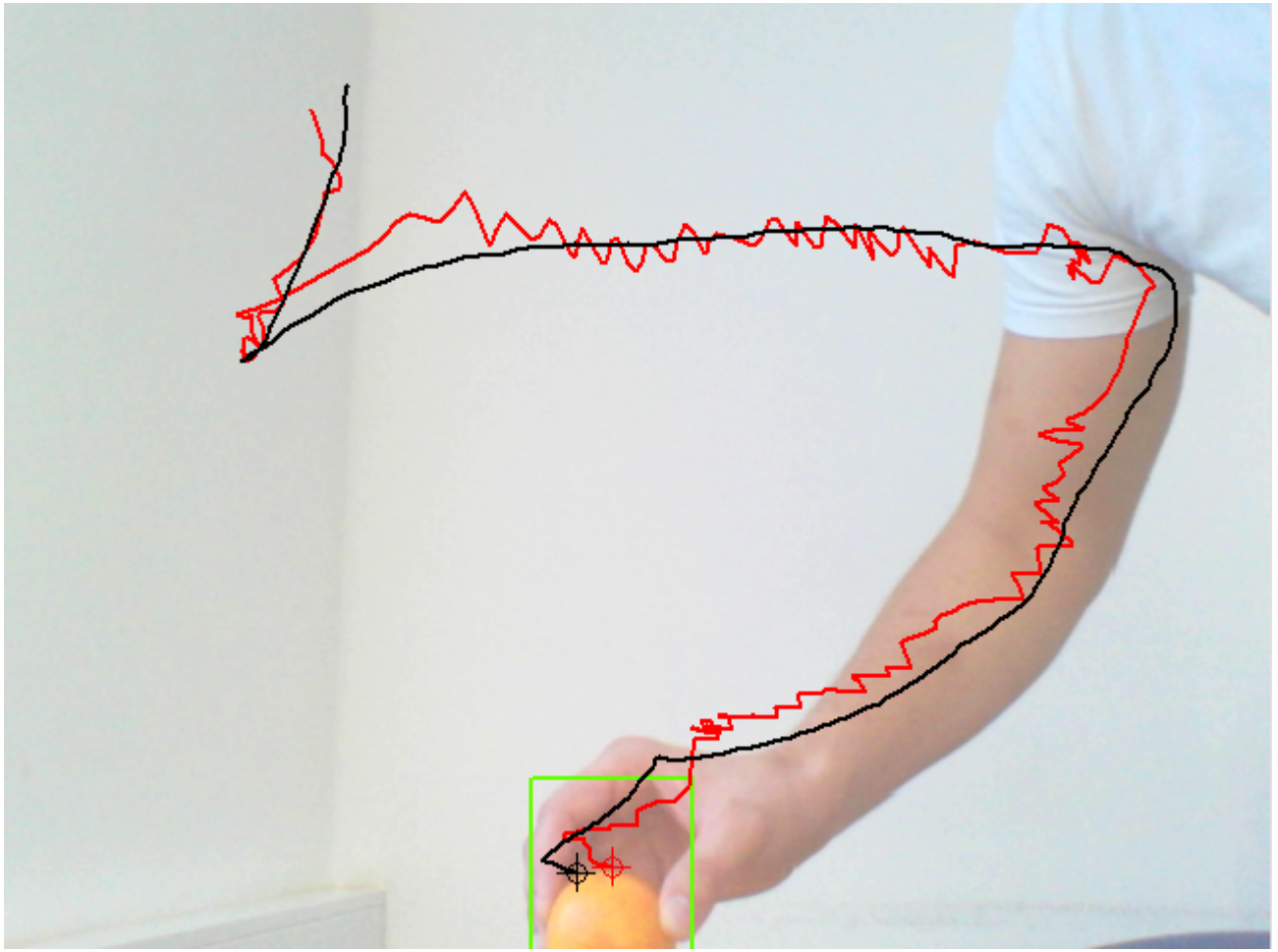
Kalman filter object tracker

- Smooth track line by set of noisy estimations
- Prediction the next points of track

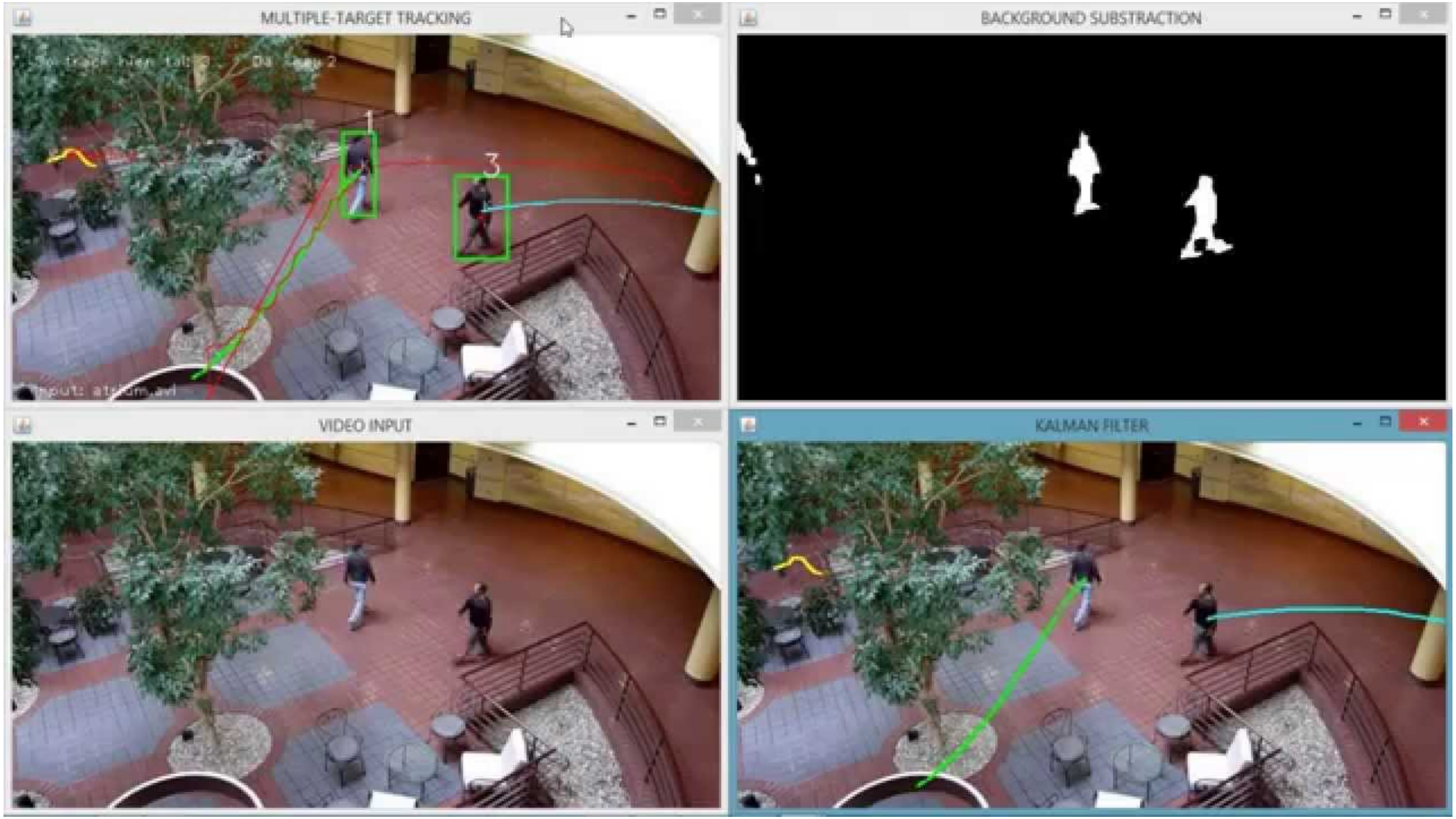
Kalman filter object tracker: smooth



Kalman filter object tracker: single object



Kalman filter object tracker: multi object



3D reconstruction

3D reconstruction

- Camera Calibration
- View Morphing
- POSIT

Camera Calibration

- Define intrinsic and extrinsic camera parameters.
- Define Distortion parameters

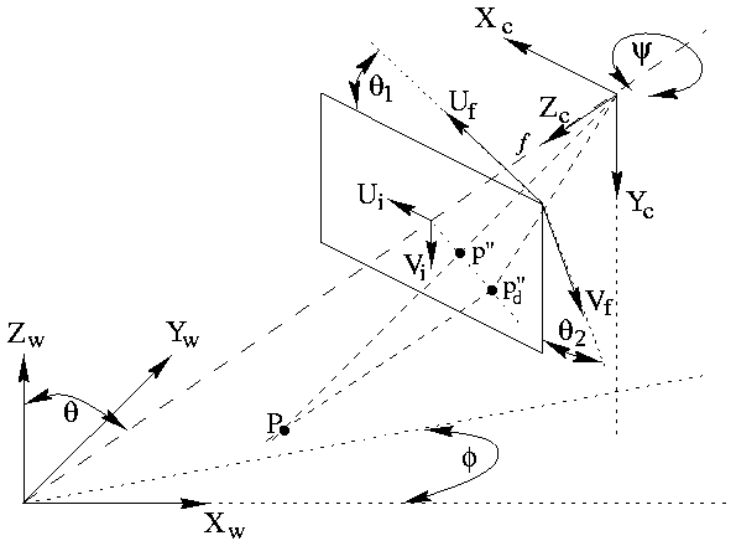
$$p = A[RT]P,$$

$$A = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}, \quad R = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & r_{33} \end{bmatrix}, \quad T = \begin{bmatrix} t_1 \\ t_2 \\ t_3 \end{bmatrix}, \quad P = [X, Y, Z], \quad p = [u, v]$$

$$\tilde{u} = u + (u - c_x) \cdot [k_1 \cdot r^2 + k_2 \cdot r^4 + 2p_1 y + p_2 (r^2 / x + 2x)],$$

$$\tilde{v} = v + (v - c_y) \cdot [k_1 \cdot r^2 + k_2 \cdot r^4 + 2p_2 x + p_1 (r^2 / y + 2y)],$$

$$r^2 = x^2 + y^2.$$

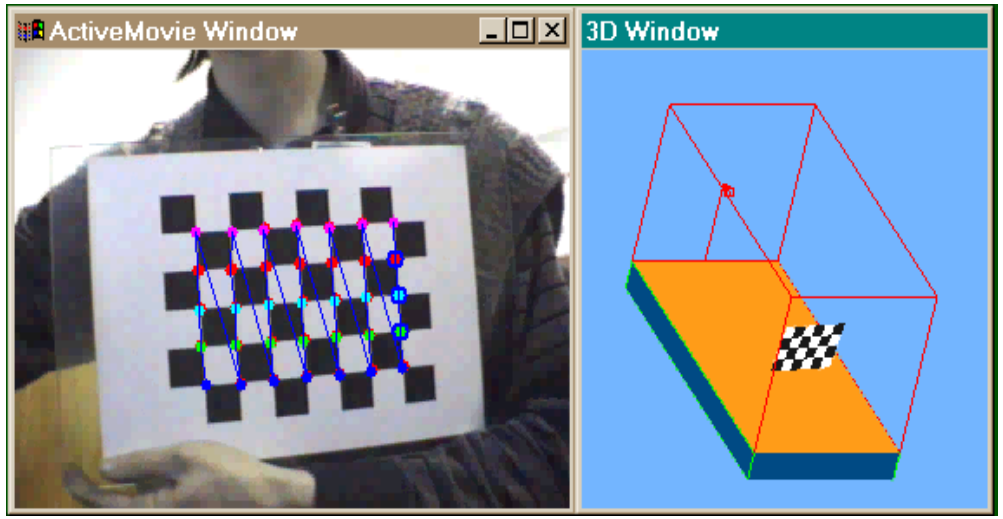


Camera Calibration

Now, camera calibration can be done by holding checkerboard in front of the camera for a few seconds.

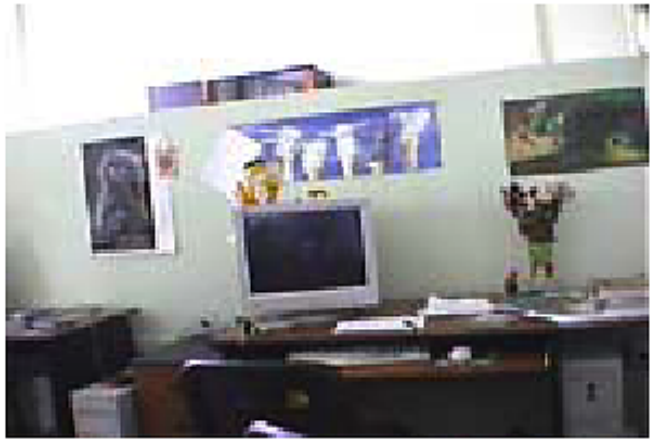
And after that you'll get:

3D view of etalon



Un-distorted image

View Morphing



Original Image From Left Camera

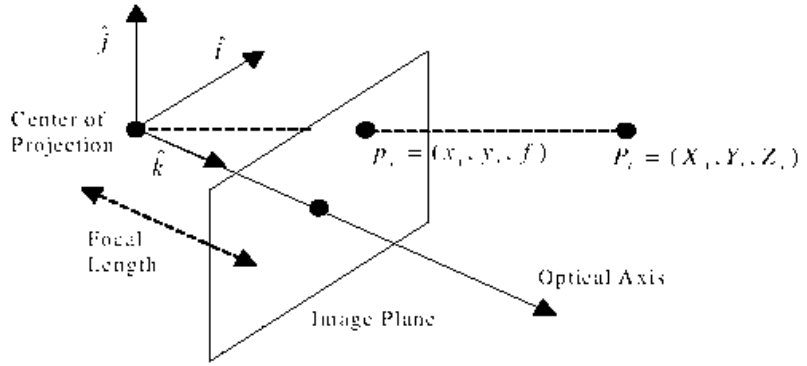
Original Image From Right Camera

Result

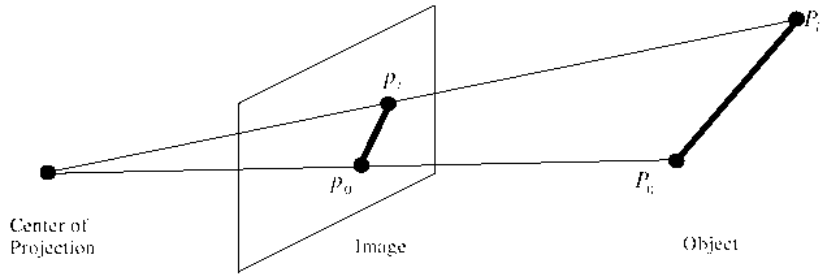
POSIT Algorithm

- Perspective projection:

$$x_i = (f / Z_i) \cdot X_i,\ y_i = (f / Z_i) \cdot Y_i$$



- Weak-perspective projection: $x_i = s \cdot X_i,\ y_i = s \cdot Y_i,\ s = f / Z.$



References

- Gunilla Borgefors. *Distance Transformations in Digital Images.*Computer Vision, Graphics and Image Processing 34, 344-371,(1986).
- G. Bradski and J. Davis. *Motion Segmentation and Pose Recognition with Motion History Gradients.* IEEE WACV'00, 2000.
- P. J. Burt, T. H. Hong, A. Rosenfeld. *Segmentation and Estimation of Image Region Properties Through Cooperative Hierarchical Computation.* IEEE Tran. On SMC, Vol. 11, N.12, 1981, pp.802-809.
- J.Canny.*A Computational Approach to Edge Detectio*n, IEEE Trans. on Pattern Analysis and Machine Intelligence, 8(6), pp.679-698 (1986).
- J. Davis and Bobick. *The Representation and Recognition of Action Using Temporal Template*s. MIT Media Lab Technical Report 402,1997.
- Daniel F. DeMenthon and Larry S. Davis. *Model-Based Object Pose in 25 Lines of Code.* In Proceedings of ECCV '92, pp. 335-343, 1992.
- Andrew W. Fitzgibbon, R.B.Fisher. *A Buyer's Guide to Conic Fittin*g.Proc.5 th British Machine Vision Conference, Birmingham, pp. 513-522, 1995.
- Berthold K.P. Horn and Brian G. Schunck. *Determining Optical Flo*w. Artificial Intelligence, 17, pp. 185-203, 1981.

References

- M.Hu.*Visual Pattern Recognition by Moment Invariant*s, IRE Transactions on Information Theory, 8:2, pp. 179-187, 1962.
- B. Jahne. *Digital Image Processing*. Springer, New York, 1997.
- M. Kass, A. Witkin, and D. Terzopoulos. *Snakes: Active Contour Model*s, International Journal of Computer Vision, pp. 321-331, 1988.
- J.Matas, C.Galambos, J.Kittler. *Progressive Probabilistic Hough Transfor*m. British Machine Vision Conference, 1998.
- A. Rosenfeld and E. Johnston. *Angle Detection on Digital Curve*s. IEEE Trans. Computers, 22:875-878, 1973.
- Y.Rubner.C.Tomasi,L.J.Guibas.*Metrics for Distributions with Applications to Image Databases*. Proceedings of the 1998 IEEE International Conference on Computer Vision, Bombay, India, January 1998, pp. 59-66.
- Y. Rubner. C. Tomasi, L.J. Guibas. *The Earth Mover's Distance as a Metric for Image Retrieval*. Technical Report STAN-CS-TN-98-86, Department of Computer Science, Stanford University, September, 1998.
- Y.Rubner.C.Tomasi.*Texture Metric*s. Proceeding of the IEEE International Conference on Systems, Man, and Cybernetics, San-Diego, CA, October 1998, pp. 4601- 4607. http://robotics.stanford.edu/~rubner/publications.html

References

- J. Serra. *Image Analysis and Mathematical Morphology*. Academic Press, 1982.
- Bernt Schiele and James L. Crowley. *Recognition without Correspondence Using Multidimensional Receptive Field Histograms*. In International Journal of Computer Vision 36 (1), pp. 31-50, January 2000.
- S. Suzuki, K. Abe. *Topological Structural Analysis of Digital Binary Images by Border Following*. CVGIP, v.30, n.1. 1985, pp. 32-46.
- C.H.Teh, R.T.Chin.*On the Detection of Dominant Points on Digital Curves*. - IEEE Tr. PAMI, 1989, v.11, No.8, p. 859-872.
- Emanuele Trucco, Alessandro Verri. *Introductory Techniques for 3-D Computer Visio*n. Prentice Hall, Inc., 1998.
- D. J. Williams and M. Shah. *A Fast Algorithm for Active Contours and Curvature Estimatio*n. CVGIP: Image Understanding, Vol. 55, No. 1, pp. 14-26, Jan., 1992. http://www.cs.ucf.edu/~vision/papers/shah/92/WIS92A.pdf.
- A.Y.Yuille, D.S.Cohen, and P.W.Hallinan. *Feature Extraction from Faces Using Deformable Templates in CVP*R, pp. 104-109, 1989.
- Zhengyou Zhang. *Parameter Estimation Techniques: A Tutorial with Application to Conic Fittin*g, Image and Vision Computing Journal, 1996.