

HUMAN VISION



It Works!!

-->Proof of existence.

- The image formation process is well understood
- The image understanding one remains mysterious

PATHWAYS TO THE BRAIN

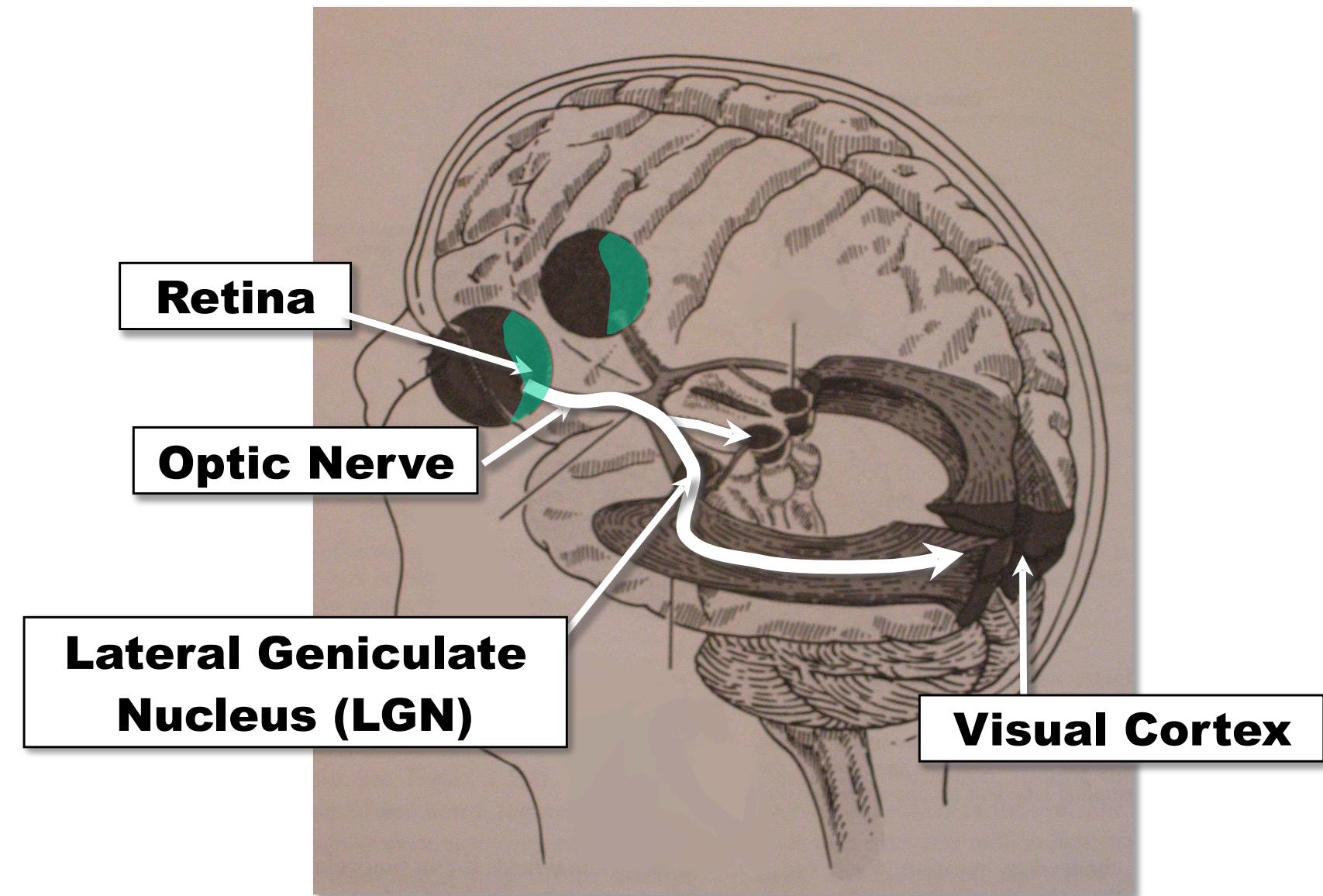
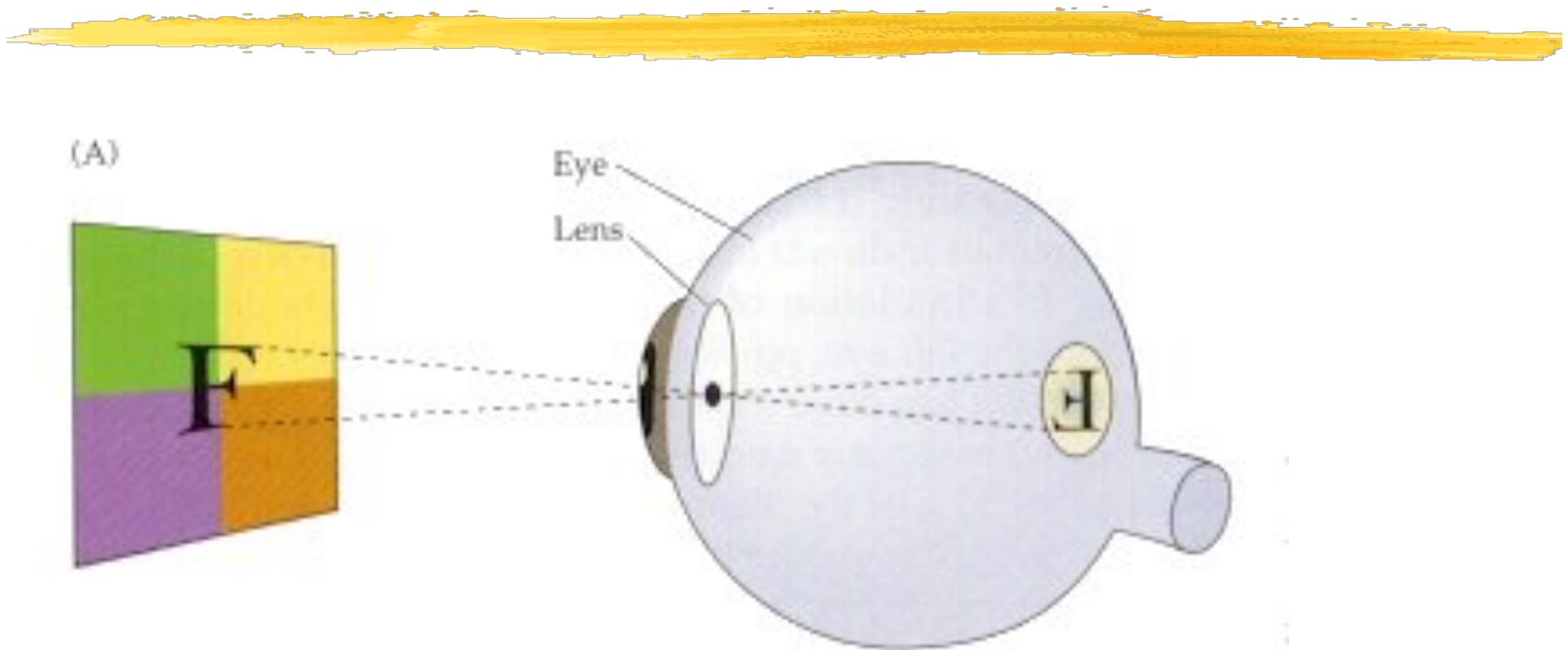
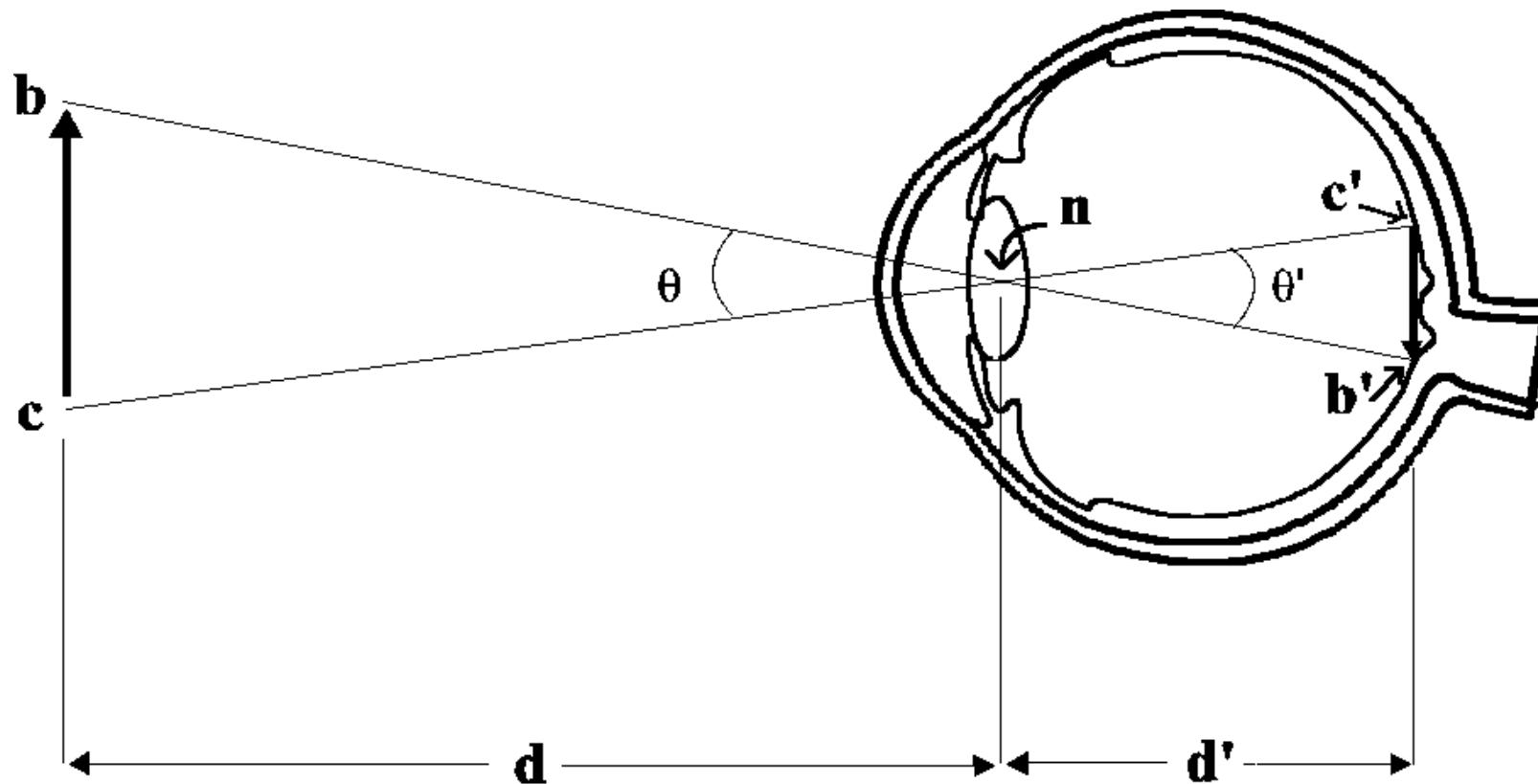


IMAGE FORMATION



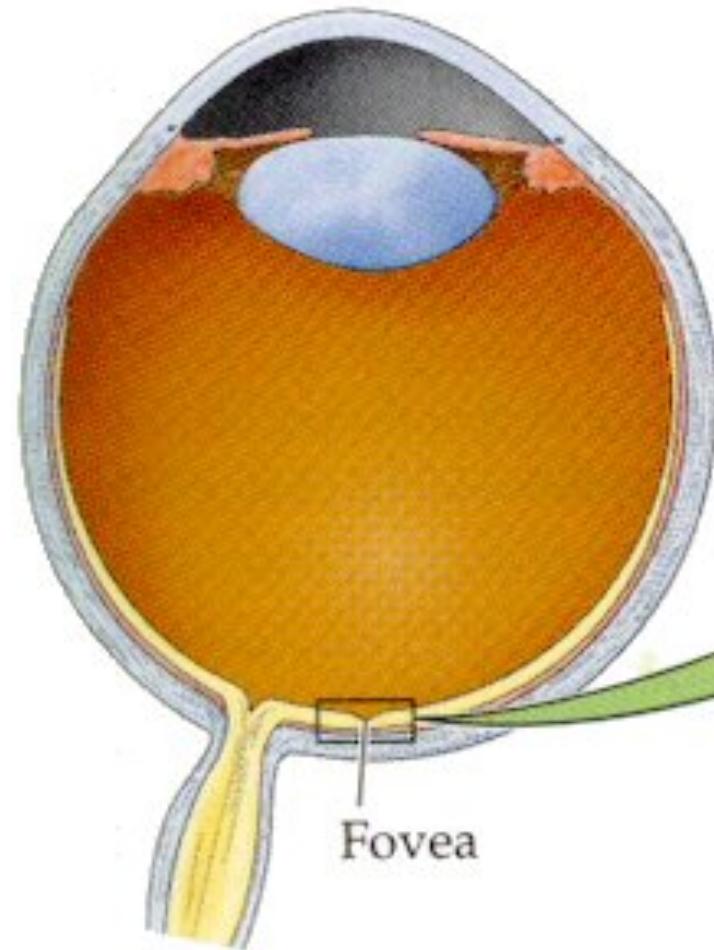
Pinhole camera model

PERSPECTIVE PROJECTION

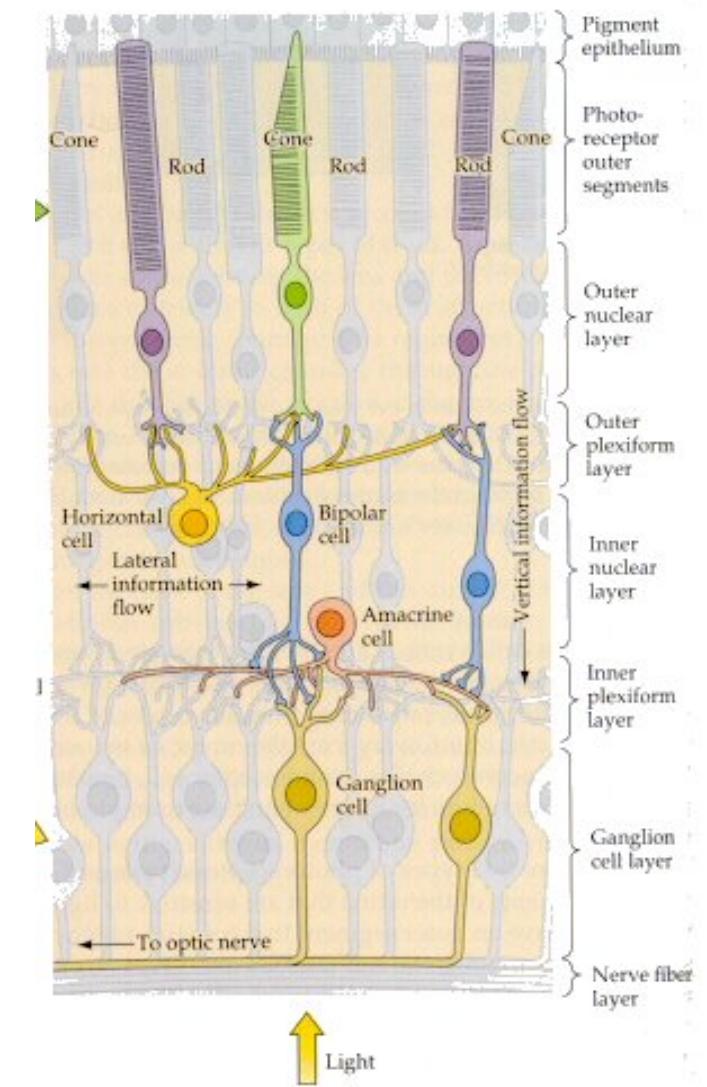
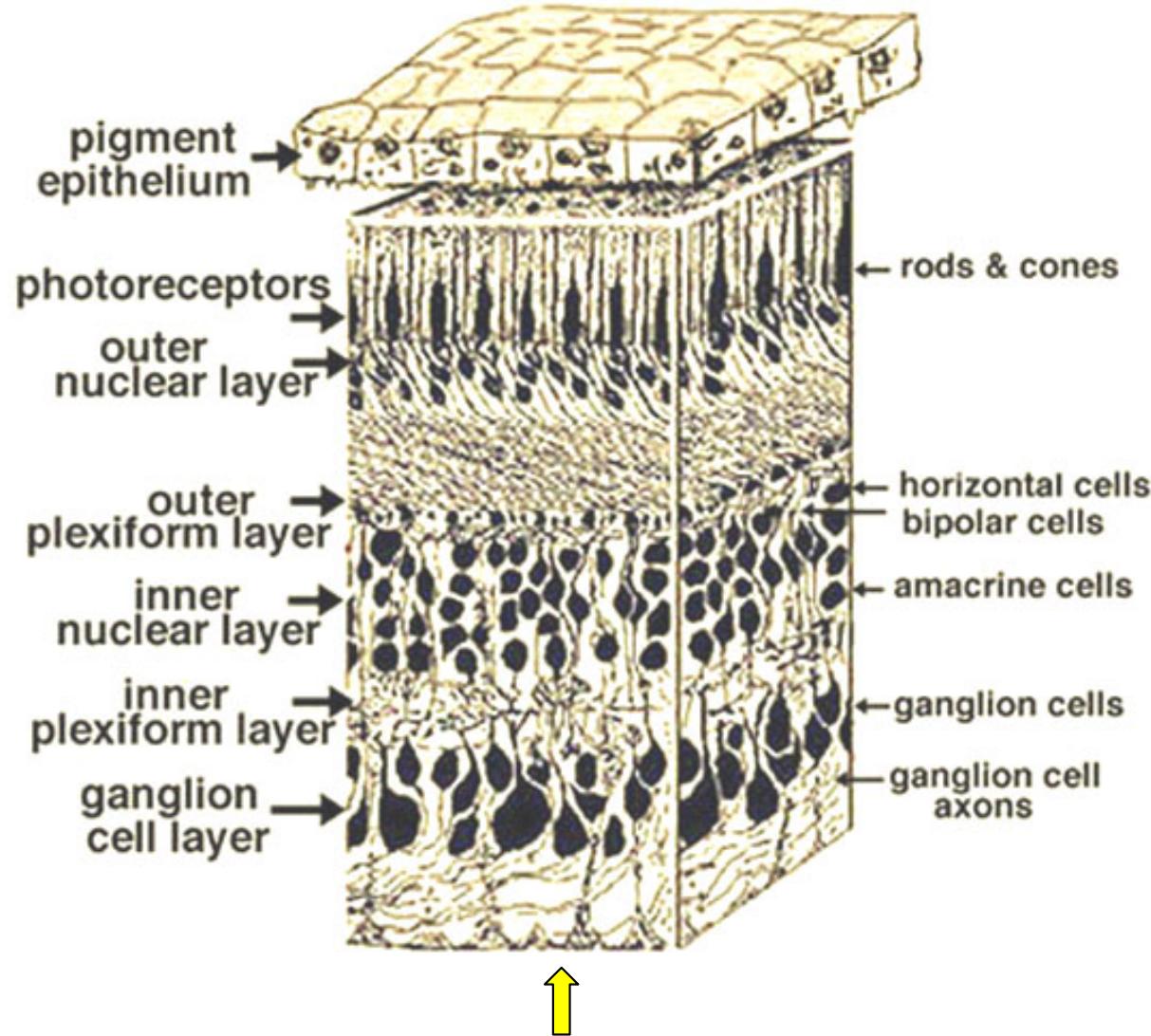


$$bc/d = b'c'/d'$$

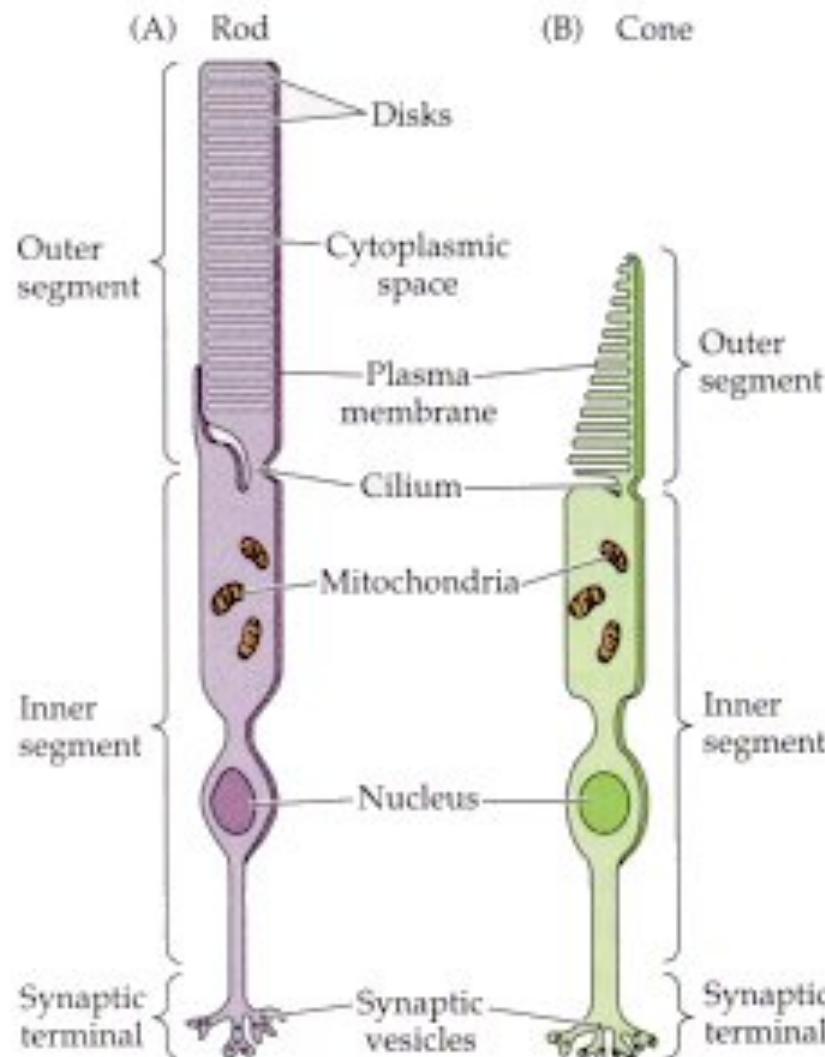
HUMAN EYE



RETINA



RODS AND CONES



Rods: Low-intensity light vision, e.g. night vision.

Cones: Color-vision with higher intensity light.

CELL DISTRIBUTION

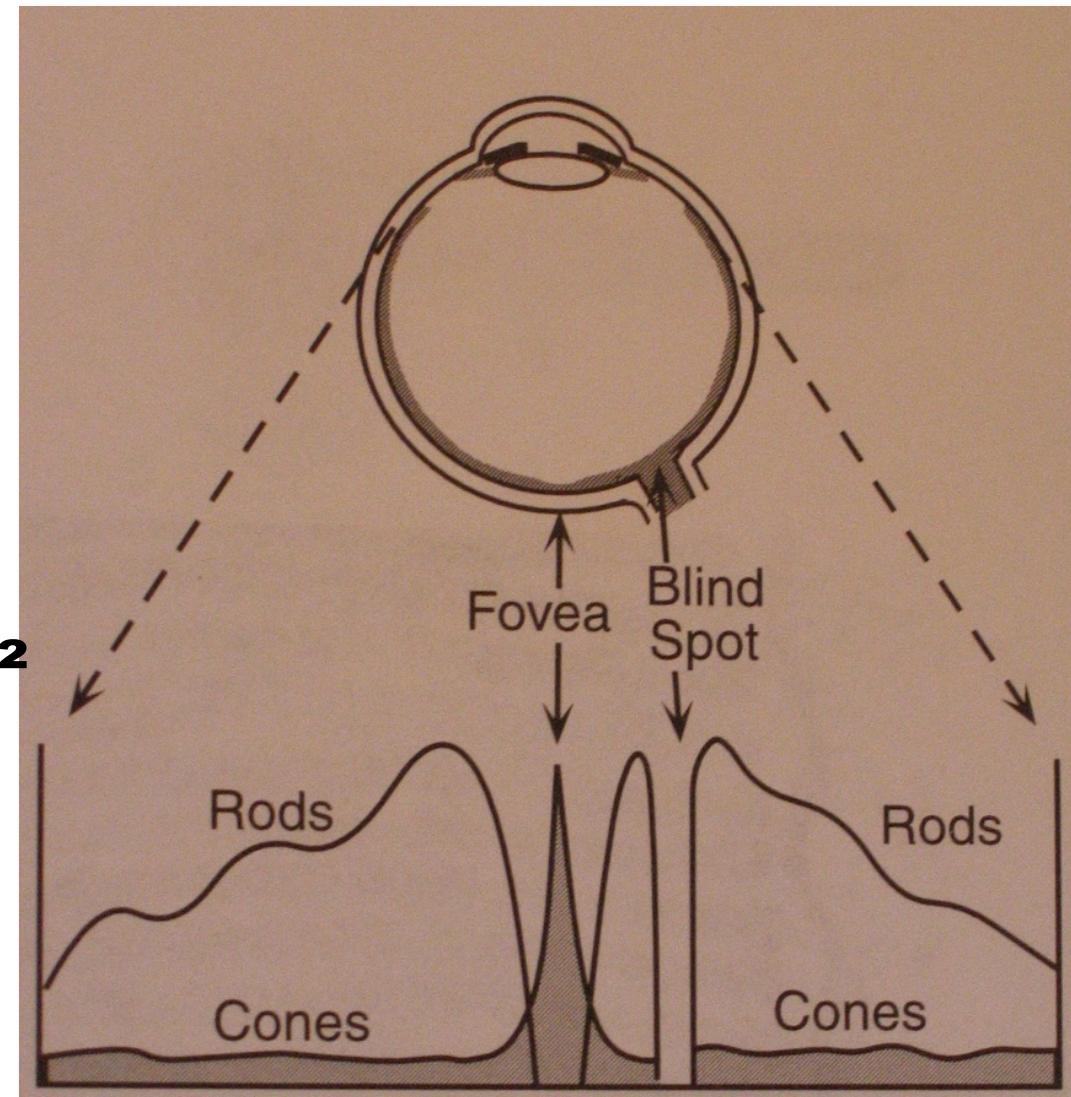


receptors/mm²

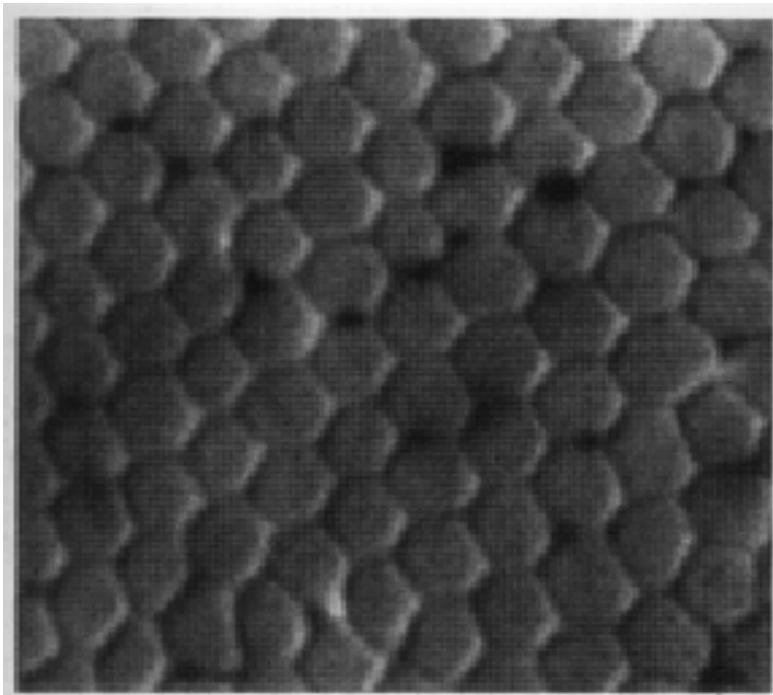
150'000

100'000

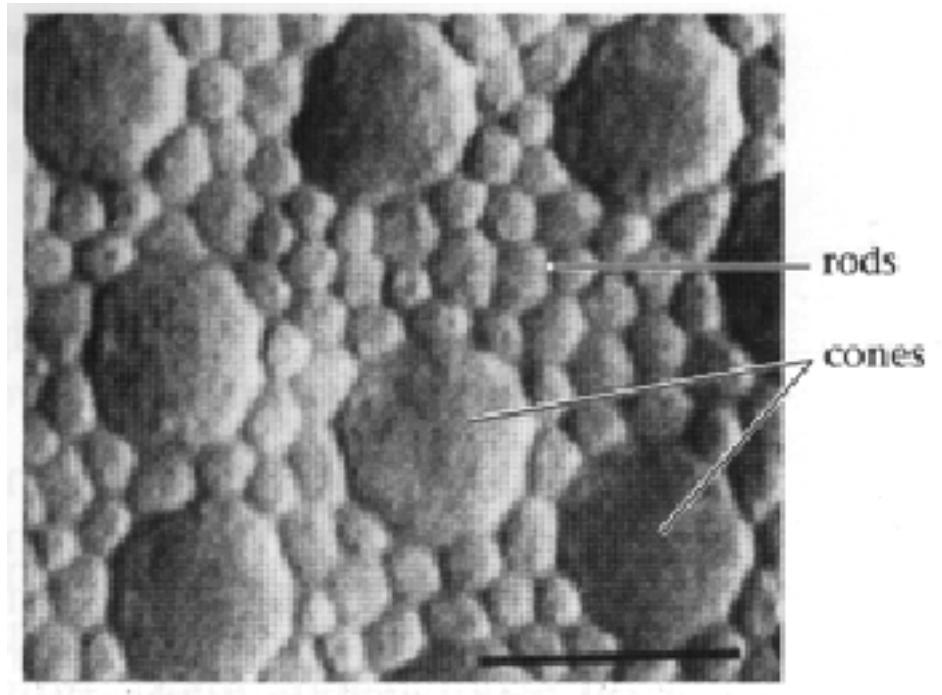
50'000



FOVEA vs PERIPHERY

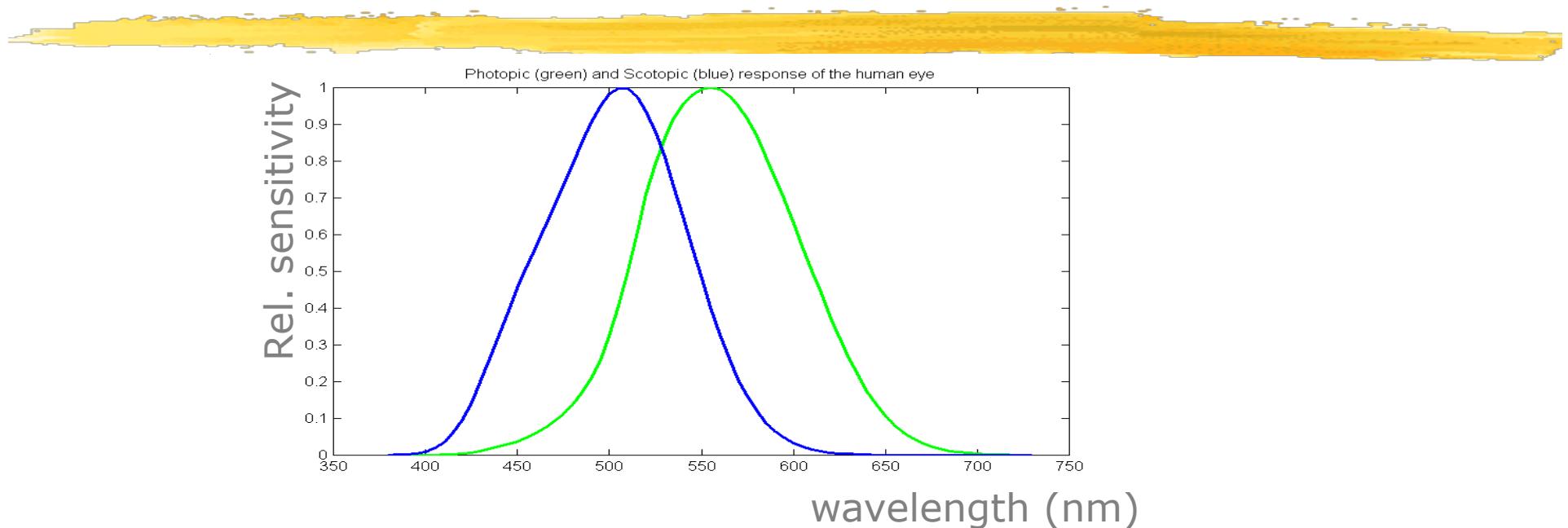


Fovea



Periphery

SCOPTIC vs PHOTOPIC



Low luminance ($< 1 \text{ cd/m}^2$):

- 120 million rods with peak spectral response around 510 nm.
- Primarily located outside the fovea.

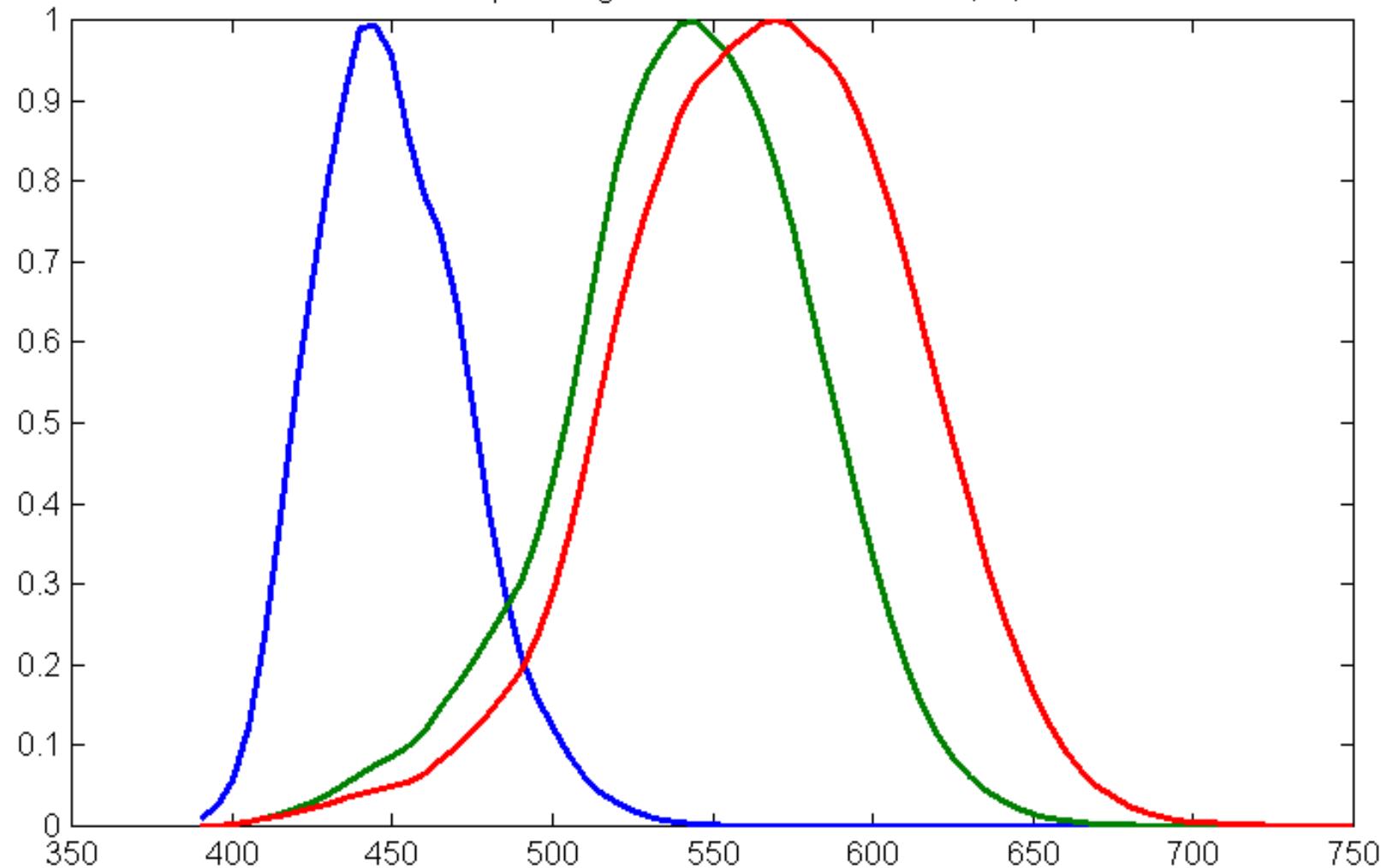
High luminance ($> 100 \text{ cd/m}^2$):

- 7 million cones per retina.
- Primarily located in the fovea.
- Three types of cones (S, M, L) with peak spectral response at different nm.
- Ratio L:M:S $\approx 40:20:1$

SENSITIVITY TO DIFFERENT WAVELENGTHS



Stockman and Sharpe 2 degree Cone fundamentals: L, M, and S cones

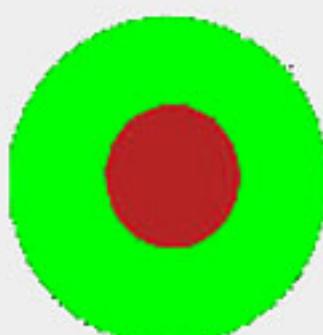


GANGLION CELLS

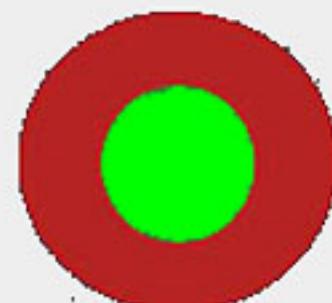
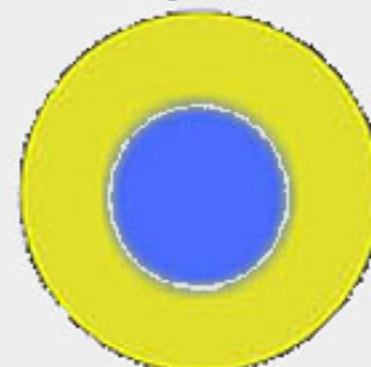
Color opponent ganglion cells



red ON/green OFF red OFF/green ON



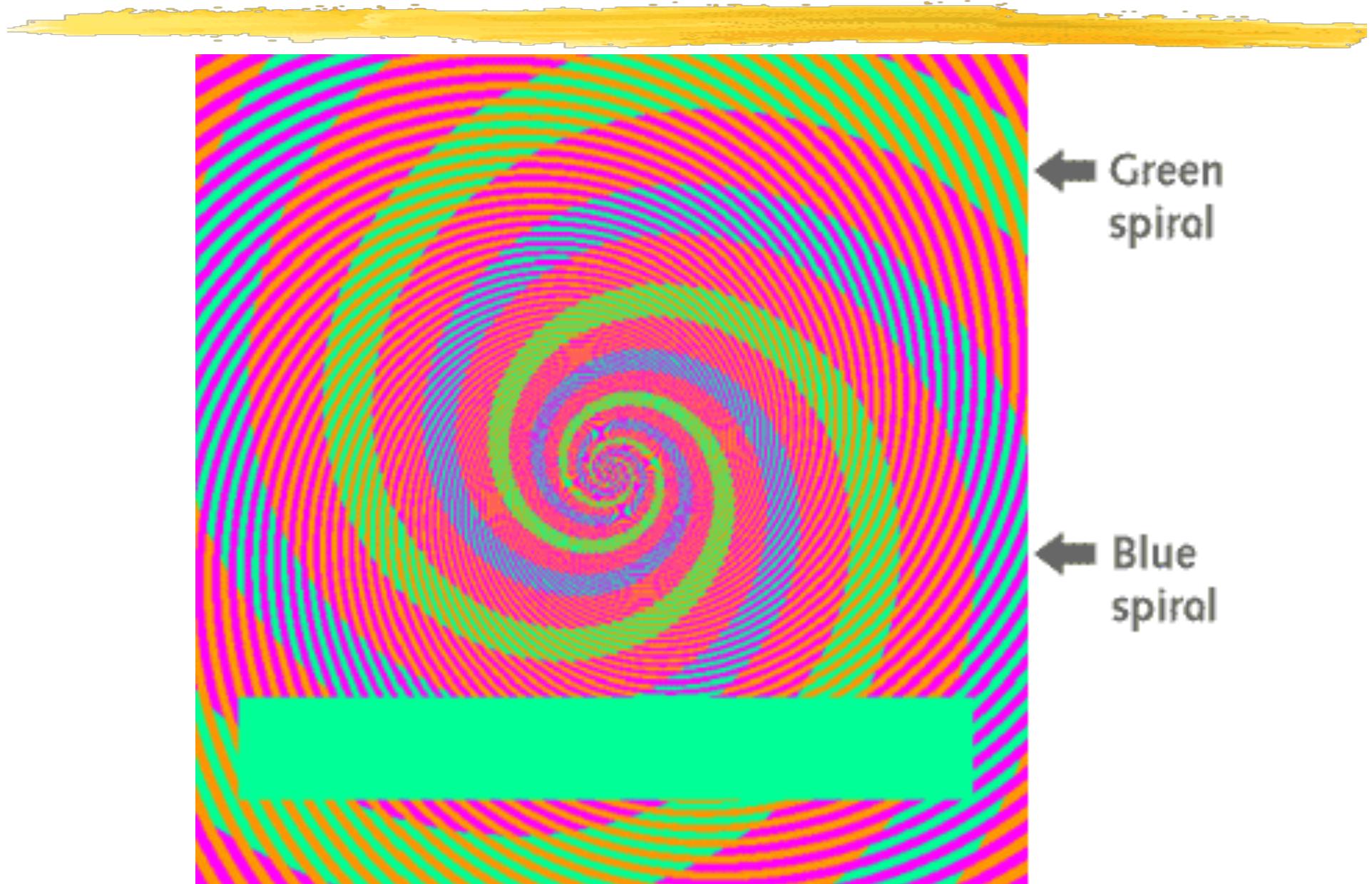
blue ON/yellow OFF



green ON/red OFF green OFF/red ON



COLOR ILLUSION

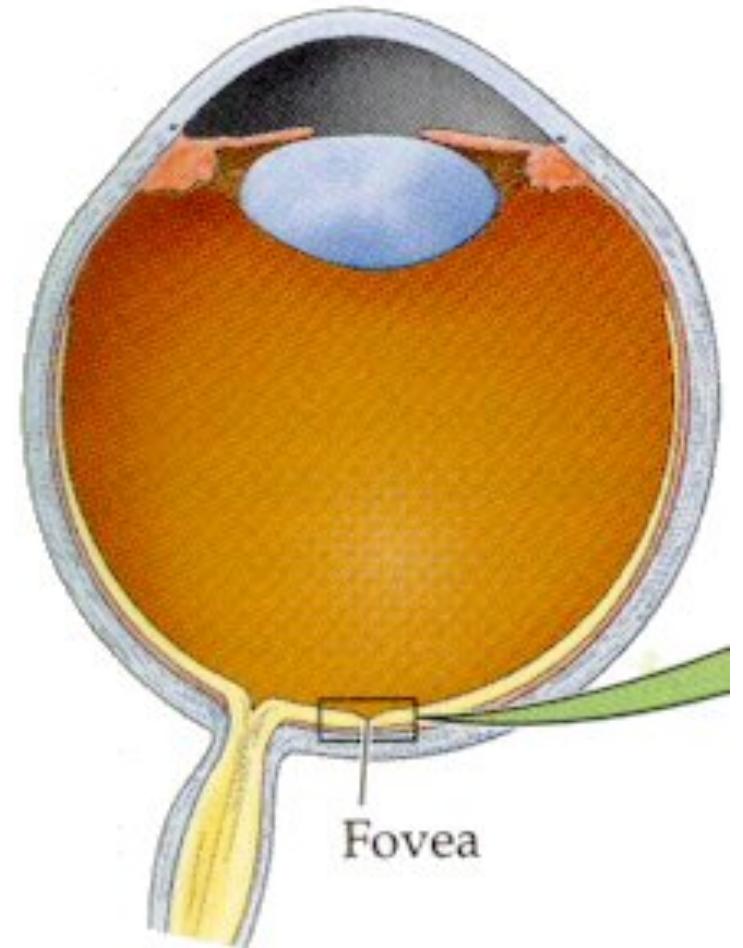


PERIPHERAL vs FOVEAL VISION

Much higher concentration of cells on the Fovea

→ Active vision:

- We find objects using our peripheral vision
- We concentrate our gaze on objects of interest.



THE HUMAN EYE IN SHORT



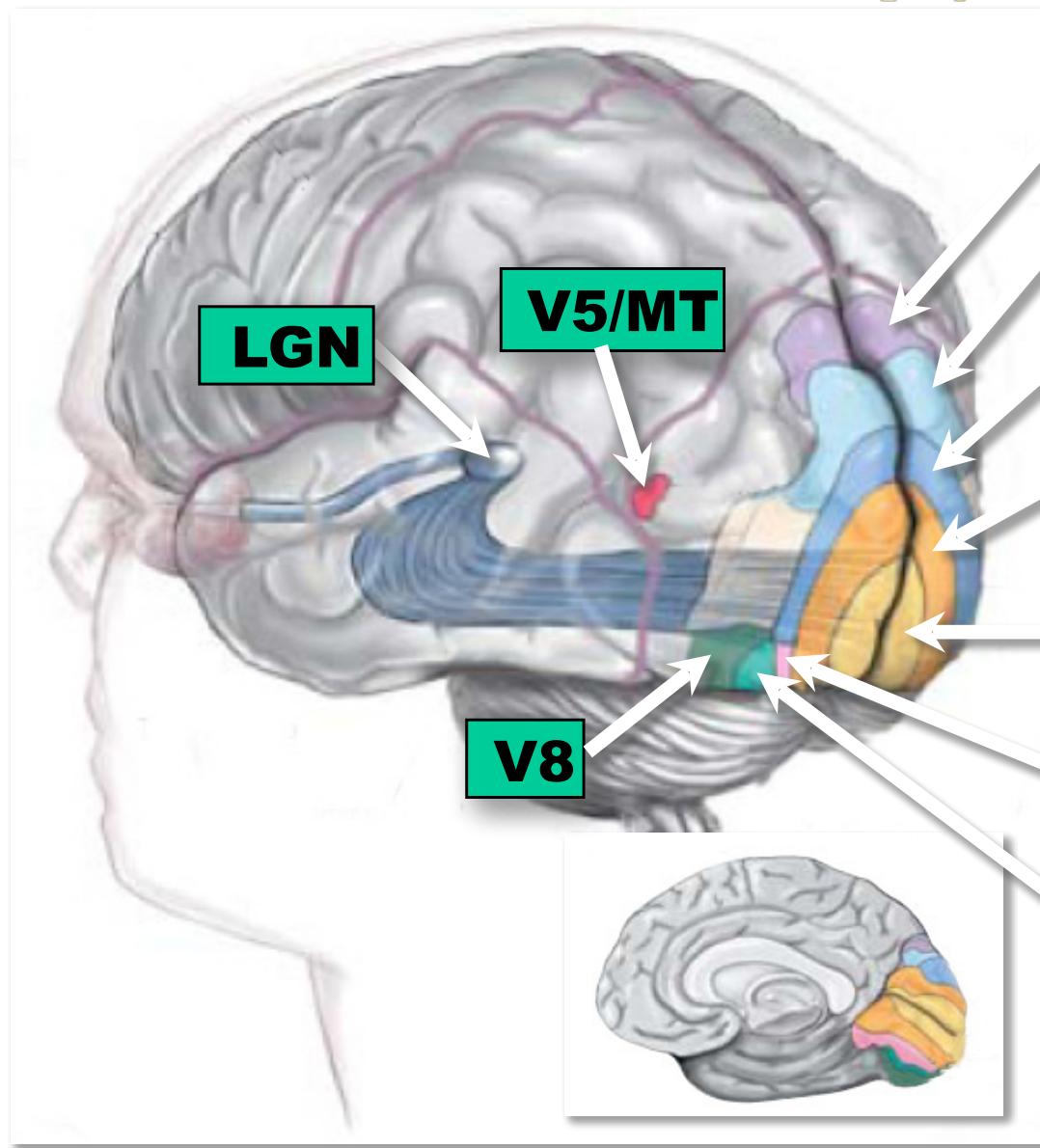
The Retina:

- rods (low-intensity light, night vision)
- cones (color-vision)
- Synapses and ganglions
- Optic nerve fibers

Sensing and low-level processing layer:

- 125 millions rods and cones feed into 1 million nerve fibers

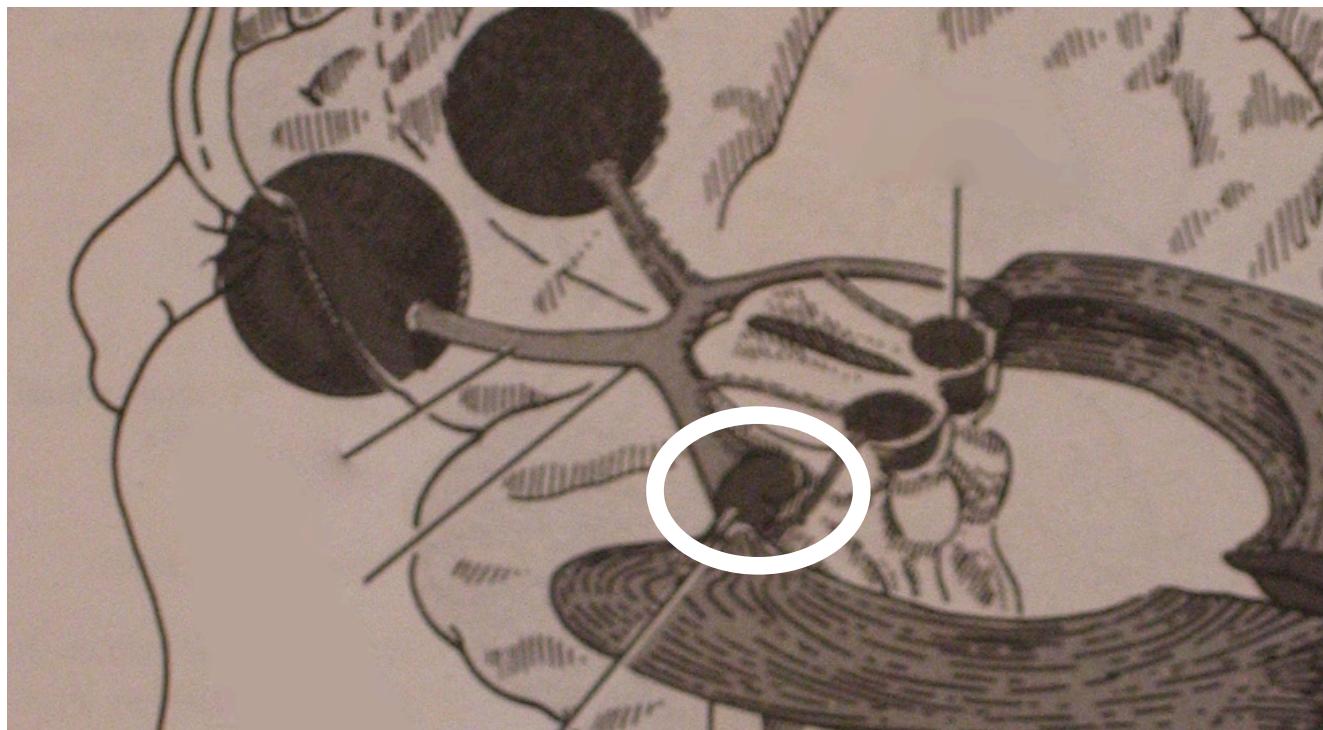
VISUAL CORTEX



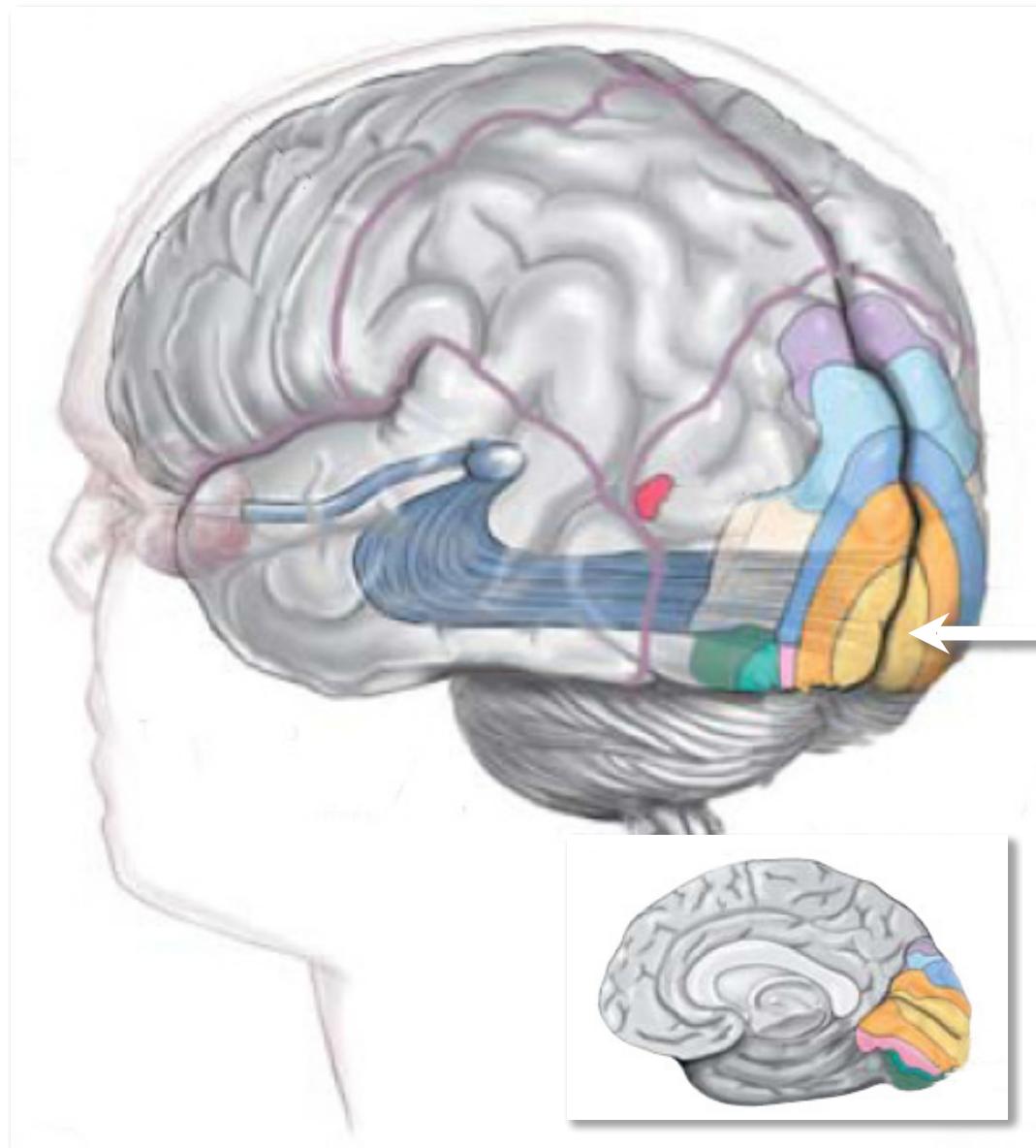
V7
V3A
V3
V2
V1
VP
V4

LATERAL GENICULATE NUCLEUS (LGN)

Receives feedbacks from V1 and V2. There is ten times more feedback than feedforward sent to V1.



PRIMARY VISUAL CORTEX (V1)



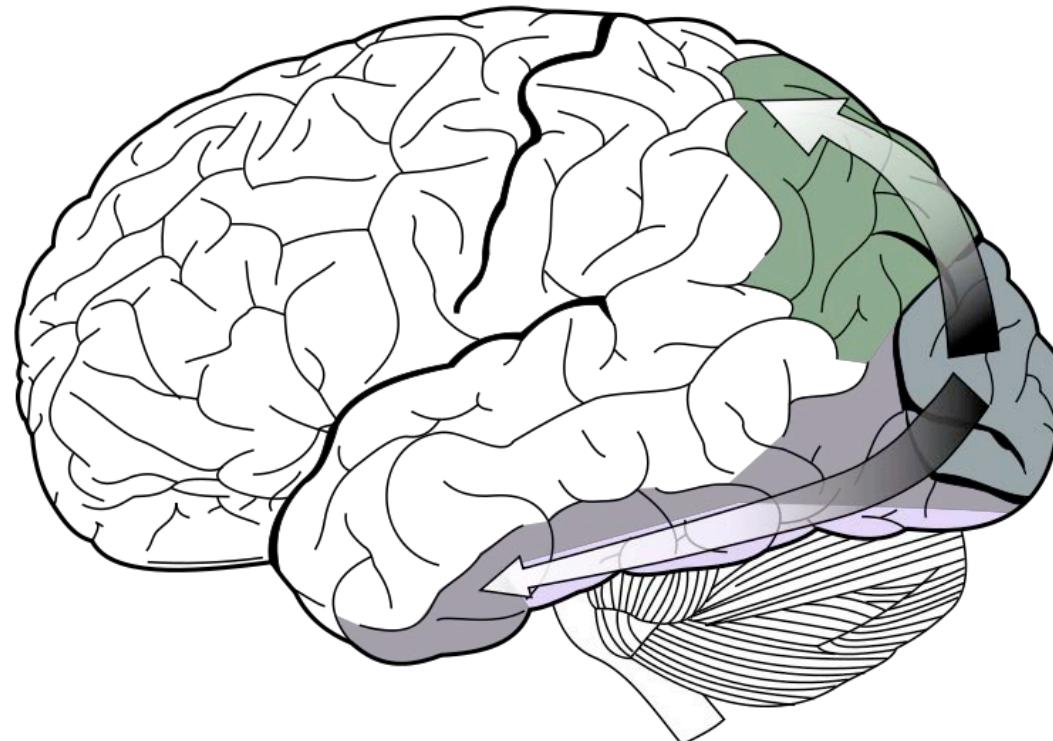
- Largest area in the visual cortex.
 - 100 times as many neurons as retinal ganglion cells
- Overcomplete representation.

FROM V1 TO THE OTHERS

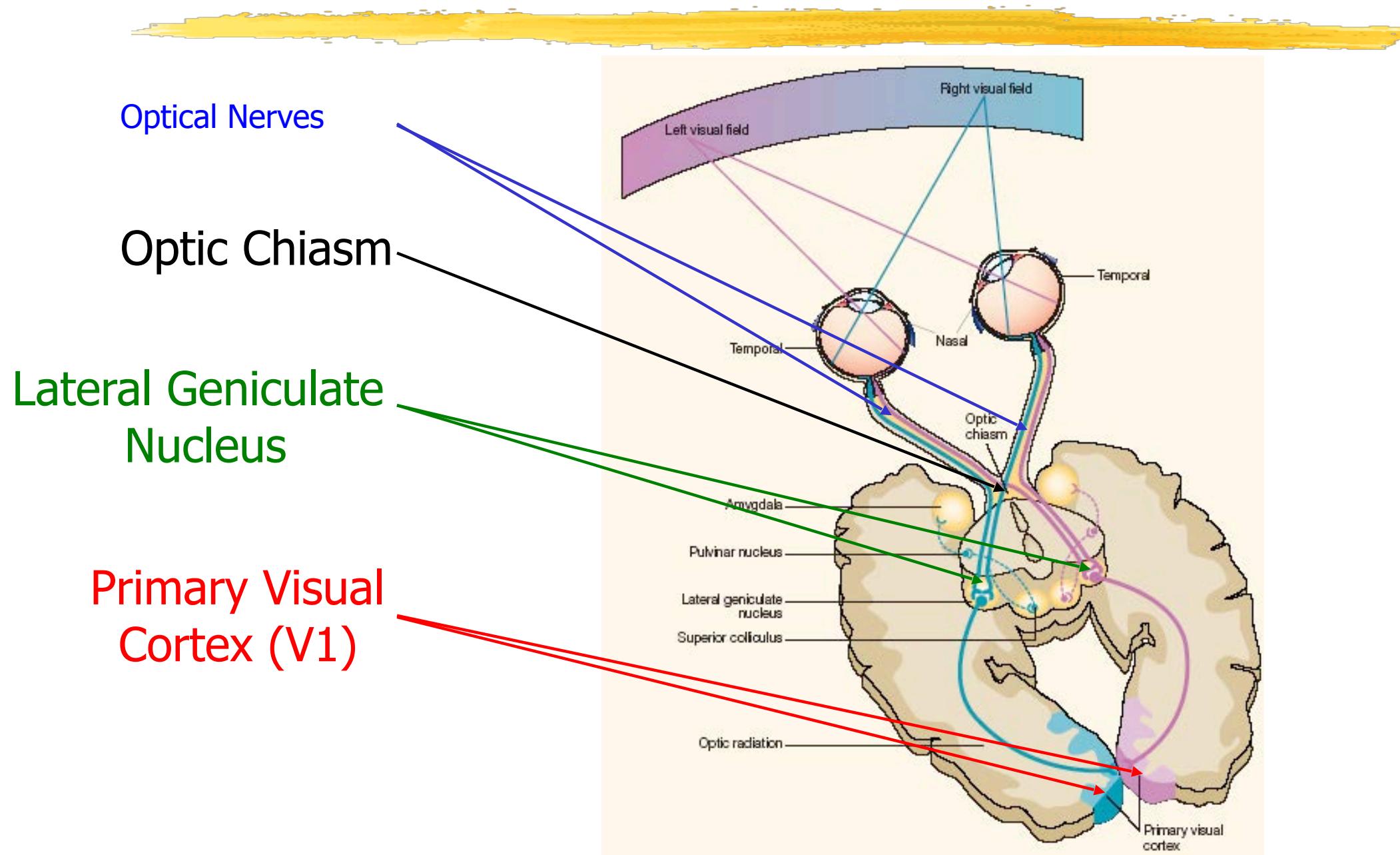
To pathways originate from V1:

- The “where” pathway: $V1 \rightarrow V2 \rightarrow V5 \rightarrow$ parietal lobe.
- The “what” pathway: $V1 \rightarrow V2 \rightarrow V3 \rightarrow V4 \rightarrow$ temporal lobe.

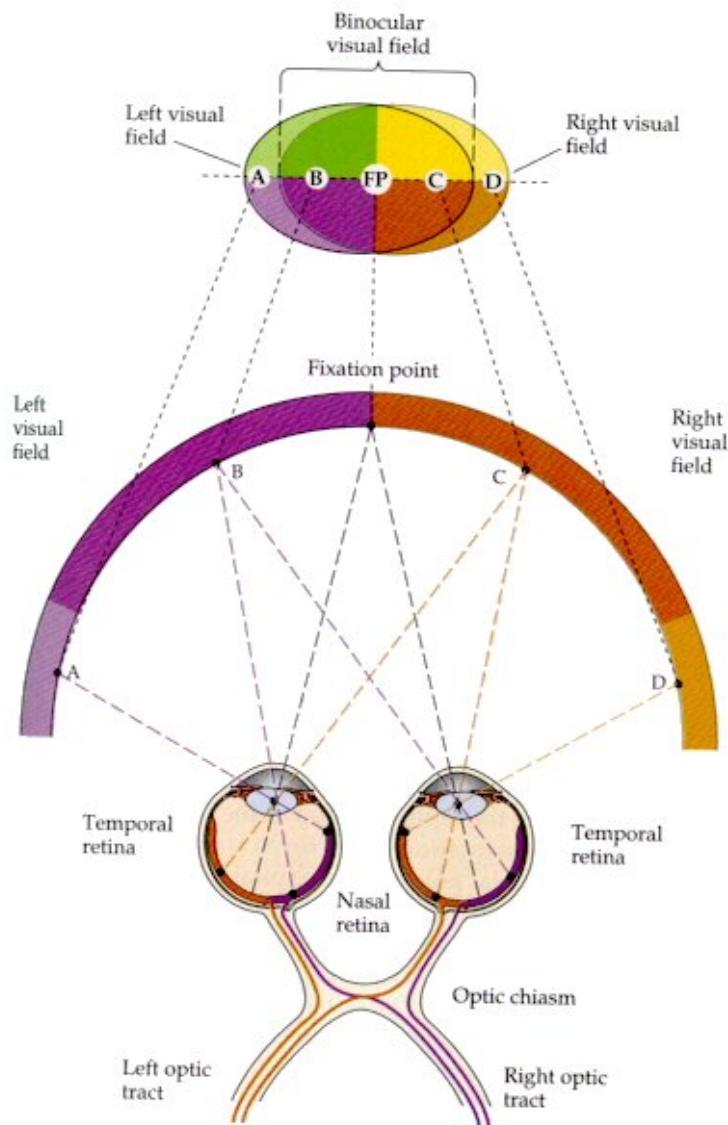
⇒ Motion Detection and Object Recognition are mostly performed in parallel but interconnections exist.



HEMISPHERICAL VISION



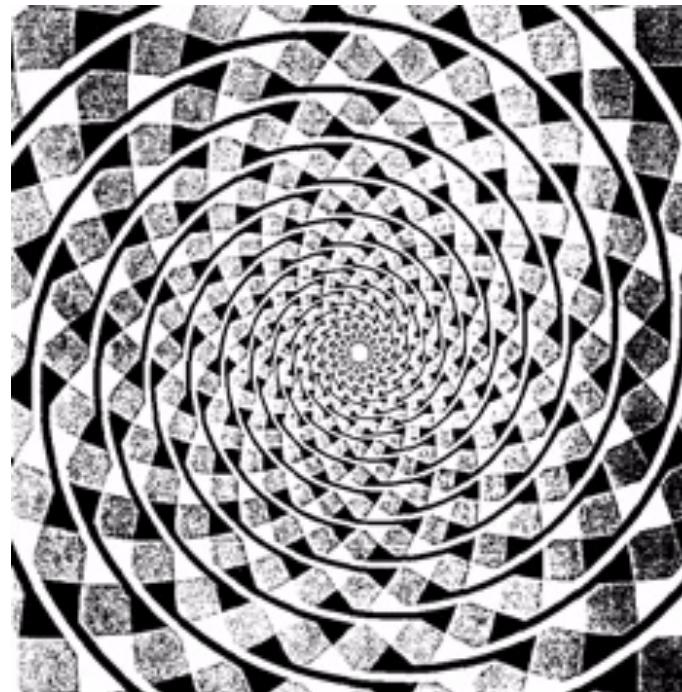
STEREOSCOPICAL VISION



Our brain is wired for stereo vision:

- Redundancy
- Depth perception

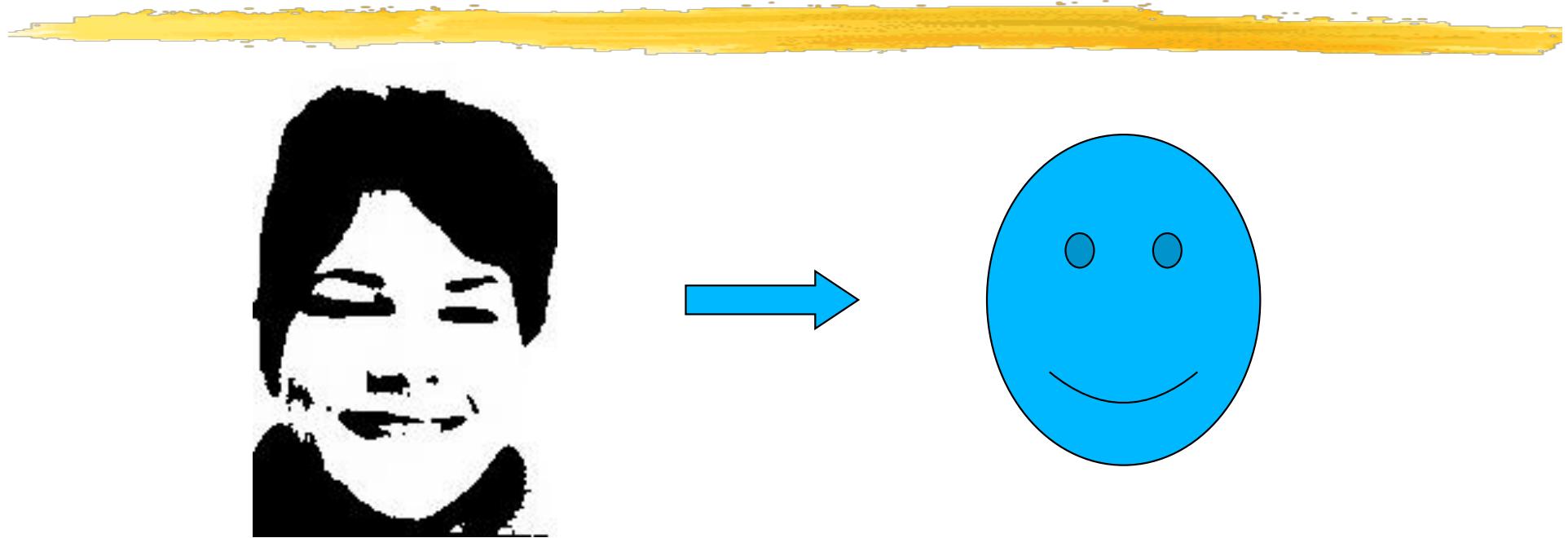
OPTICAL ILLUSIONS



Every image is the image of thing merely to him who knows how to read it, and who is enabled by the aid of the image to form an idea of the thing.

Handbook of Physiological Optics
H. von Helmholtz

CONTROLLED HALLUCINATION?



Perhaps, but very cleverly implemented in “wetware”.

→ How can we emulate it in hardware?

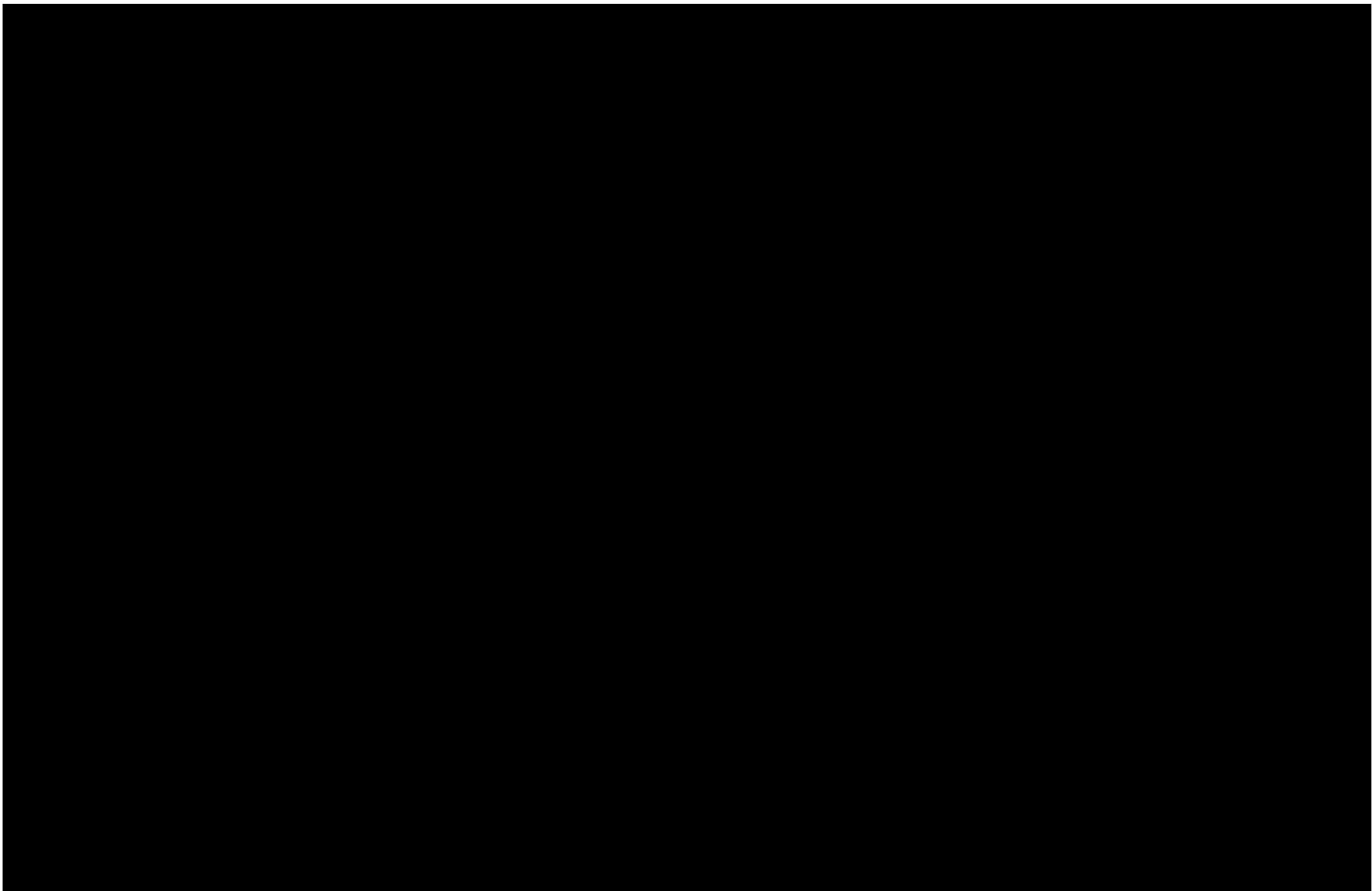
RECOGNIZE AND CLASSIFY ANIMAL -- NO ANIMAL



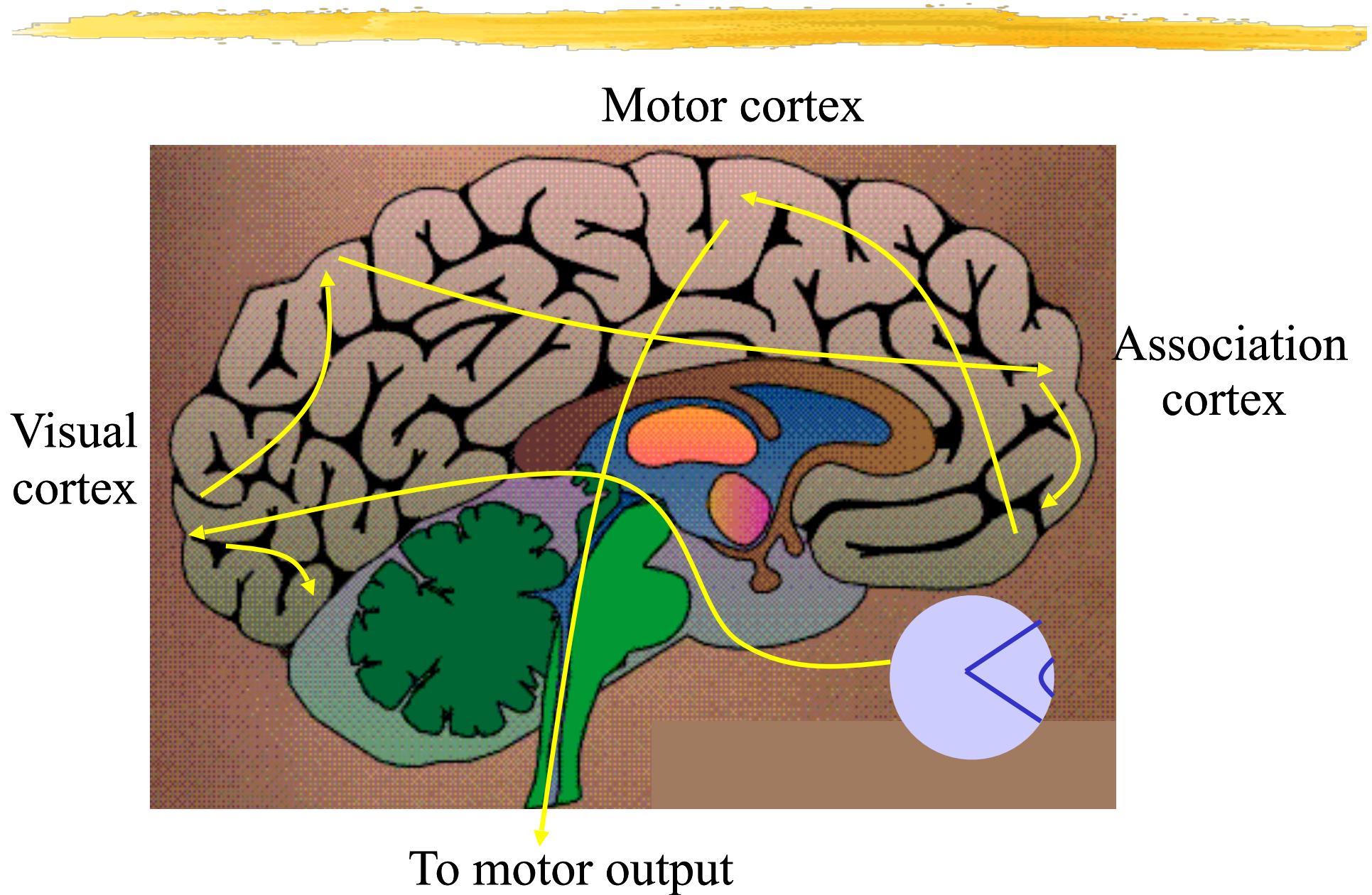
Subjects must raise their hand if they see an animal:

- 60 images
- 1 image per second

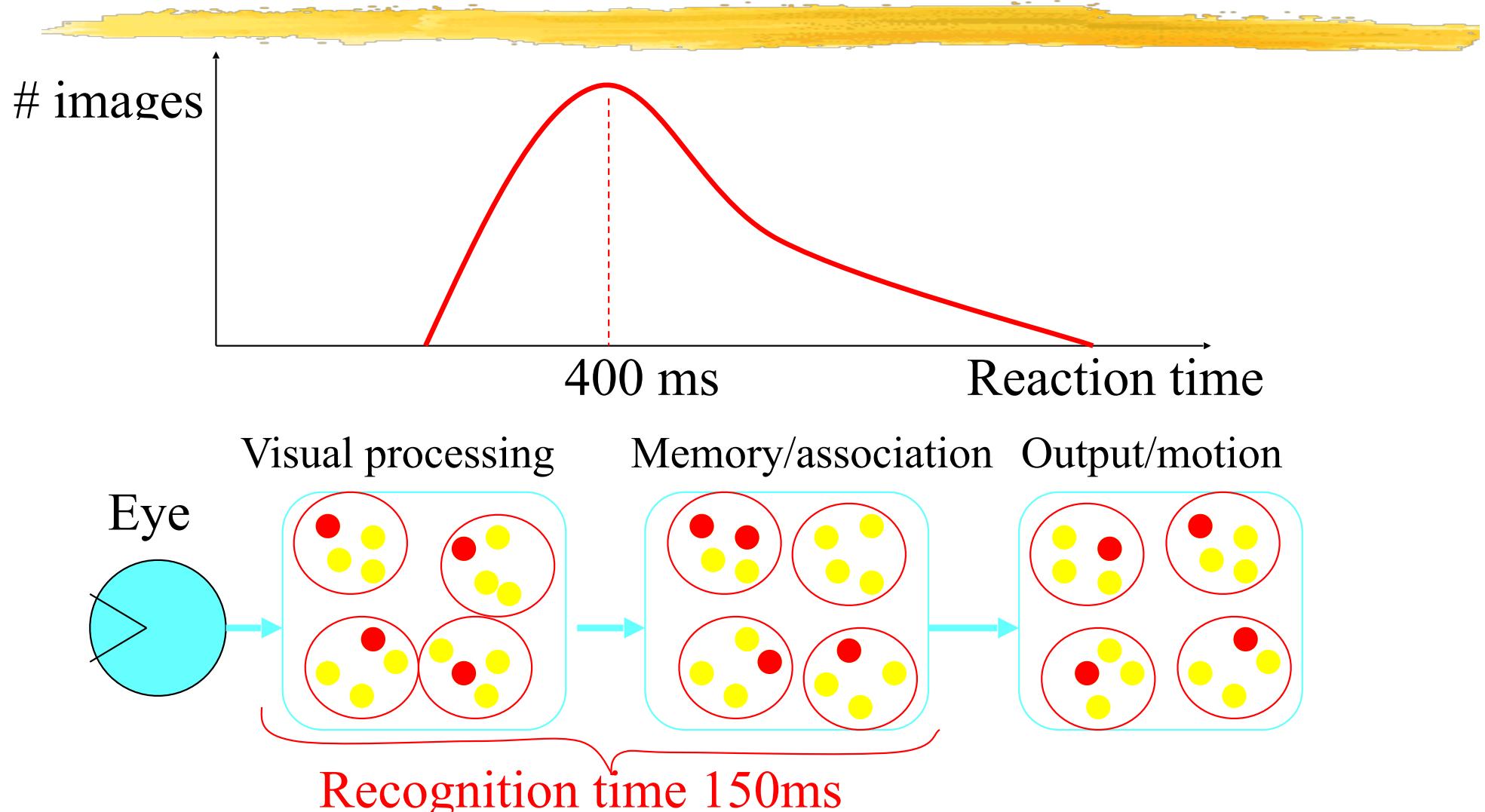
→ Measure their reaction time.



BRAIN PATHWAYS

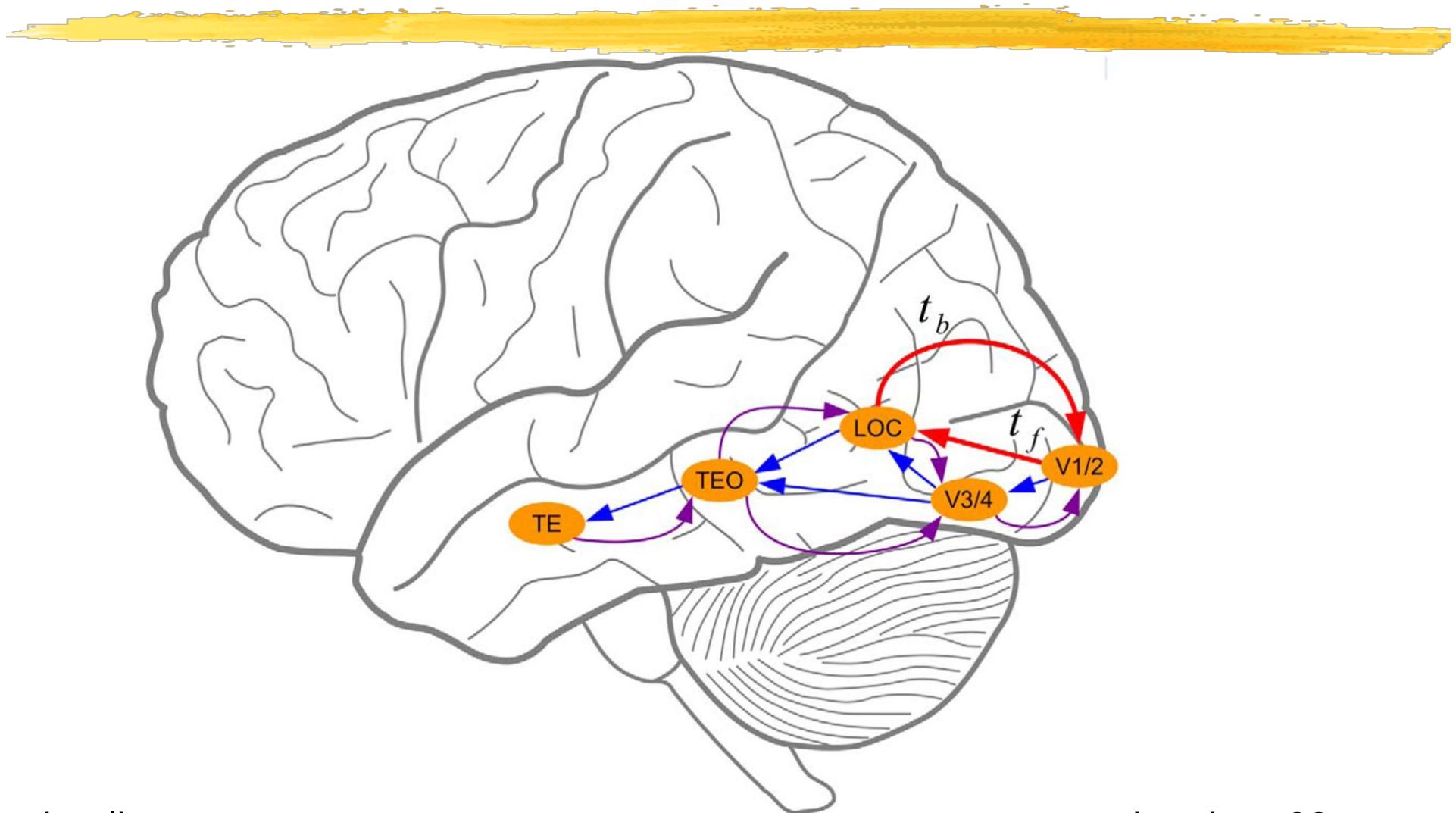


REACTION TIME



→ **Suggests** purely feed-forward processing because there is not enough time for feedback loops.

RECURRENT PATHWAYS



"Shape stimuli are optimally reinforcing each other when separated in time by ~60 ms, suggesting an underlying recurrent circuit with a time constant (feedforward + feedback) of 60 ms."

Drewes et al., *Journal of Neuroscience*, 2016

HUMAN vs COMPUTER VISION



The camera replaces the eye:

- Eye lens → Camera Optics
- Cones and Rods → CCD array
- Ganglion cells → Filter banks

The computer replaces the brain:

- But how?

COMPUTER VISION

Pascal Fua

EPFL CVLab

CH-1015 Lausanne

Switzerland

<http://cvlab.epfl.ch/~fua>

COMPUTER VISION



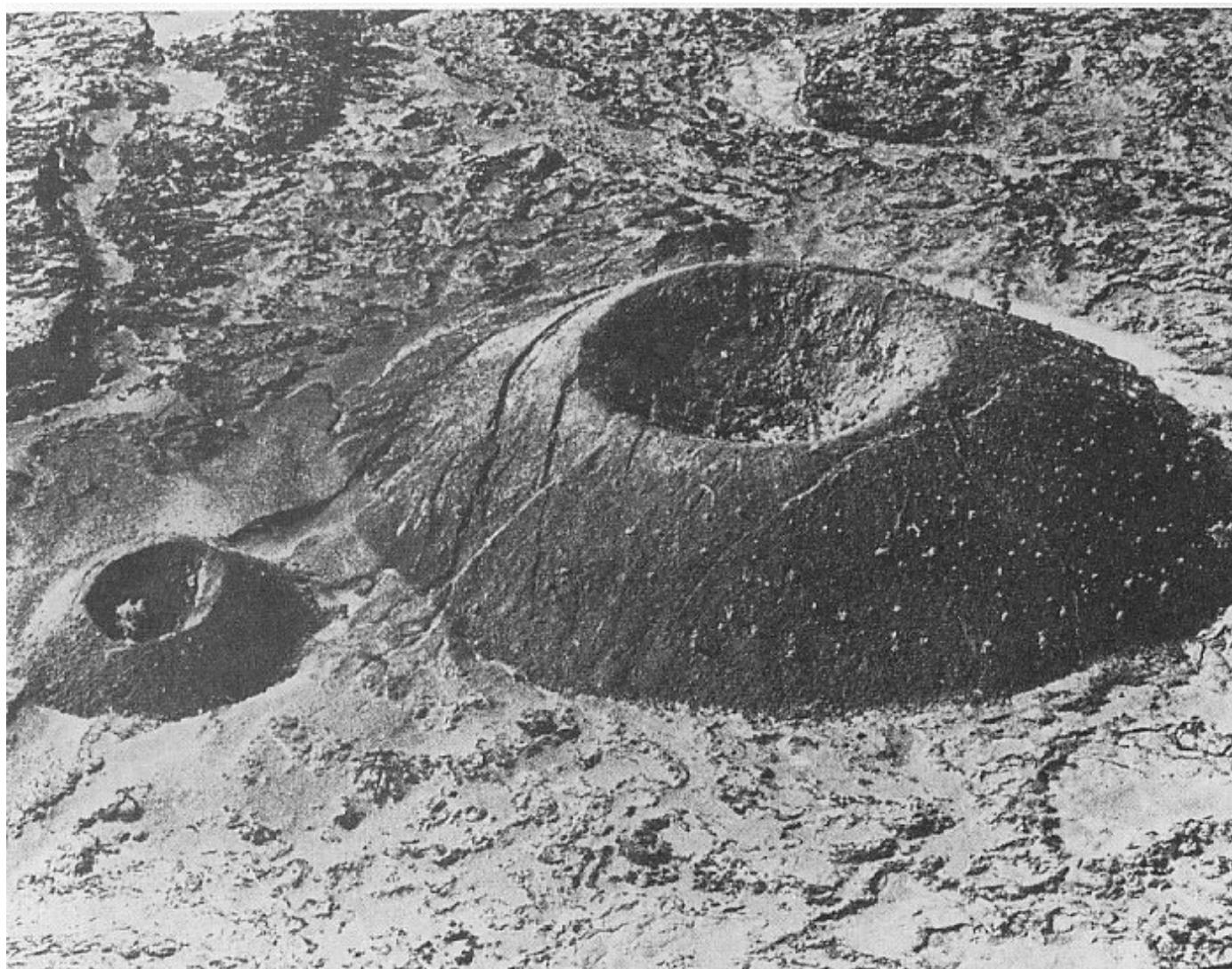
Goal: Inferring the properties of the world from one or more images

- Photographs
- Video Sequences
- Medical images
- Microscopy

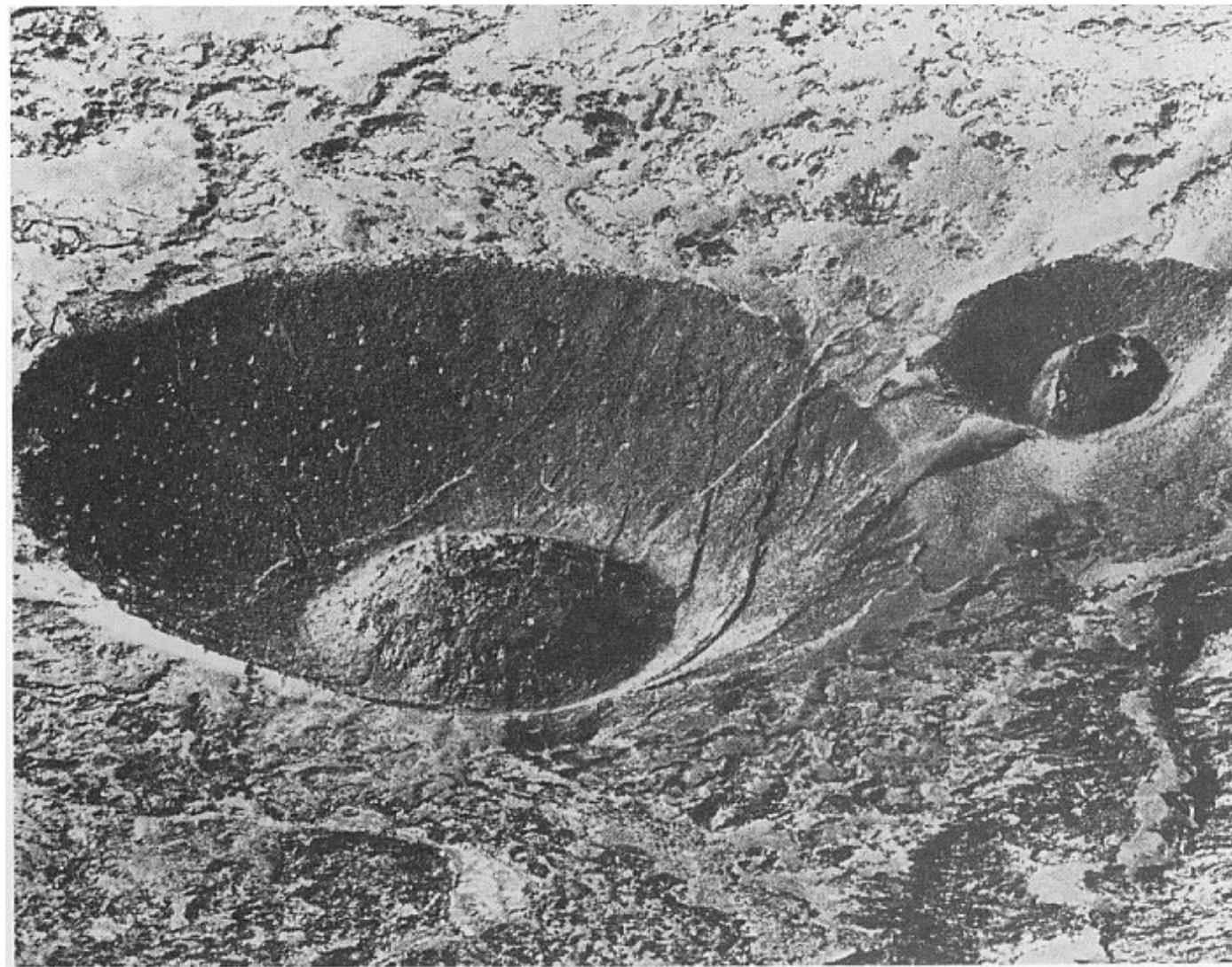


→ Image Understanding

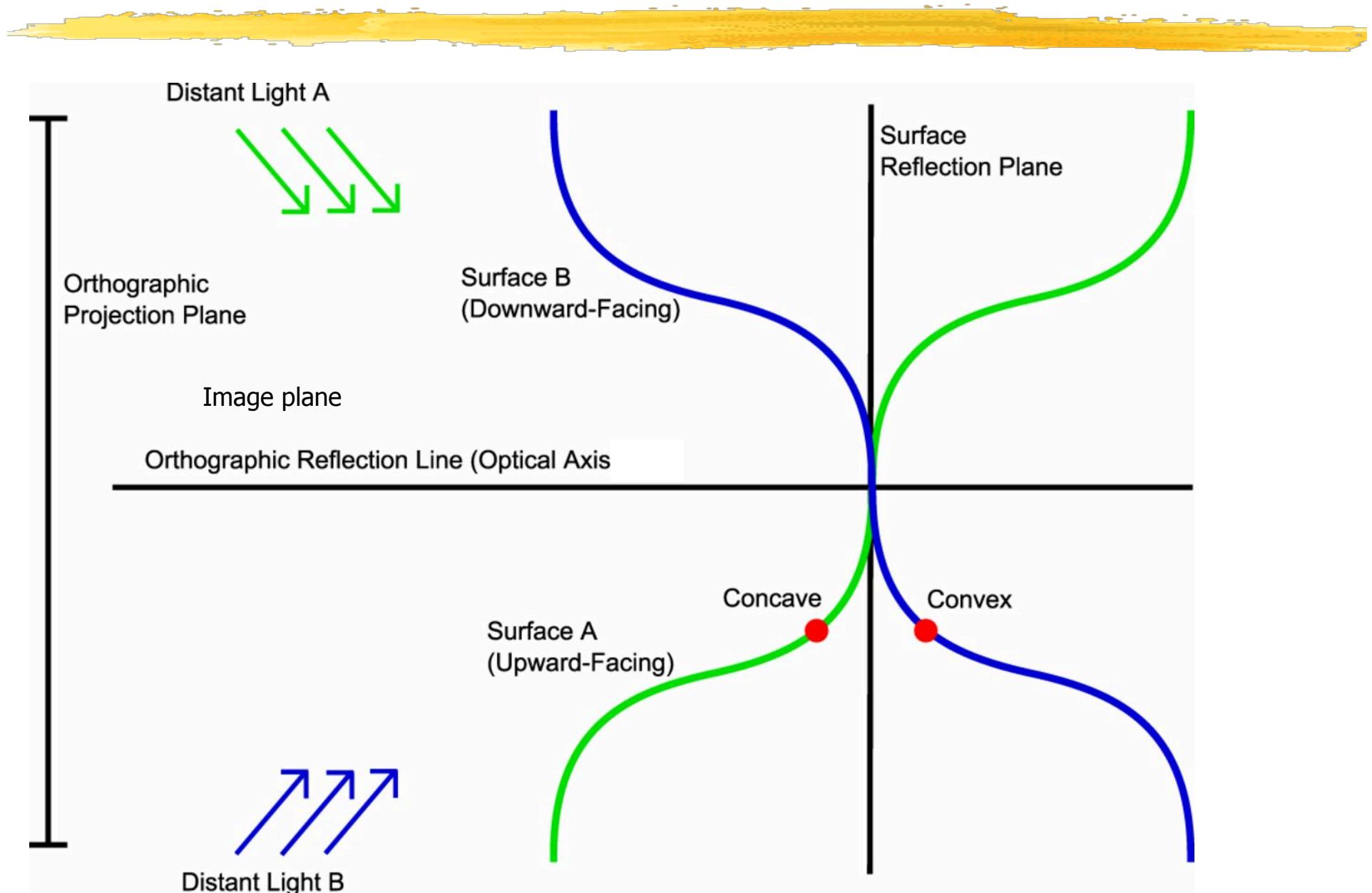
WHAT DO YOU SEE?



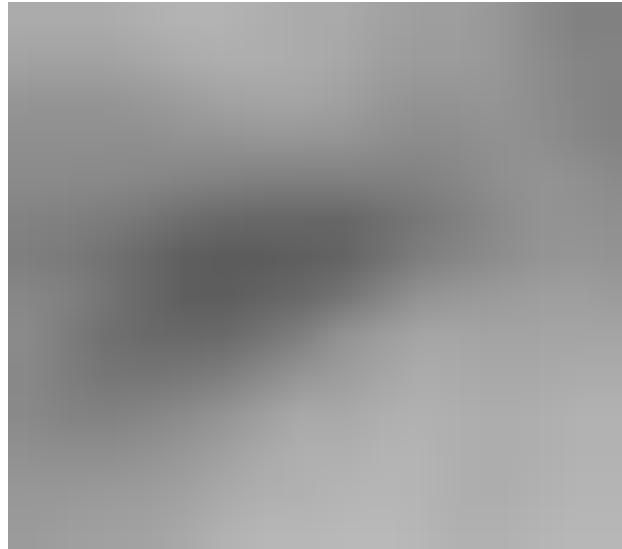
AND NOW?



CONCAVE vs CONVEX

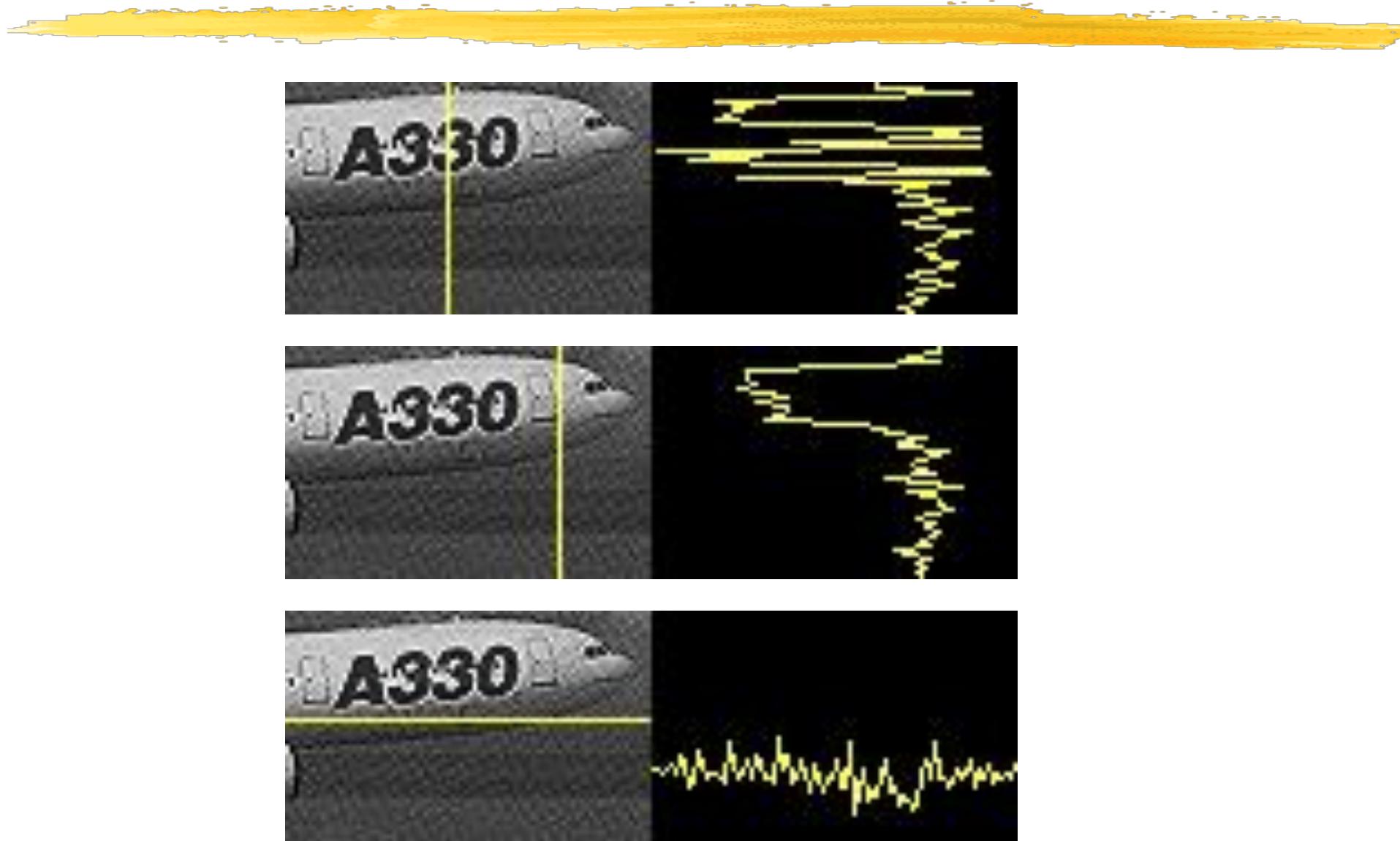


A POWERFUL MECHANISM

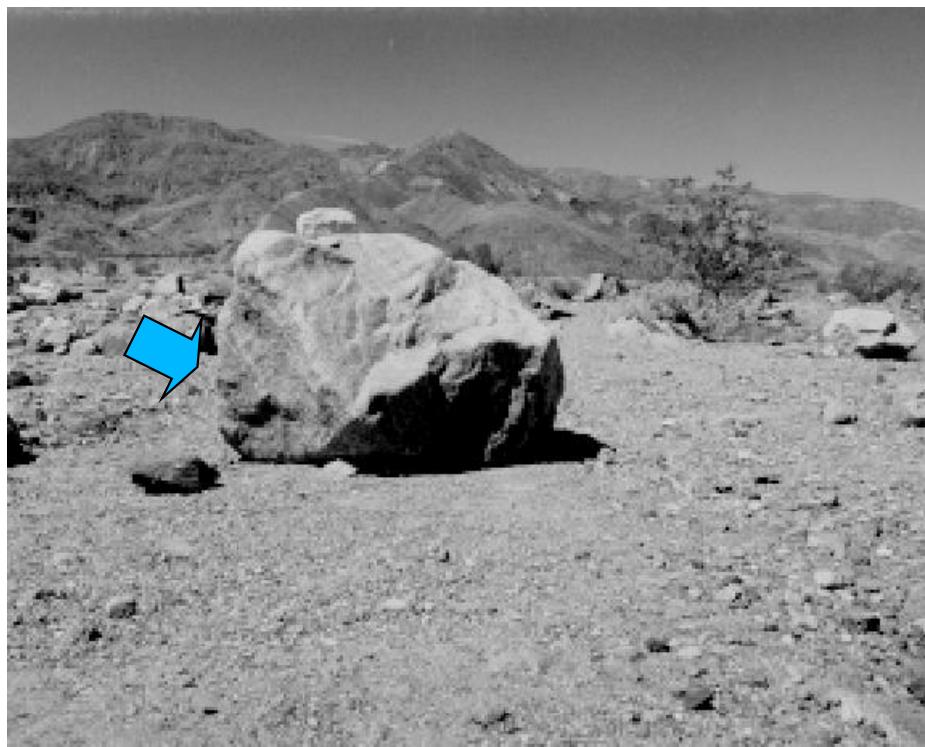


136 134 161 159 163 168 171 173 173 171 166 159 157 155
152 145 136 130 151 149 151 154 158 161 163 163 159 151
145 149 149 145 140 133 145 143 145 145 145 146 148 148
148 143 141 145 145 141 136 136 135 135 136 135 133
131 131 129 129 133 136 140 142 142 138 130 128 126 120
115 111 108 106 106 110 120 130 137 142 144 141 129 123
117 109 098 094 094 094 100 110 125 136 141 147 147 145
136 124 116 105 096 096 100 107 116 131 141 147 150 152
152 152 137 124 113 108 105 108 117 129 139 150 157 159
159 157 157 159 135 121 120 120 121 127 136 147 158 163
165 165 163 163 163 166 136 131 135 138 140 145 154 163
166 168 170 168 166 168 170 173 145 143 147 148 152 159
168 173 173 175 173 171 170 173 177 178 151 151 153 156
161 170 176 177 177 179 176 174 174 176 177 179 155 157
161 162 168 176 180 180 180 182 180 175 175 178 180 180

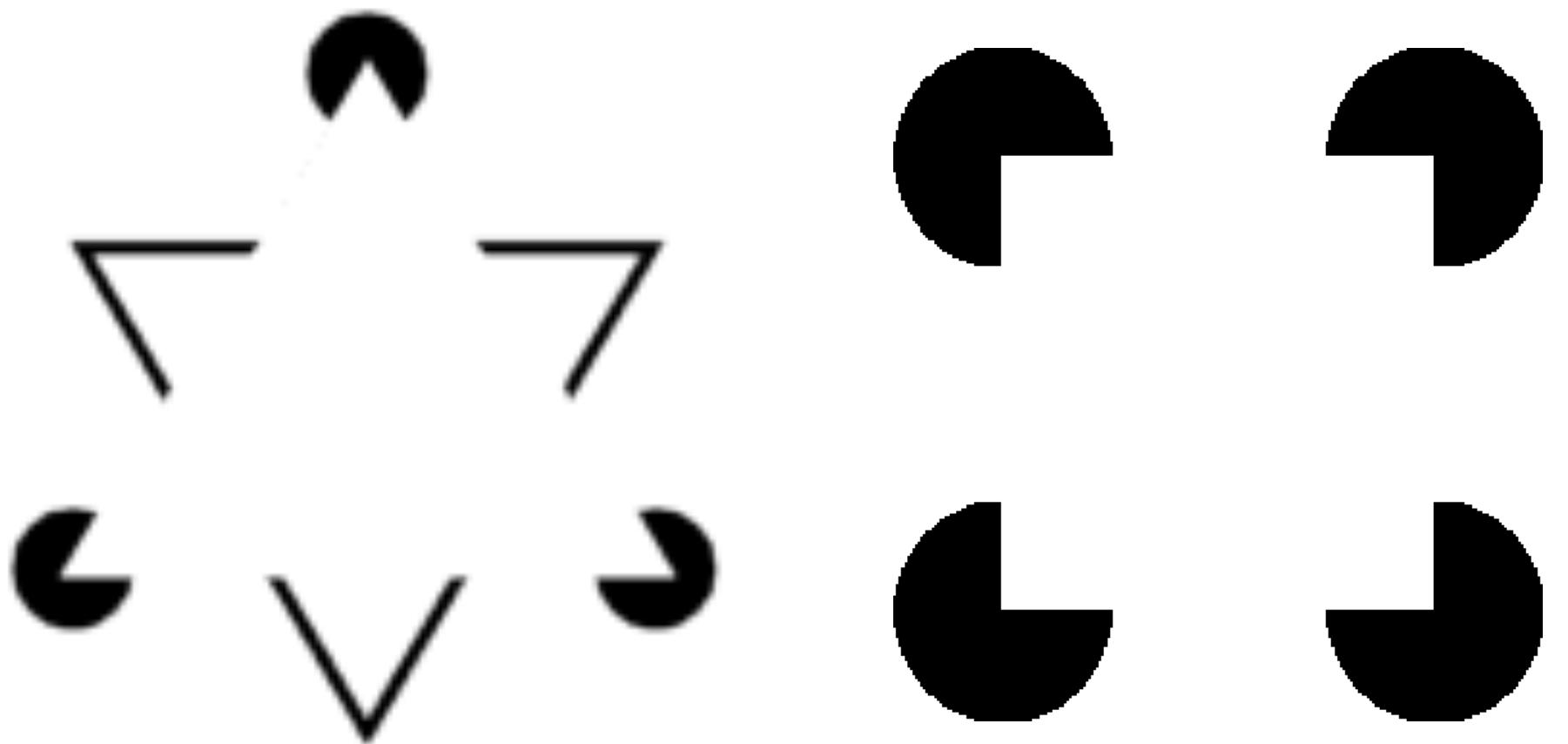
A POWERFUL MECHANISM



CONTEXT & MODELS



ILLUSORY CONTOURS



KANIZSA'S TRIANGLE



Closure of Good Form Hypothesis: Illusory contours represent an example of the closure of good form (Osgood 1953), (Pastore 1971), (Kanizsa 1976, 1979).

Figural-Cue Hypothesis: Illusory contours are responses to partial figural cues in the same way that meaning is abstracted from simple outline drawings or cartoons (Gregory 1972), (Piggins 1975), (Rock and Anson 1979).

Cues-to-Depth Hypothesis: Illusory contours are produced by the monocular depth cue of interposition to perceive a plane in depth (Coren 1972).

Organizational-Attentional Effects Hypothesis: Emphasizes that illusory contours are not totally stimulus-bound (Bradley and Dumais 1975), (Bradley and Petry 1977), (Kennedy 1976).

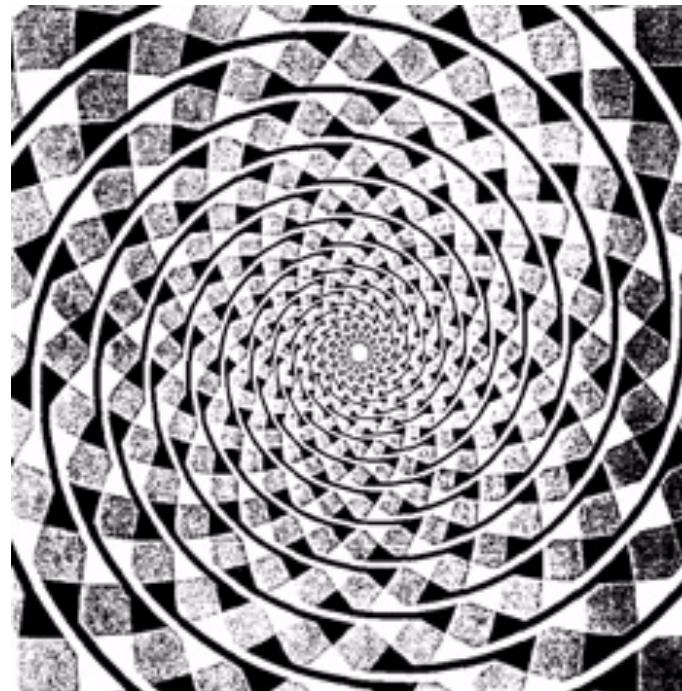
Retinal-Smearing Hypothesis: The edge effects created by the inducing areas are smeared over the retina during the course of normal eye movements to produce illusory contours (Kennedy and Chattaway 1975). This theory has been found to be untenable.

Brightness-Contrast Hypothesis: Illusory contour formation is secondary to the perception of brightness differences between the illusory figure and its background (Brigner and Gallagher 1974), (Frisby and Clatworthy 1975), (Day and Jory 1978, 1979, 1980).

Feature Analyzers Hypothesis: Illusory contours result from the partial triggering of contour-specific neural units by the physically present edge along the inducing areas (Stadler and Dieker 1972), (Smith and Over 1975, 1979). Others have suggested neural networks that generate continuous contours (filling-in contours) from discontinuous stimulus (Ullman 1976), (Grossberg and Mingolla 1985).

Spatial-Frequency-Analysis Hypothesis: Existence of a stimulus correlate for illusory contours based on a Fourier analysis of the stimulus display (Ginsburg 1975).

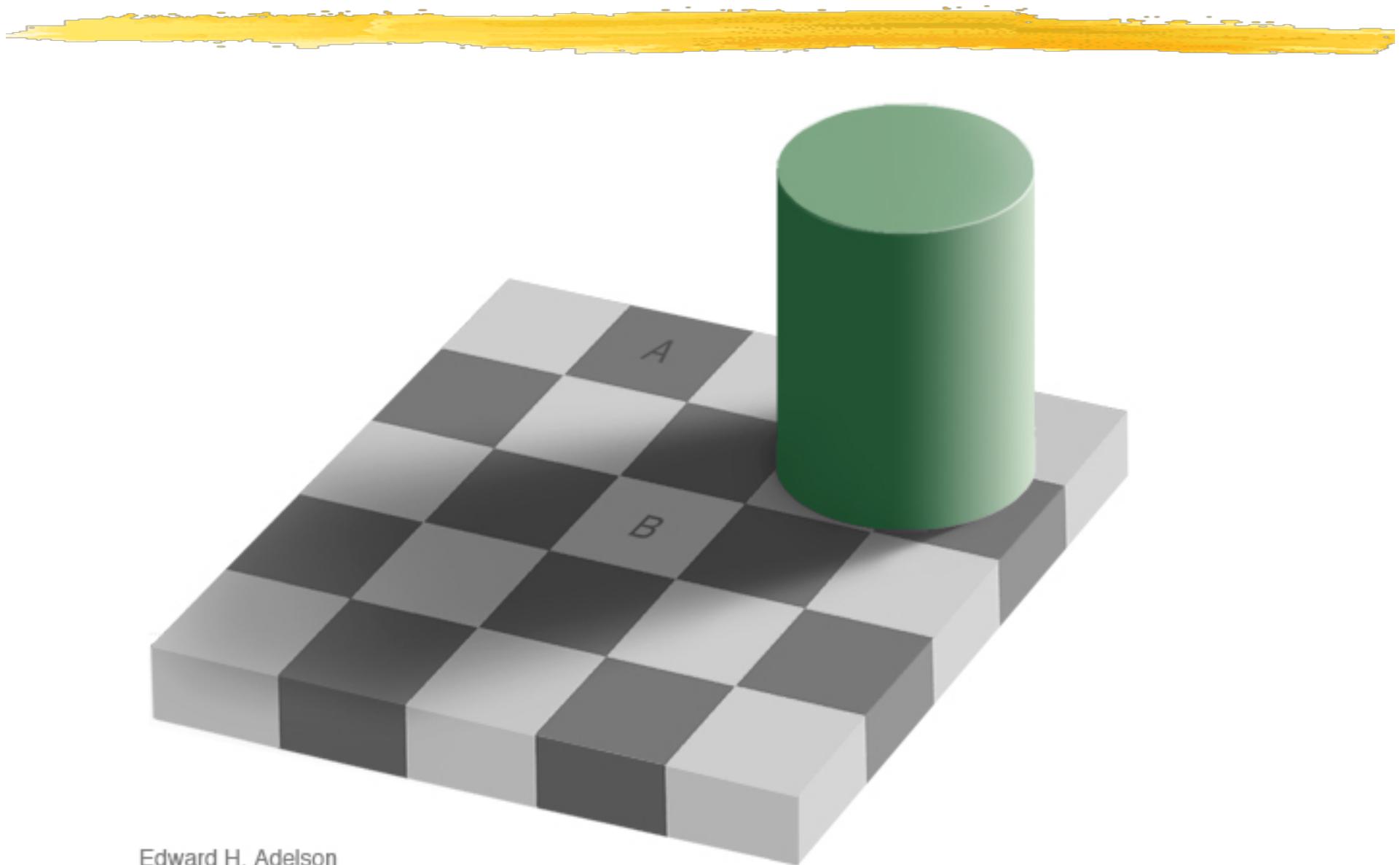
OPTICAL ILLUSIONS



Every image is the image of thing merely to him who knows how to read it, and who is enabled by the aid of the image to form an idea of the thing.

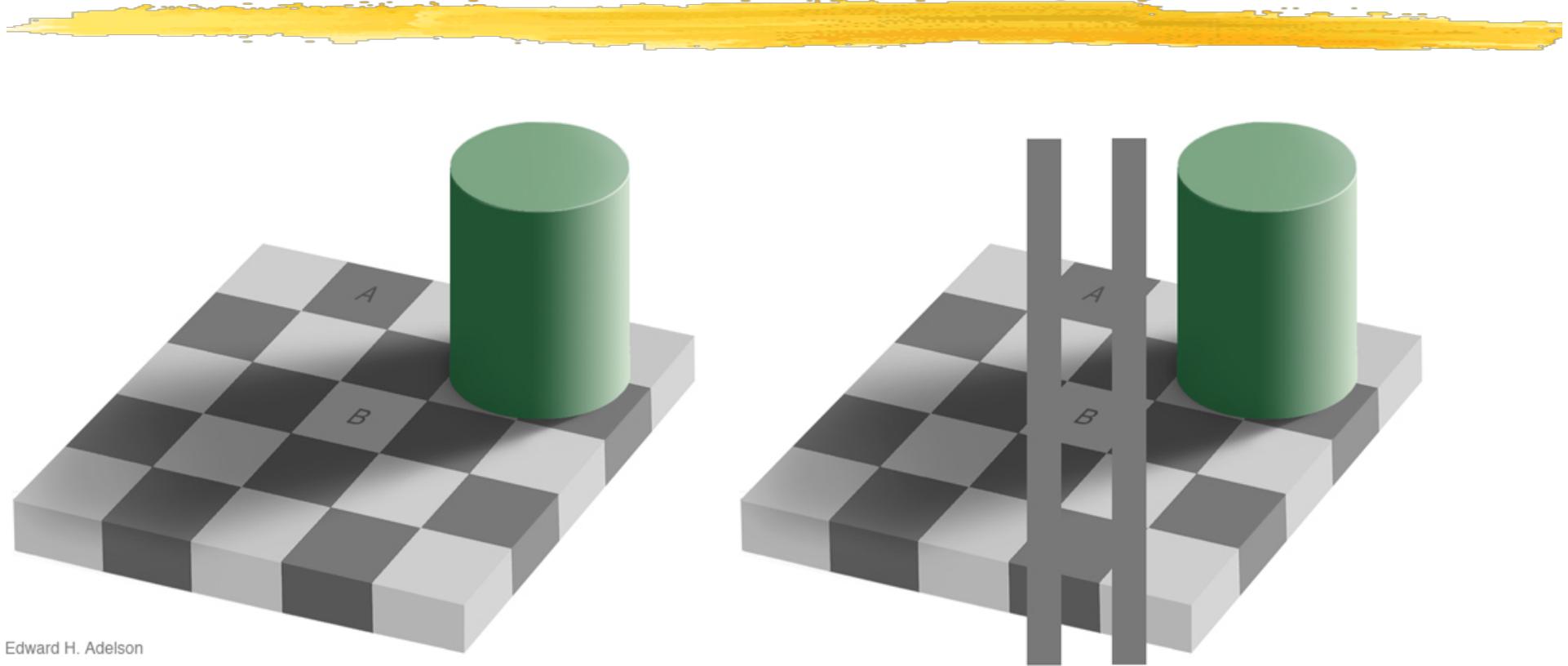
Handbook of Physiological Optics
H. von Helmholtz

PHOTOMETRIC ILLUSION



Edward H. Adelson

NO MORE ILLUSION



Edward H. Adelson

The human eye measures relative rather than absolute intensity values.

CHALLENGES



Vision involves dealing with:

- Noisy images
- Many-to-one mapping
- Aperture problem

→ Requires:

- Assumptions about the world
- Object models
- Training data

COMPUTERS vs HUMANS



True image understanding seems to involve a great deal of human intelligence:

- Automated systems are still very far from achieving human performances;
- But can be very effective in a sufficiently constrained context.

→ **Good interfaces key to effective systems**

APPLICATIONS



Cartography:

- Maps from aerial and satellite images

Robotics:

- Autonomous navigation
- Visual servoing

Industrial inspection

- Quality control

Security applications

- Access control
- Surveillance

Databases

- Retrieval and Annotation

Medical Imagery

- Microscopy

CARTOGRAPHY ON MARS



CARTOGRAPHY ON EARTH

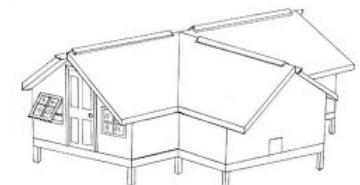


Image © 2010 TerraMetrics
Image © 2010 DigitalGlobe

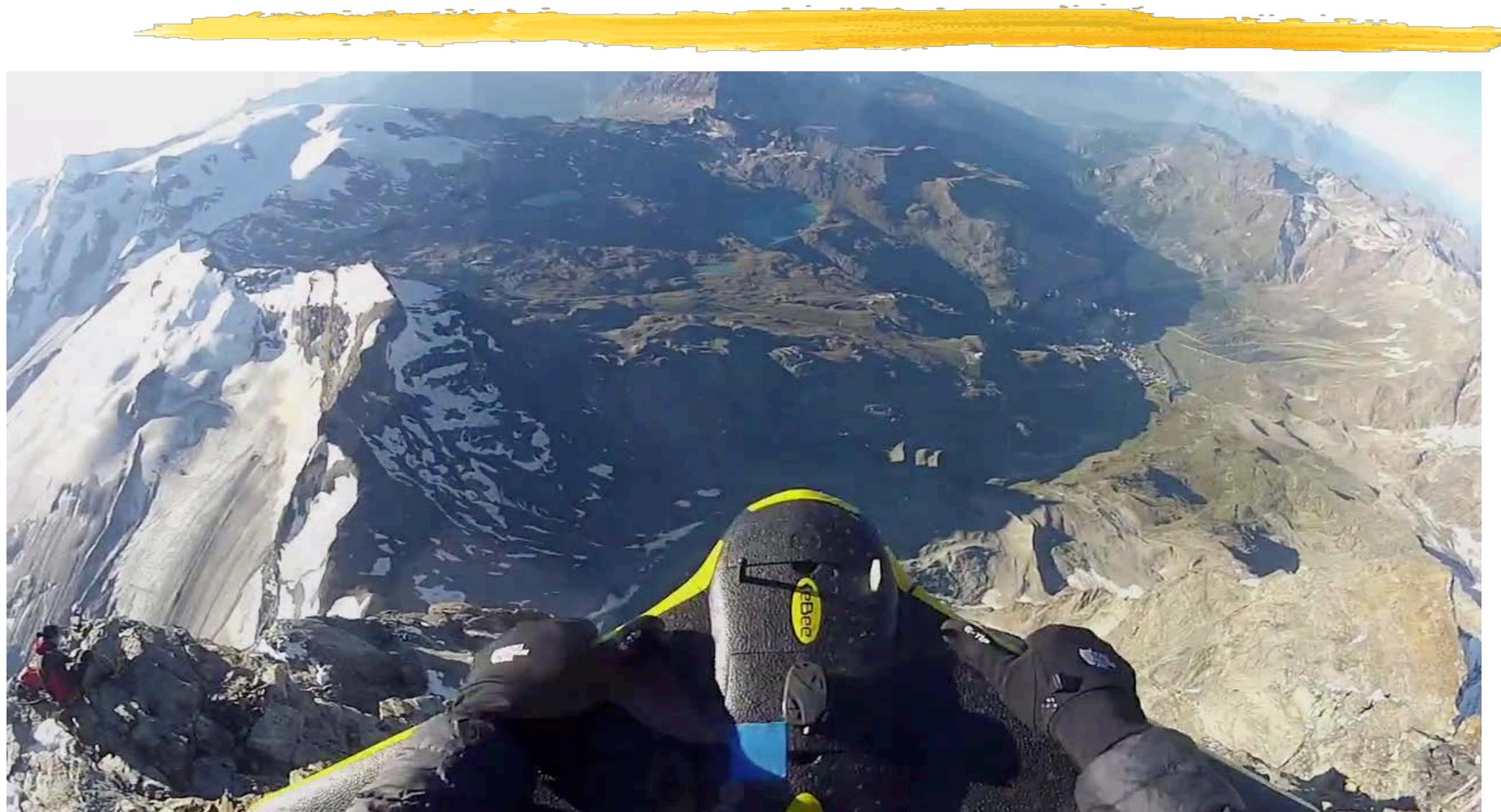
Data SIO, NOAA, U.S. Navy, NGA, GEBCO

Google
©2010

ACQUISITION PLATFORMS



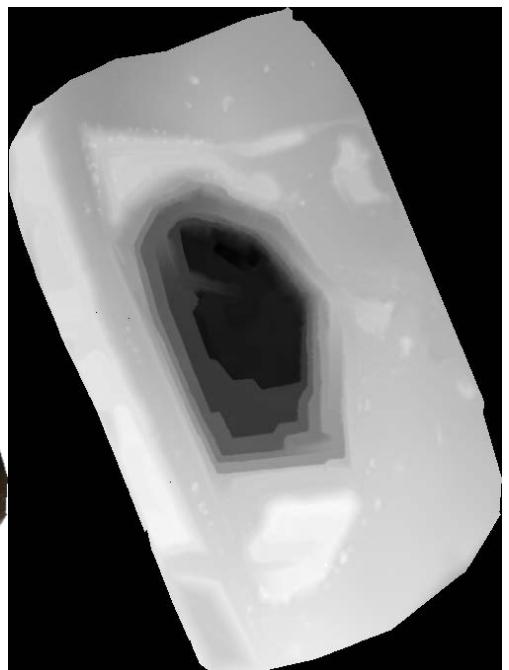
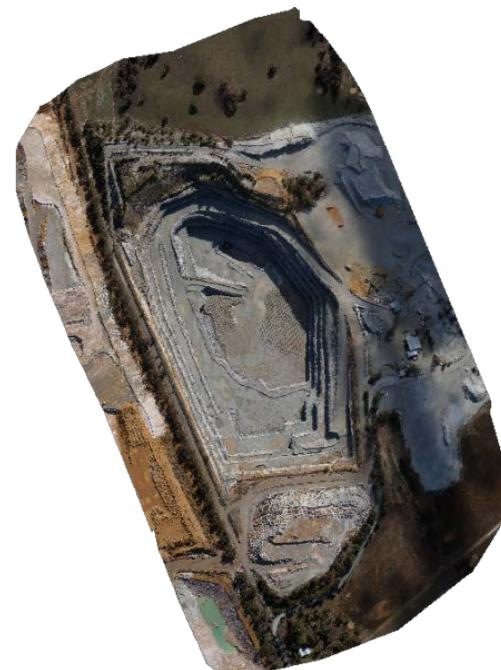
VIRTUAL MATTERHORN



Drone: <https://www.sensefly.com>

Mapping: <http://pix4d.com/>

MINING SITE



- Fully automated.
- Accurate.
- Inexpensive.

GCP statistics

	X[m]	Y[m]	Z[m]
RMS	0.086	0.074	0.053
σ	0.040	0.061	0.053

SAILS AND WINGS



Useful for

- analysis of structural behavior under realistic strains,
- guiding design choices.

APPLICATIONS



Cartography:

- Maps from aerial and satellite images

Robotics:

- Autonomous navigation
- Visual servoing

Industrial inspection

- Quality control

Security applications

- Access control
- Surveillance

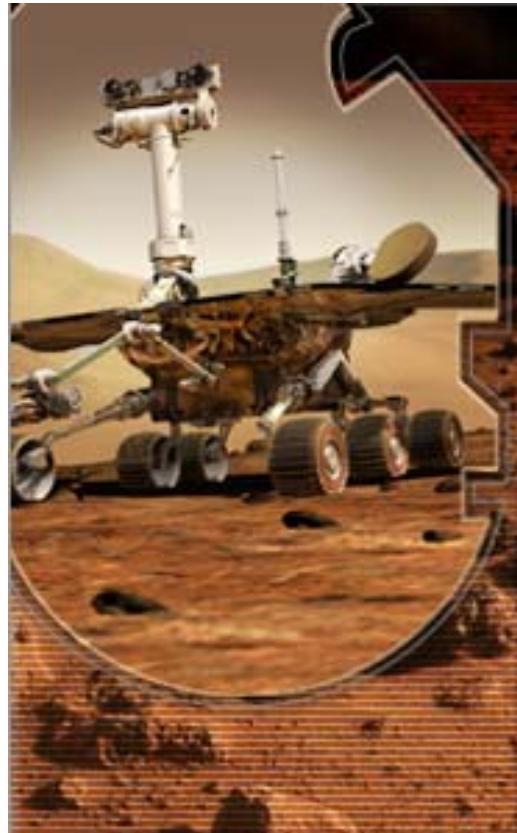
Databases

- Retrieval and Annotation

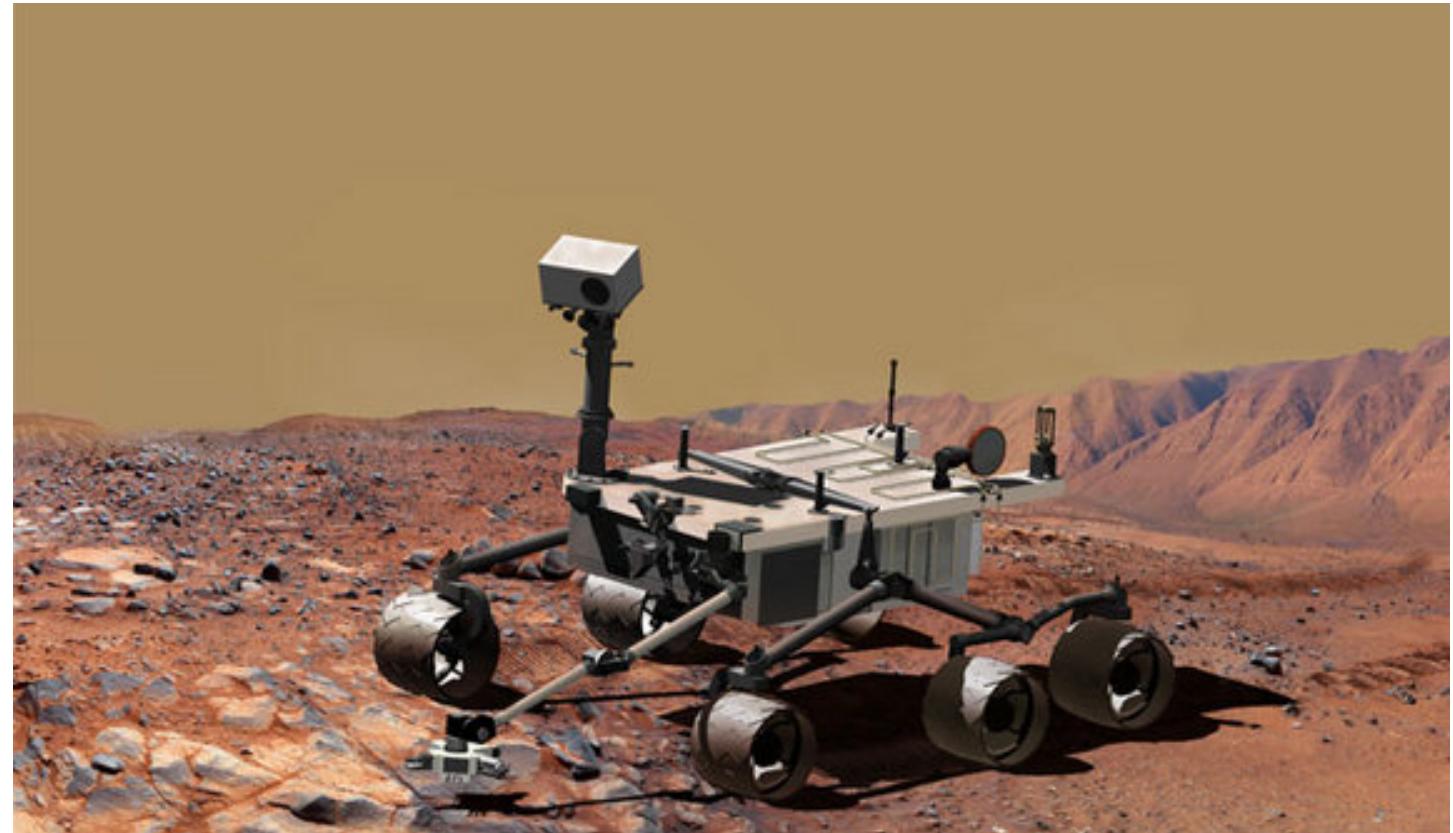
Medical Imagery

- Microscopy

MARS ROVERS

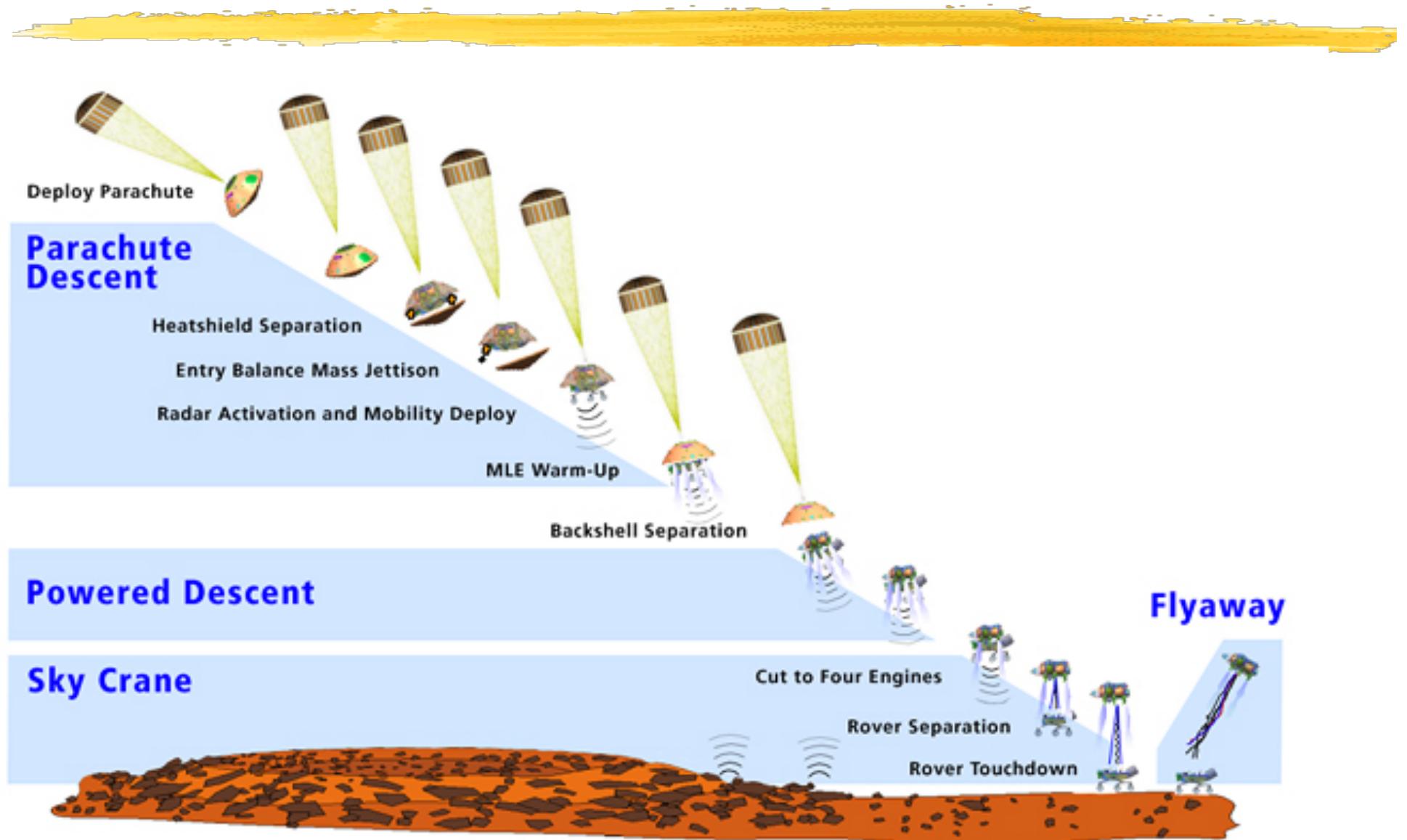


Opportunity
Launched 2003
Landed 2004



Curiosity
Launched 2011
Landed 2012

DESCENT PROFILE



POWERED DESCENT



- Track feature points.
- Estimate drift.
- Combine with IMU output.
- Fire retro rockets as needed.

- Relatively simple computations.
- Space hardened hardware now fast enough.

EARTH ROVERS



1985
DARPA ALV



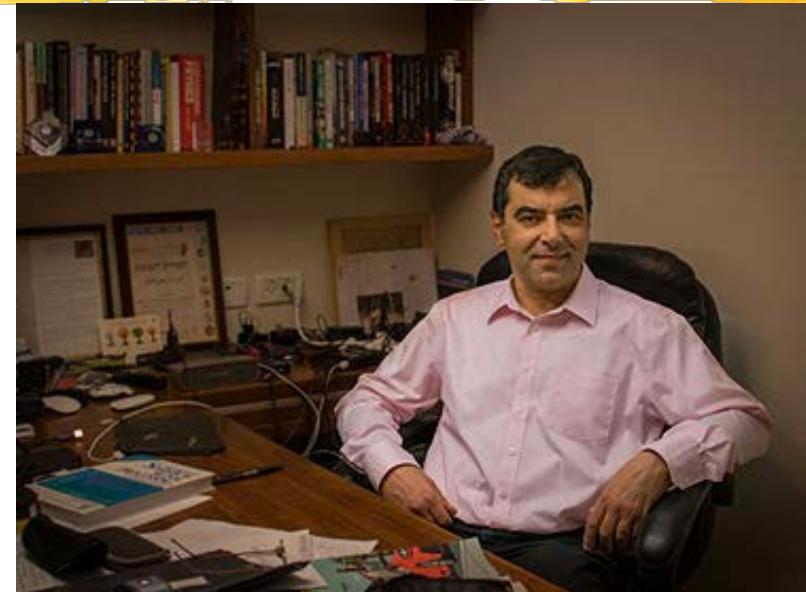
2007
DARPA Urban Challenge



2016
Google Cars

- Much more computing power.
- More reliable sensors.
- Detailed maps and models of the environment.

MOBILEYE



A. Shashua, co-founder 1999.

Intel Buys Mobileye in \$15.3 Billion Bid to Lead Self-Driving Car Market.

NEW YORK TIMES, 13/03/17

APPLICATIONS



Cartography:

- Maps from aerial and satellite images

Robotics:

- Autonomous navigation
- Visual servoing

Industrial inspection

- Quality control

Security applications

- Access control
- Surveillance

Databases

- Retrieval and Annotation

Medical Imagery

- Microscopy

VISUAL INSPECTION OF ASSEMBLED DEVICES



Software embedded in the camera to find and read serial numbers

- Localization
- Illumination changes
- Generality

APPLICATIONS



Cartography:

- Maps from aerial and satellite images

Robotics:

- Autonomous navigation
- Visual servoing

Industrial inspection

- Quality control

Security applications

- Access control
- **Surveillance**

Databases

- Retrieval and Annotation

Medical Imagery

- Microscopy

LICENCE PLATES

Appian Technologies PLC



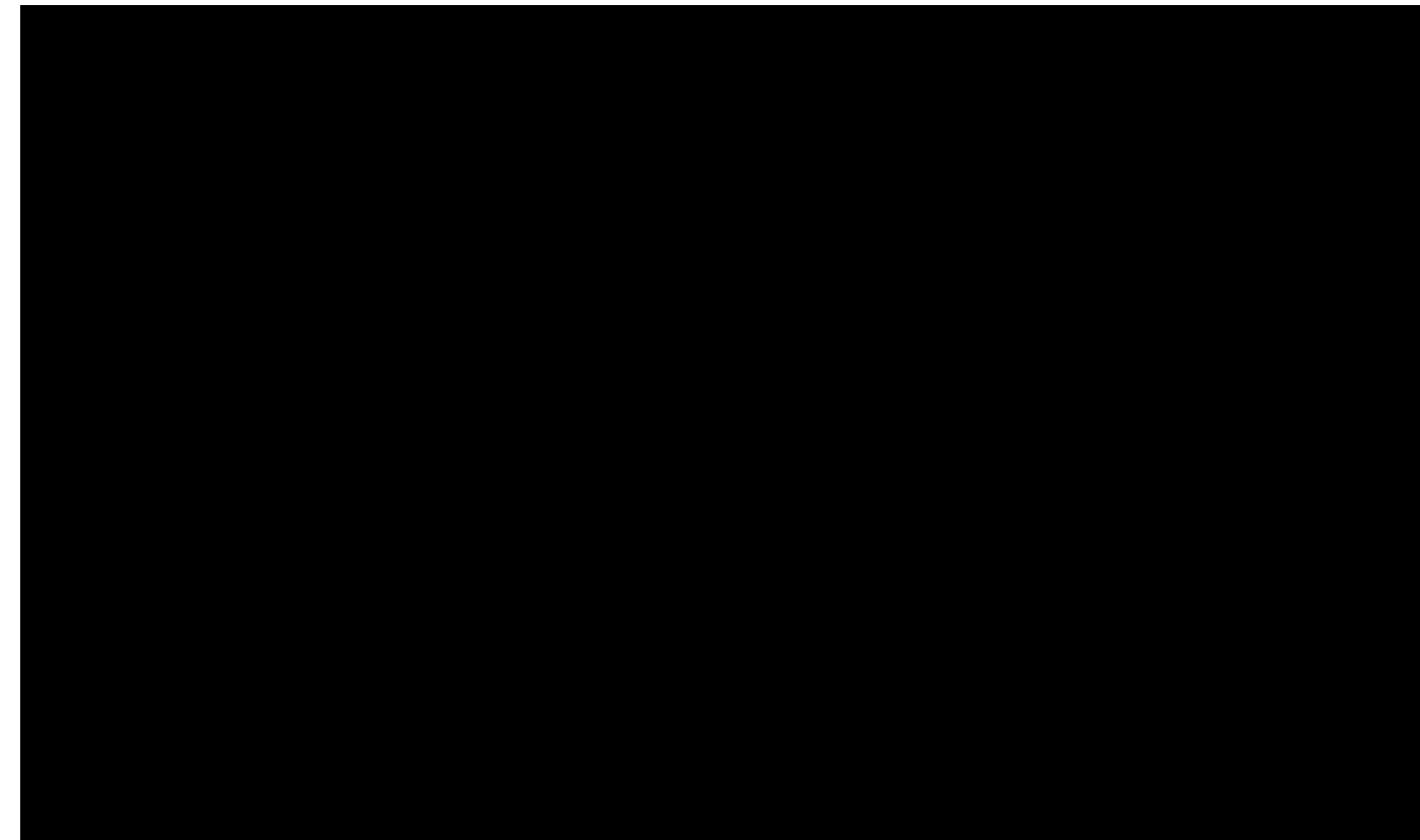
http://www.appian-tech.com/products/anpr_overview.html

TRACKING PEOPLE



... and the ball → Behavioral analysis.

TECH TRANSFER



APPLICATIONS



Cartography:

- Maps from aerial and satellite images

Robotics:

- Autonomous navigation
- Visual servoing

Industrial inspection

- Quality control

Security applications

- Access control
- Surveillance

Databases

- Retrieval and Annotation

Medical Imagery

- Microscopy

IMAGE RETRIEVAL



Google Images Aerie...-2-966.jpg × федеральная политехническая школа лозанна

All **Images** Maps More Settings Tools

About 25,270,000,000 results (0.78 seconds)

 Image size:
402 × 186
No other sizes of this image found.

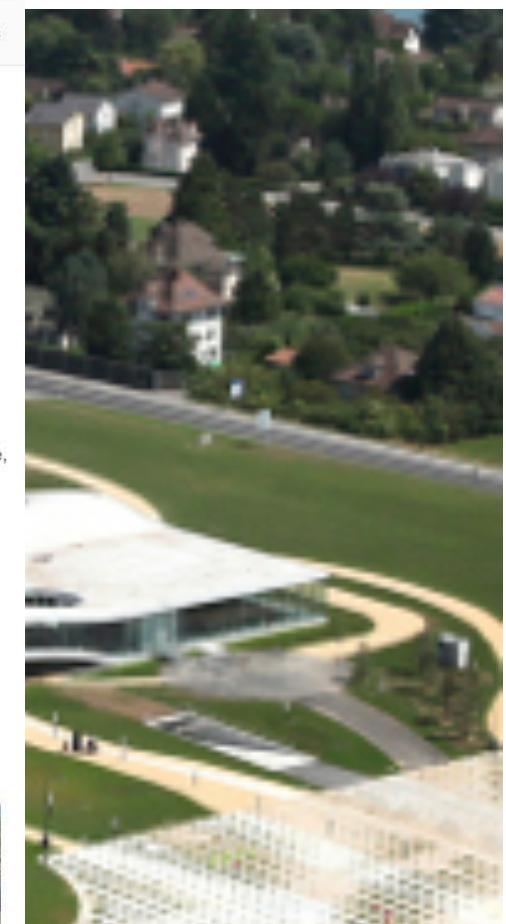
Best guess for this image: [федеральная политехническая школа лозанны](#)

École Polytechnique Fédérale de Lausanne - Wikipedia
https://en.wikipedia.org/wiki/École_Polytechnique_Fédérale_de_Lausanne ▾
The École polytechnique fédérale de Lausanne (EPFL) is a research institute and university in Lausanne, Switzerland, that specializes in natural sciences and engineering. It is one of the two Swiss Federal Institutes of Technology, and it has three ... The environment at modern day EPFL is highly international with the school ...

Федеральная политехническая школа Лозанны (EPFL) - École ...
<https://www.educationindex.ru/.../ecole-polytechnique-federale-de-l...> ▾ [Translate this page](#)
Федеральная политехническая школа Лозанны (EPFL) — один из двух политехнических институтов в Швейцарии. Наш университет является самым ...

Visually similar images

A grid of 12 smaller images showing various views of the EPFL campus and its surrounding urban environment, used to demonstrate visual similarity search results.



APPLICATIONS



Cartography:

- Maps from aerial and satellite images

Robotics:

- Autonomous navigation
- Visual servoing

Industrial inspection

- Quality control

Security applications

- Access control
- Surveillance

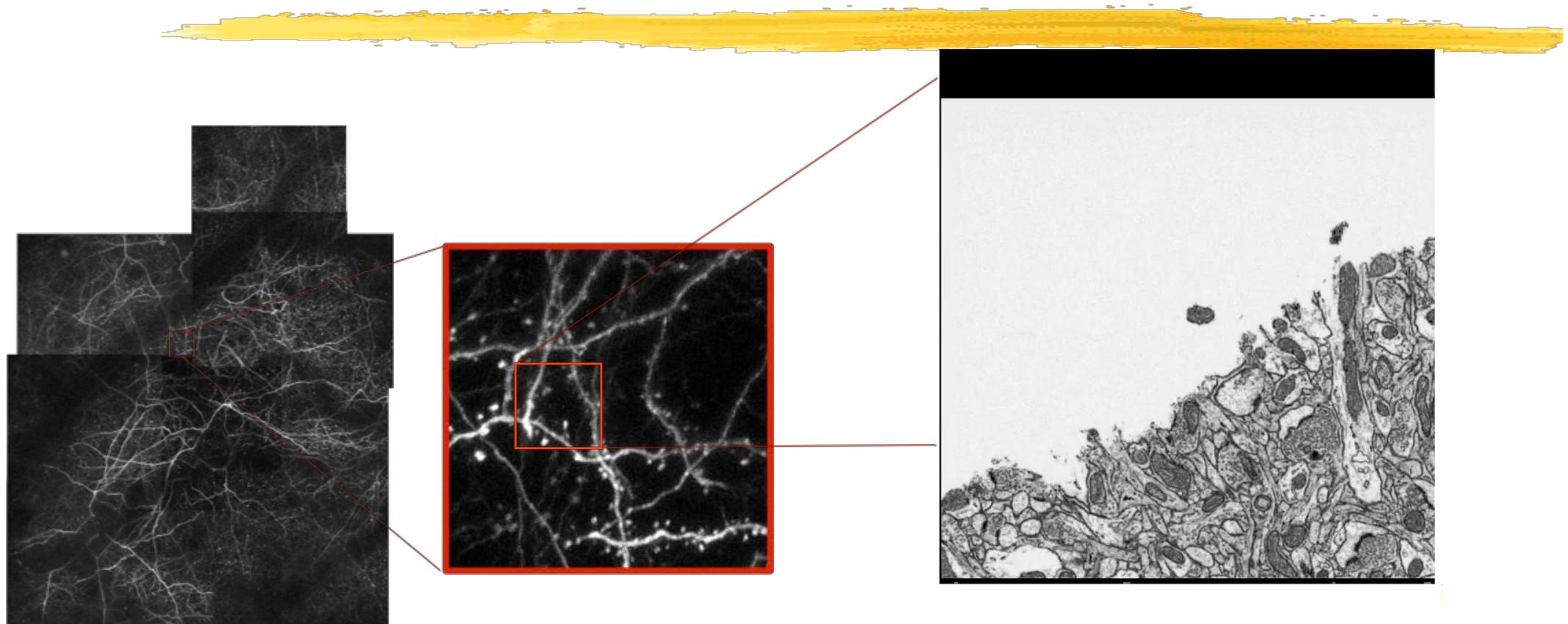
Databases

- Retrieval and Annotation

Medical Imagery

- Microscopy

MICROSCOPY

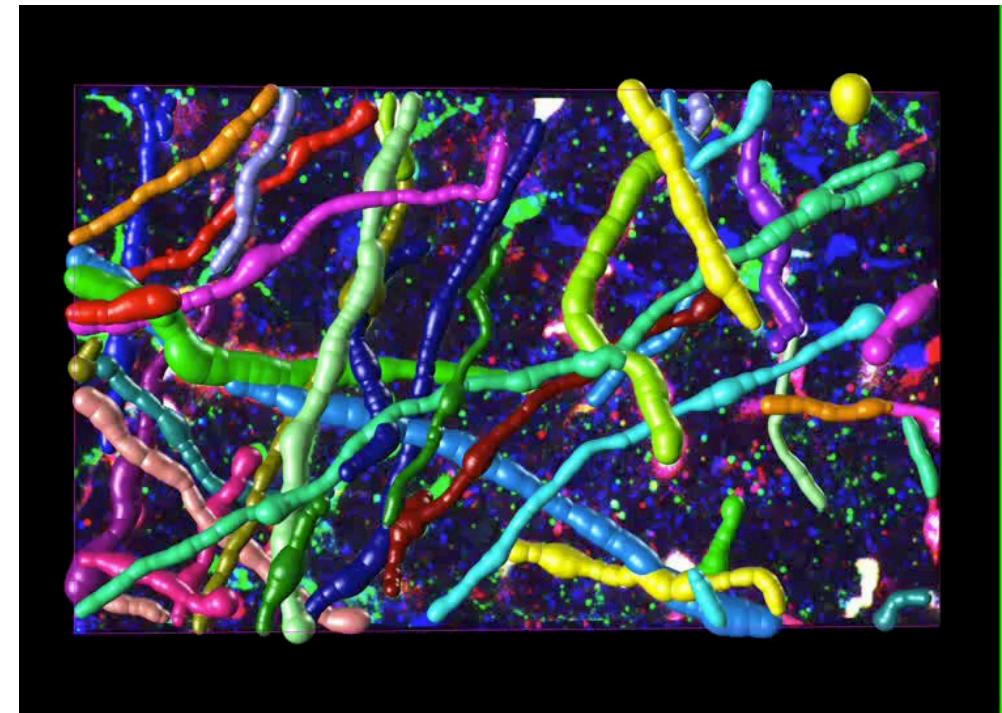
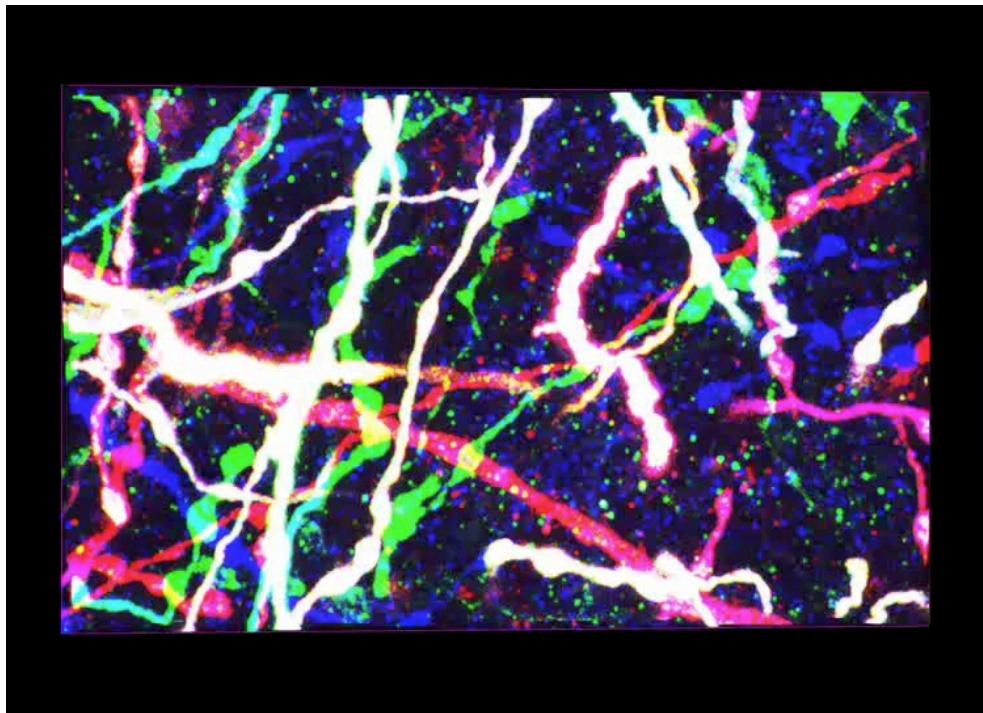


Fluorescent neurons *in vivo* in the adult mouse brain Imaged through a cranial window using a 2-photon microscope.

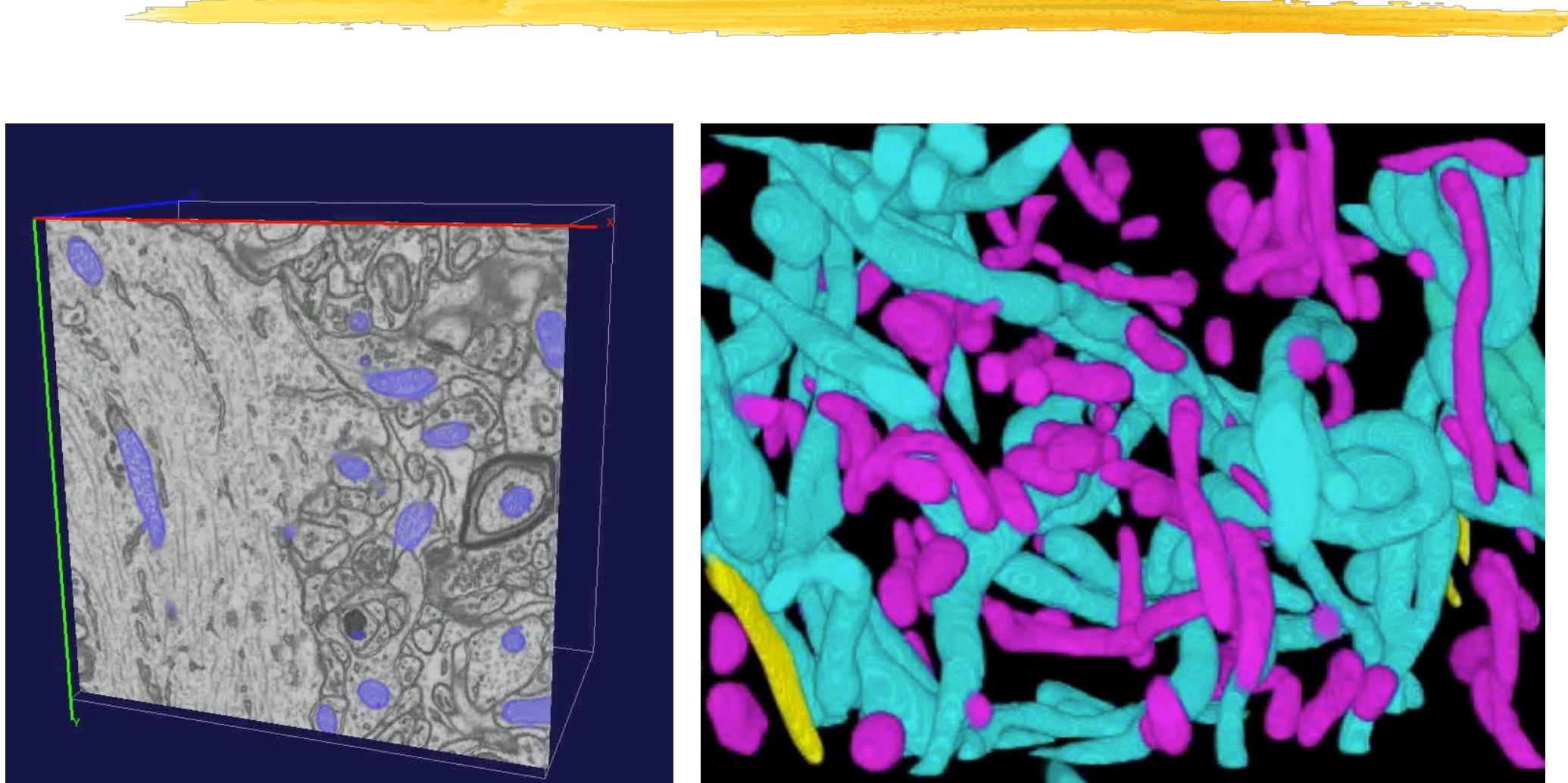
Electron Microscopy Image Stack at five nanometer resolution.

Courtesy of G. Knott

DELINEATING DENDRITIC TREES



FINDING MITOCHONDRIA



GOOGLE EARTH FOR THE BRAIN



- A human brain contains approximately 100 billion neurons and 100 trillion synapses.
- It would take 1000 Exabytes to store an uncompressed digitization at 5nm resolution.

amazon x 500!

→ Seriously big data!

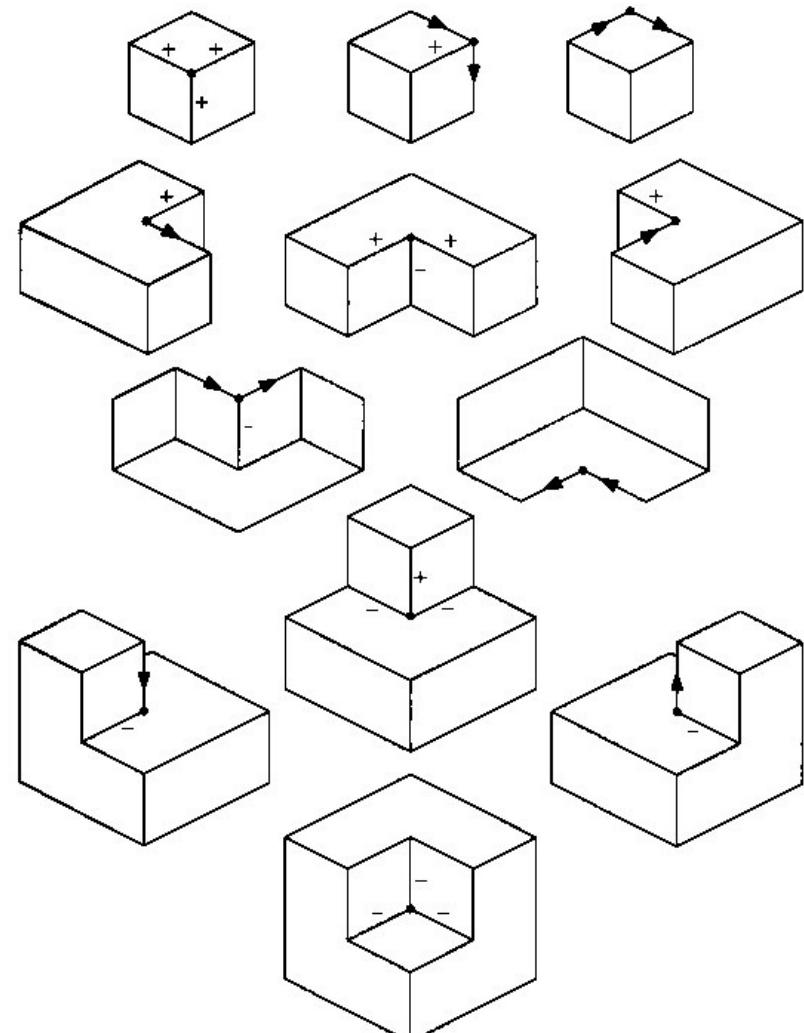
KINECT



→ Image whose pixel values are distances

HOW IT BEGAN

- Computer Vision started in 1965 at MIT as a short term project.
 - A world of perfect blocks and strong assumptions.
- The real world is not like that!



HISTORICAL PERSPECTIVE



1960s: Beginnings in artificial intelligence, image processing and pattern recognition.

1970s: Foundational work on image formation.

1980s: Vision as applied mathematics, geometry, multi-scale analysis, control theory, optimization.

1990s: Physics-based models, Probabilistic reasoning.

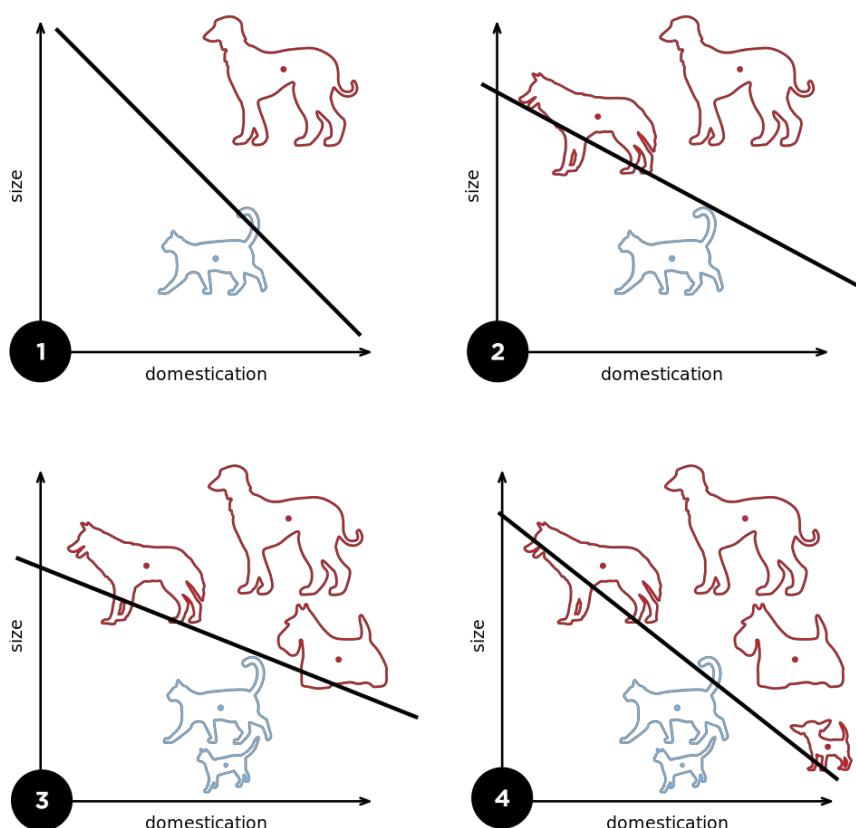
2000s: Machine learning.

2010s: Deep Learning.

2020s: ?????

--> Improved understanding and successful applications in graphics, mapping, biometrics, and others but still far from human performance.

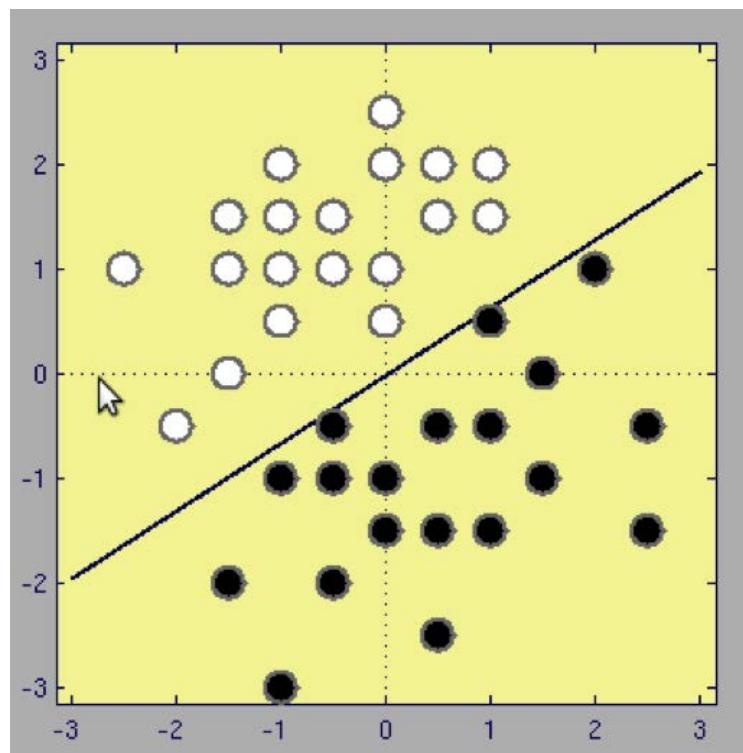
LINEAR CLASSIFICATION



$$f(\mathbf{x}; \mathbf{w}, b) = \begin{cases} 1 & \text{if } \mathbf{w} \cdot \mathbf{x} + b \geq 0, \\ 0 & \text{otherwise.} \end{cases}$$

Learning: Finding \mathbf{w} and b so that the number of misclassified instances in the training set is minimized.

PERCEPTRON



- Center the \mathbf{x}_i s so that $b = 0$.
- Pick a random \mathbf{w}_0 .
- Iteratively, pick a random index i .
 - If \mathbf{x}_i is correctly classified, do nothing.
 - Otherwise, $\mathbf{w}_{t+1} = \mathbf{w}_t + y_i \mathbf{x}_i$.

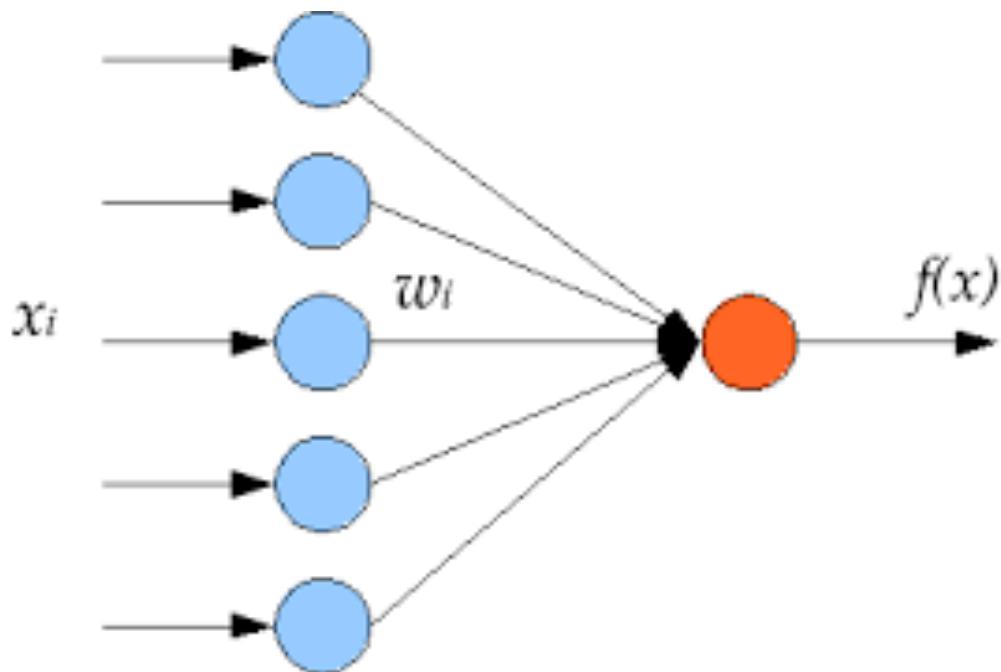
MORE ANCIENT HISTORY

- The perceptron is a simple algorithm, but imagine coding it on this IBM 704, which Frank Rosenblatt used to implement it in 1957.

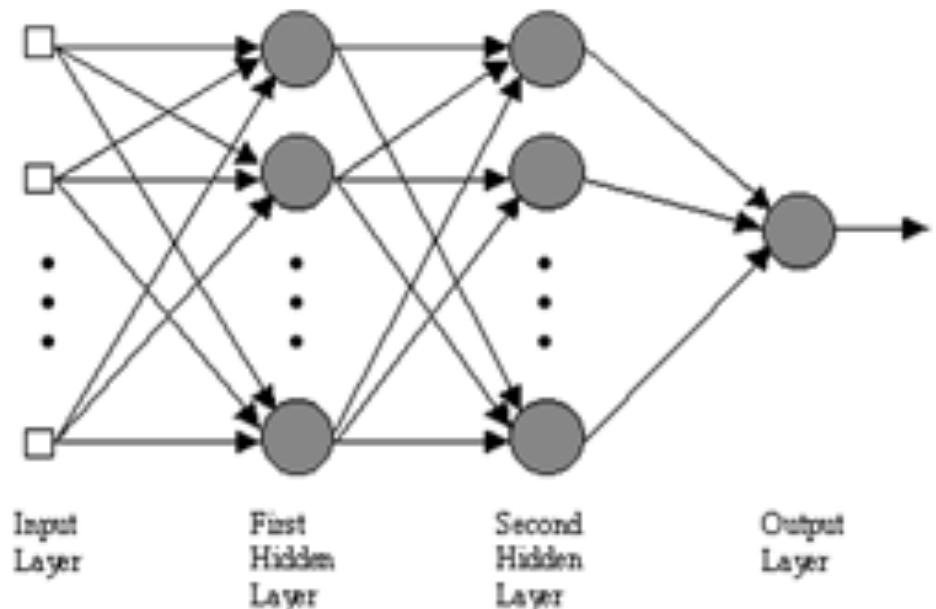


- There was much initial enthusiasm. But, it was later realized there were serious limitations, such as the linear separability requirement.
- The perceptron eventually evolved to have multiple layers and smooth activation functions.

LAYERED PERCEPTRONS



Single Layer

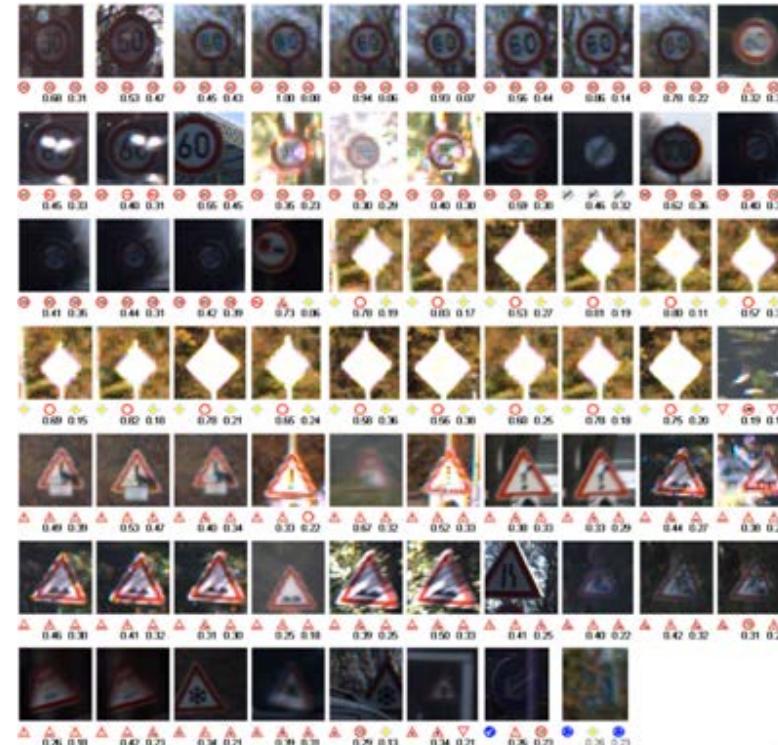
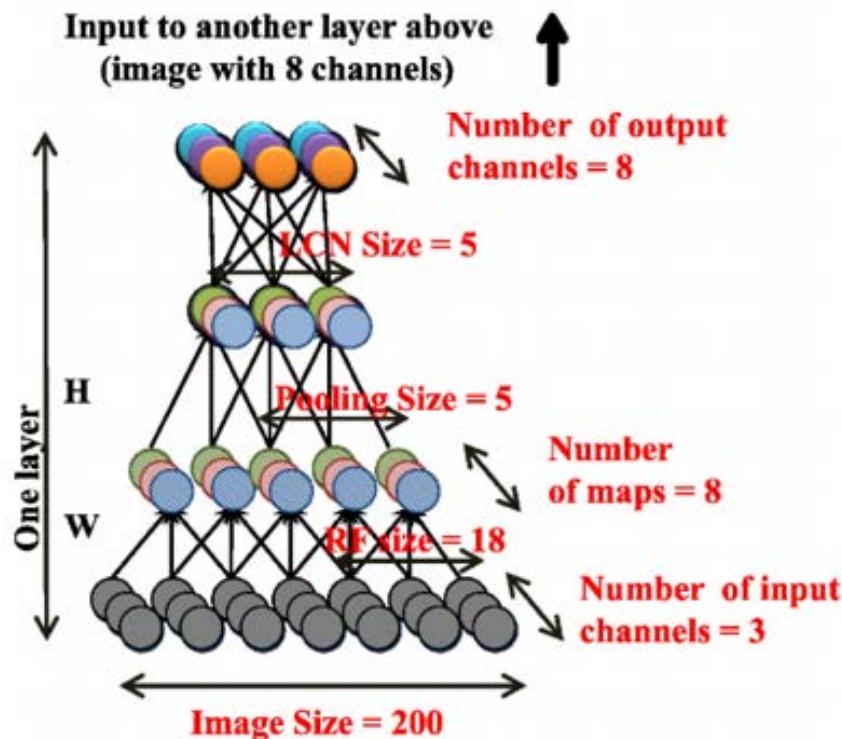


Multiple Layers

Requires many wide layers to be effective:

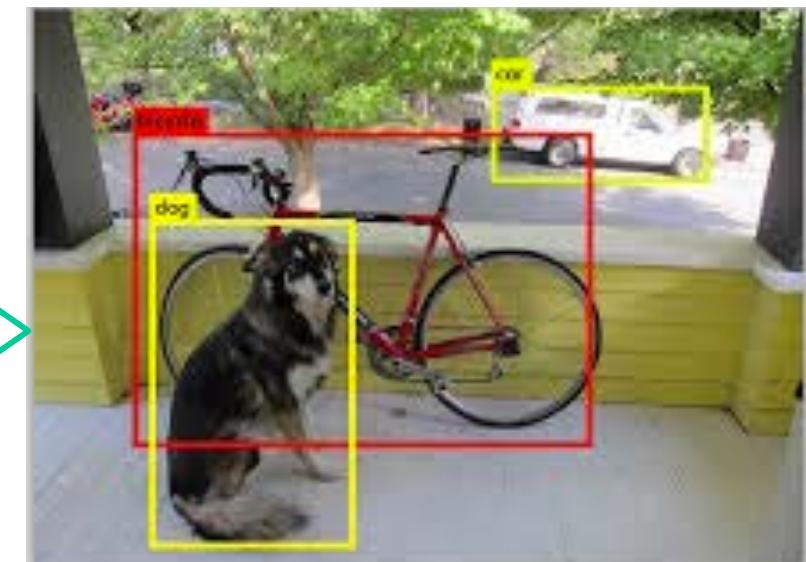
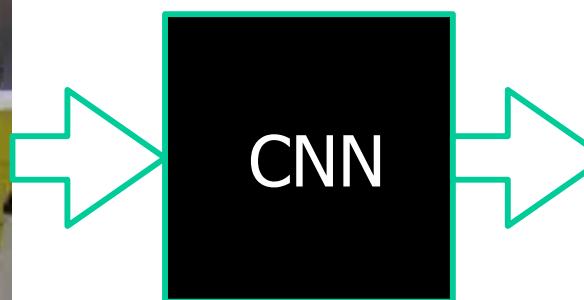
- Mostly impractical in the 1980s due to computational limitations.
- Shown to be highly effective in 2012 by using GPUs.

CONVOLUTIONAL NETWORKS



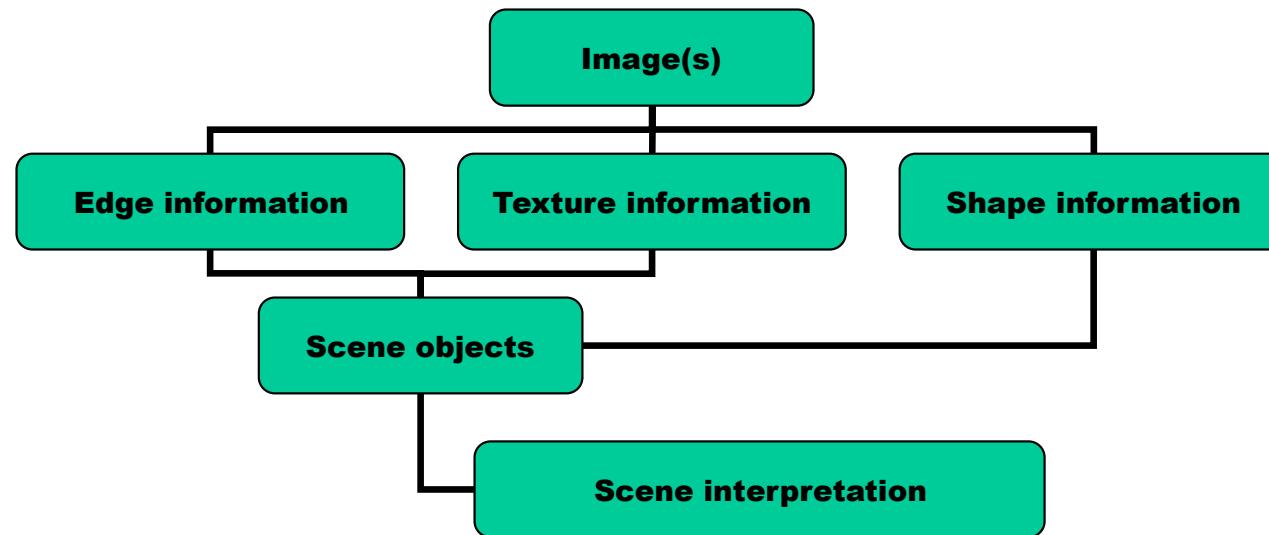
Multi-Layer Perceptron --> Deep Convolutional Network:
Effective use not only of GPUs but also of the geometry of images.

COMPUTER VISION IN THE ERA OF DEEP NETS



- Extremely effective in many cases but does not shed much light on the vision process.
- The best algorithms combine Deep Nets with more traditional techniques.

A TEACHABLE SCHEME



Decomposition of the vision process into smaller manageable and implementable steps.

- > Paradigm followed in this course
- > May not be the one humans use

COURSE OUTLINE



Introduction:

- Definition
- Image formation

Extracting features:

- Contours
- Texture
- Regions

Shape recovery:

- From one image
- Using additional images

COURSE ORGANIZATION



- Formal lectures every week (Friday)
- Exercises every other week (Tuesday)
- Written examination

WEBSITES



Projects:

- <http://cvlab.epfl.ch/projects>

Research activities:

- <http://cvlab.epfl.ch/research>

COURSE MATERIAL



Textbooks:

- R. Szeliski, Computer Vision: Computer Vision: Algorithms and Applications, 2010.
- A. Zisserman and R. Hartley, Multiple View Geometry in Computer Vision, Cambridge University Press, 2003.

Web pages:

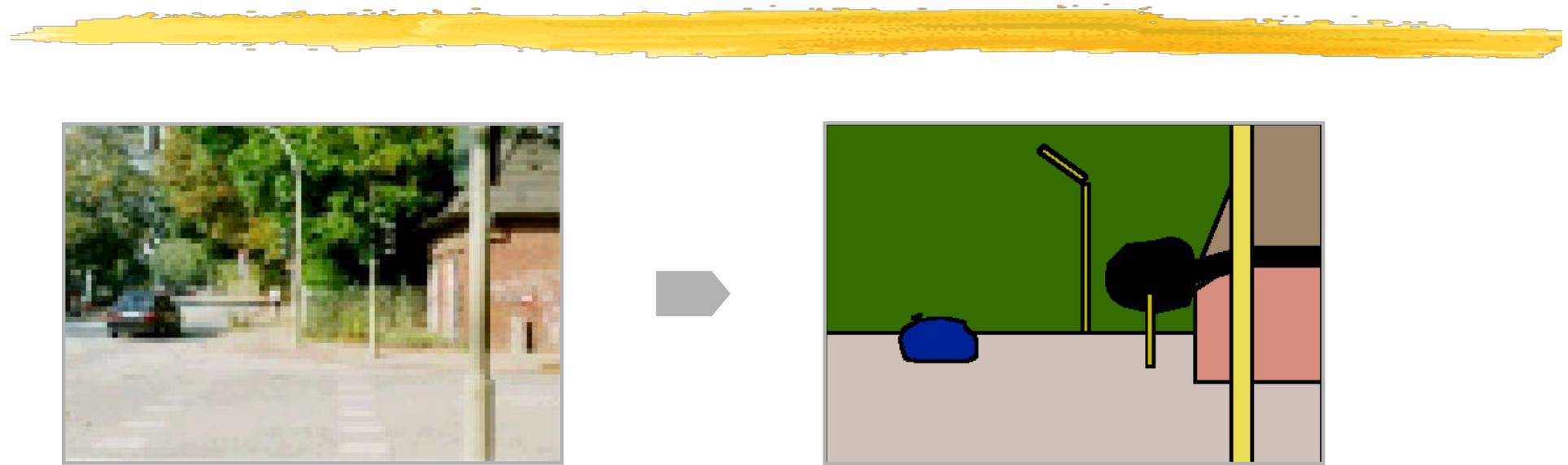
- moodle.epfl.ch (Computer Vision, introcv)
- cvlab.epfl.ch (Teaching & Projects)

REGIONS

- Defining the problem
- Interactive methods
- Automated algorithms



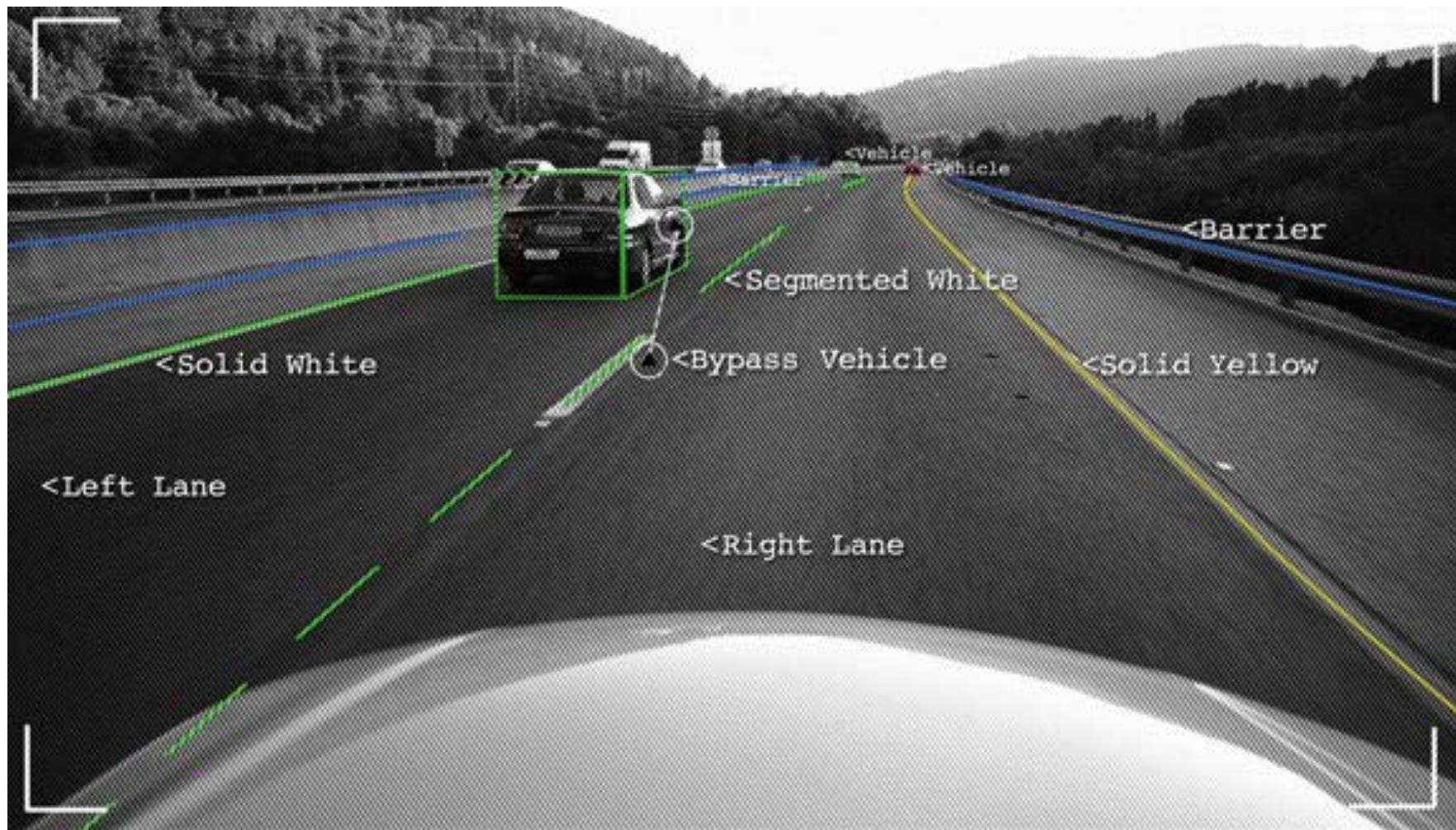
REGION SEGMENTATION



Ideal region: Set of pixels with the same statistical properties and corresponding to the same object.

Purpose: Should help with recognition, tracking, image database retrieval, and image compression among other high-level vision tasks.

AUTOMATED DRIVING



IN THEORY



Look for an image partition such that:

$$I = \bigcup_{i=1}^m S_i$$

$$S_i \cap S_j = 0, \forall i \neq j$$

$$H(S_i) = True, \forall i$$

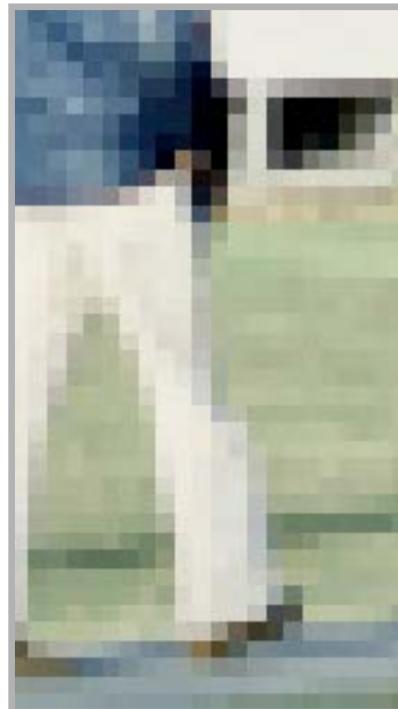
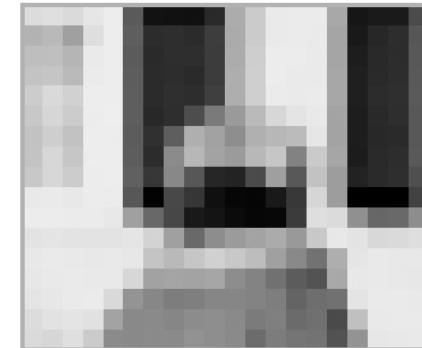
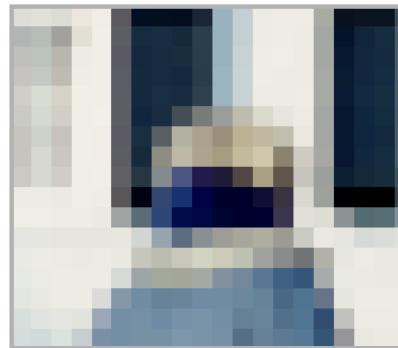
$$H(S_i \cup S_j) = False, \text{ if } S_i \text{ and } S_j \text{ are adjacent.}$$

where H measures homogeneity.

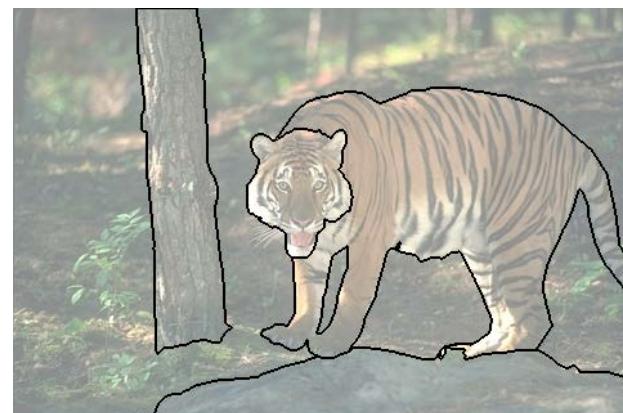
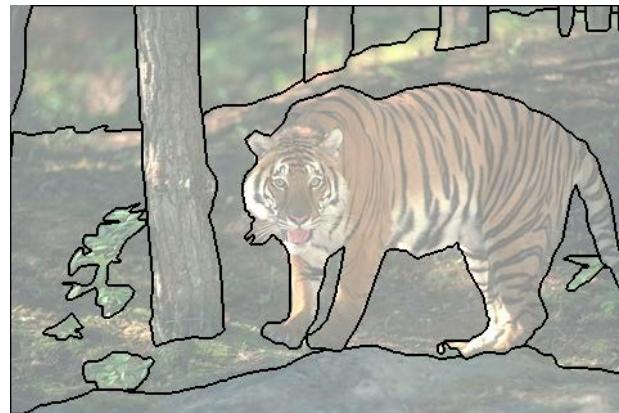
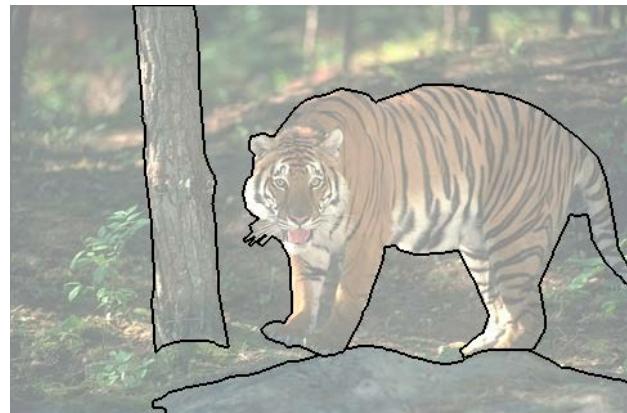
IN PRACTICE



CONTEXT IS ESSENTIAL



MULTIPLE ANSWERS



Segmentations hand-drawn by 5 different people.

HOMOGENEOUS OR NOT?



MITOCHONDRIA

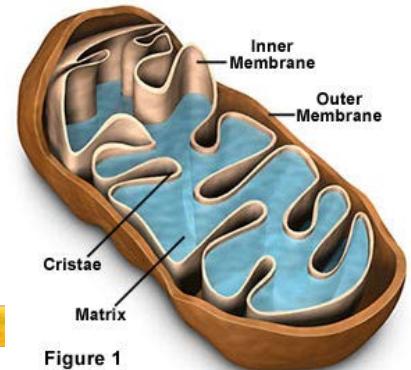
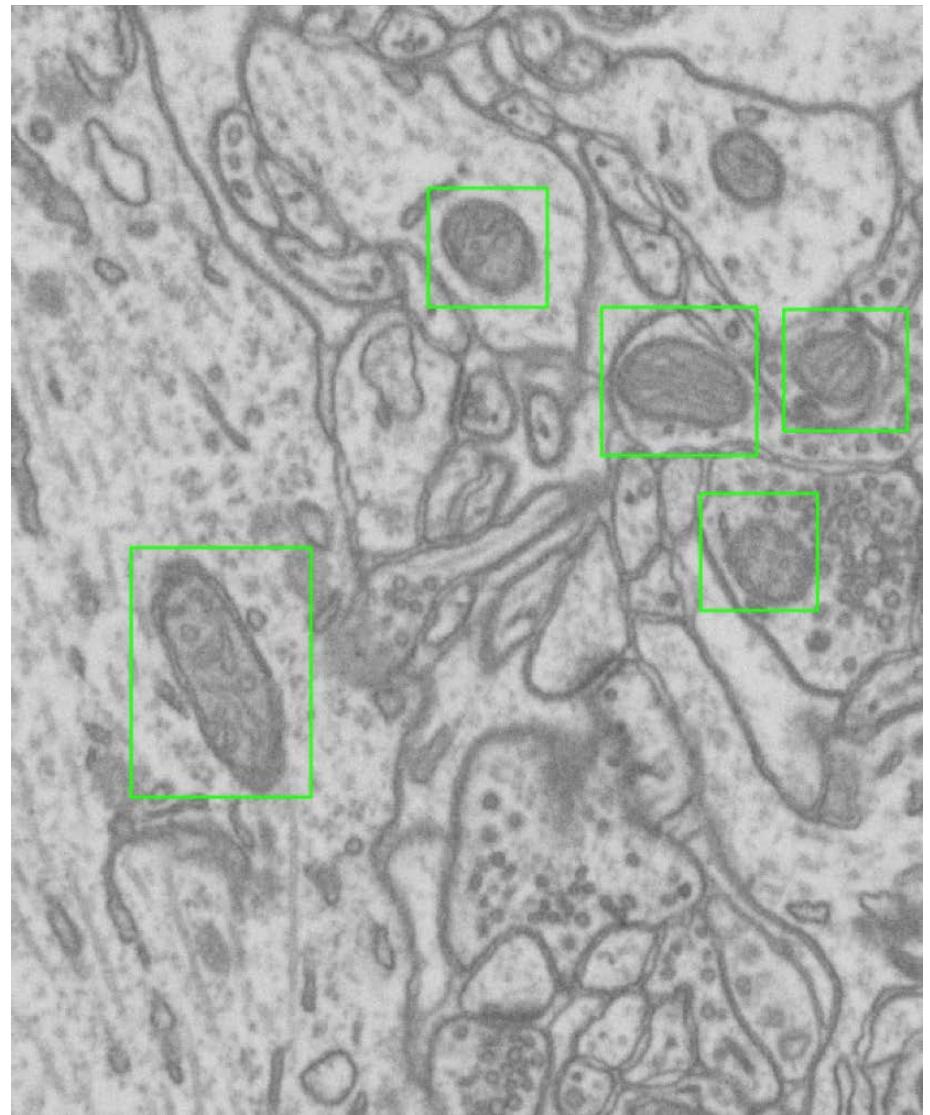
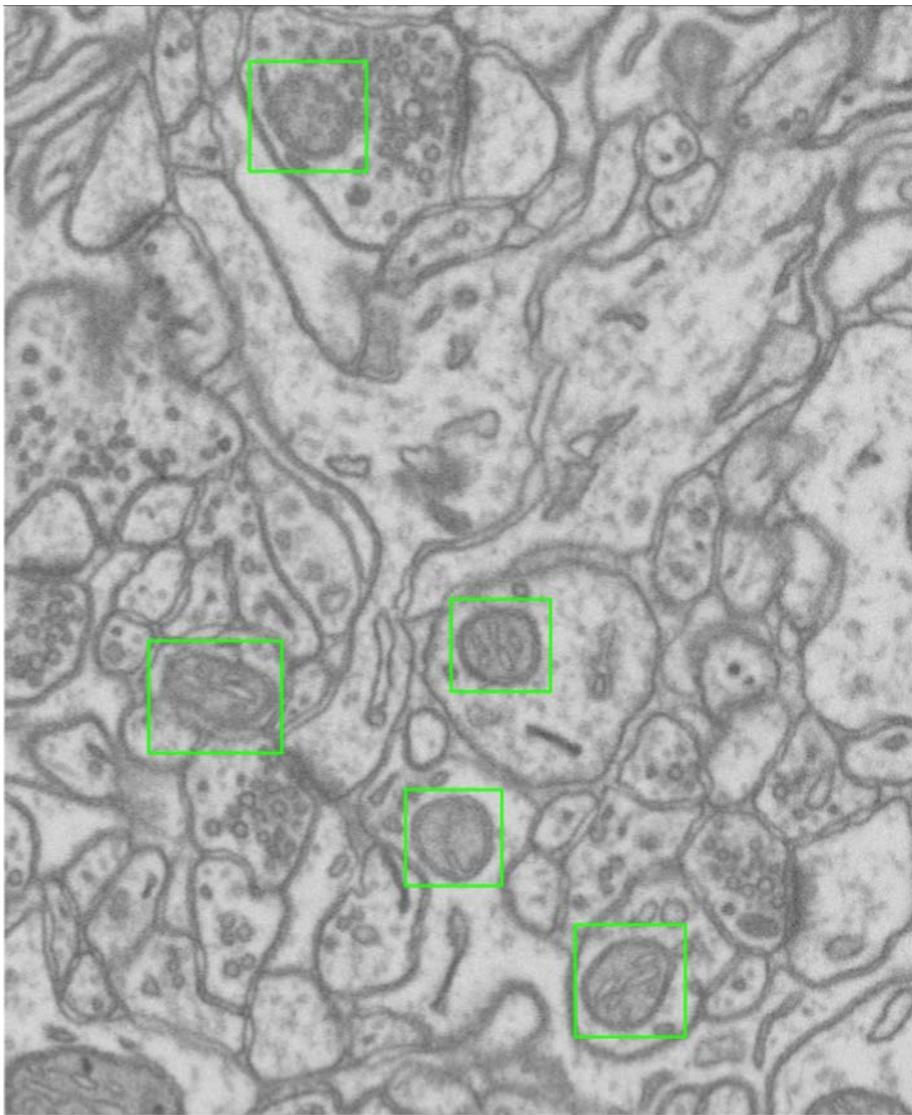
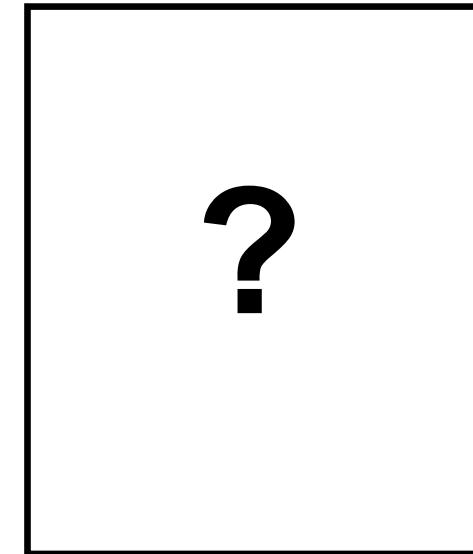
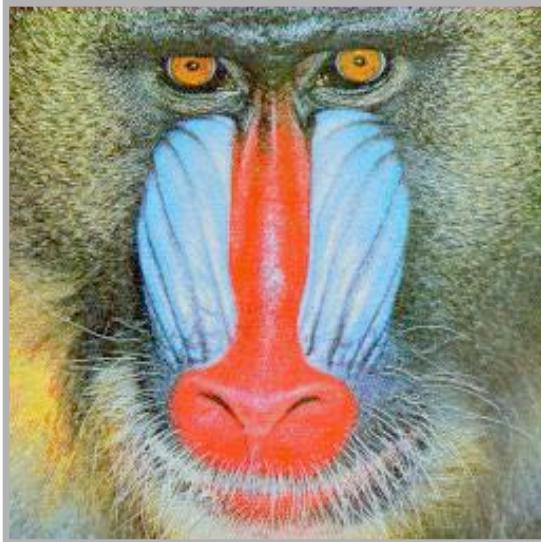


Figure 1



DERIVED IMAGES



Homogeneity can be evaluated in the original image data or in 'derived' images:

- Gray level images
- Color images from R, G, B
- Textural images
- Displacement images from motion analysis
- 3D depth images

IN THEORY



Merge:

- Start with a partition that satisfies Eq. 3.
- Satisfy Eq. 4 by merging regions.

Split:

- Start with a partition that satisfies Eq. 4.
- Split regions until they all satisfy Eq. 3.

Homogeneity:

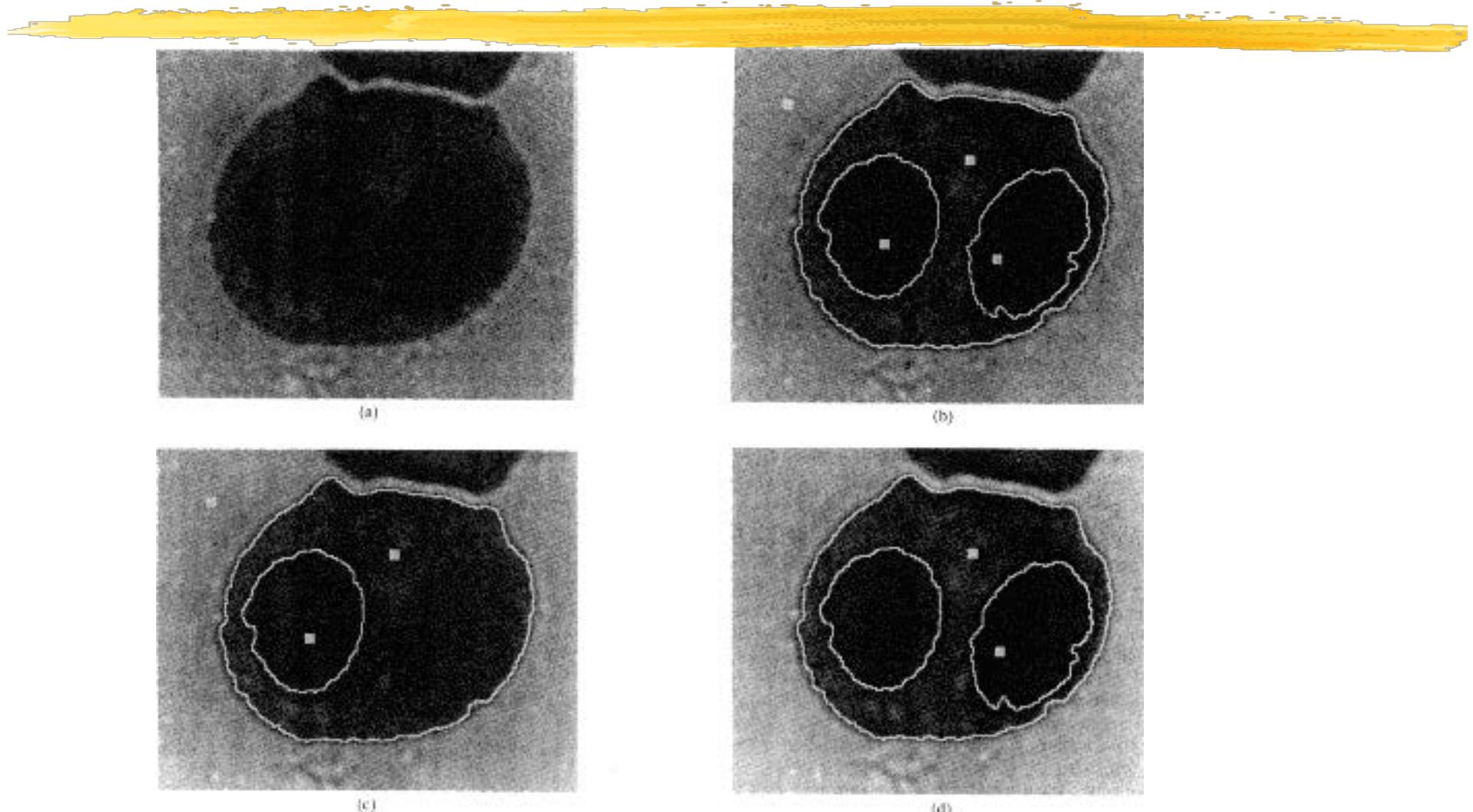
- Uniform gray-level or color statistics.
- Regions to which a parametric surface can be fitted.

IN PRACTICE



- Region Growing
- Histogram splitting.
- K-Means.
- Mean Shift.
- Graph theoretic methods.
- Convolutional Neural Nets

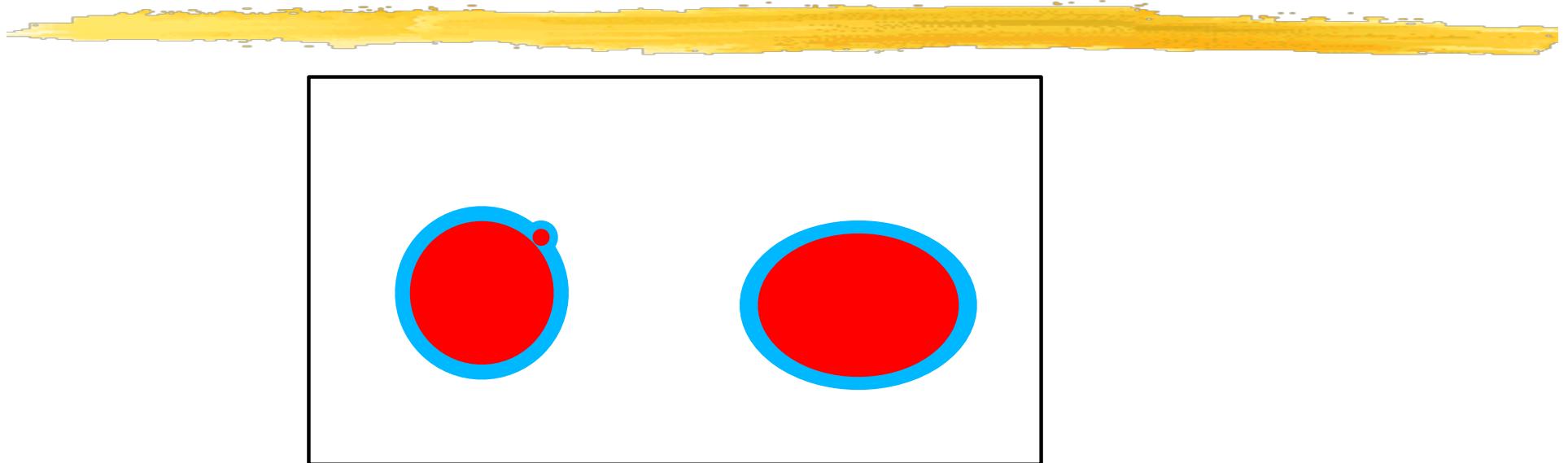
MERGE: REGION GROWING



Interactive Segmentation of a Cell

Adams and Bischof, PAMI'94

REGION GROWING



Given a set of regions A_1, \dots, A_N , consider

$$T = \left\{ x \notin \bigcup_{i=1}^N A_i \text{ such that } N(x) \mid \left(\bigcup_{i=1}^N A_i \right) \neq \emptyset \right\},$$

the set of unlabeled pixels that are neighbors of already labeled ones.

- Define a metric, e.g. $\delta(x) = \left| g(x) - \operatorname{mean}_{y \in A_{i(x)}} [g(y)] \right|$.
- Represent T as a sorted list according to this metric, the *SSL*.

REGION GROWING



While SSL is not empty do

 Remove first pixel y from SSL.

If all already labeled neighbors of y, other than boundary pixels, have the same label

then

 Set y to this label.

 Update running mean of corresponding region.

 Add neighbors of y that are neither already set nor already in the SSL to the SSL according to their distance value.

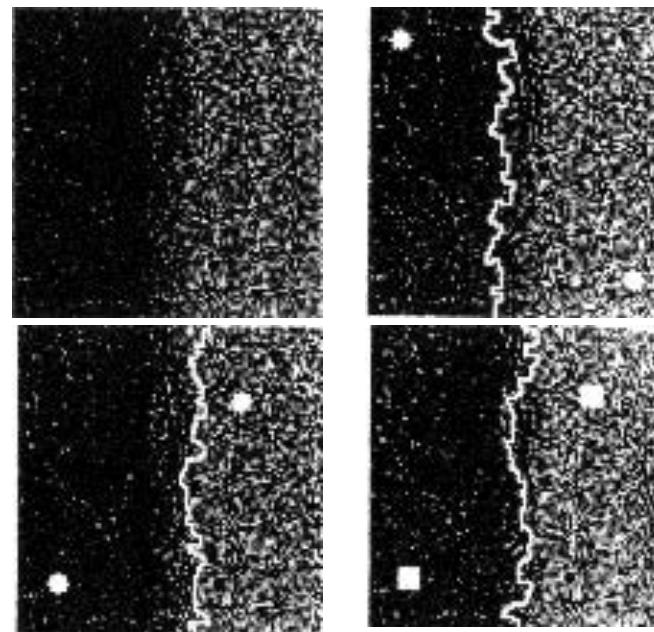
else

 Flag y as a boundary pixel.

fi

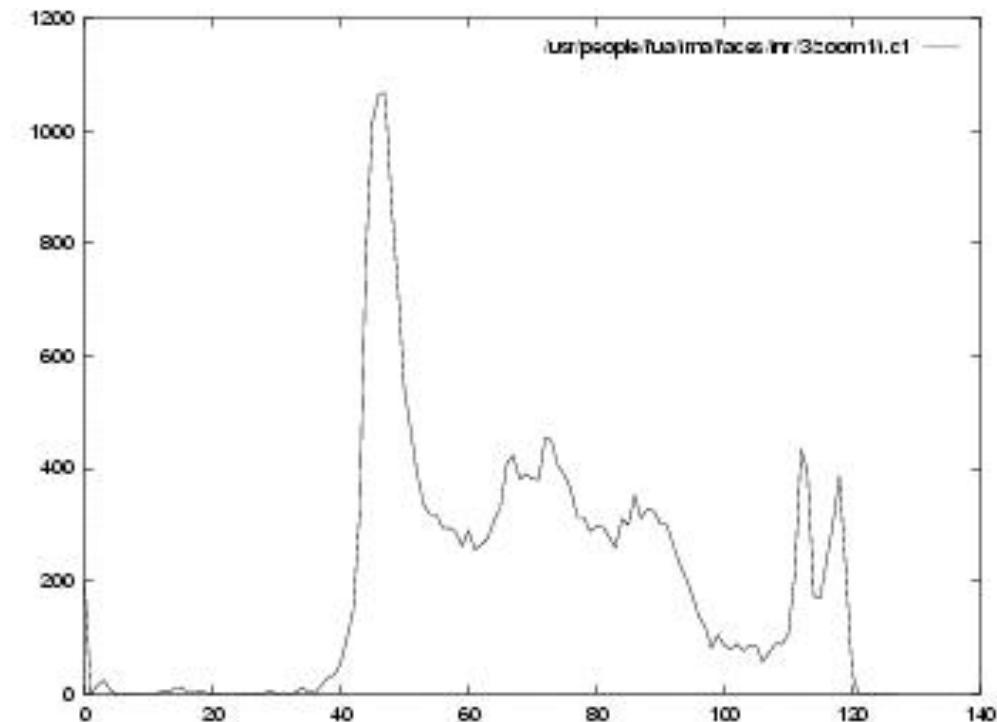
od

LIMITATIONS



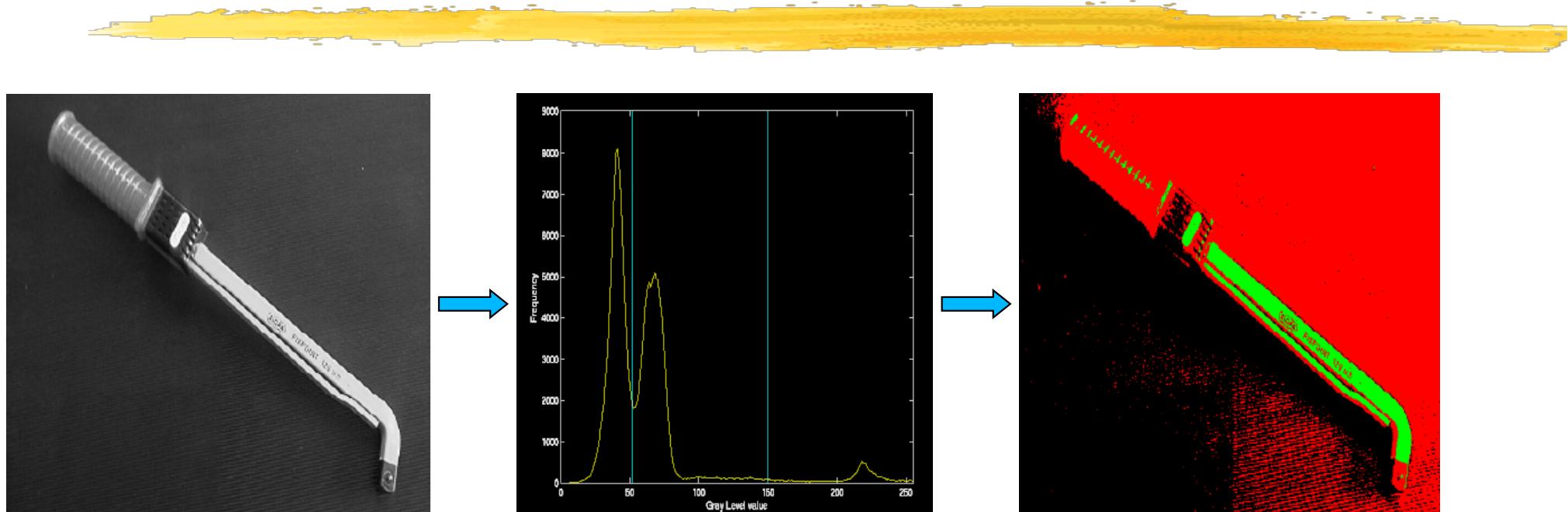
- In general, the result depends on the order in which the pixels are taken into consideration.
- The homogeneity measure is noise sensitive.

SLIT: IMAGE HISTOGRAM



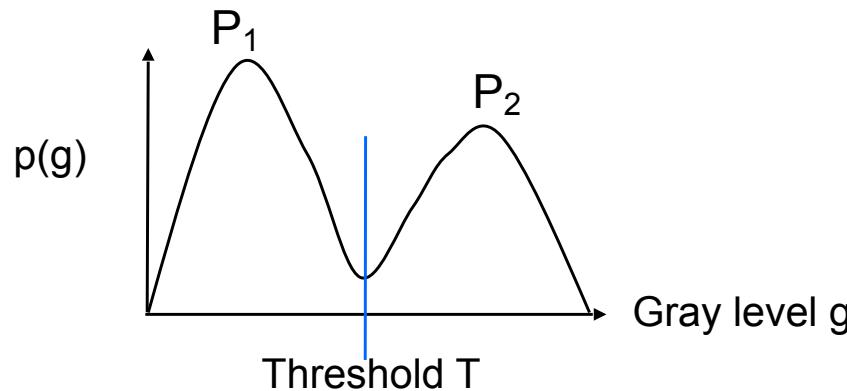
Number of pixels that have a given gray level.

HISTOGRAM SPLITTING



- Groups of similar pixels appear as bumps in the brightness histogram
- Split the histogram at local-minima
- Label pixels according to which bump they belong to

RECURSIVE SPLITTING

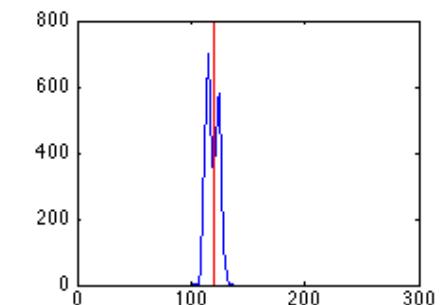
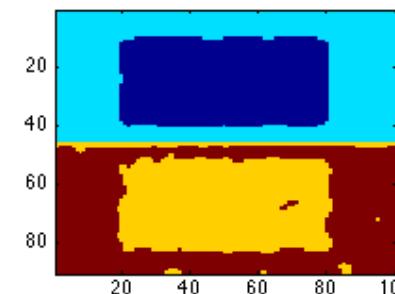
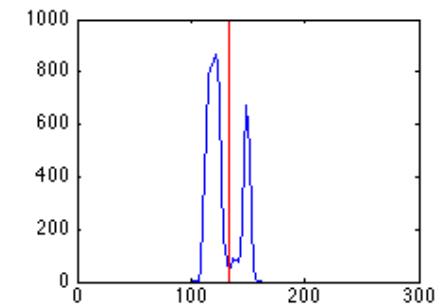
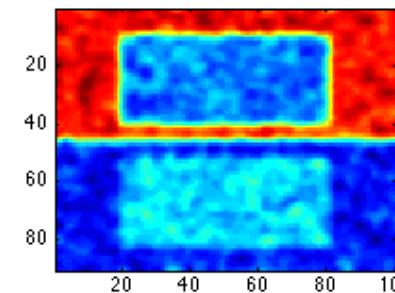


- Compute image histogram.
- Smooth histogram.
- Look for peaks separated by deep valleys.
- Group pixels into connected regions.
- Smooth these regions.
- Iterate.

RECURSIVE SPLITTING

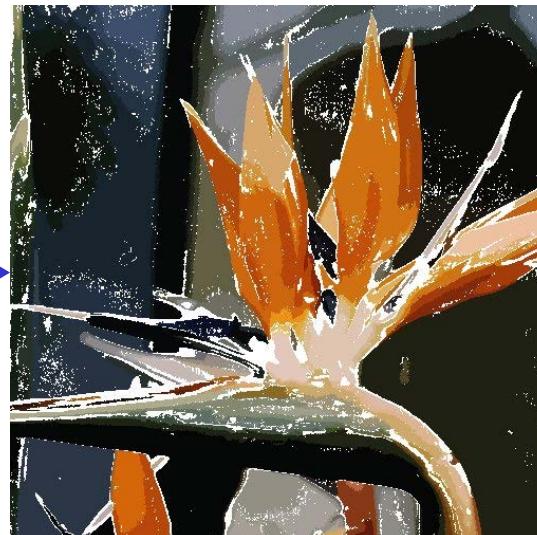
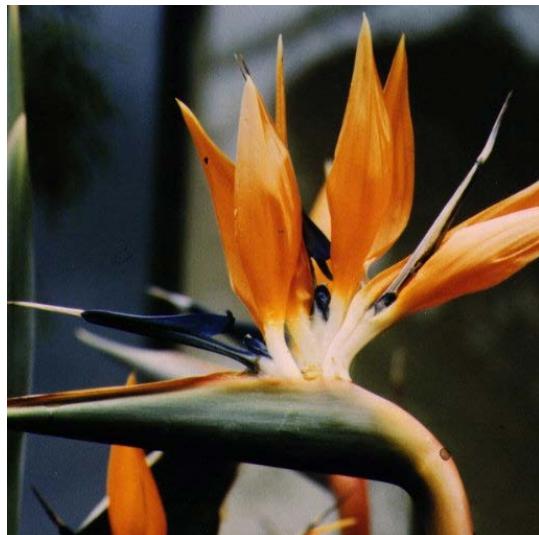
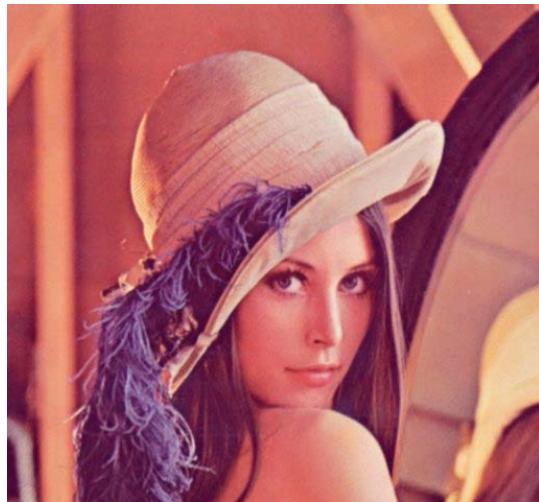


- A first threshold is used to segment the dark pixels.
- This yields two regions, the bottom half of the picture and the dark rectangle at the top.
- The bottom half of the picture can now be more easily segmented into two regions.

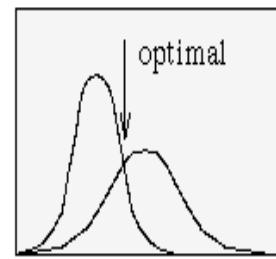
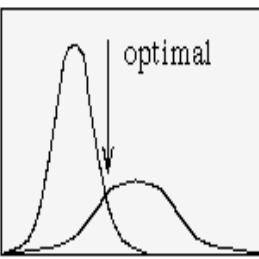
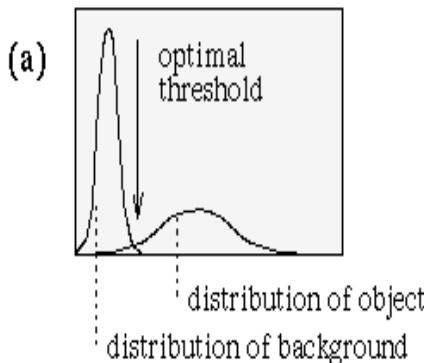


-> Decisions can be deferred until enough information becomes available.

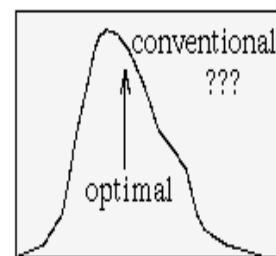
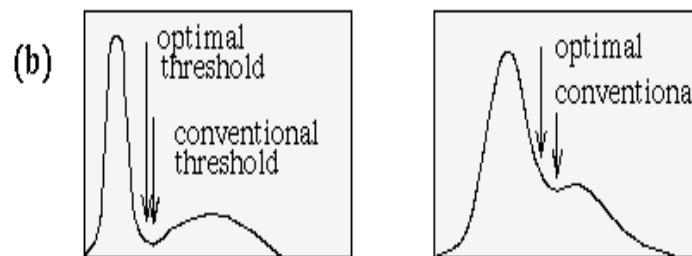
OVERSEGMENTATION



THRESHOLDS



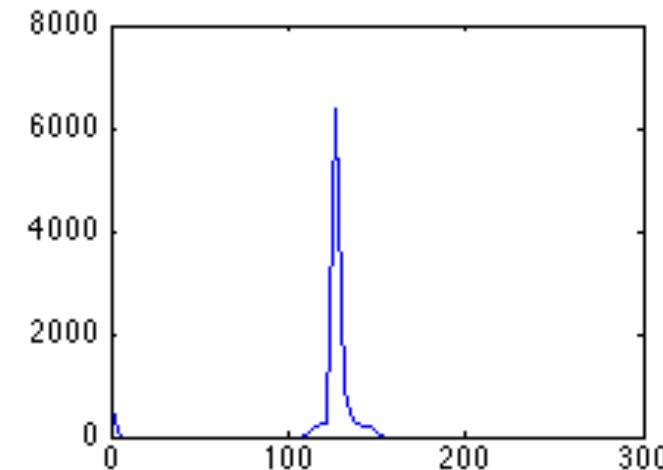
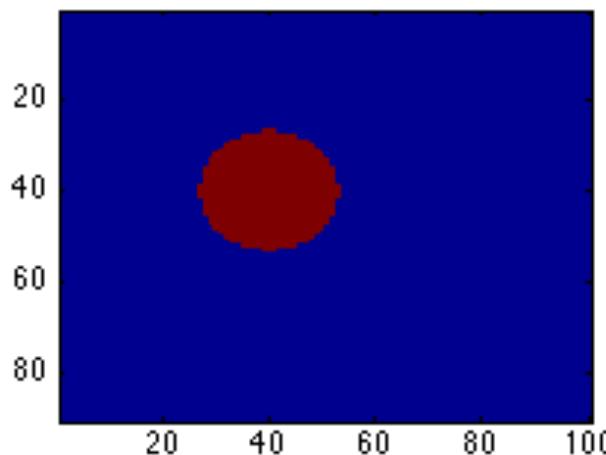
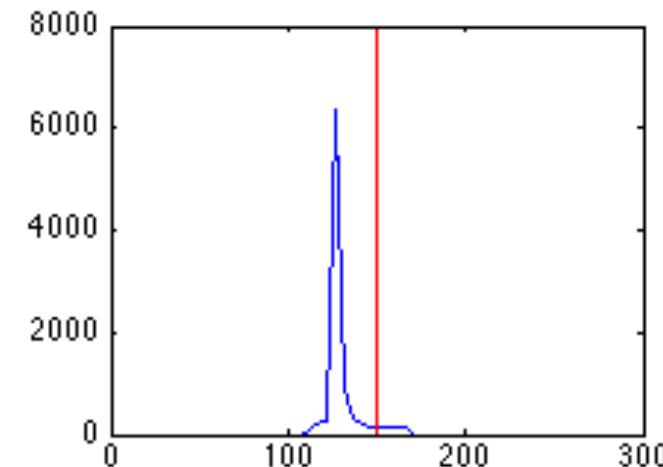
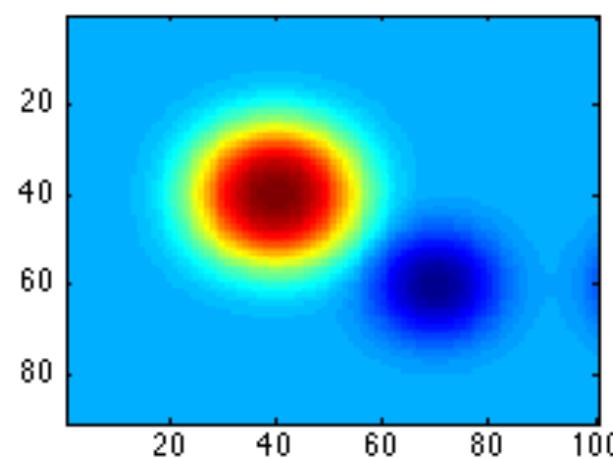
**Probability
distributions**



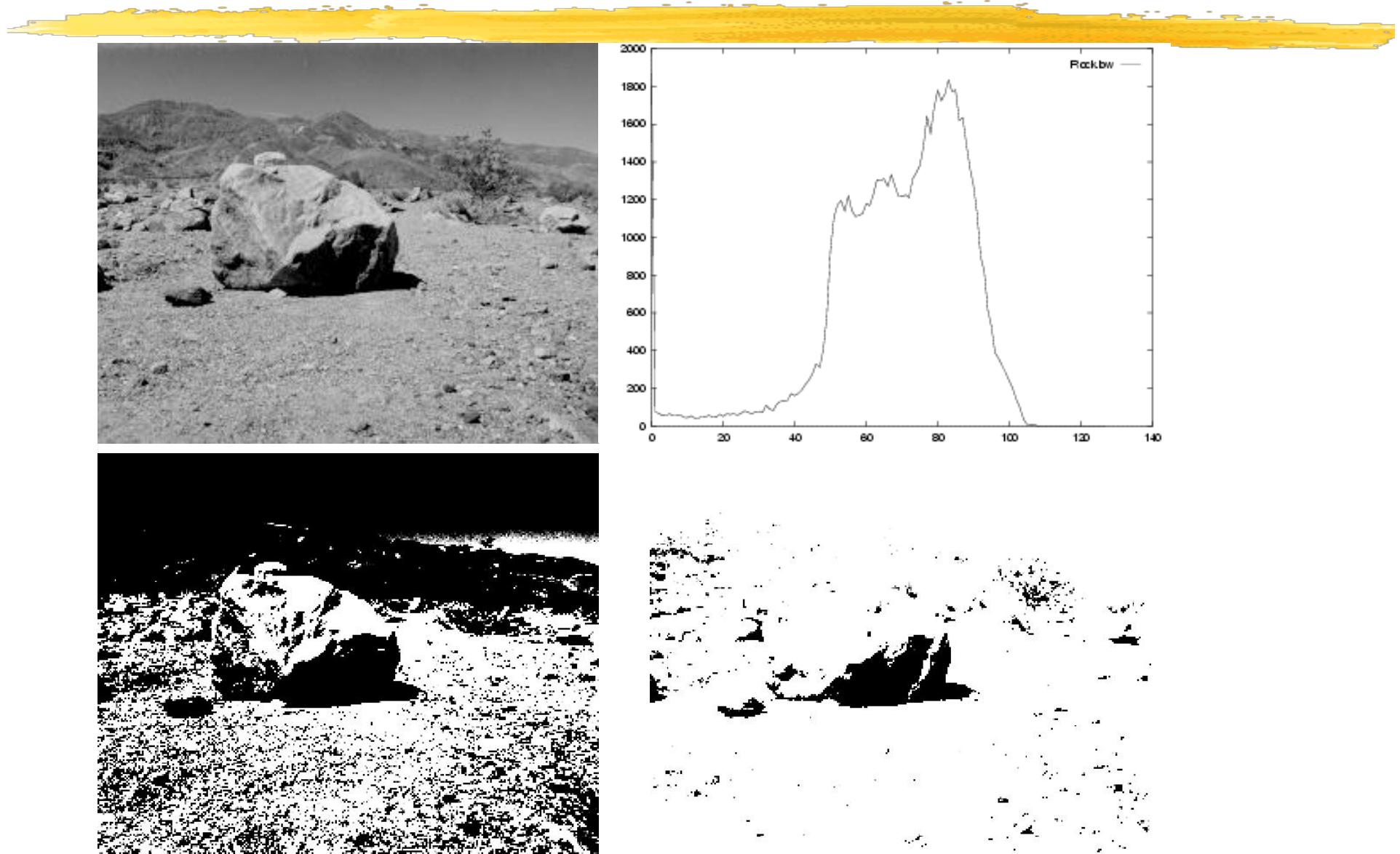
**Corresponding
histograms**

Choosing optimal thresholds is a difficult optimization problem.

NO VALID THRESHOLD



NO VALID THRESHOLD



WEAKNESSES



- Histograms do not account for neighborhood relationships.
- Thresholds are hard to find.
- Some edges can have gray levels on both sides that belong to the same histogram peak.

ILLUMINATION PROBLEMS

Sonnet for Lena

O dear Lena, your beauty is so vast
It is hard sometimes to describe it fast.
I thought the entire world I would impress
If only your portrait I could compress.
Alas! First when I tried to use VQ
I found that your cheeks belong to only you.
Your silky hair contains a thousand lines
Hard to match with sums of discrete cosines.
And for your lips, sensual and tactful
Thirteen Crays found not the proper fractal.
And while these setbacks are all quite severe
I might have fixed them with hacks here or there
But when filters took sparkle from your eyes
I said, 'Damn all this. I'll just digitize.'

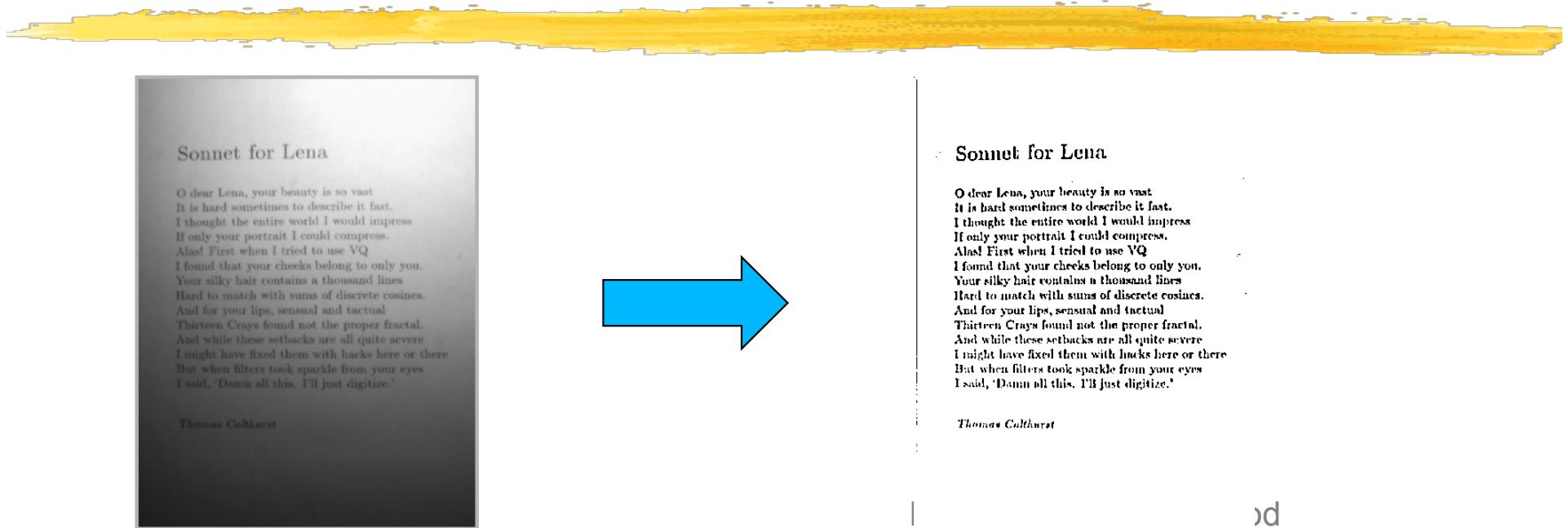
Thomas Colthurst

Sonnet [etc]

O dear Lena
It is hard sometimes to...
I thought the entire world I...
If only your portrait I...
Alas! First when I tried to use VQ
I found that your cheeks belong to only you.
Your silky hair contains a thousand lines
Hard to match with sums of discrete cosines.
And for your lips, sensual and tactful
Thirteen Crays found not the proper fractal.
And while these setbacks are all quite severe
I might have fixed them with hacks here or there
But when filters took sparkle from your eyes
I said, 'Damn all this. I'll just digitize.'

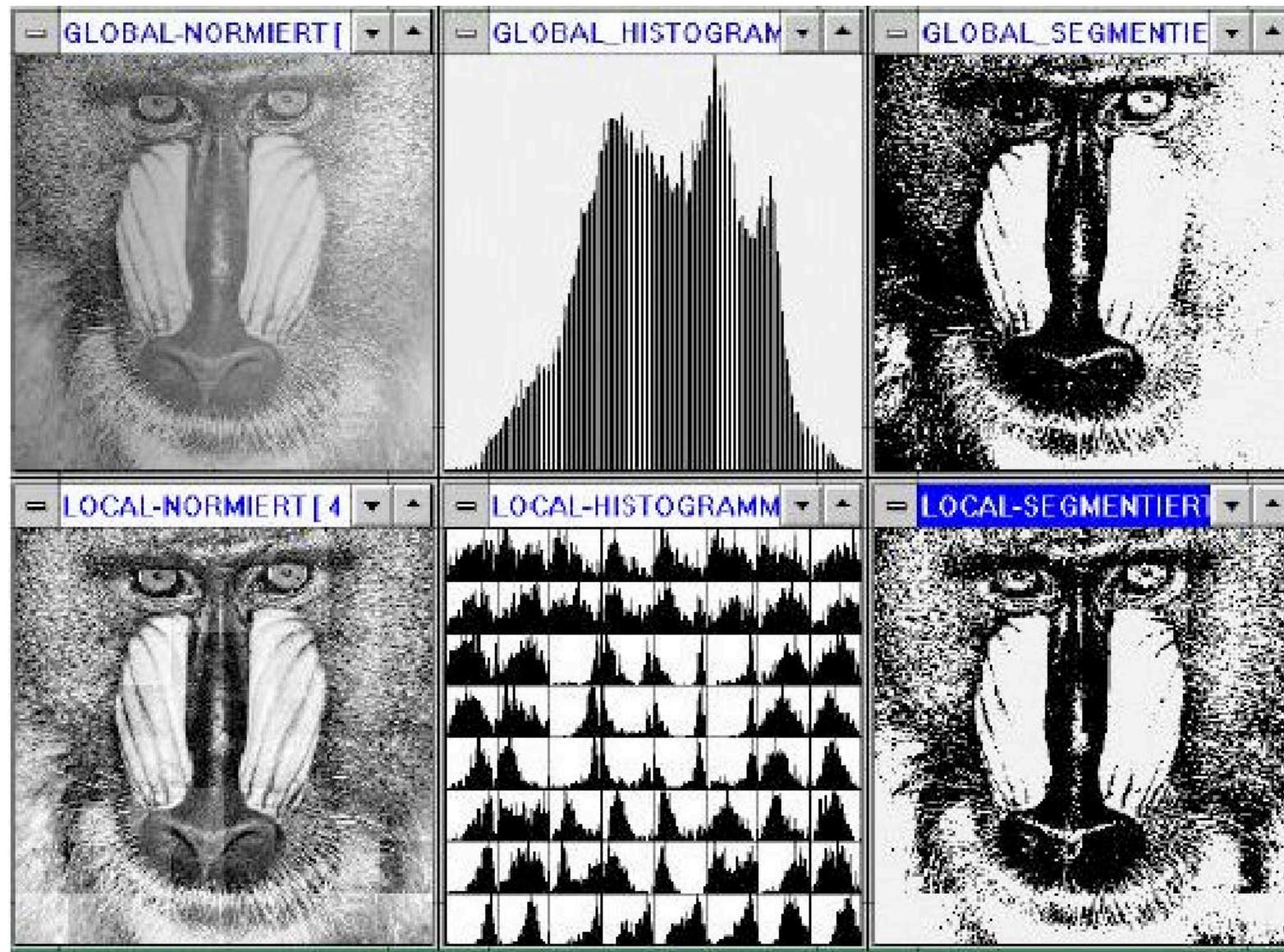
No global threshold -> Local ones are required.

LOCAL THRESHOLDING

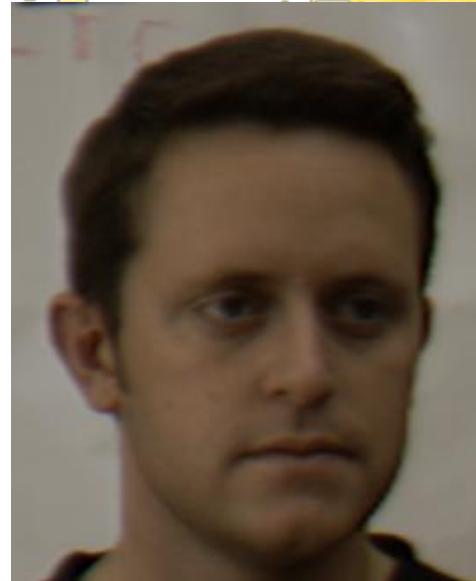


- Examine statistically pixel values in local neighborhood around pixel to be thresholded.
- Use local statistic as threshold.
- Possibilities include mean, median, or mean of max and min value.

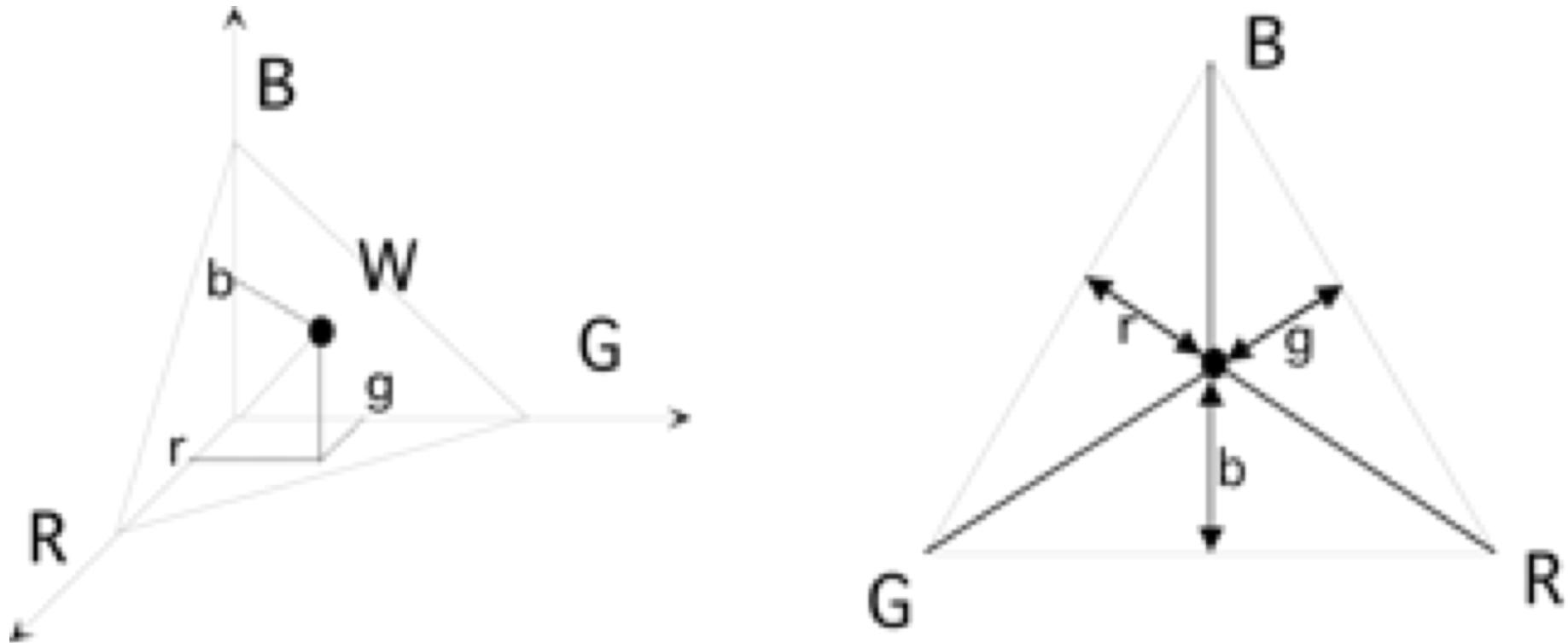
IMPROVED RESULTS



USING COLOR

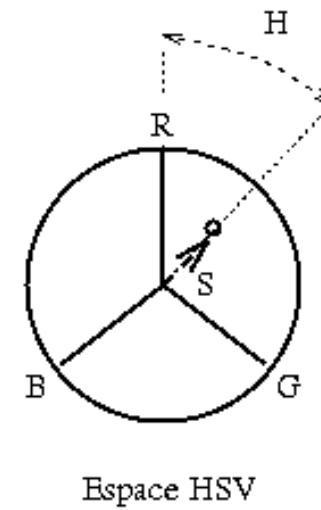
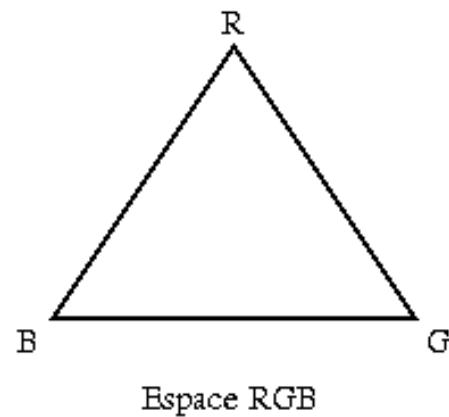


RGB CHROMATICITY DIAGRAM



The Maxwell triangle involves projecting the colors in RGB space onto $R+G+B=1$ plane.
→ Chromaticity independently of luminance.

COLOR SPACE



Hue



Saturation



Value



HSV SPACE



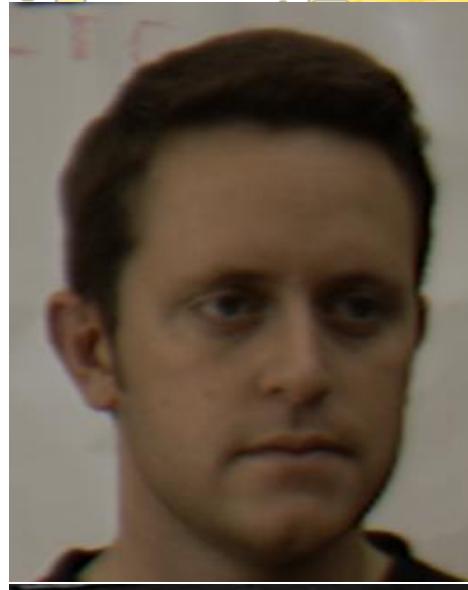
Normalized colors:

$$\begin{aligned} r &= \frac{R}{R + G + B} \\ g &= \frac{G}{R + G + B} \\ b &= \frac{B}{R + G + B} \end{aligned}$$

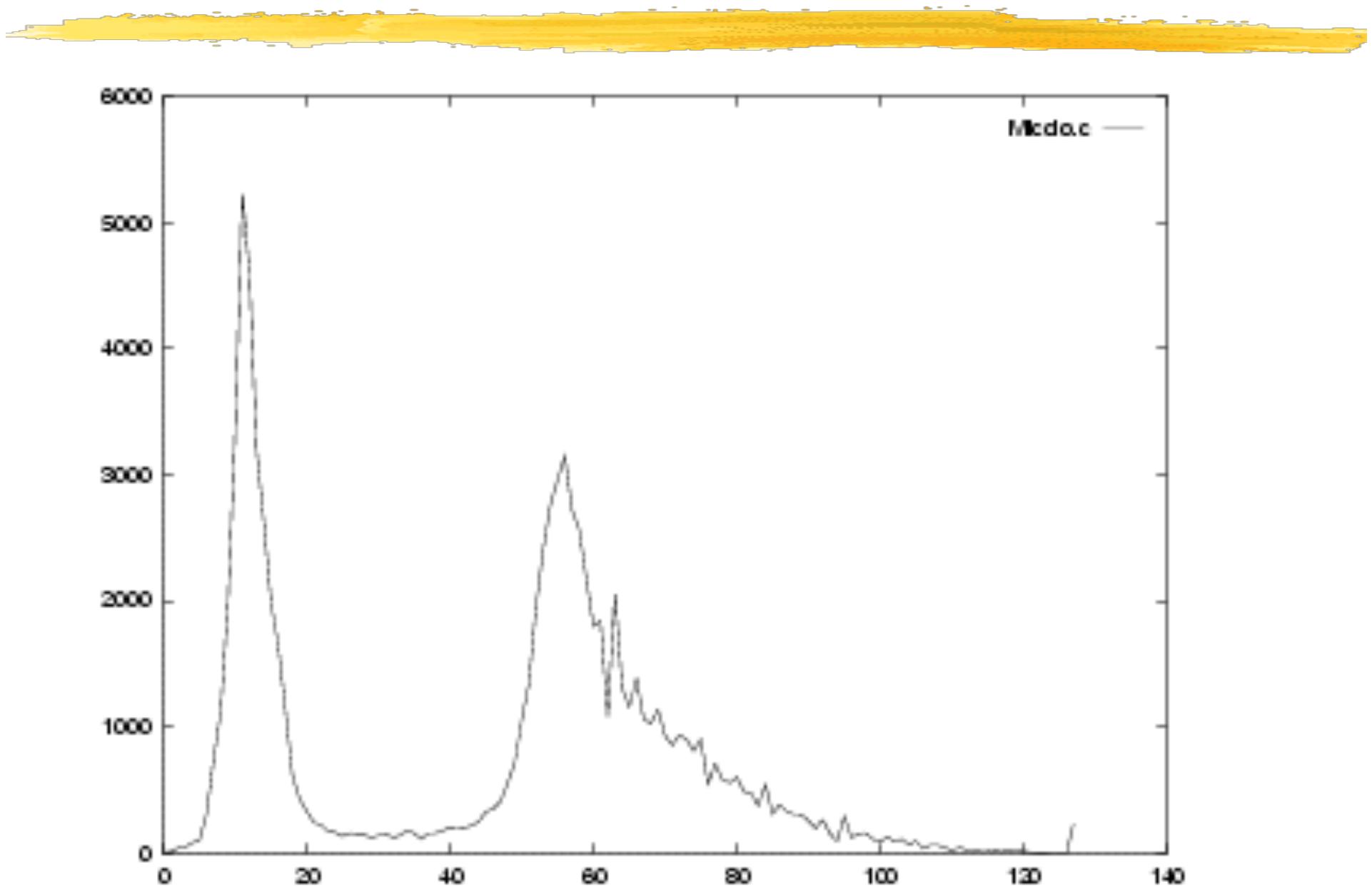
Hue/Saturation/Value

$$\begin{aligned} V &= R + G + B \\ S &= 1 - \frac{3 \min(r, g, b)}{I} \\ H &= \arccos\left(\frac{0.5(2r - g - b)}{\sqrt{(r - g)^2 + (r - g)(g - b)}}\right) \text{ if } b < g \end{aligned}$$

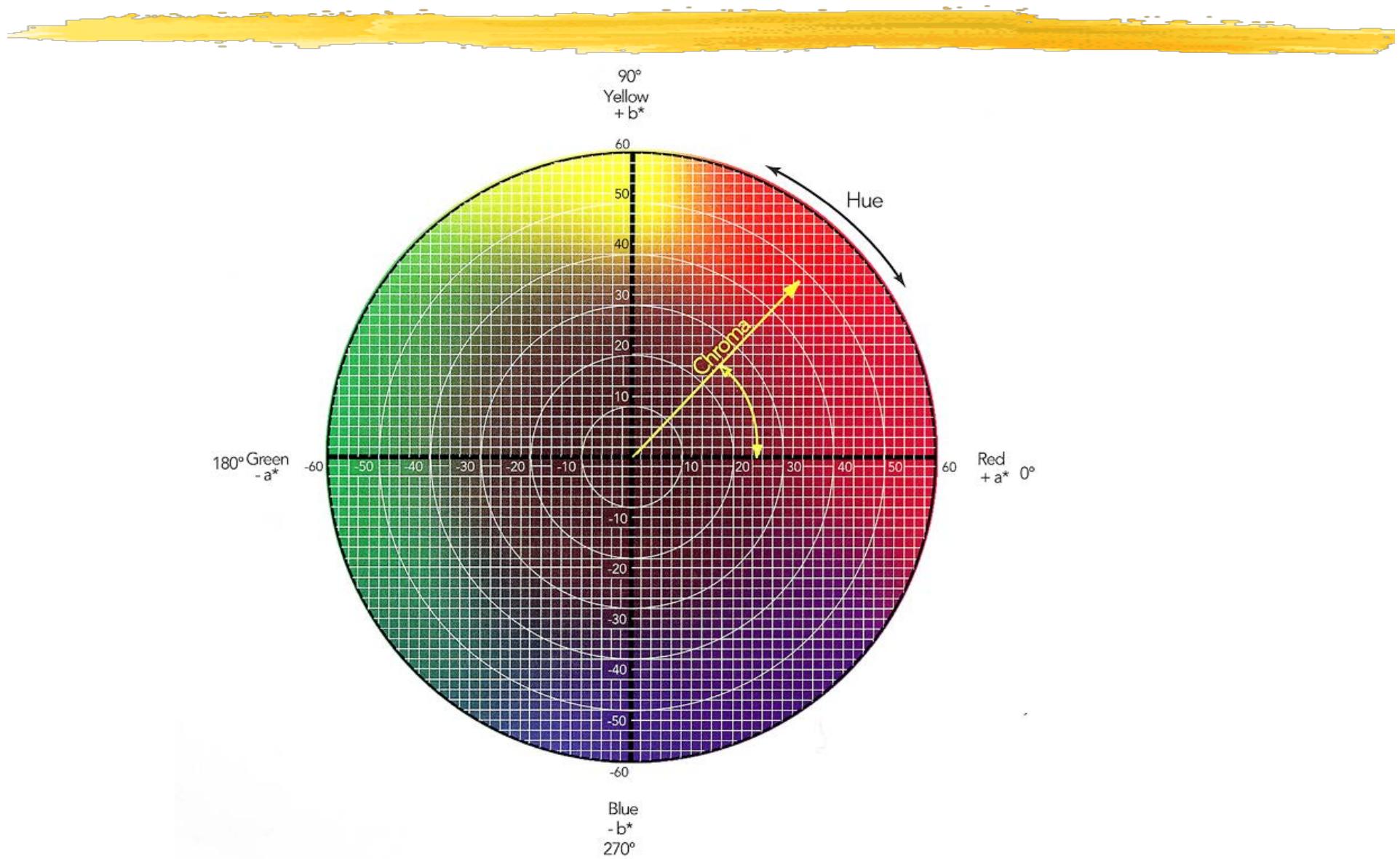
HSV IMAGES



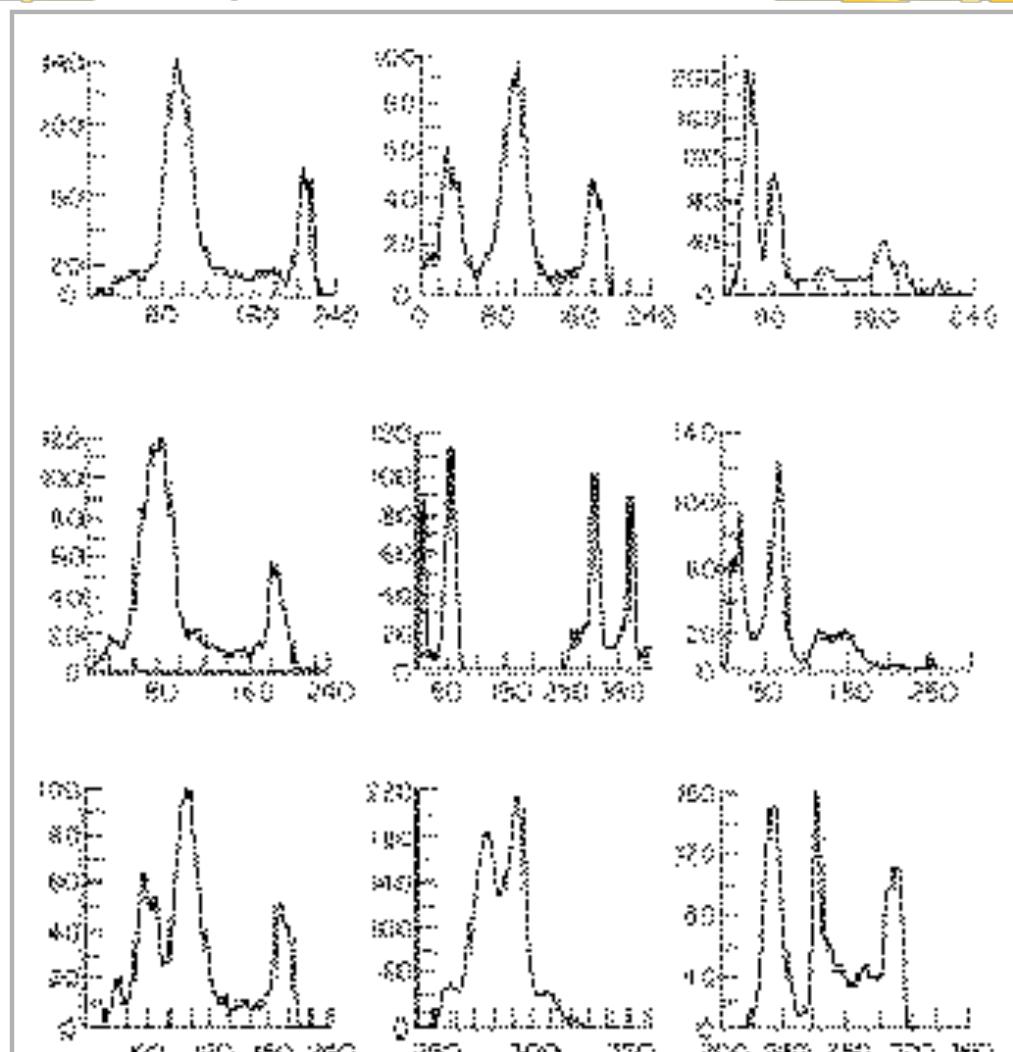
IMPROVED HISTOGRAM



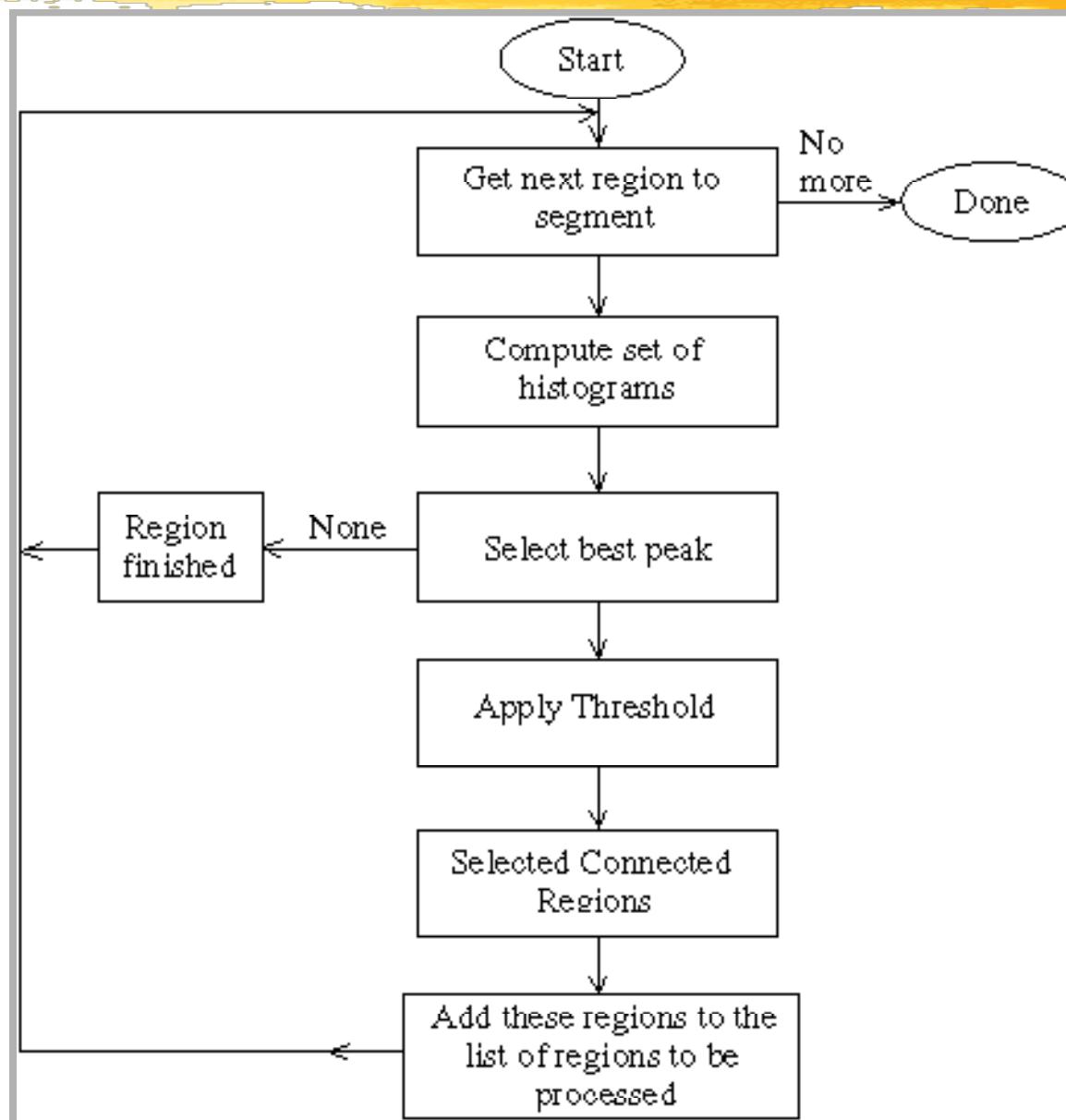
CIE LAB COLOR SPACE



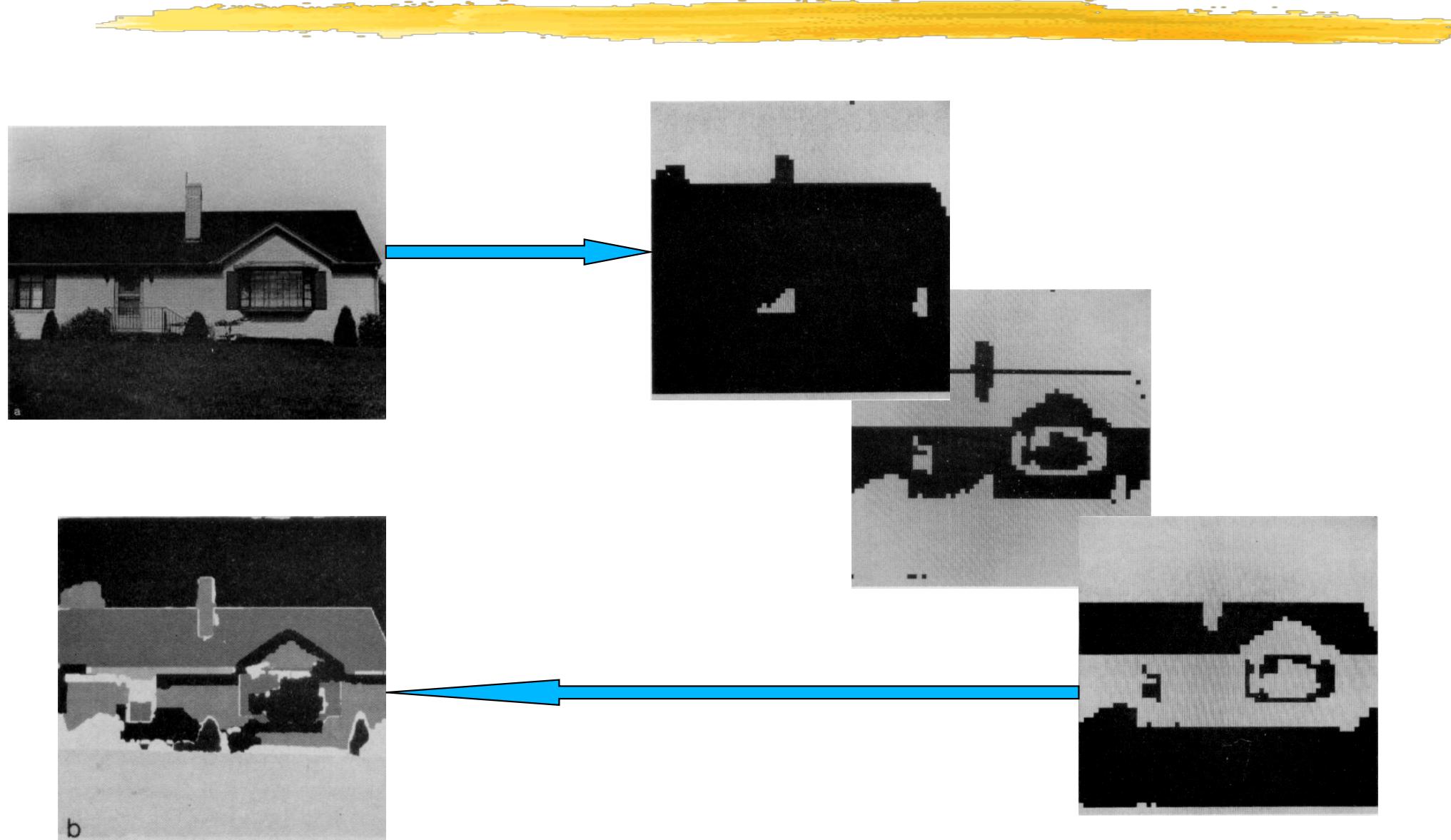
MULTIPLE HISTOGRAMS



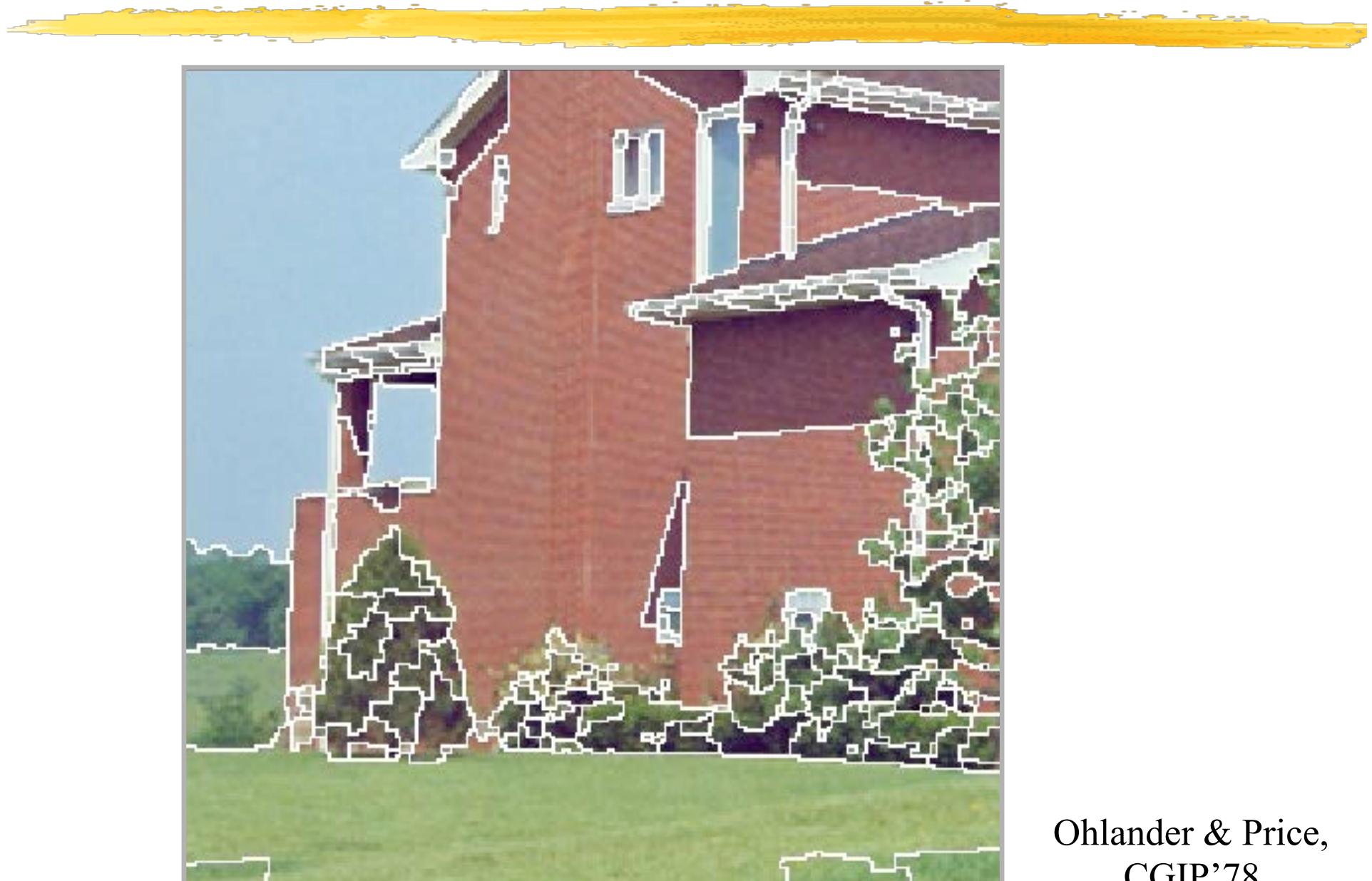
ALGORITHM



HIERARCHICAL SEGMENTATION



COLOR SEGMENTATION



Ohlander & Price,
CGIP'78

MERGING REGIONS

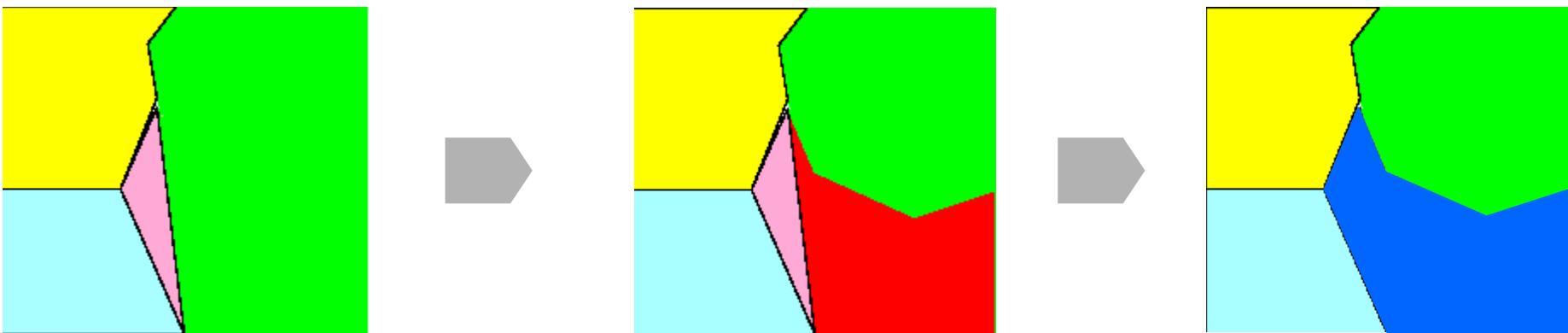


Oversegmentation: Too many small regions.

Merging: Each region is merged with the most similar one until no regions smaller than a threshold remain.



SPLIT AND MERGE



Split, merge, and split again on the basis of a homogeneity criterion.

RECURSIVE MERGING



- Create an image partition.
- Compute an adjacency graph.
- For each image region:
 - Test its similarity with it neighbors.
 - Group the most similar ones.
- Iterate until no more regions can be grouped.

FISHER'S CRITERION

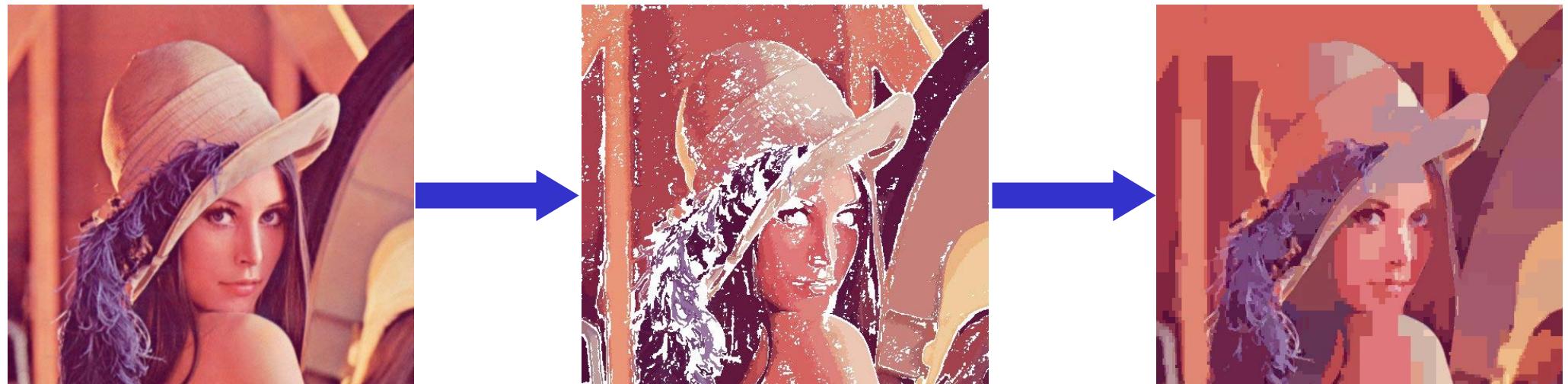


Discrimination between regions of different means and standard deviations can be done using

$$\frac{\mu_1 - \mu_2}{\sqrt{\sigma_1^2 + \sigma_2^2}} > \lambda$$

where λ is a threshold. If two regions have good separation in the means and low variance, then we can separate them.

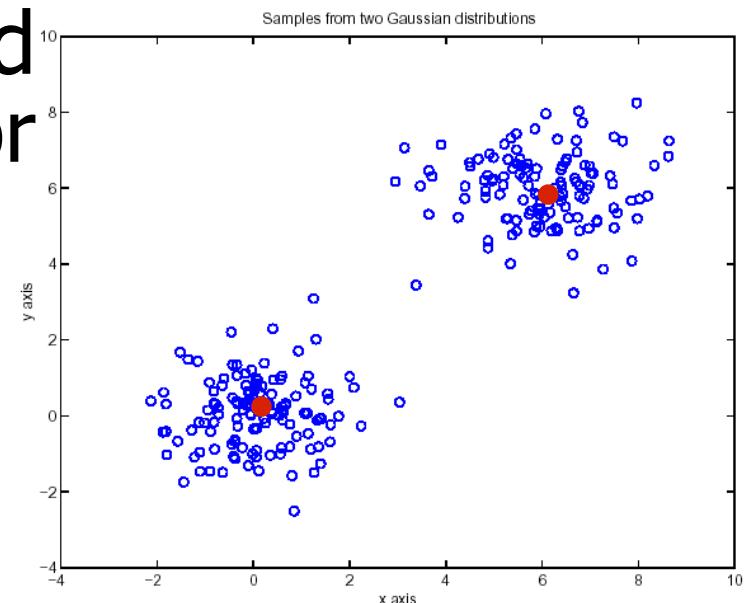
RESULTS



K-MEANS CLUSTERING

Assuming there are k regions and each one is described by a vector x_j , define:

- An objective function that measures the compactness of these regions.
- An optimization method that finds the most compact ones.



For a set of points in space, x_j is a coordinate vector. For black and white images, there are 1 or 3. For color images there are 3 or 5.

OBJECTIVE FUNCTION

$$\Phi(\text{clusters, data}) = \sum_{i \in \text{clusters}} \left\{ \sum_{j \in i^{\text{'th cluster}}} (\mathbf{x}_j - \mathbf{c}_i)^T (\mathbf{x}_j - \mathbf{c}_i) \right\}$$

If the allocation of points to clusters were known, we could compute the best centers easily.

But there are far too many combinations for an exhaustive search.

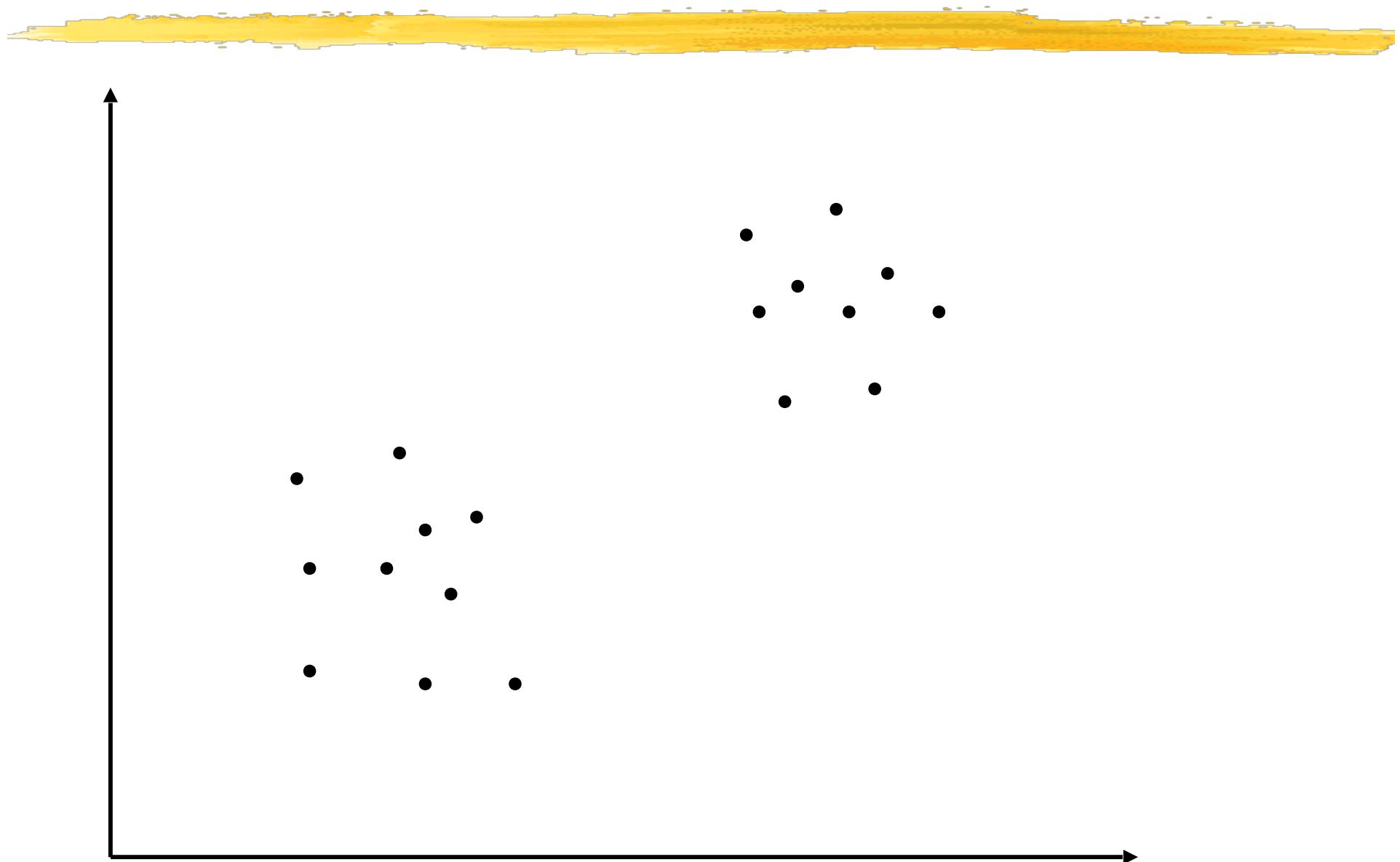
- Define an algorithm that alternates
 - Assume centers are known, allocate points
 - Assume allocation is known, compute centers

ALGORITHM

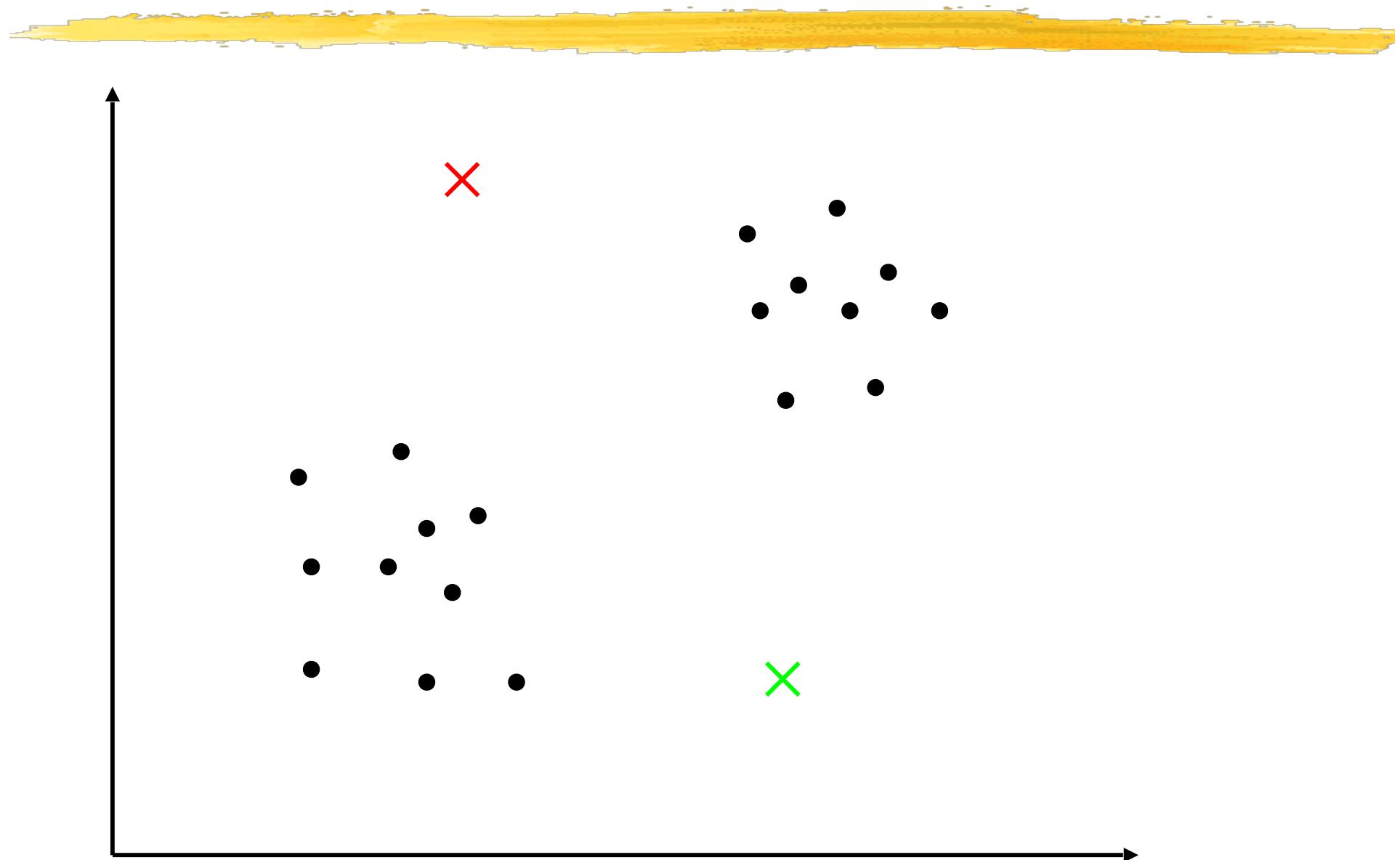


- Choose, for example randomly, k points that will serve as cluster centers.
- Until their positions stabilize:
 1. Associate each pixel to the cluster whose center is closest;
 2. Recompute the centers by averaging the elements of each cluster.
- Extract connected components.

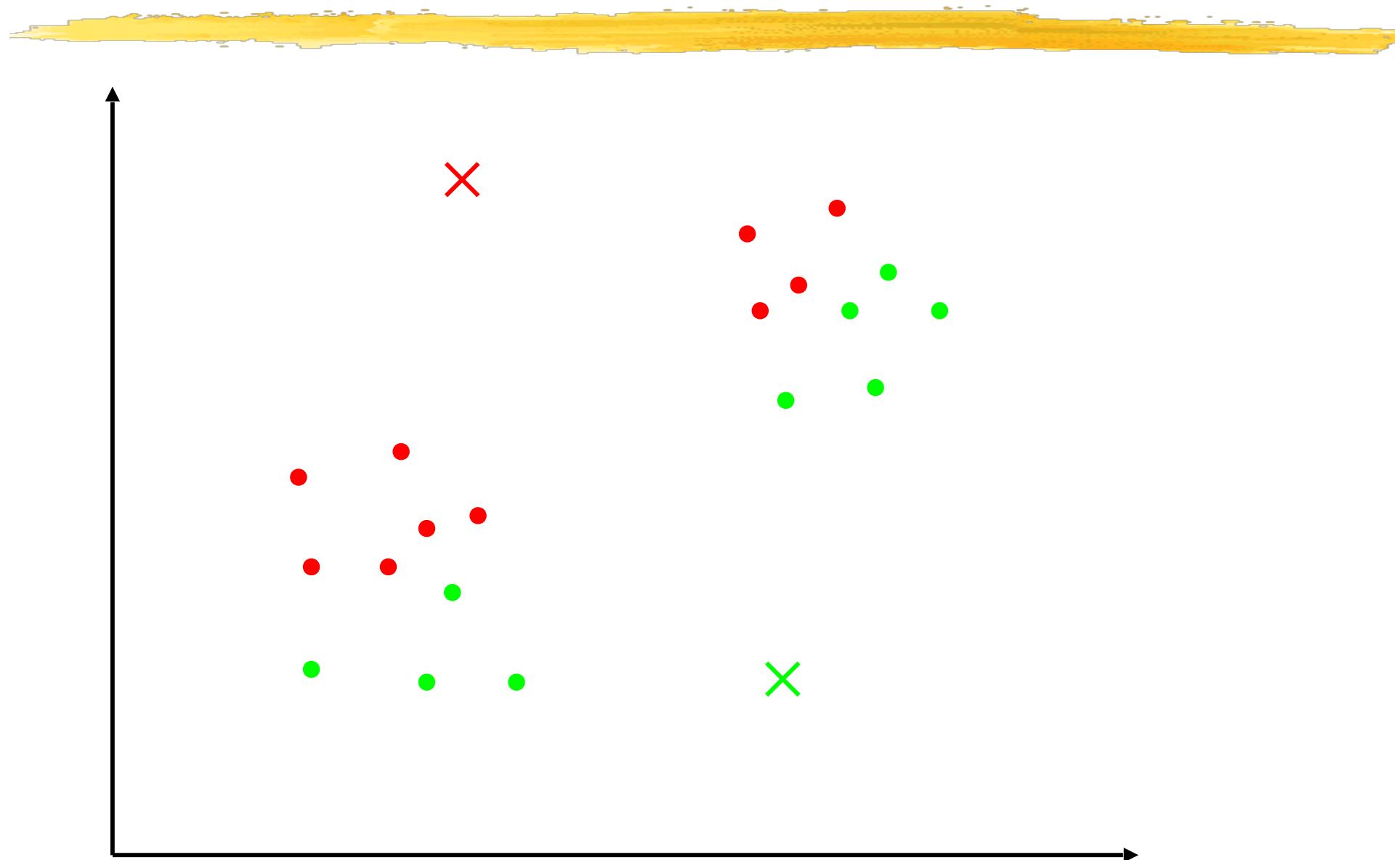
K-MEANS CLUSTERING



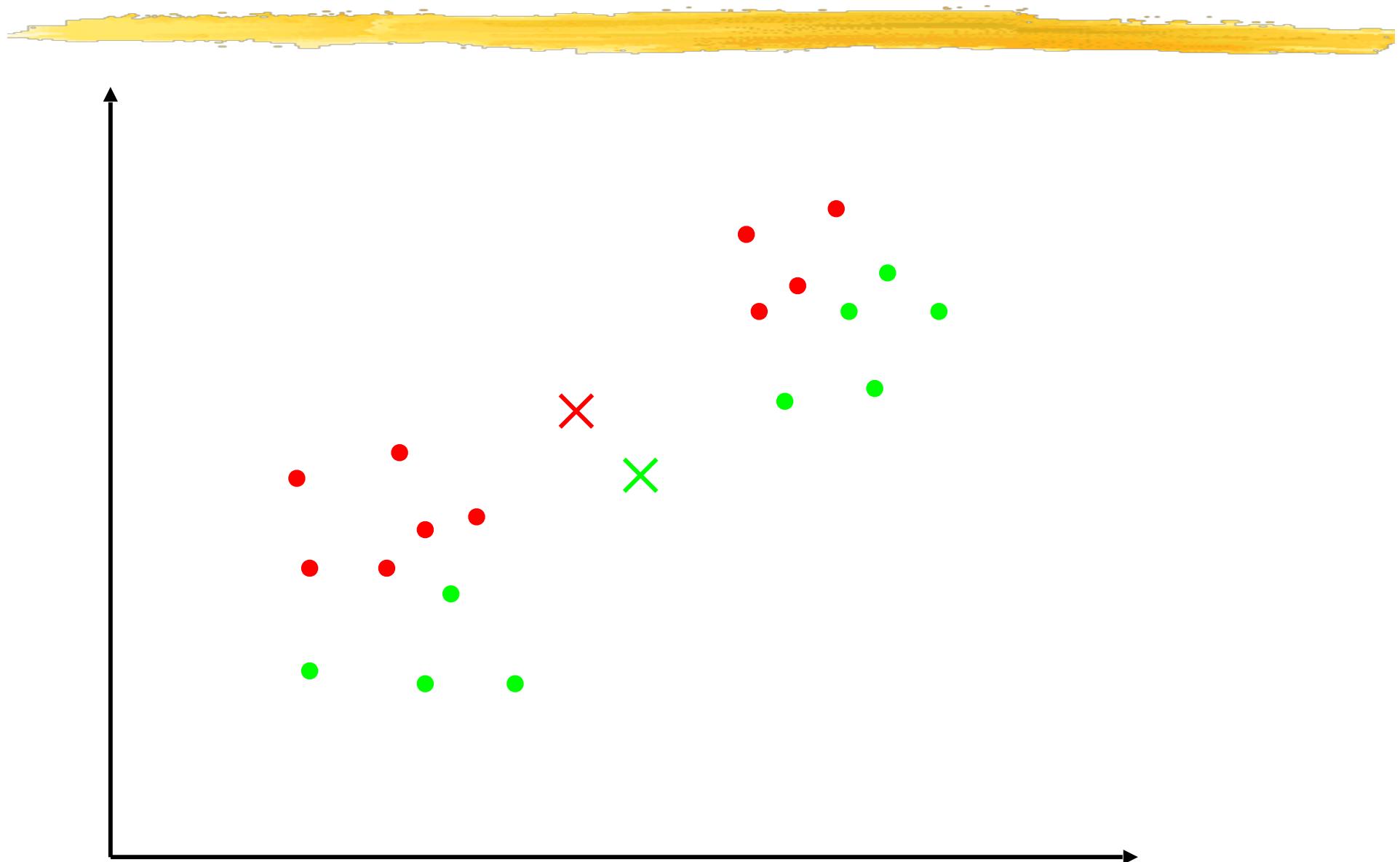
K-MEANS CLUSTERING



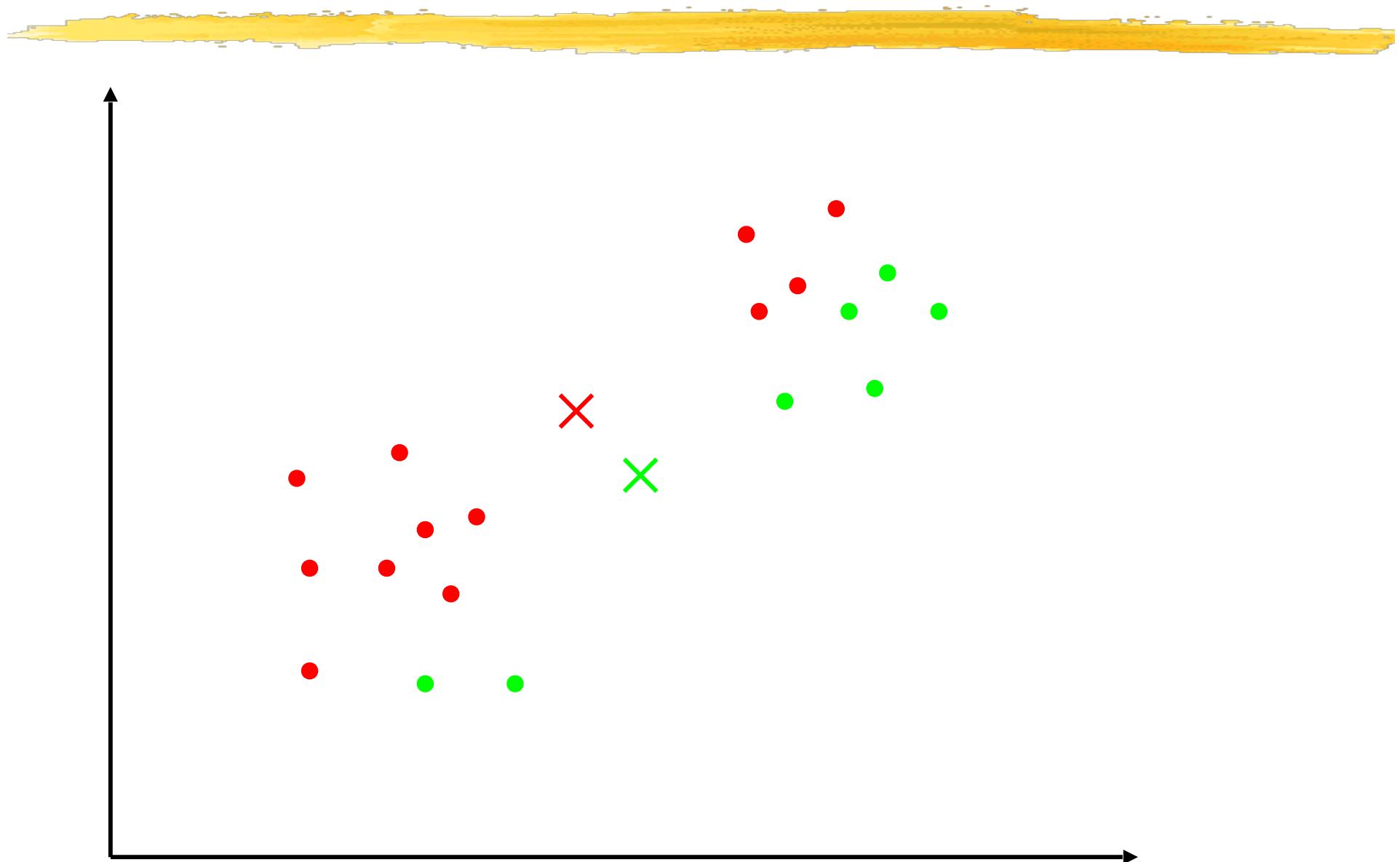
K-MEANS CLUSTERING



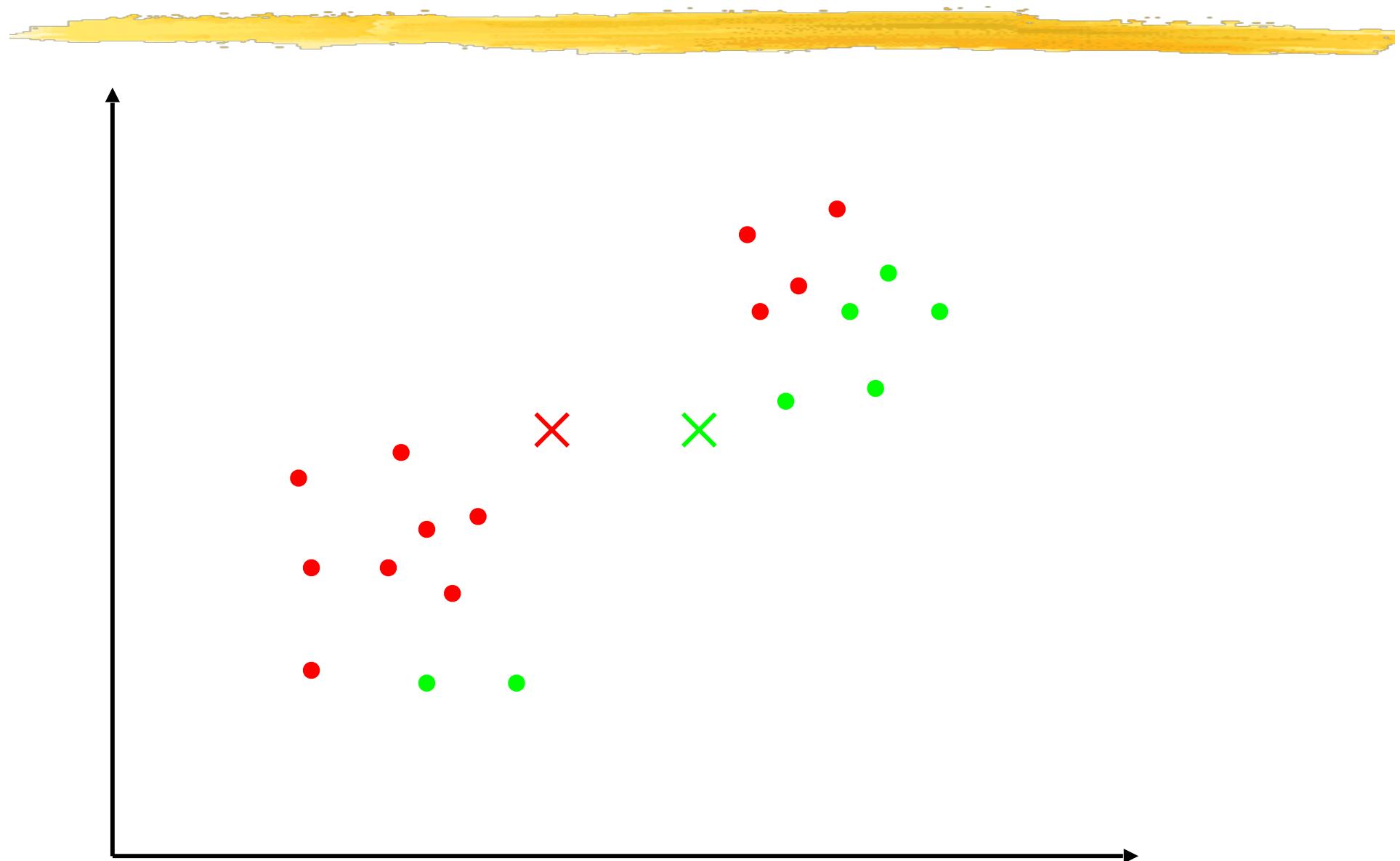
K-MEANS CLUSTERING



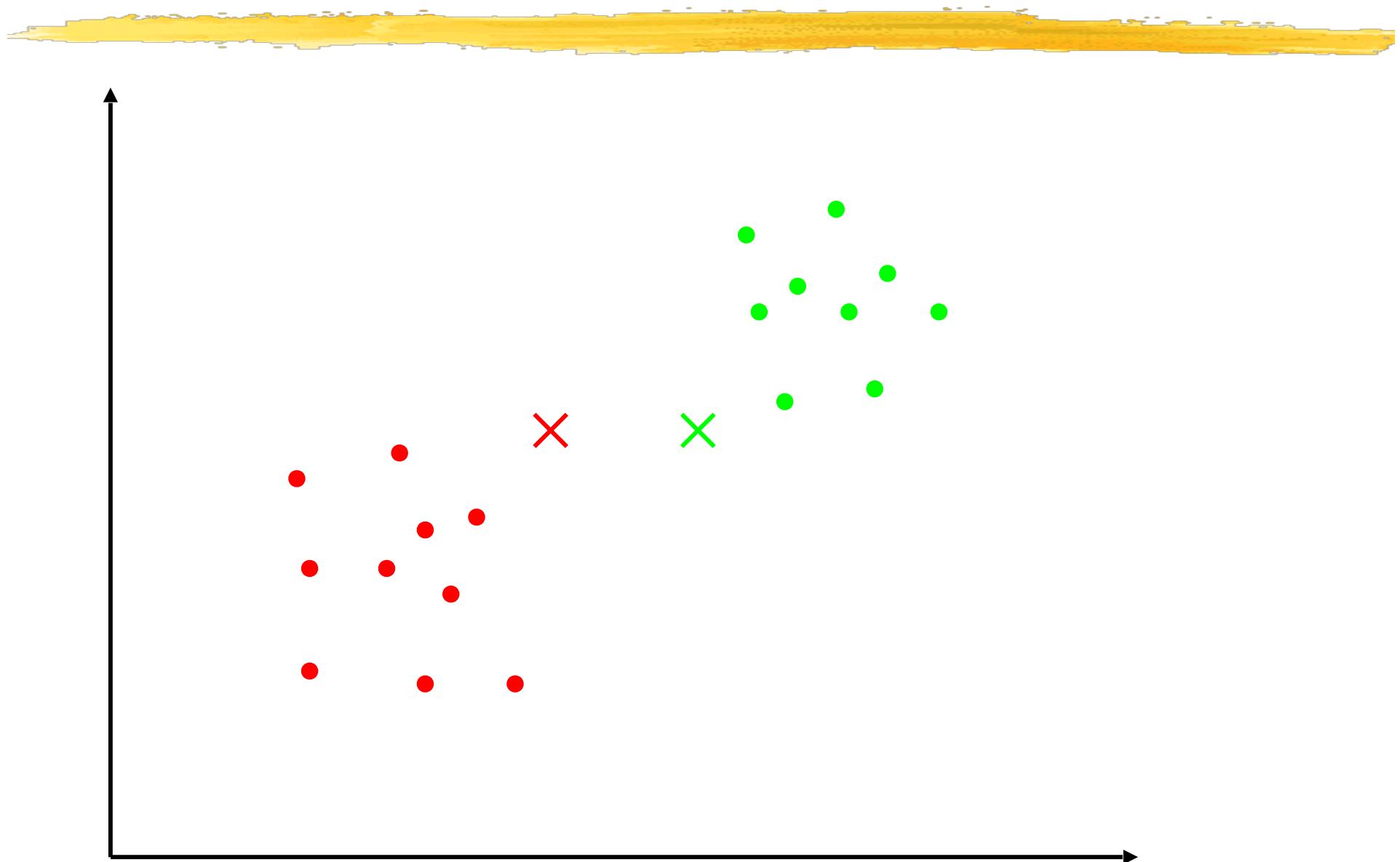
K-MEANS CLUSTERING



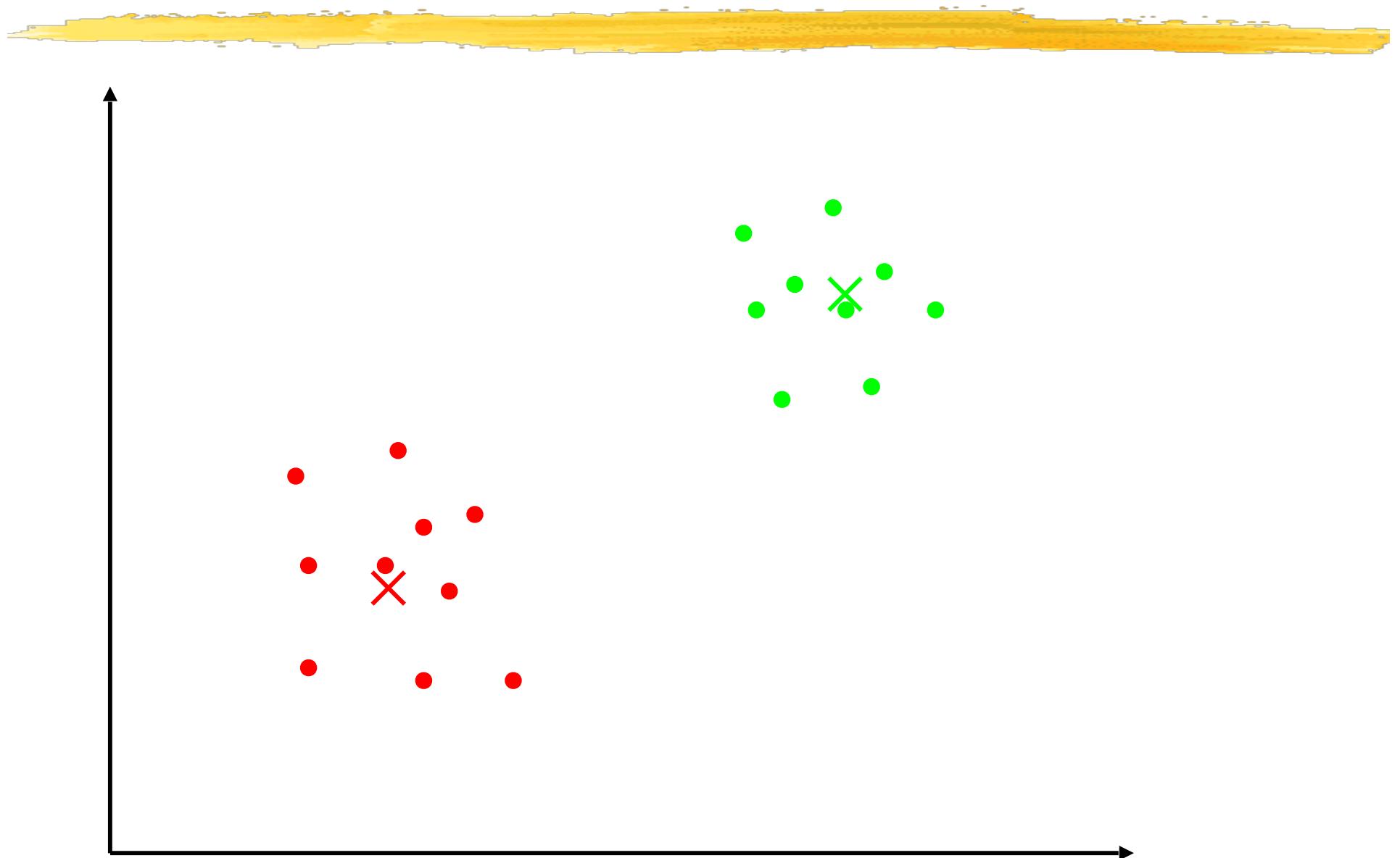
K-MEANS CLUSTERING



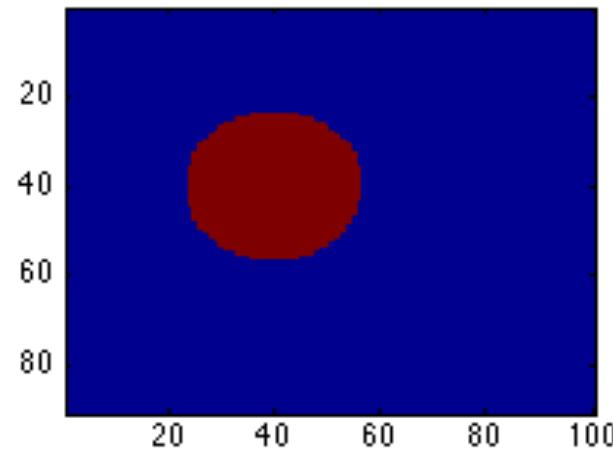
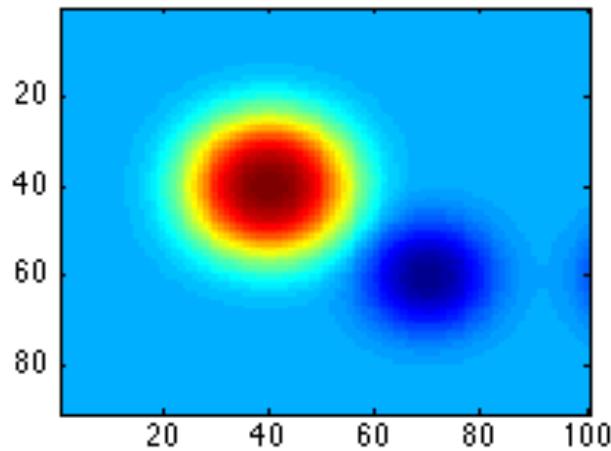
K-MEANS CLUSTERING



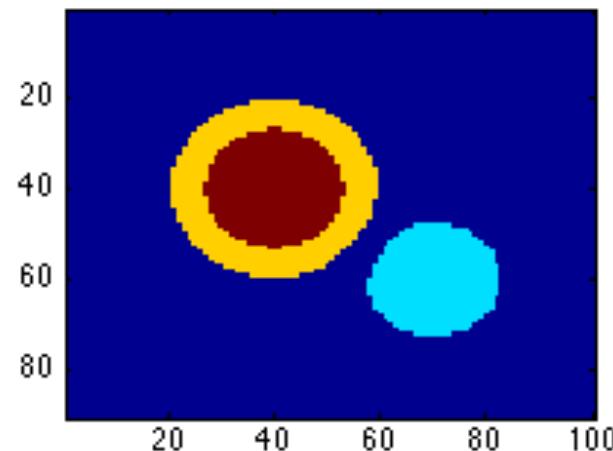
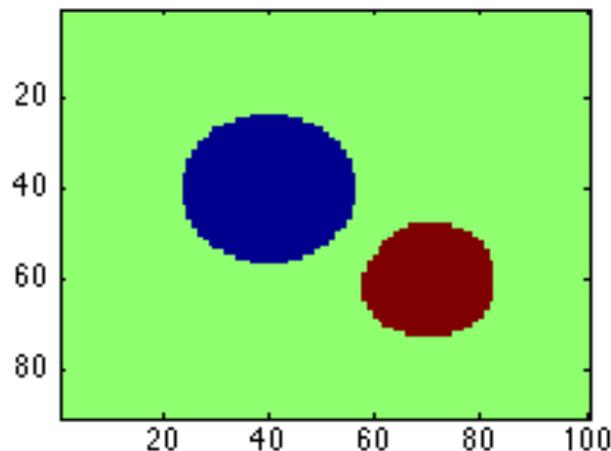
K-MEANS CLUSTERING



SYNTHETIC CASE



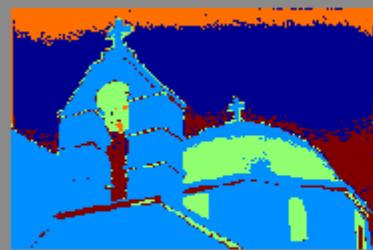
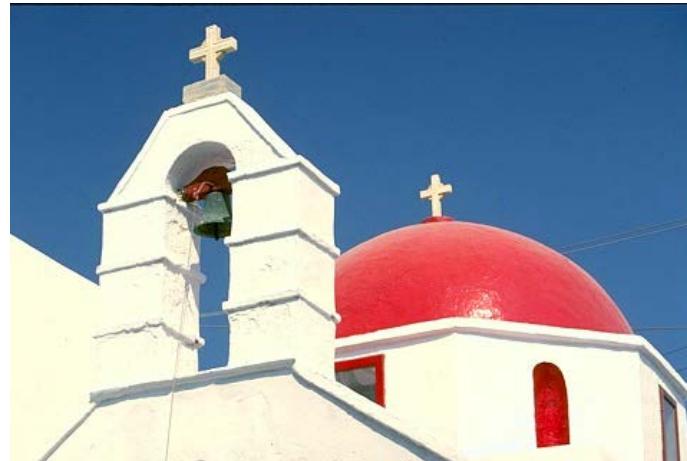
K=2



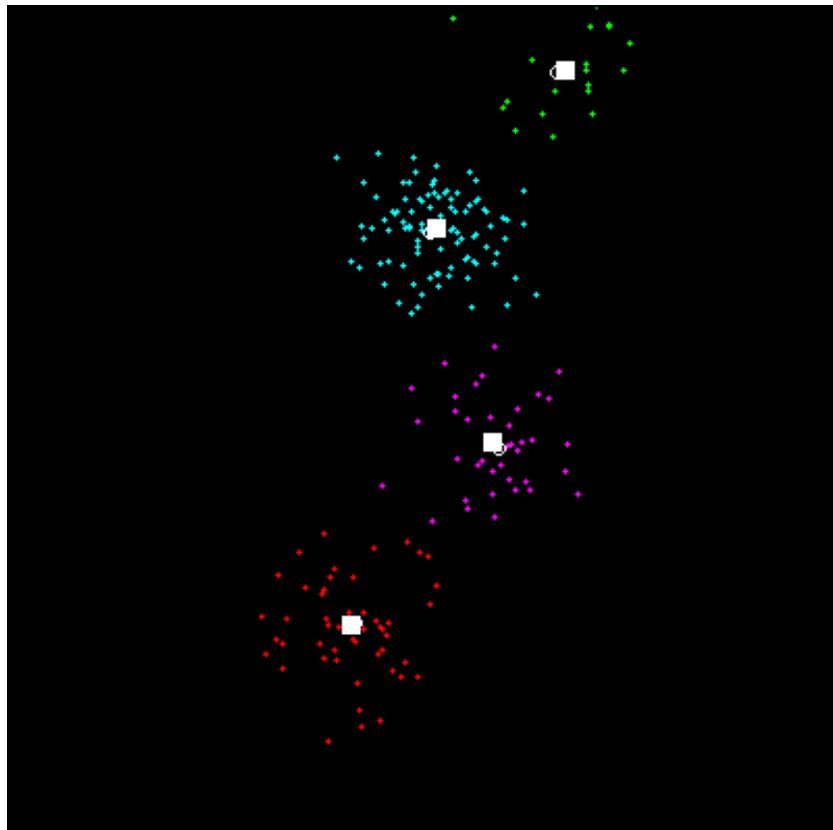
K=3

K=4

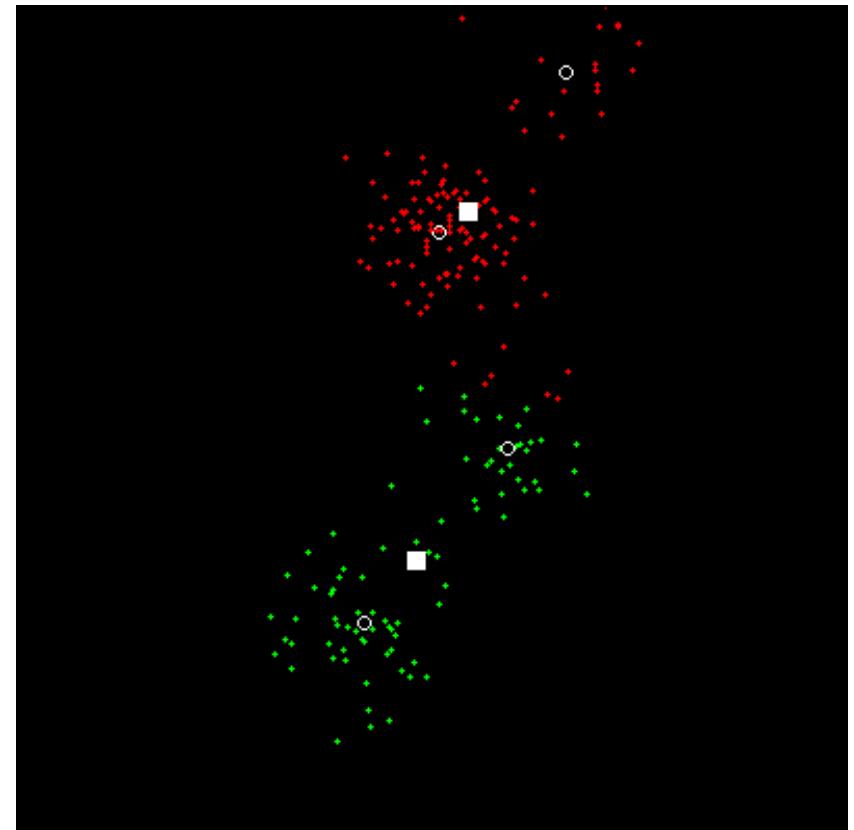
8 ITERATIONS FOR K=5



INITIAL CONDITIONS MATTER

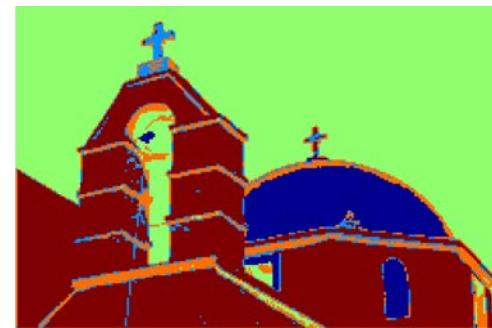
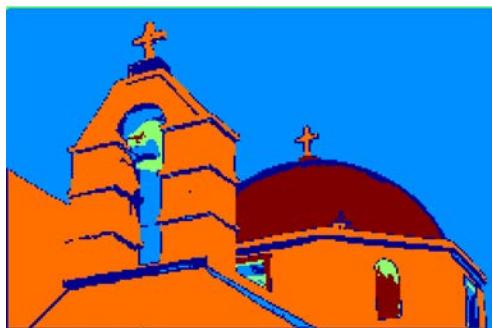


Initially, the points are assigned randomly to each one of the clusters.

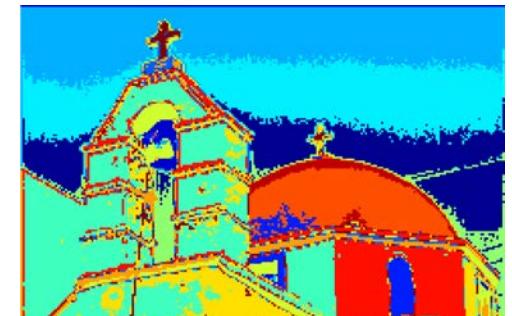


Initially, the points are assigned to the closest cluster.

K-MEANS RESULTS



Random Initialization



K=3

K=5

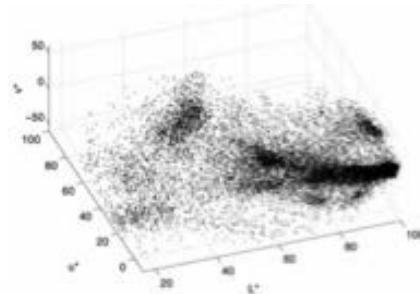
K=8

K=15

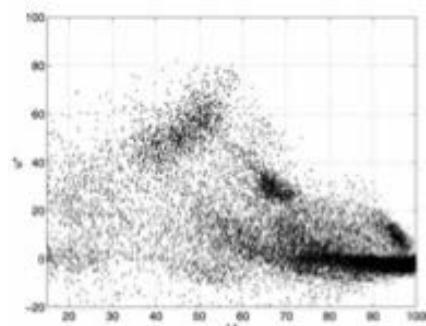
FROM IMAGES TO PROBABILITY DENSITY FUNCTION



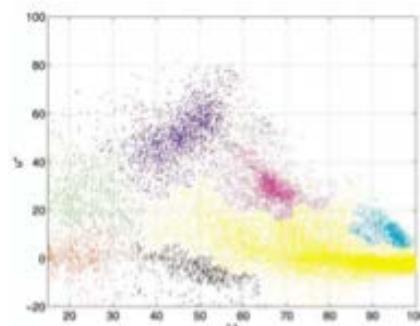
(a)



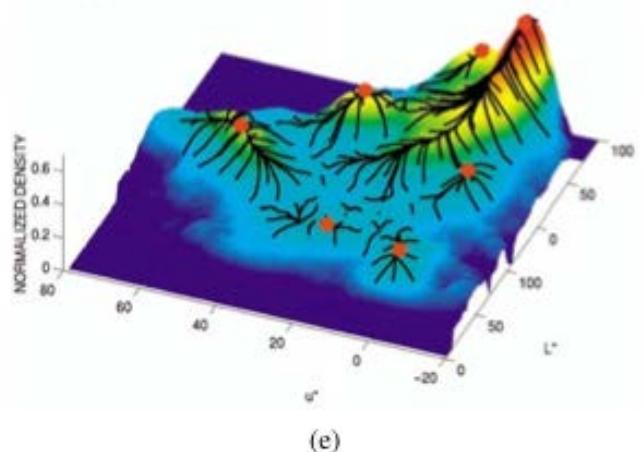
(b)



(c)



(d)



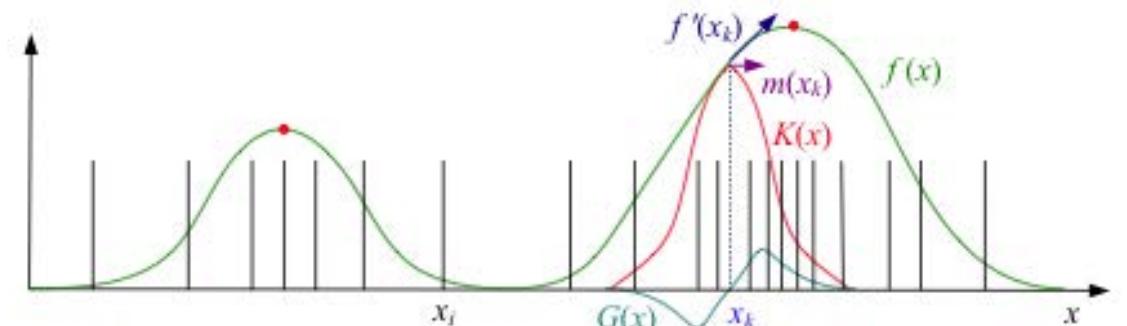
(e)

- Image pixels can be thought of samples of a probability distribution function.
 - Regions then become major peaks in that probability density function.
 - A way to estimate this probability density function is needed.

PROBABILITY DENSITY FUNCTION AND ITS GRADIENT



$$f(\mathbf{x}) = \sum_i K\left(\frac{\|\mathbf{x} - \mathbf{x}_i\|^2}{h^2}\right)$$

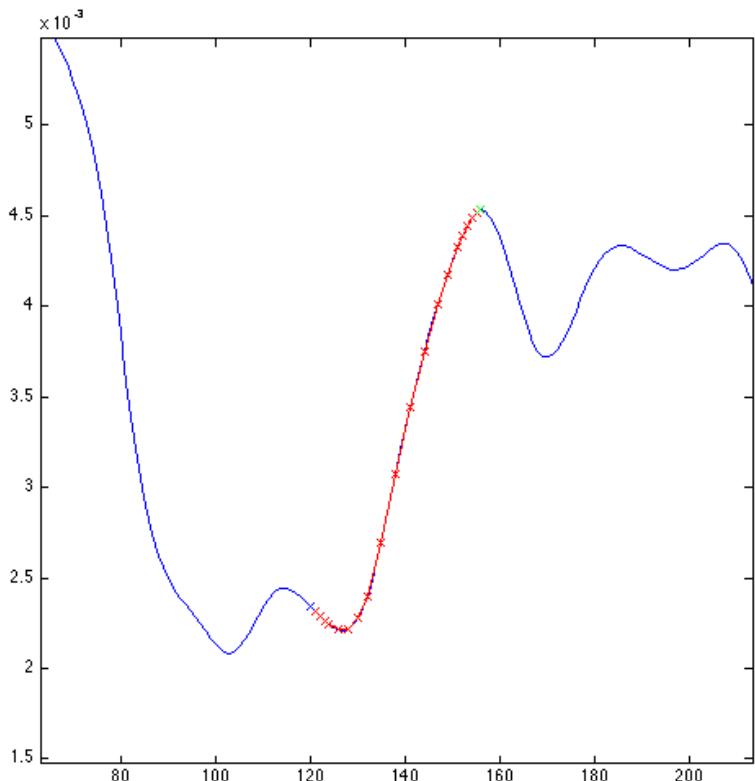


$$\nabla f(\mathbf{x}) = \sum_i (\mathbf{x}_i - \mathbf{x}) G(\mathbf{x} - \mathbf{x}_i) \text{ with } G(\mathbf{x} - \mathbf{x}_i) = -K'\left(\frac{\|\mathbf{x} - \mathbf{x}_i\|^2}{h^2}\right)$$

$$= \left[\sum_i G(\mathbf{x} - \mathbf{x}_i) \right] \mathbf{m}(\mathbf{x}) \text{ with } \mathbf{m}(\mathbf{x}) = \frac{\sum_i \mathbf{x}_i G(\mathbf{x} - \mathbf{x}_i)}{\sum_i G(\mathbf{x} - \mathbf{x}_i)} - \mathbf{x}$$

- **m(x)** is known as the mean shift because it is the difference between the weighted mean of the values of the neighbors of \mathbf{x} and that of \mathbf{x} itself.
- **m(x)** is the direction of steepest ascent.

1D MEAN-SHIFT PROCEDURE



$$\mathbf{y}_{k+1} = \mathbf{y}_k + \mathbf{m}(\mathbf{y}_k) = \frac{\sum_i \mathbf{x}_i G(\mathbf{y} - \mathbf{x}_i)}{\sum_i G(\mathbf{y} - \mathbf{x}_i)}$$

$$k_e(r) = \max(0, 1 - r) \Rightarrow g(r) = \begin{cases} 0 & \text{if } r < 1 \\ 1 & \text{otherwise} \end{cases}$$

$$k_n(r) = \exp(-r/2) \Rightarrow g(r) = \frac{1}{2} \exp(-r/2)$$

3D MEAN-SHIFT PROCEDURE

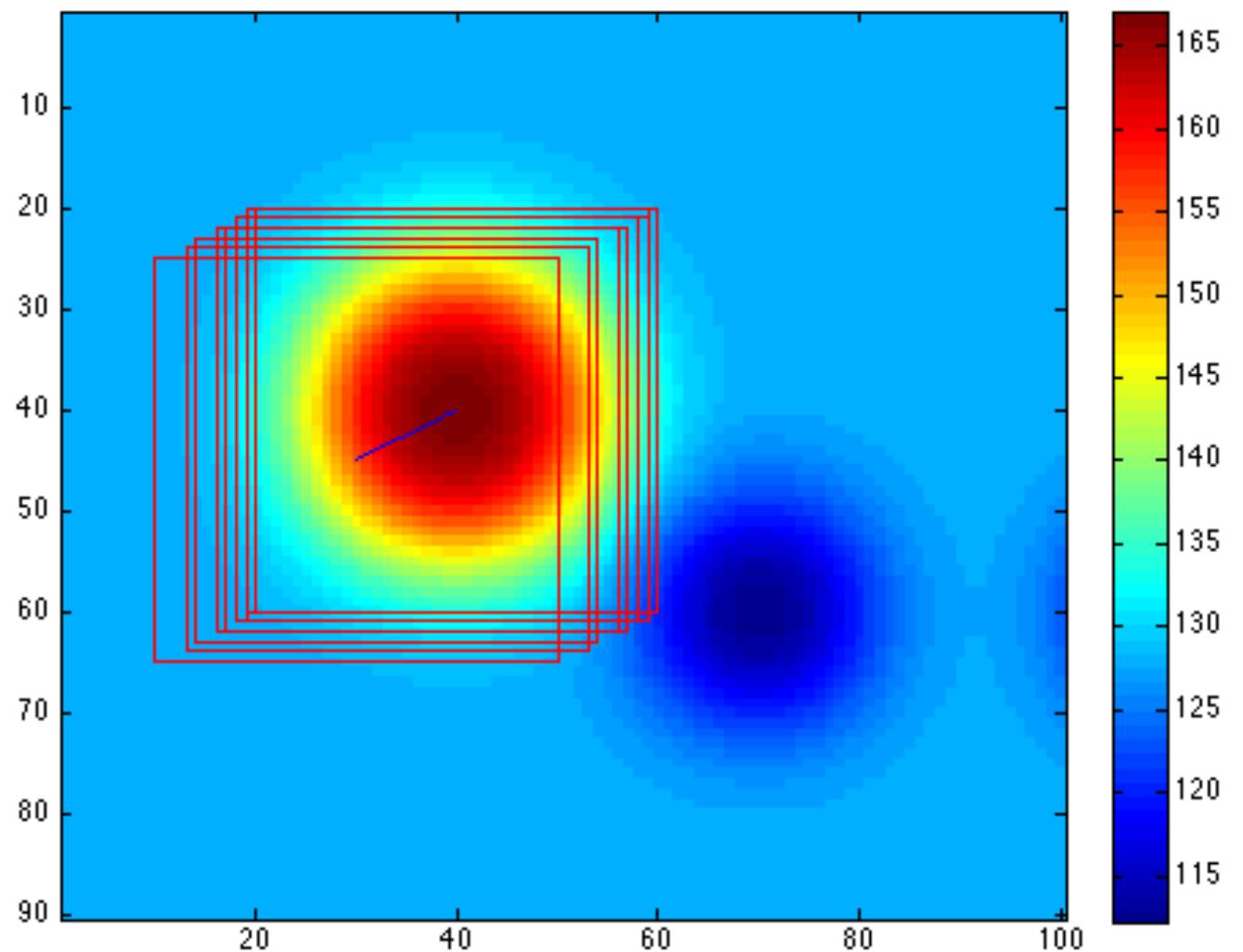


$$\mathbf{y}_{k+1} = \mathbf{y}_k + \mathbf{m}(\mathbf{y}_k) = \frac{\sum_i \mathbf{x}_i G(\mathbf{y} - \mathbf{x}_i)}{\sum_i G(\mathbf{y} - \mathbf{x}_i)}$$

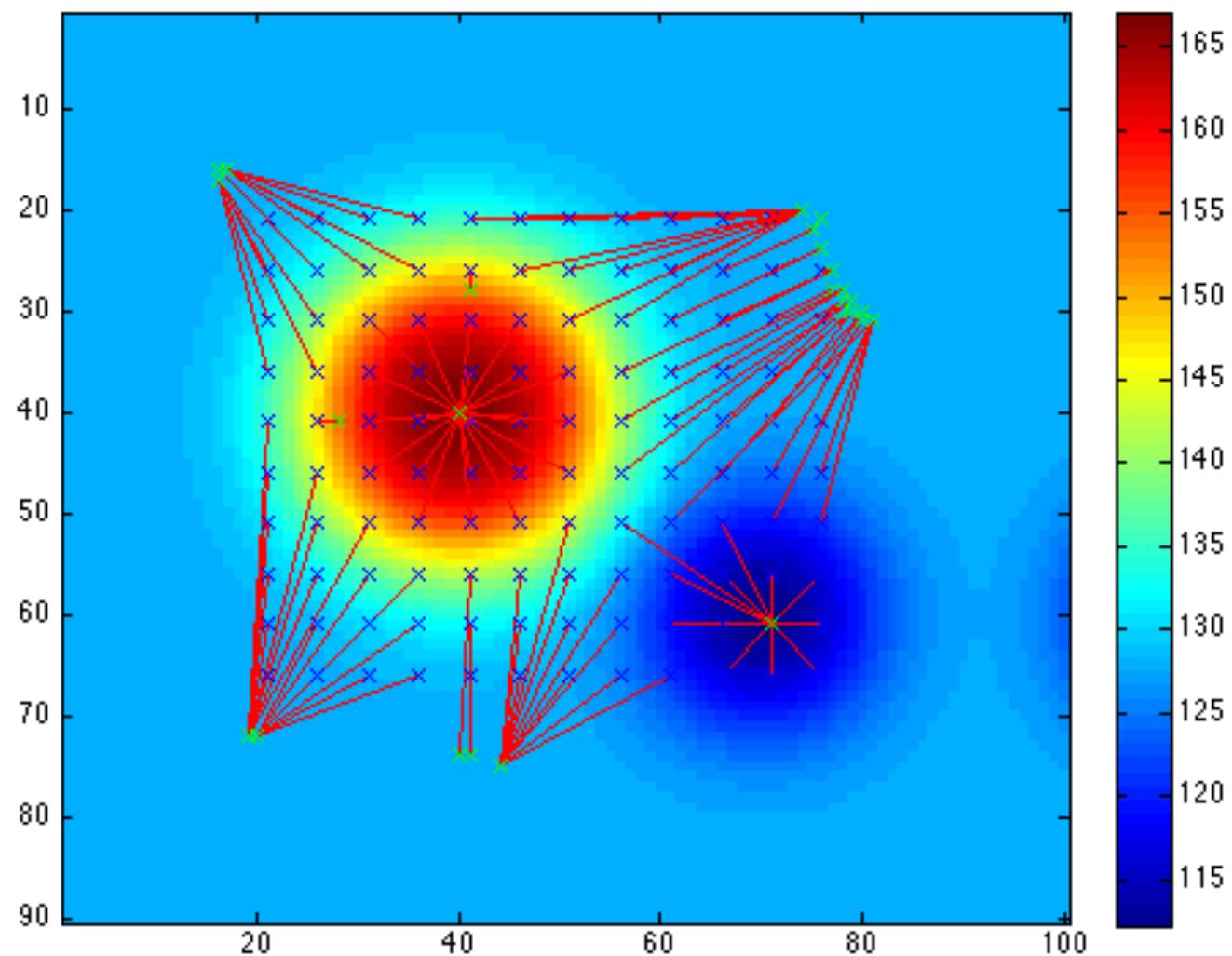
$$\mathbf{x} = \begin{bmatrix} u \\ v \\ g \end{bmatrix}$$

$$K(\mathbf{x}) = k_n \left(\frac{u^2 + v^2}{h_s^2} \right) k \left(\frac{g^2}{h_r^2} \right)$$

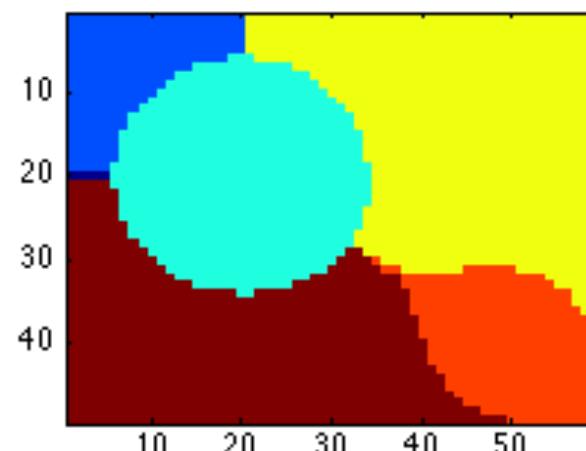
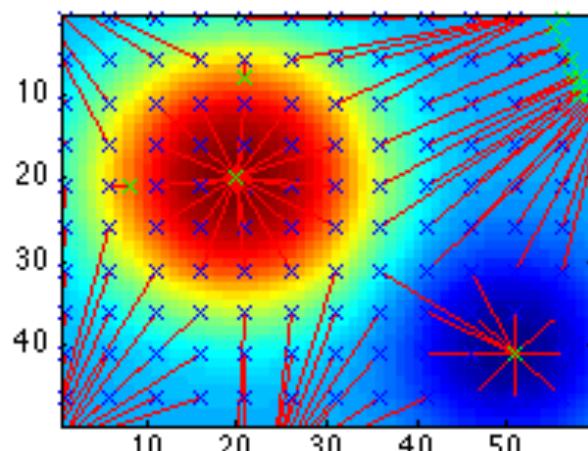
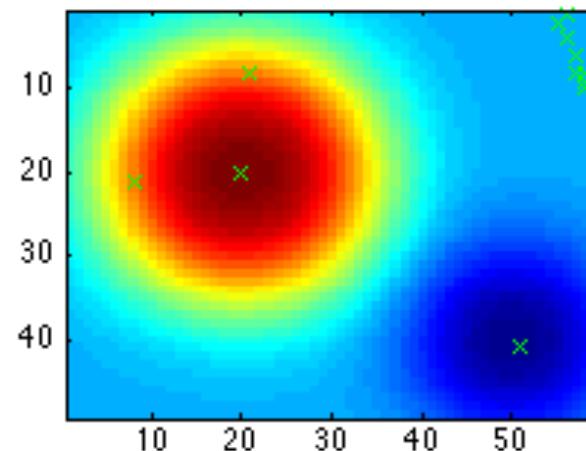
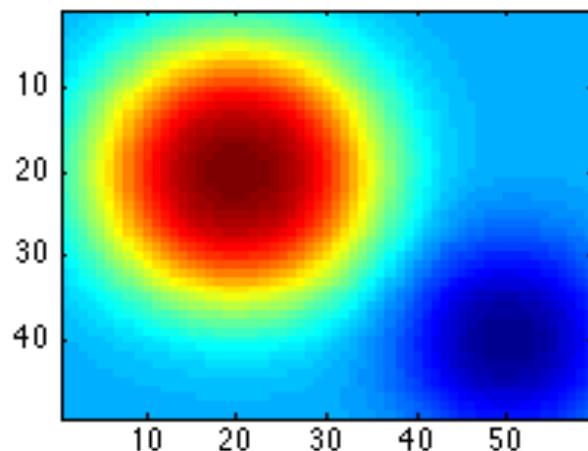
$$G(\mathbf{x}) = \exp \left(-\left(\frac{u^2 + v^2}{h_s^2} + \frac{g^2}{h_r^2} \right) \right)$$



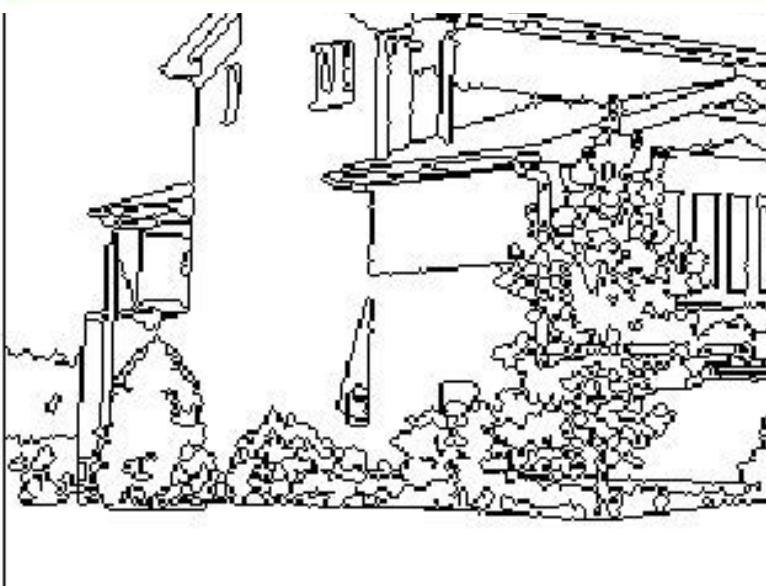
MEAN SHIFT MODES



MEAN SHIFT CLUSTERING



5D MEAN-SHIFT



$$\mathbf{y}_{k+1} = \mathbf{y}_k + \mathbf{m}(\mathbf{y}_k) = \frac{\sum_i \mathbf{x}_i G(\mathbf{y} - \mathbf{x}_i)}{\sum_i G(\mathbf{y} - \mathbf{x}_i)}$$

$$\mathbf{x} = \begin{bmatrix} u \\ v \\ R \\ G \\ V \end{bmatrix} \text{ or } \begin{bmatrix} u \\ v \\ L \\ a \\ b \end{bmatrix}$$

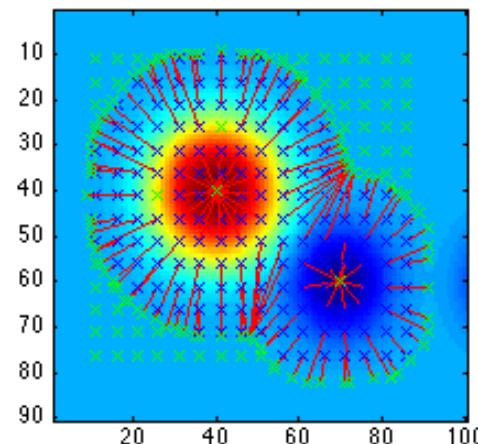
$$G(\mathbf{x}) = \exp\left(-\left(\frac{u^2 + v^2}{h_s^2} + \frac{R^2 + G^2 + B^2}{h_r^2}\right)\right)$$

Ohlander & Price,
CGIP'78

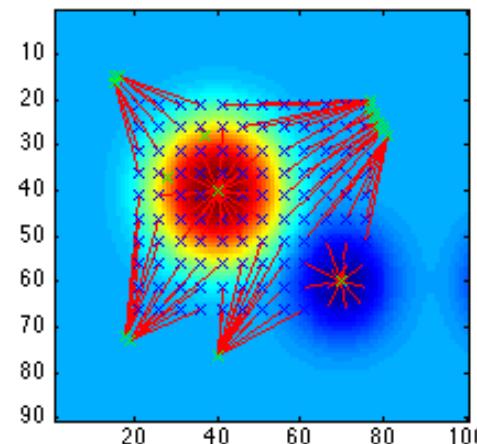
Comaniciu & Meers,
PAMI'02

PARAMETERS

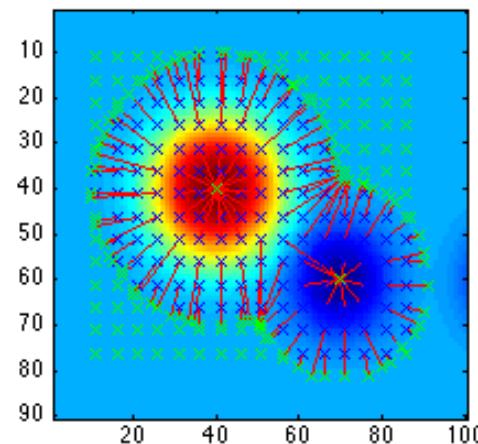
Yellow layer



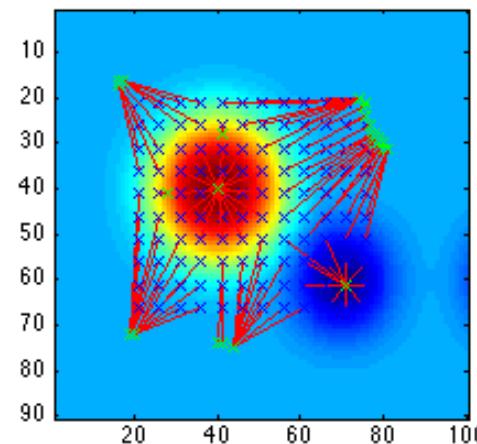
$$h_s = 5, h_r = 5$$



$$h_s = 10, h_r = 5$$

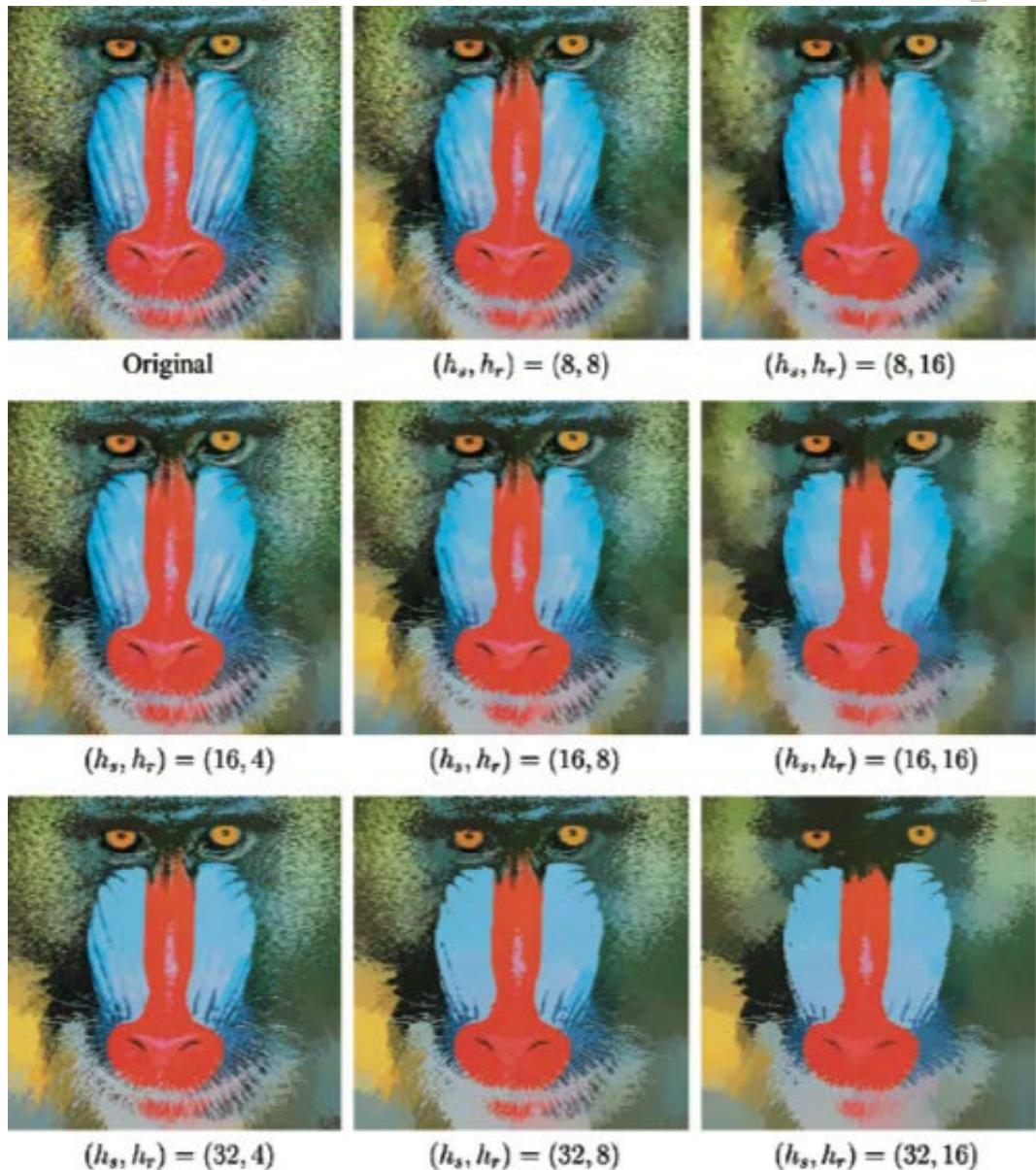


$$h_s = 10, h_r = 10$$



$$h_s = 10, h_r = 10$$

PARAMETERS



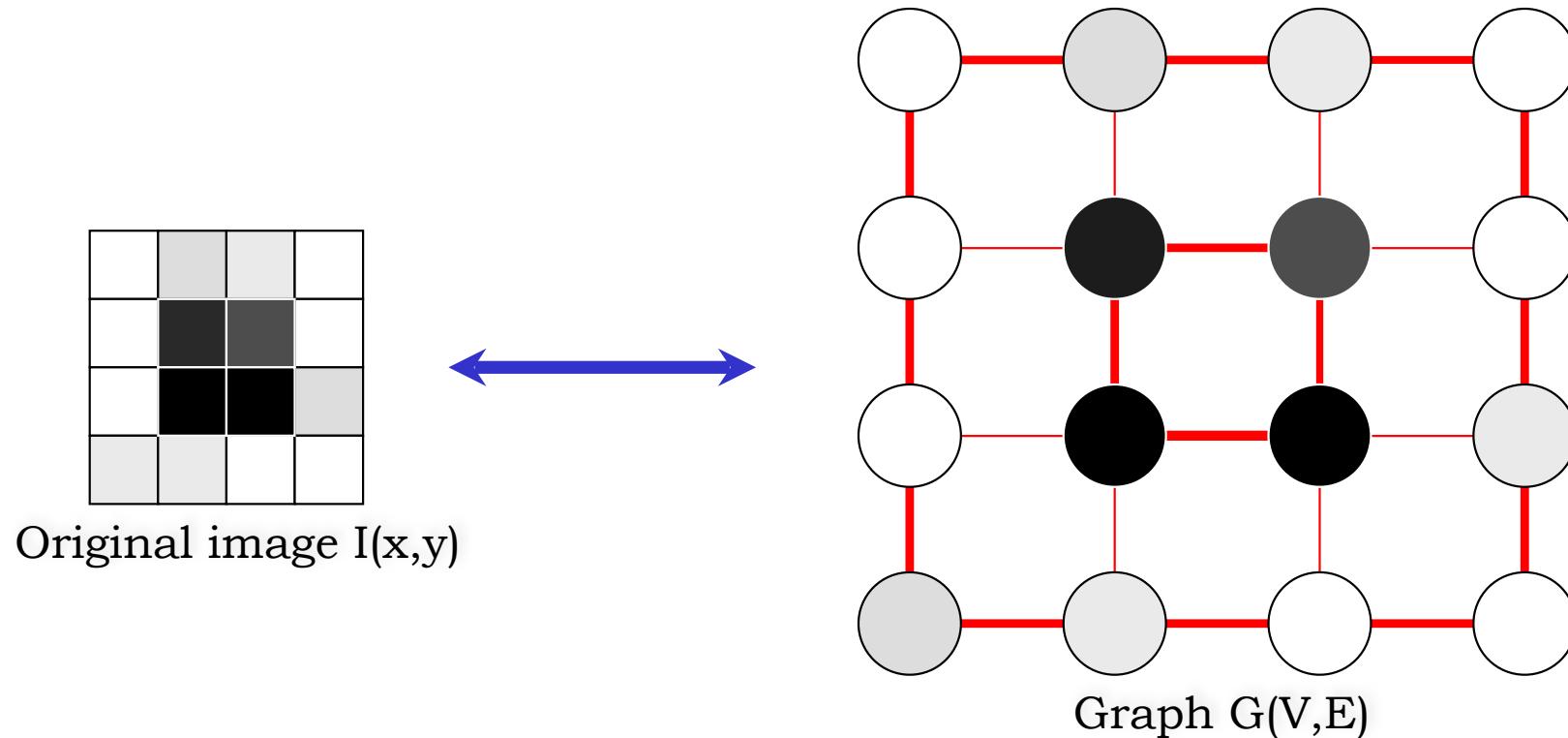
- h_s and h_r control the amount of smoothing in the spatial and color domains.
- They can be taken to be the ones that give the most stable response to small perturbations.

MORE RESULTS



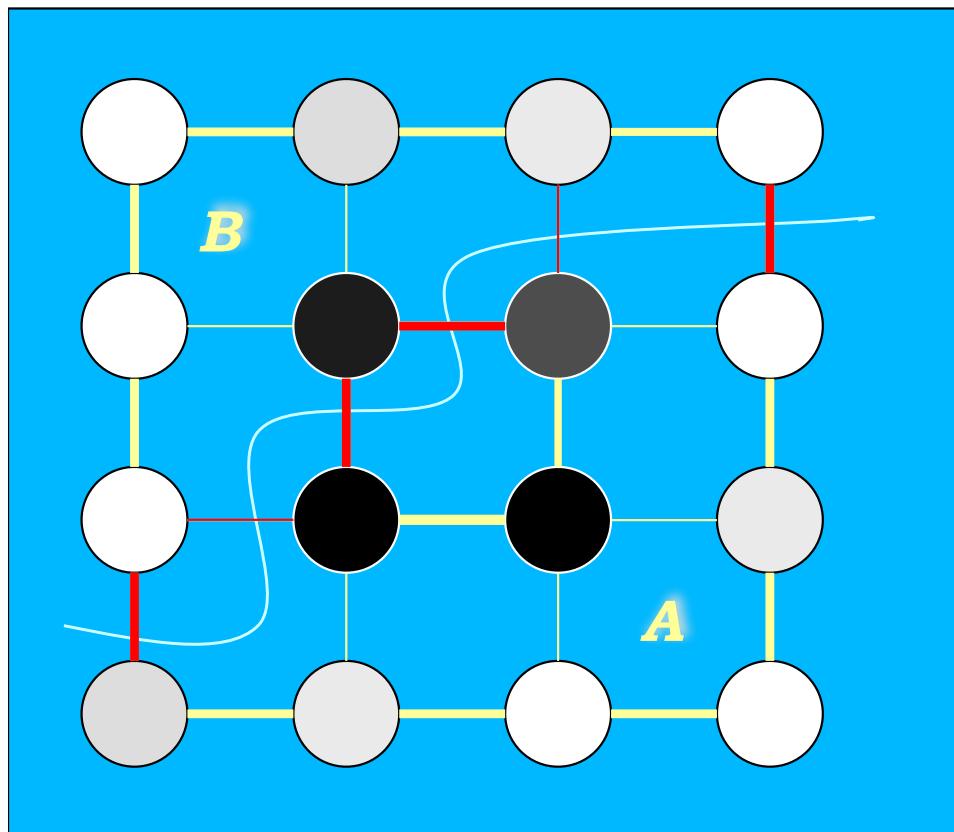
IMAGES AS GRAPHS

An image $I(x,y)$ is equivalent to a graph $G(V,E)$



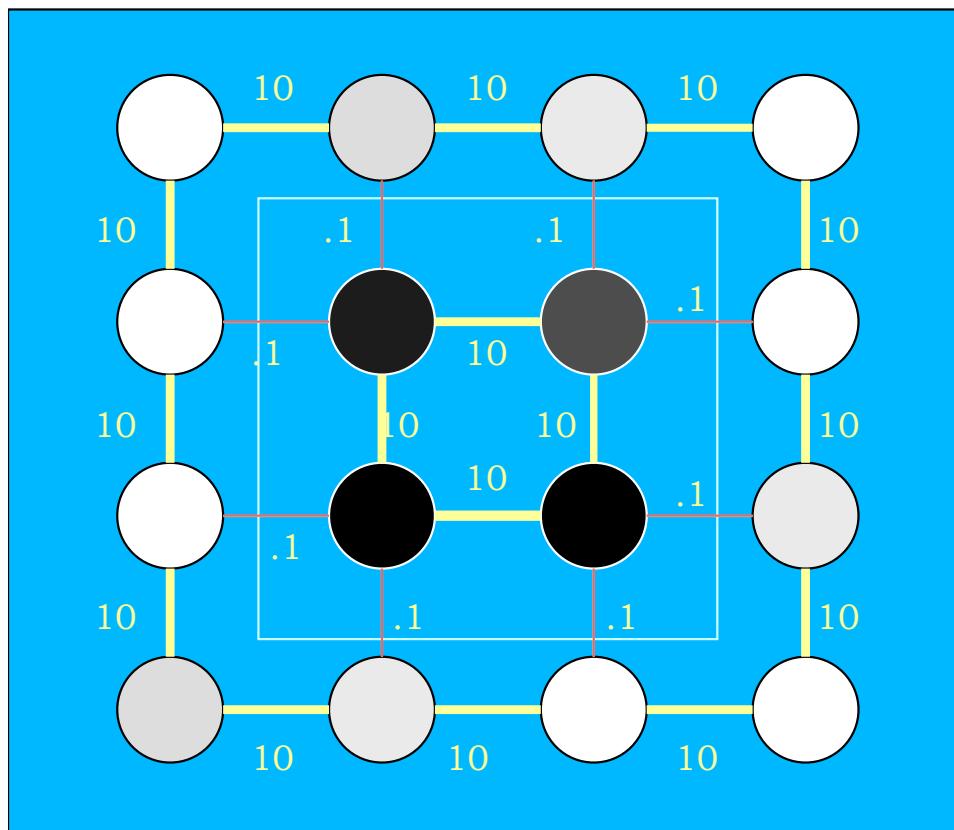
- V is a set of vertices or nodes that represent individual pixels.
- E is a set of edges linking neighboring nodes together. The weight or strength of the edge is proportional to the similarity between the vertices it joins together.

GRAPH-CUT



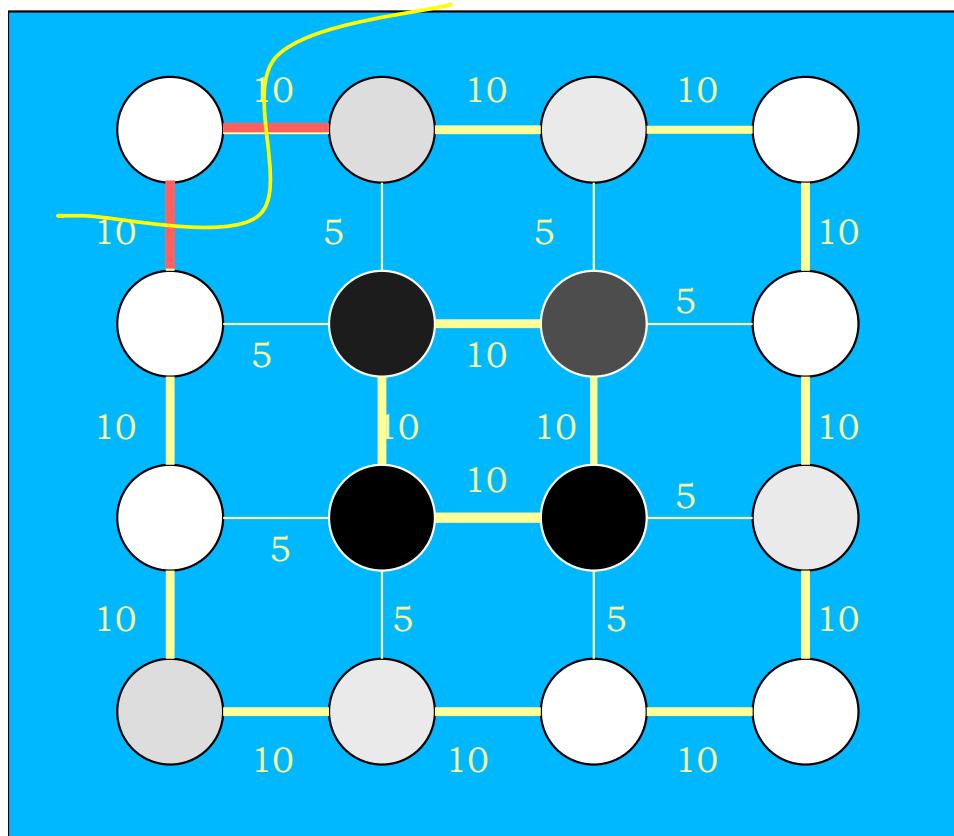
A cut through a graph is defined as the total weight of the links that must be removed to divide it into two separate components.

MIN-CUT



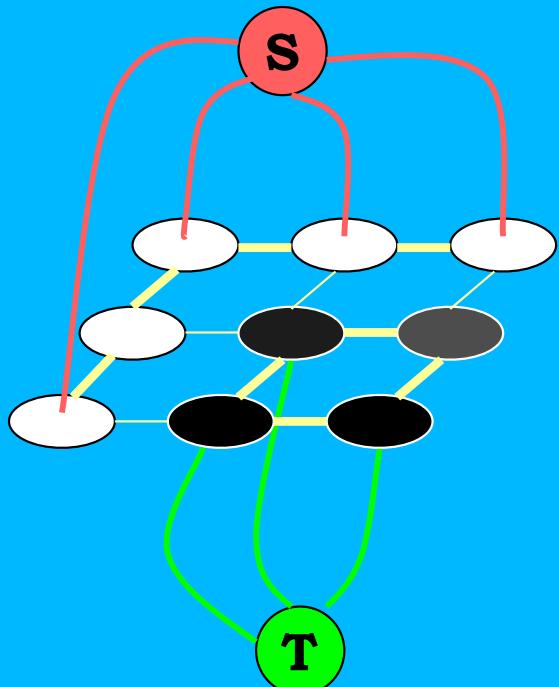
- Find the cut through the graph that has the overall minimum weight, which can be done effectively.
- It should be the subset of edges of least weight that can be removed to partition the graph.
- Since weight encodes similarity, this should be equivalent to partitioning the graph along the boundary of least similarity.

TRIVIAL CUT



- Has a preference for short cuts, which may sometime result in trivial solutions.
- Must be constrained to avoid them.

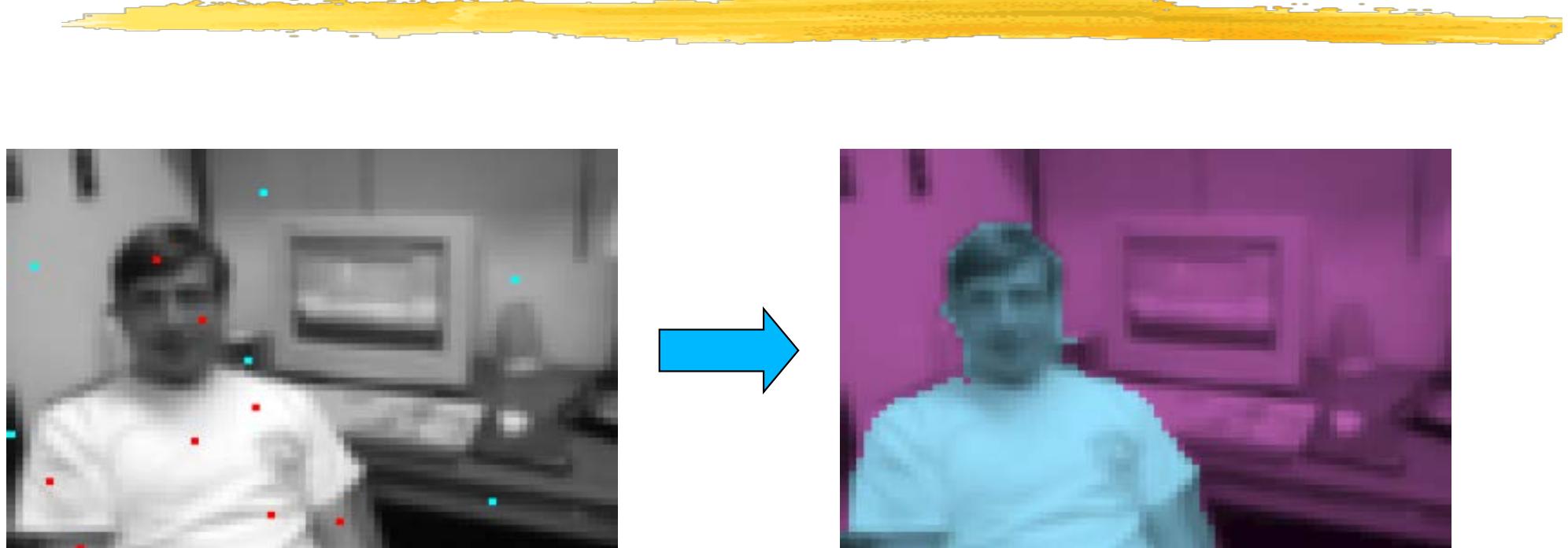
ST MIN-CUT



- Introduce two special nodes called source (S) and sink (T)
- S and T are linked to some image nodes by links of very large weight that will never be selected in a cut.
- Find the minimum cut separating the source from the sink.

--> The problem becomes deciding how to connect S and T to the image nodes.

ST MIN-CUT RESULT

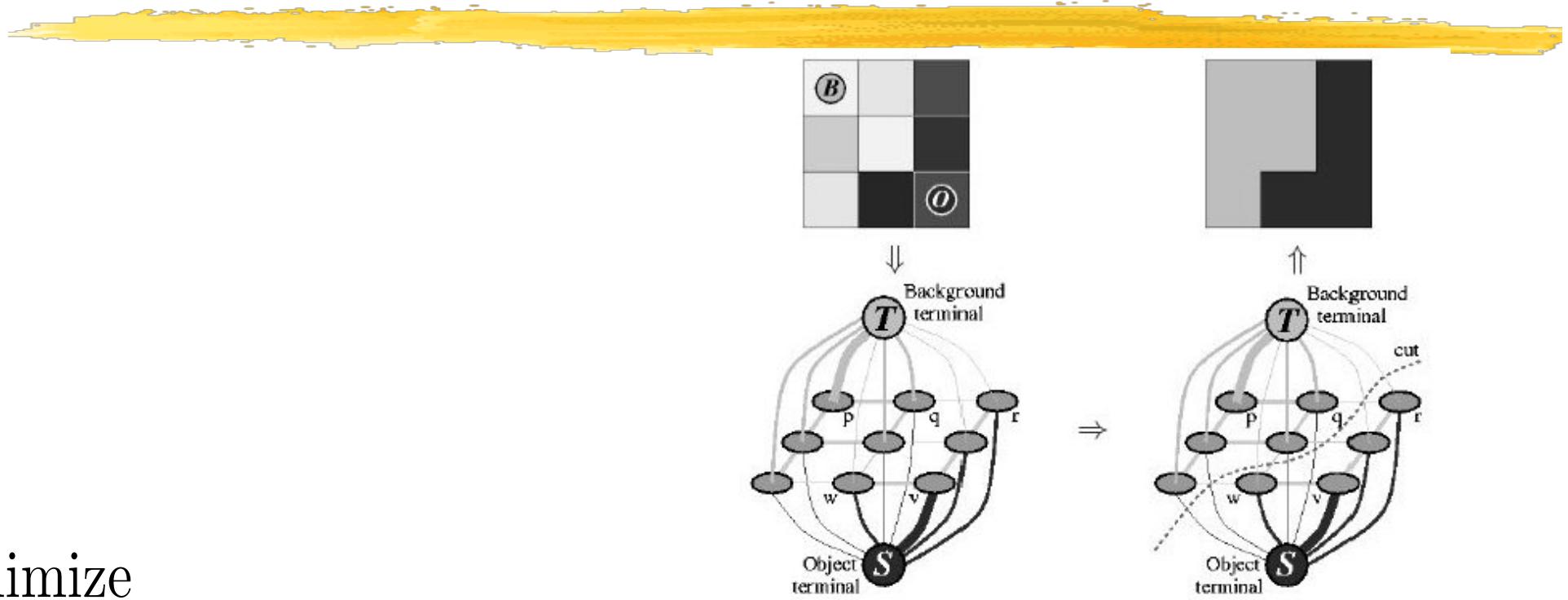


User-selected pixels connected to S (red)
and pixels connected to T (cyan)

Minimum S-T cut

--> If we have a good initial 'guess' to tell us how to link the source and sink to the image, we will get an optimal segmentation.

GRAPH CUT

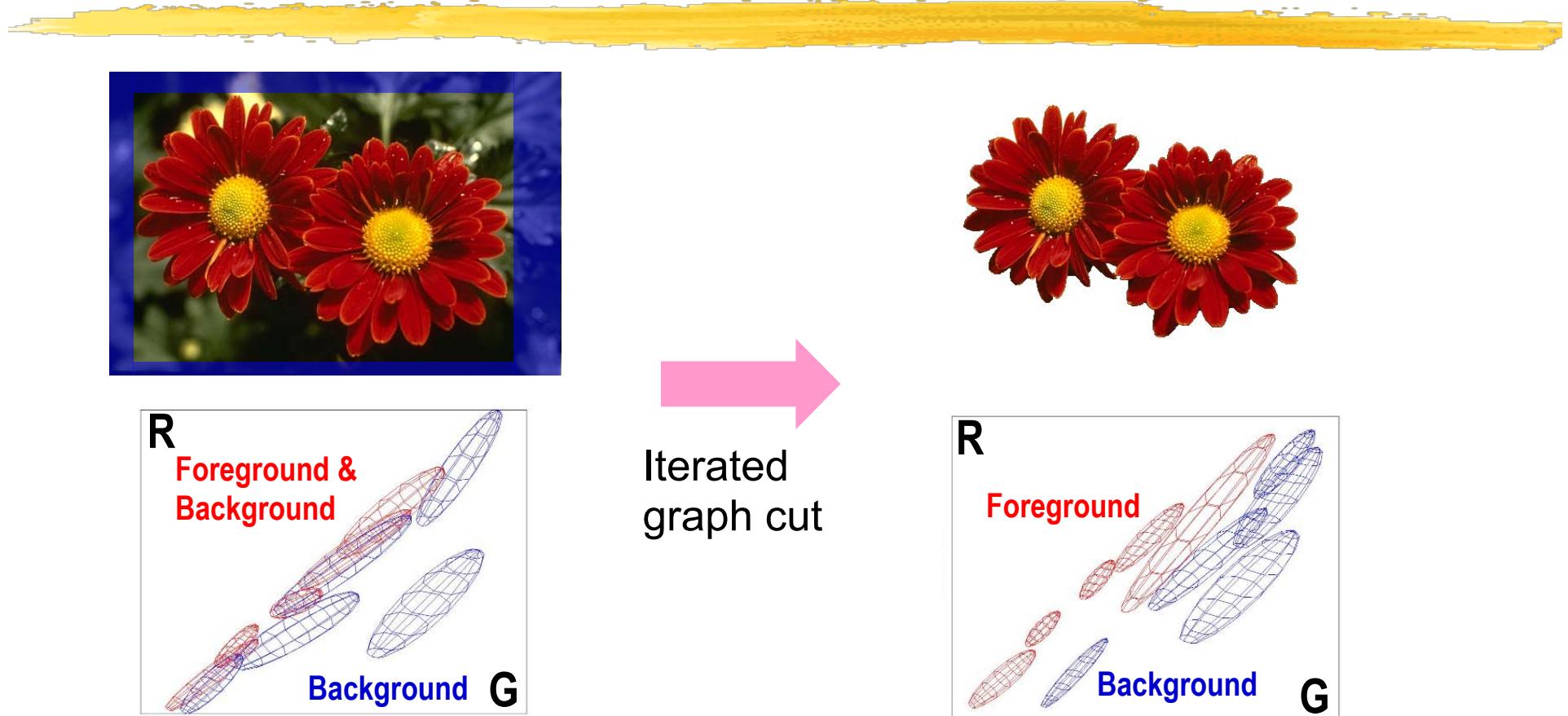


Minimize

$$E(y|x, \lambda) = \sum_i \underbrace{\psi(y_i|x_i)}_{\text{unary term}} + \lambda \sum_{(i,j) \in \mathcal{E}} \underbrace{\phi(y_i, y_j|x_i, x_j)}_{\text{pairwise term}},$$

with respect to y .

INTERACTIVE FOREGROUND EXTRACTION



- K-means to learn color distributions
- Graph cuts to infer the segmentation

GrabCut
Rother & al. SIGGRAPH 04

RELATIVELY EASY EXAMPLES



GrabCut
Rother & al. SIGGRAPH 04

MORE DIFFICULT EXAMPLES



Camouflage &
Low Contrast



Initial
Rectangle

Fine structure



Initial
Result

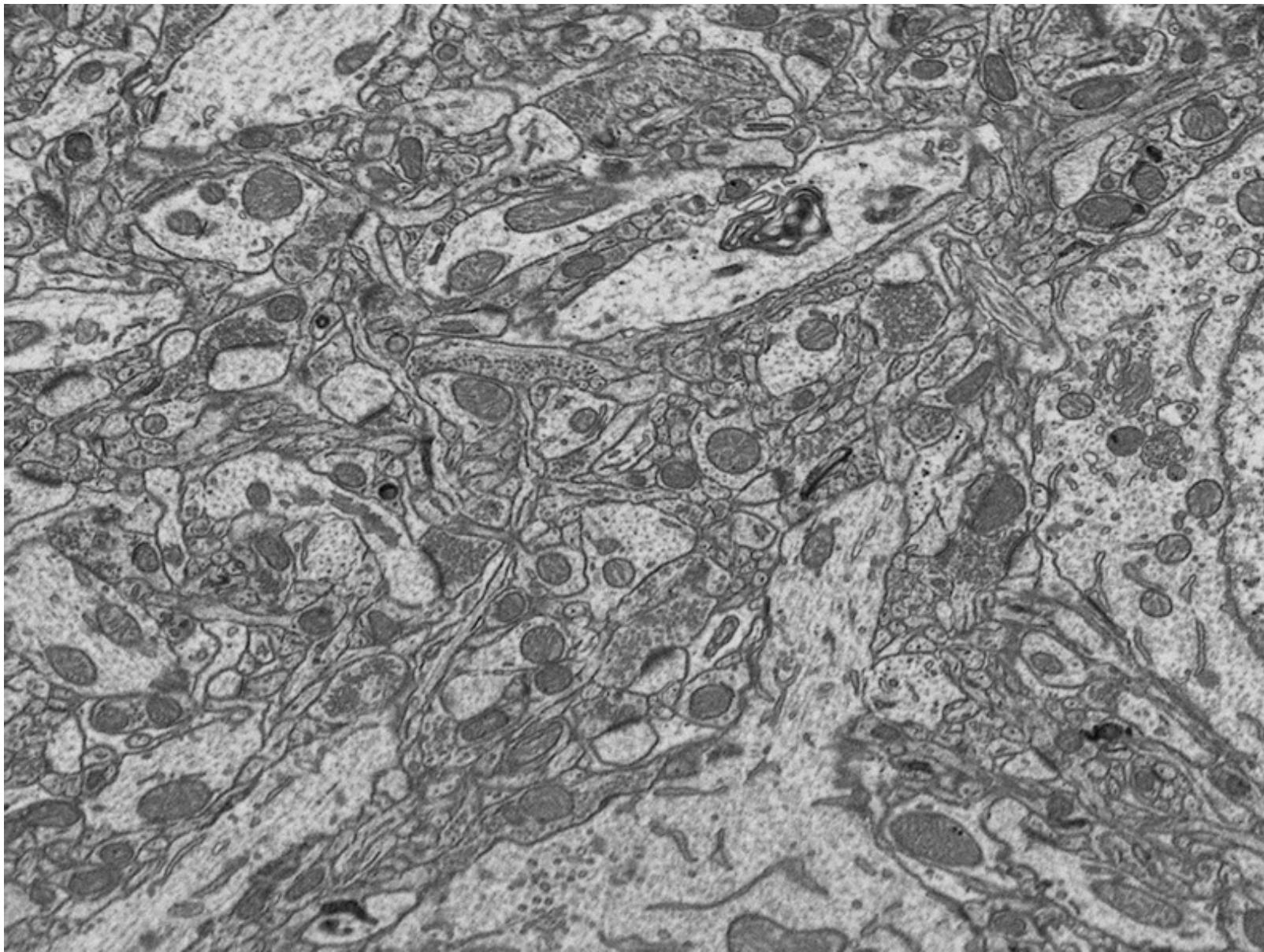


No telepathy

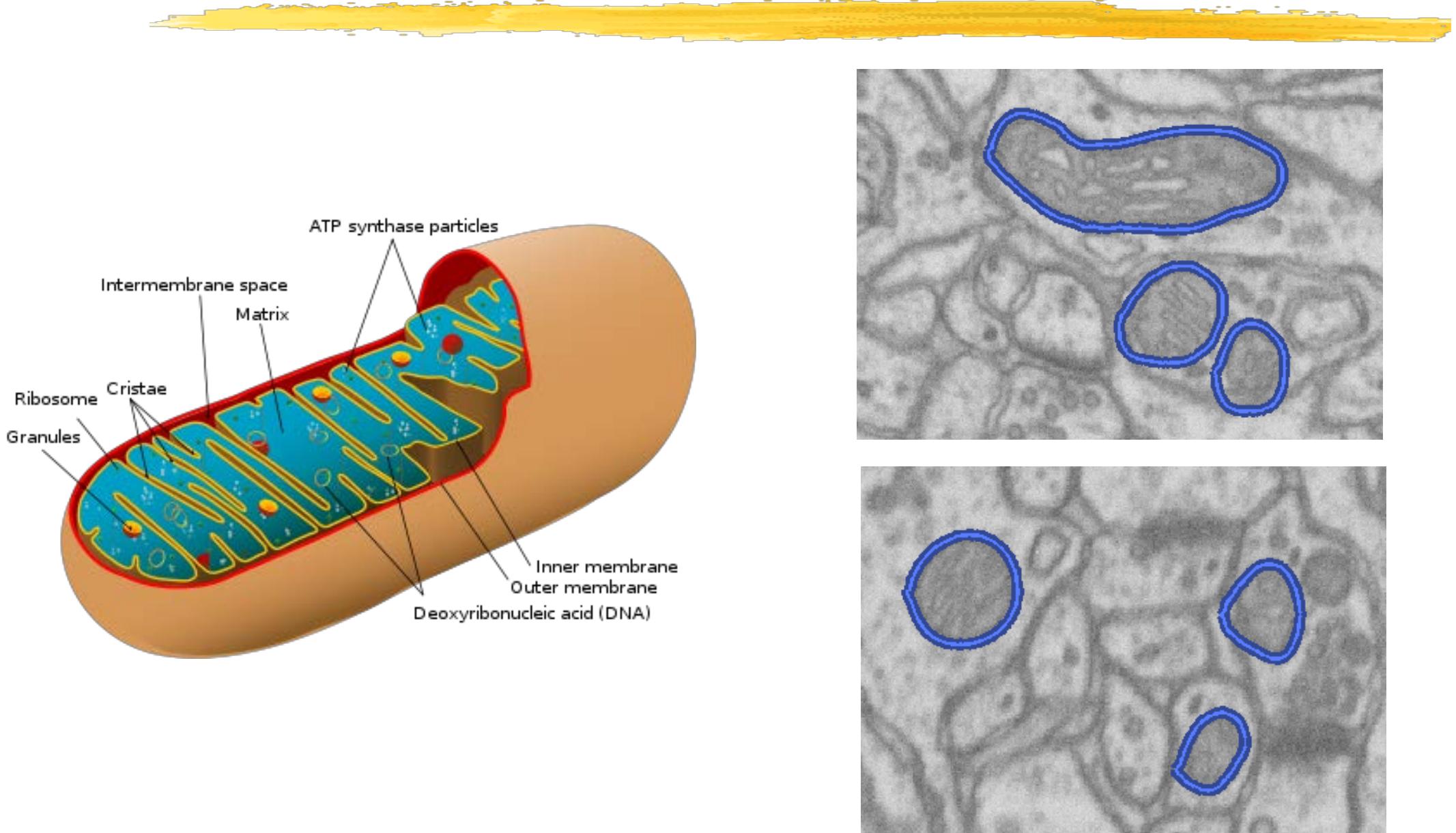


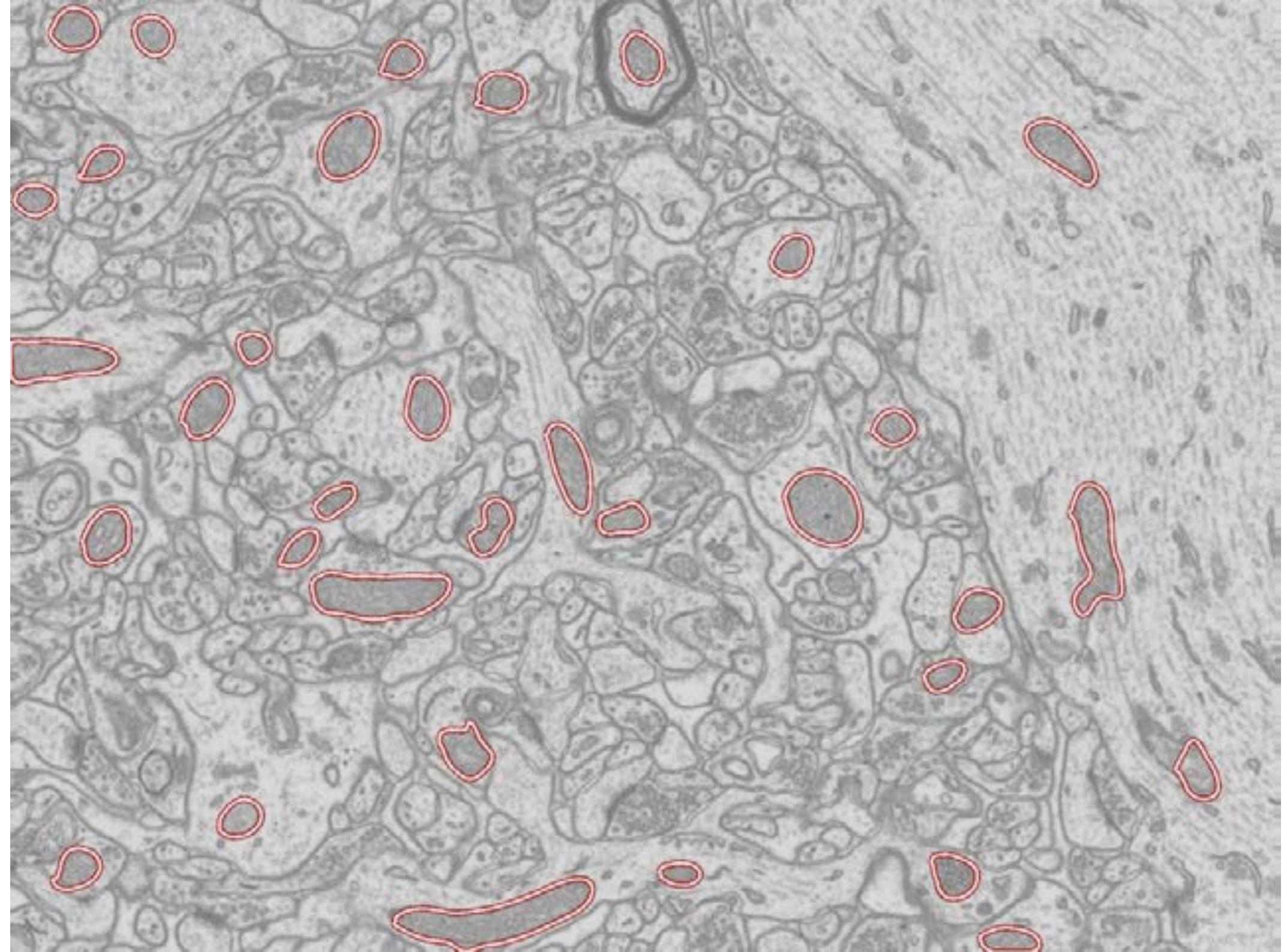
GrabCut
Rother & al. SIGGRAPH 04

ELECTRON MICROSCOPY



MITOCHONDRIA





ALGORITHM



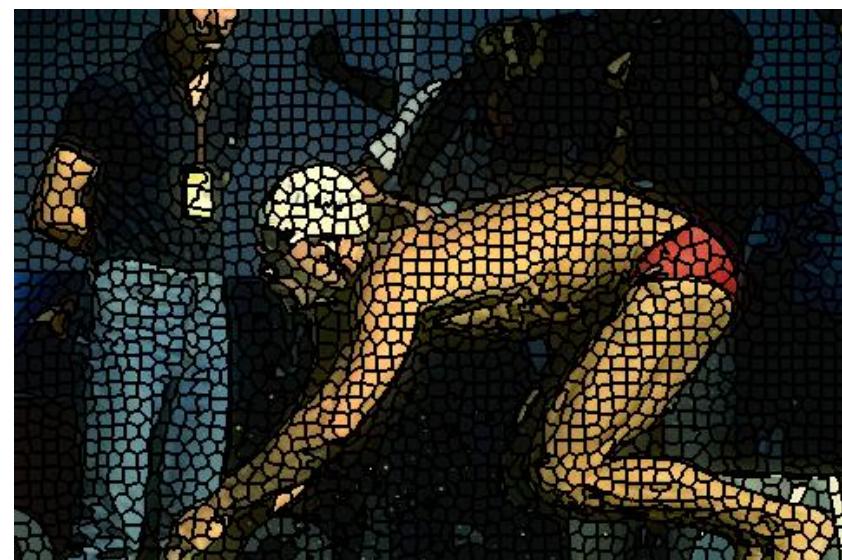
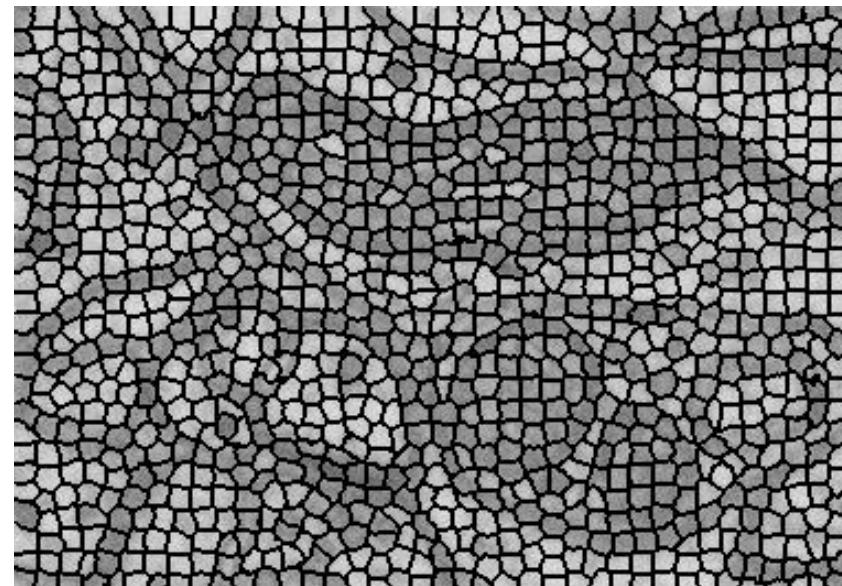
1. Superpixels oversegmentation
2. Feature extraction
 - Ray features
 - Gray level histograms
3. SVM classification
4. Graph cuts segmentation

SUPERPIXELS

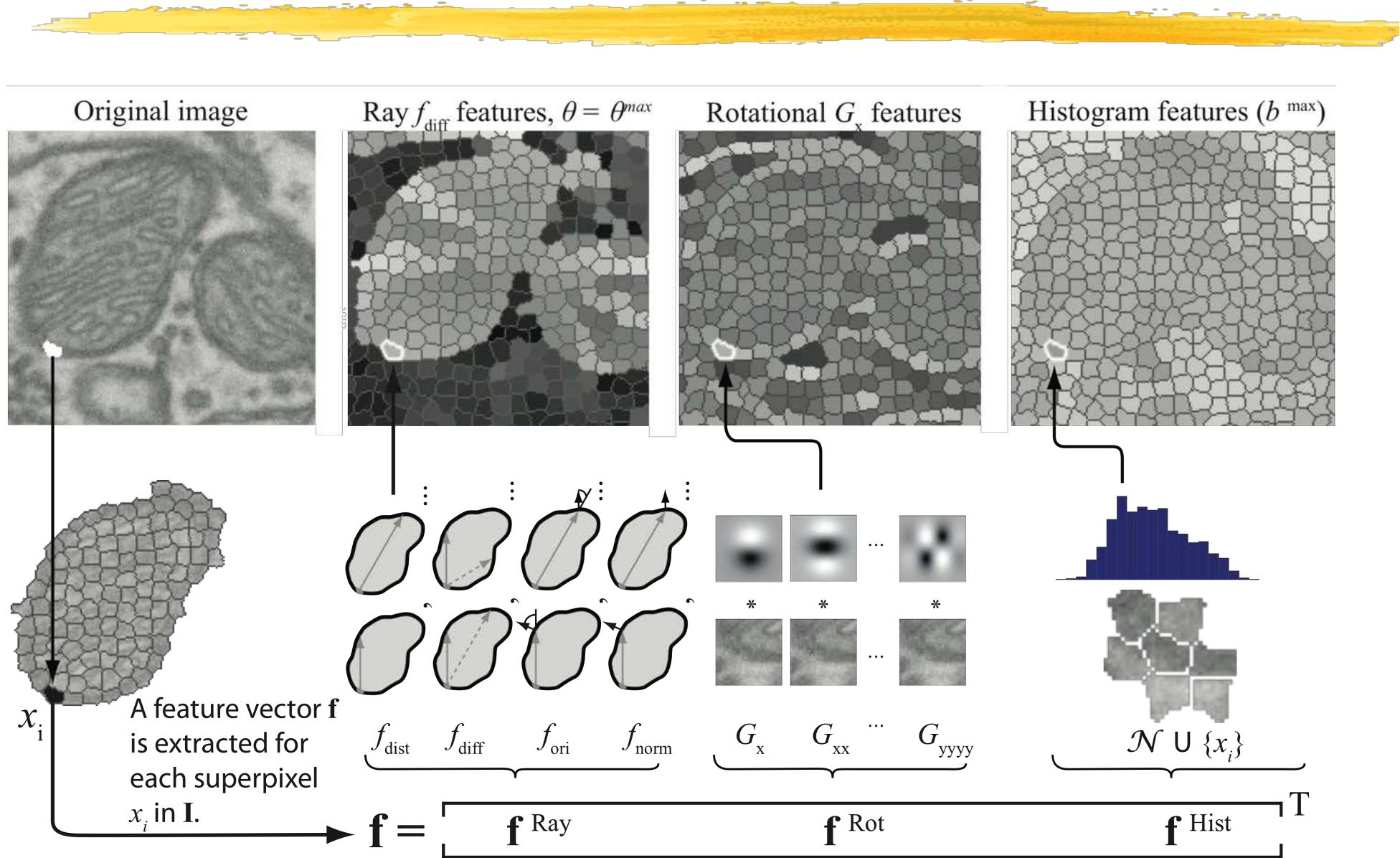


Run K-Means algorithm with regularly spaced seeds on a grid and using a distance that is a weighted sum of distances in image space and in gray level/color space.

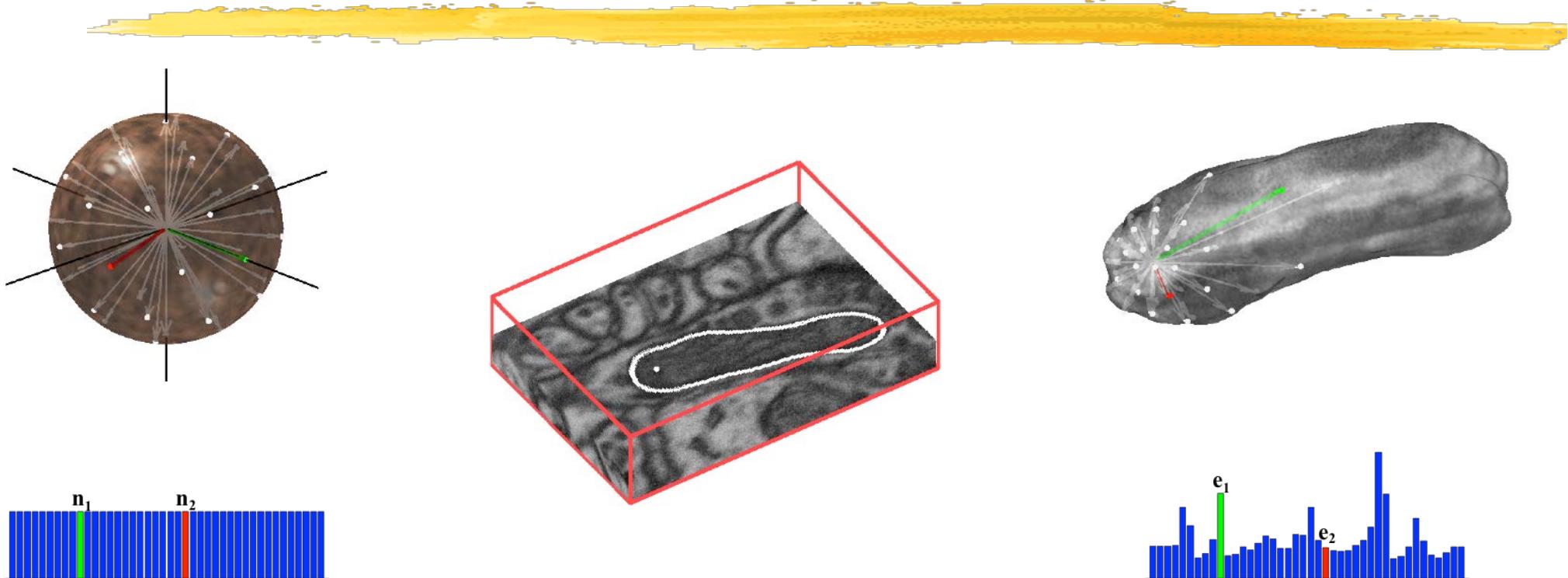
GRAY LEVELS OR COLORS



MITOCHONDRIA

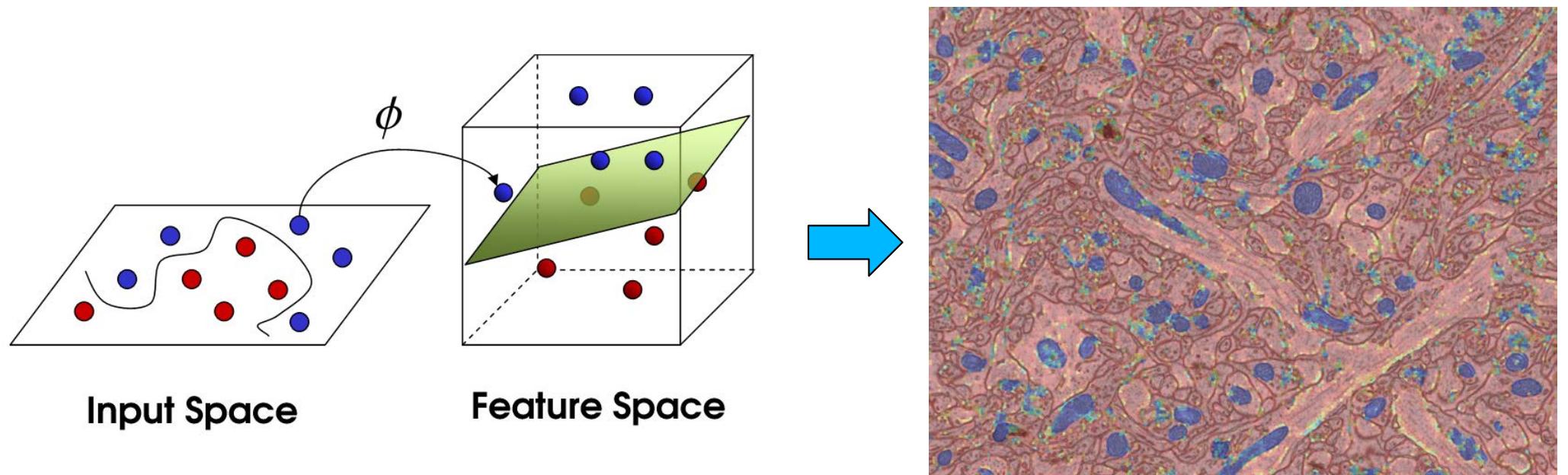


RAY FEATURES



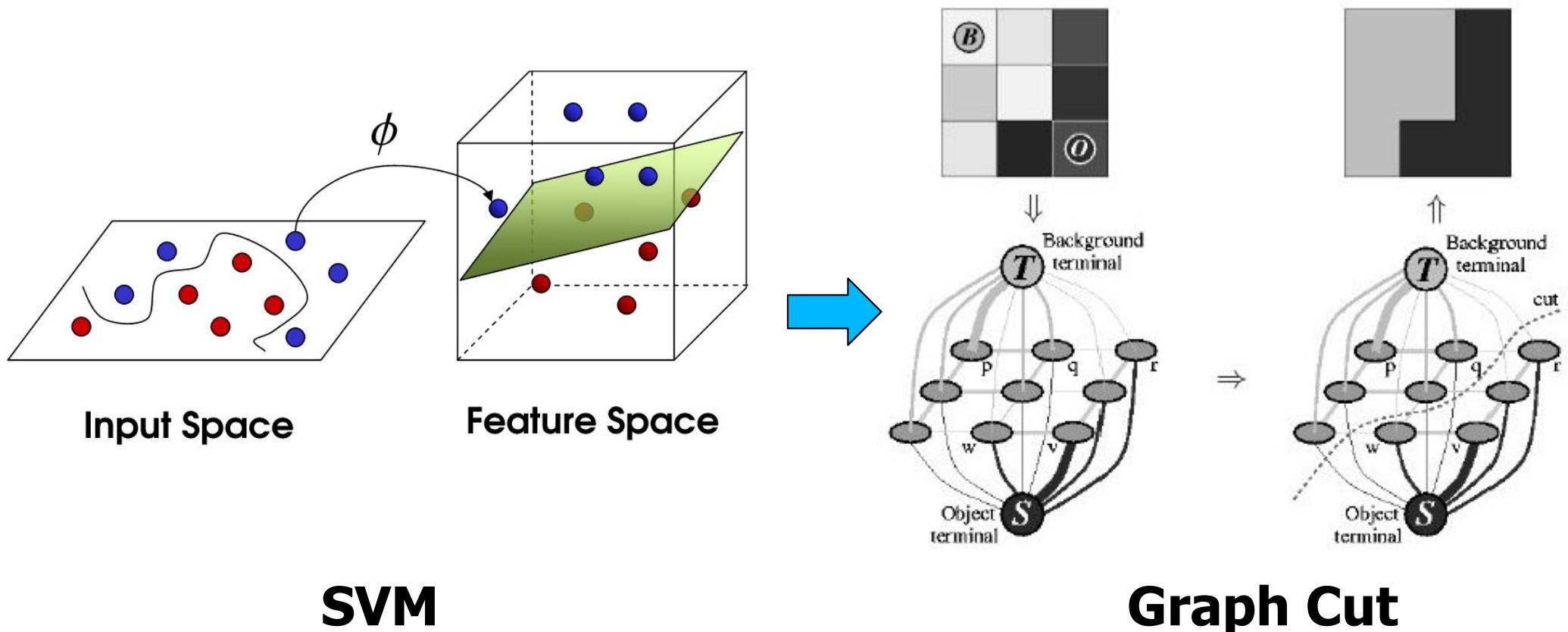
- Compute statistics of distances to nearest boundary.
- Adds global information to a purely local measure.

SVM CLASSIFICATION



- The features incorporate the filter responses among other things.
- The probability of a superpixel belonging to a mitochondria is estimated from the SVM output.

FROM SVM TO GRAPH CUT

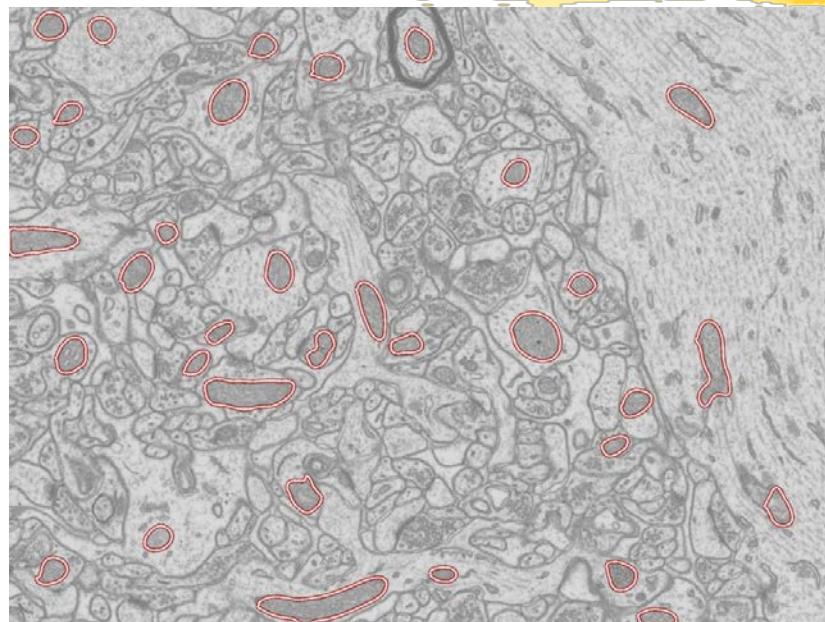


SVM

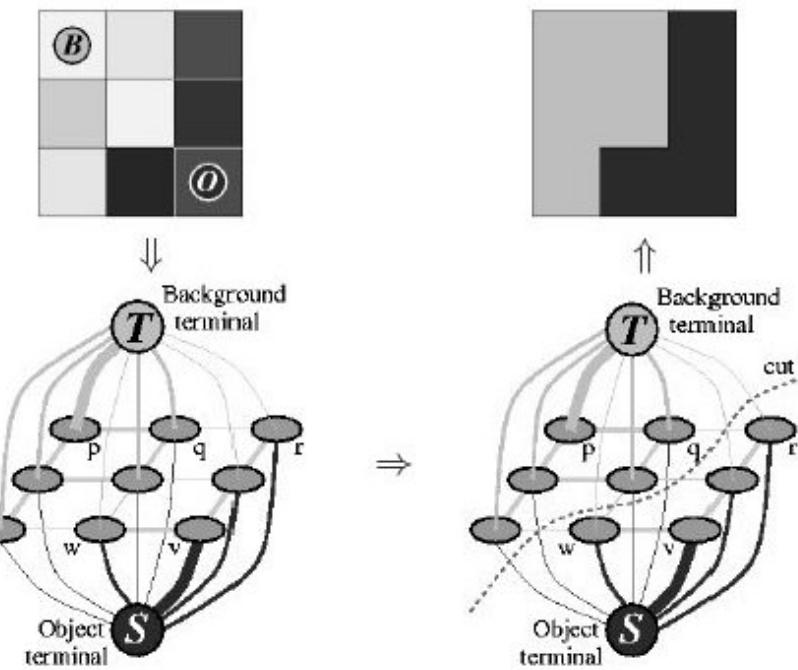
Graph Cut

- The weights of the edges connecting superpixels to the source of the sink depend on the output of the SVM.
- Those of edges connecting superpixels among themselves depend on the output of a different SVM trained to recognize boundaries.

GRAPH CUT



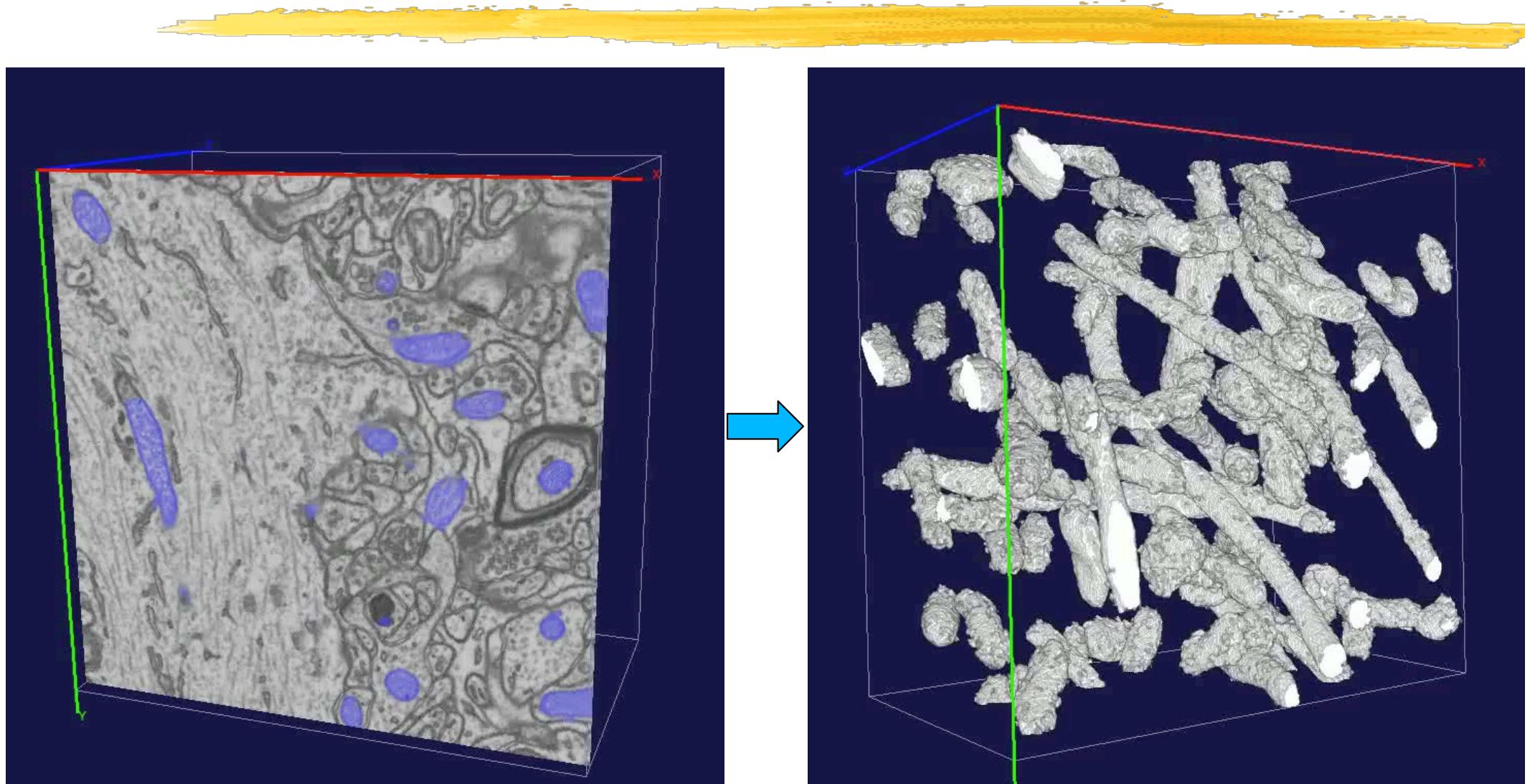
Minimize



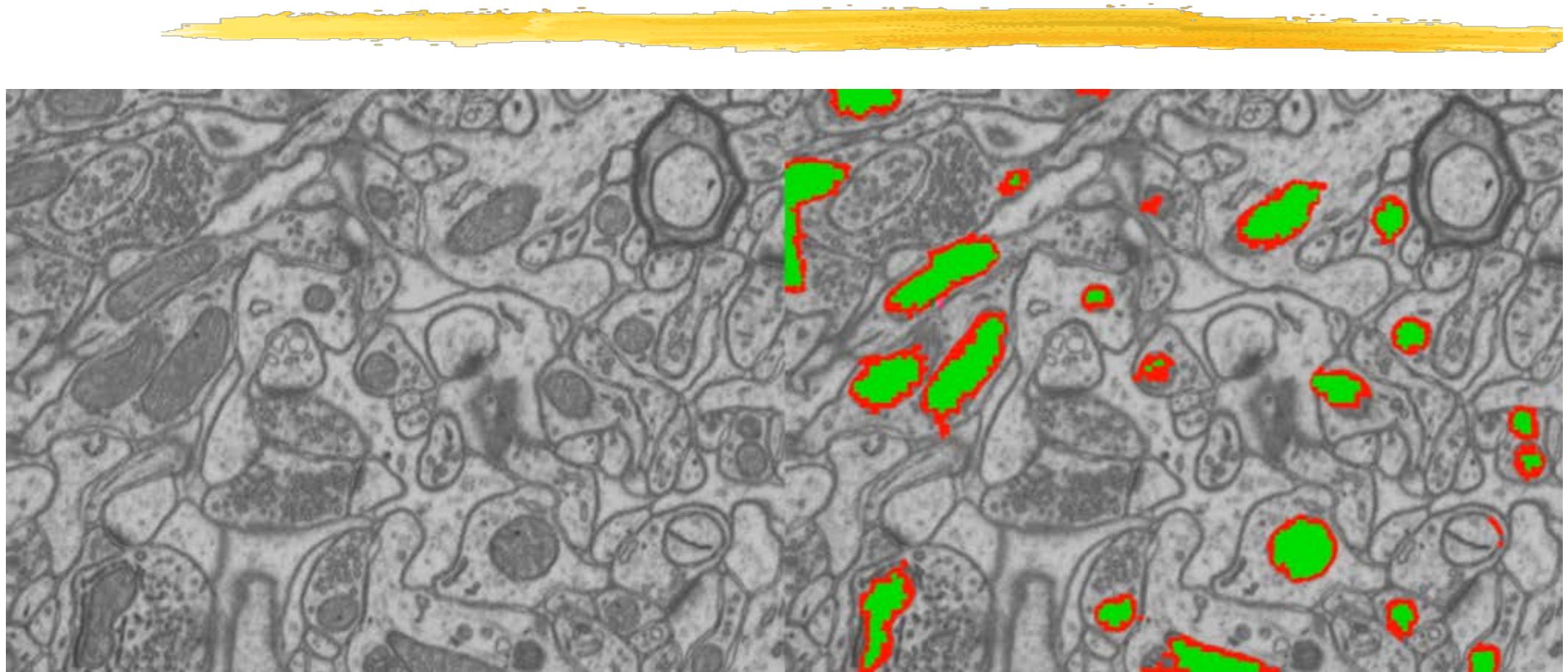
$$E(y|x, \lambda) = \sum_i \underbrace{\psi(y_i|x_i)}_{\text{unary term}} + \lambda \sum_{(i,j) \in \mathcal{E}} \underbrace{\phi(y_i, y_j|x_i, x_j)}_{\text{pairwise term}},$$

with respect to y .

3D MITOCHONDRIA

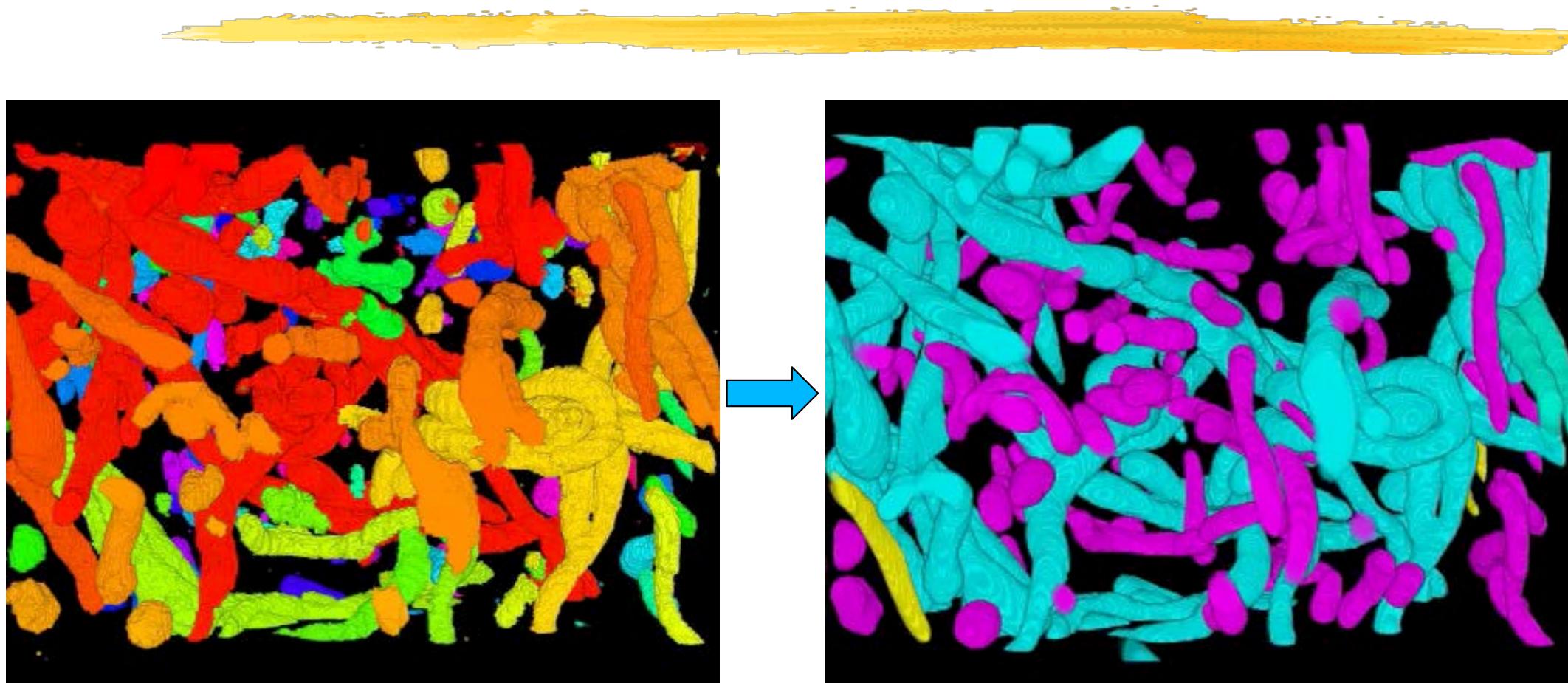


MODELING MEMBRANES



Explicitly model membranes as separate regions and exploit the fact that the inside is enclosed within them to retain the graph cut formulation.

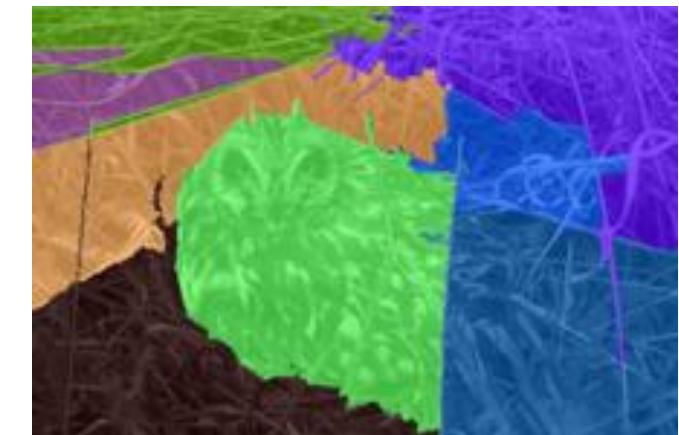
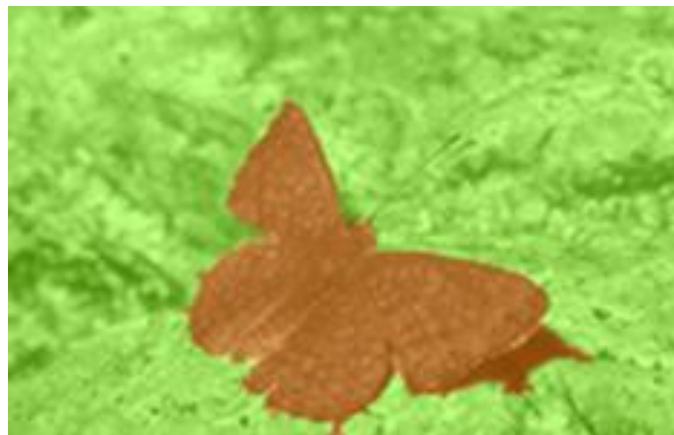
SAVING TIME



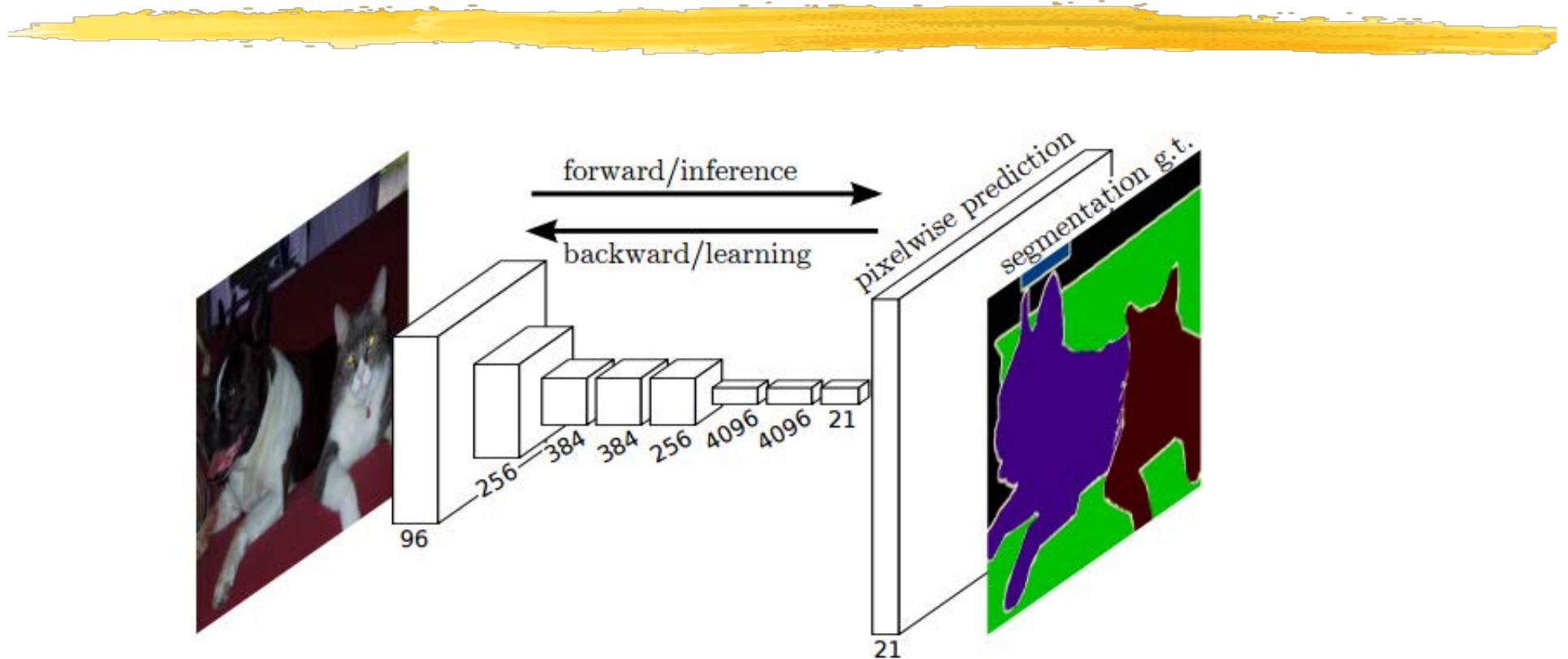
$3.21 \mu\text{m} \times 3.21 \mu\text{m} \times 1.08 \mu\text{m}$: 53 mitochondria

By hand: 6 hours. Semi-automatically: 1.5 hours

MORE GRAPH CUT RESULTS

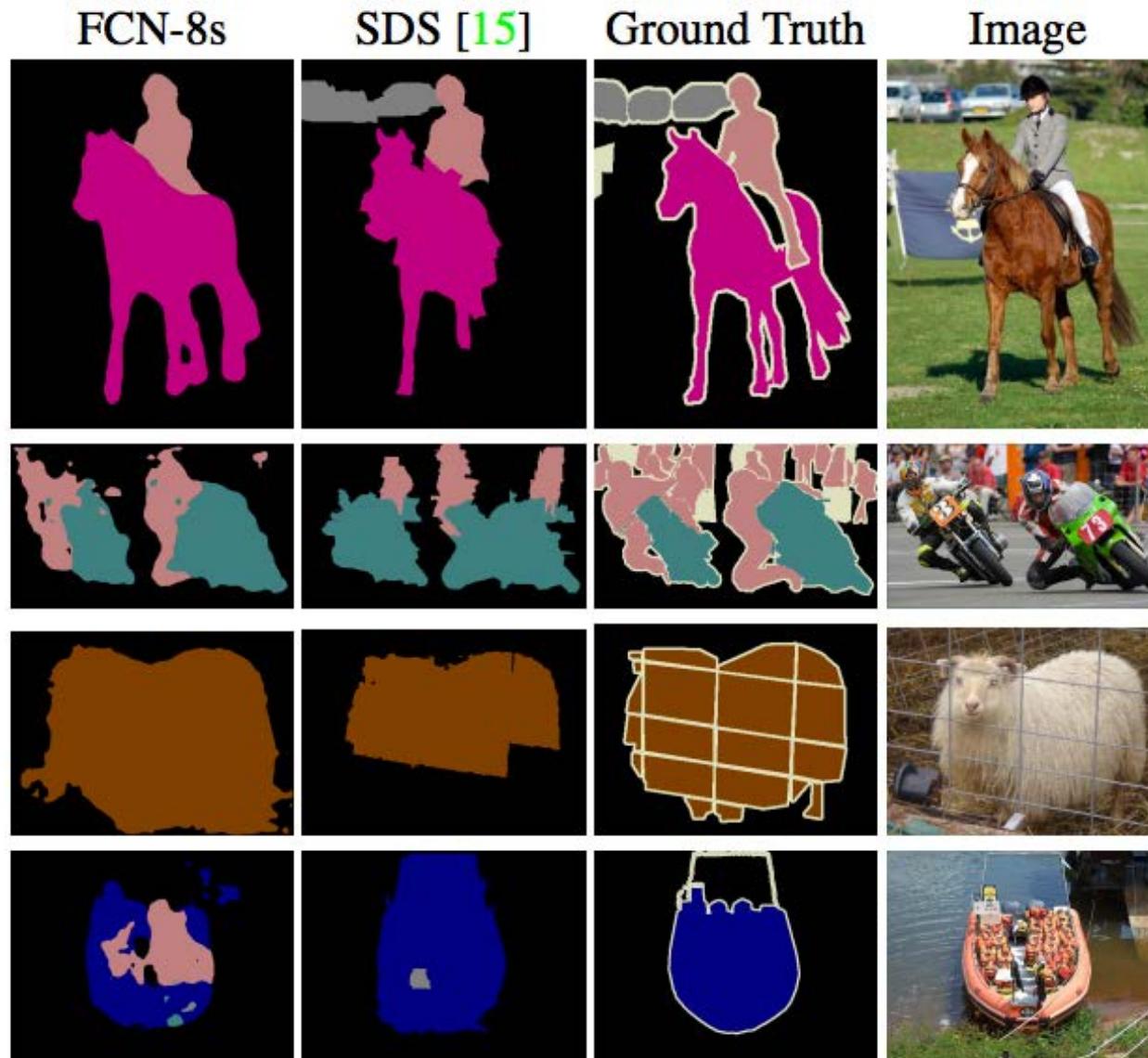


CONVOLUTIONAL NEURAL NETS



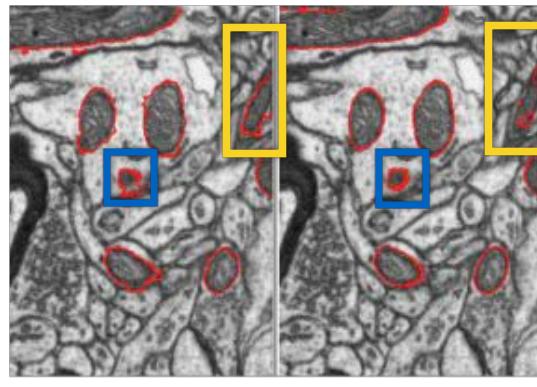
- Connect input layer to output one made of segmentation labels.
- Need layers that both downscale and upscale.
- Connect the lower layers directly to the upper ones.

IMPROVED SEGMENTATIONS



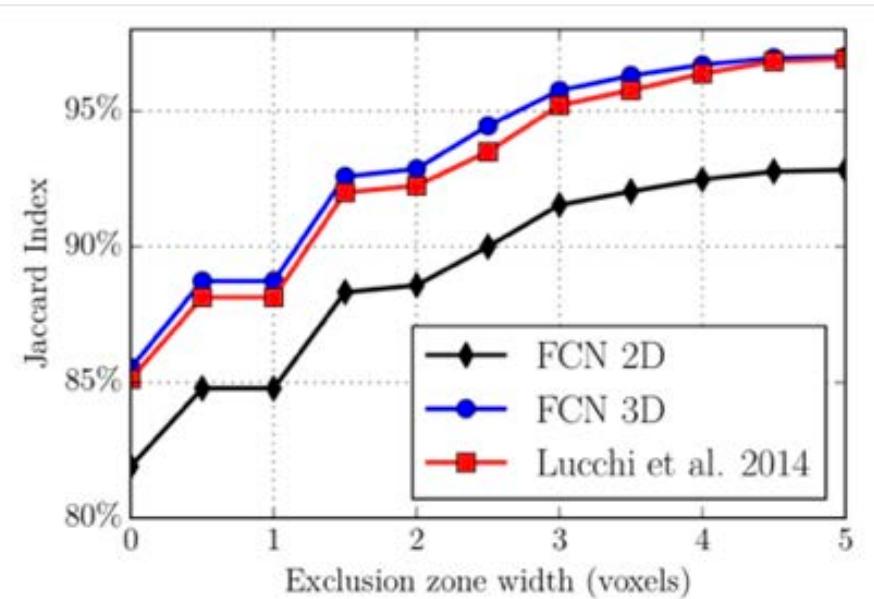
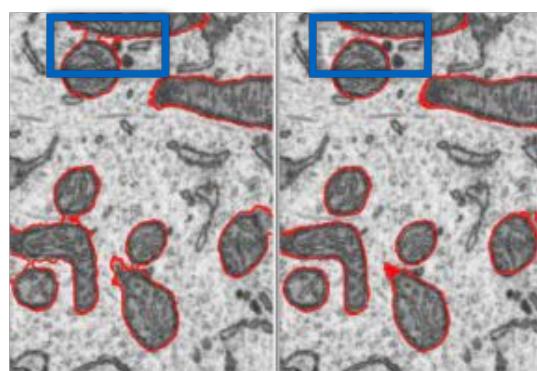
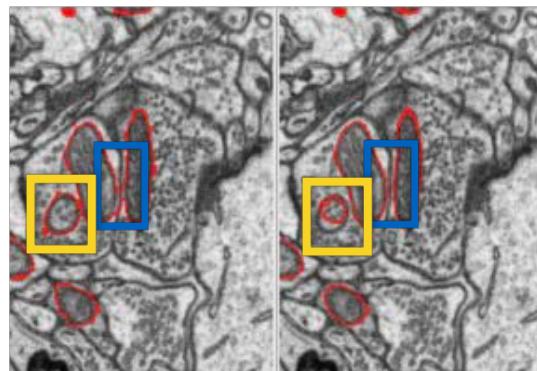
DEEP NETS vs GRAPH CUT

Context Features + CRF U-Net 3D

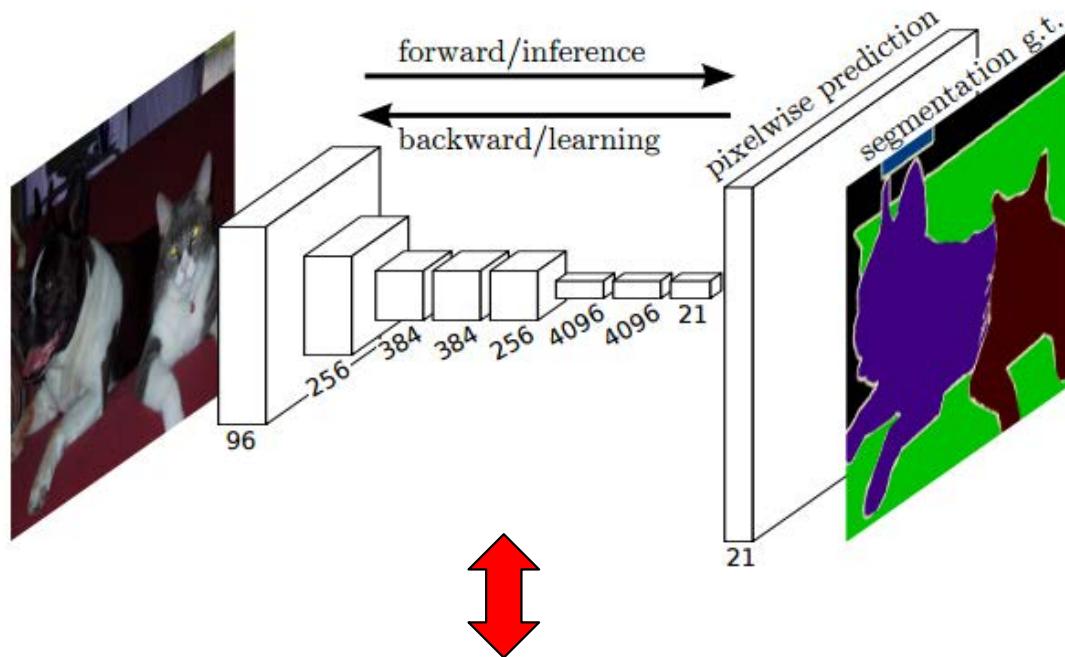


Striatum Mitochondria

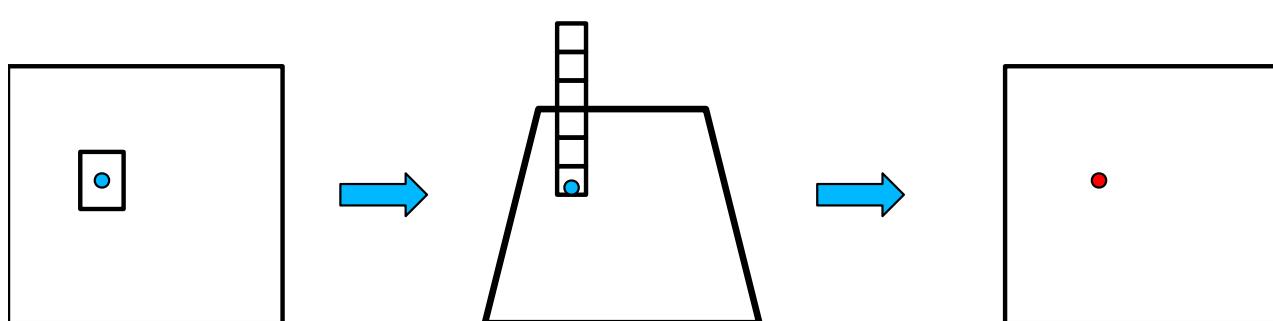
Method	Jaccard Index
Context F. + CRF	84.6%
U-Net 2D	82.4%
U-Net 3D	86.1%



A PARTIAL EXPLANATION?

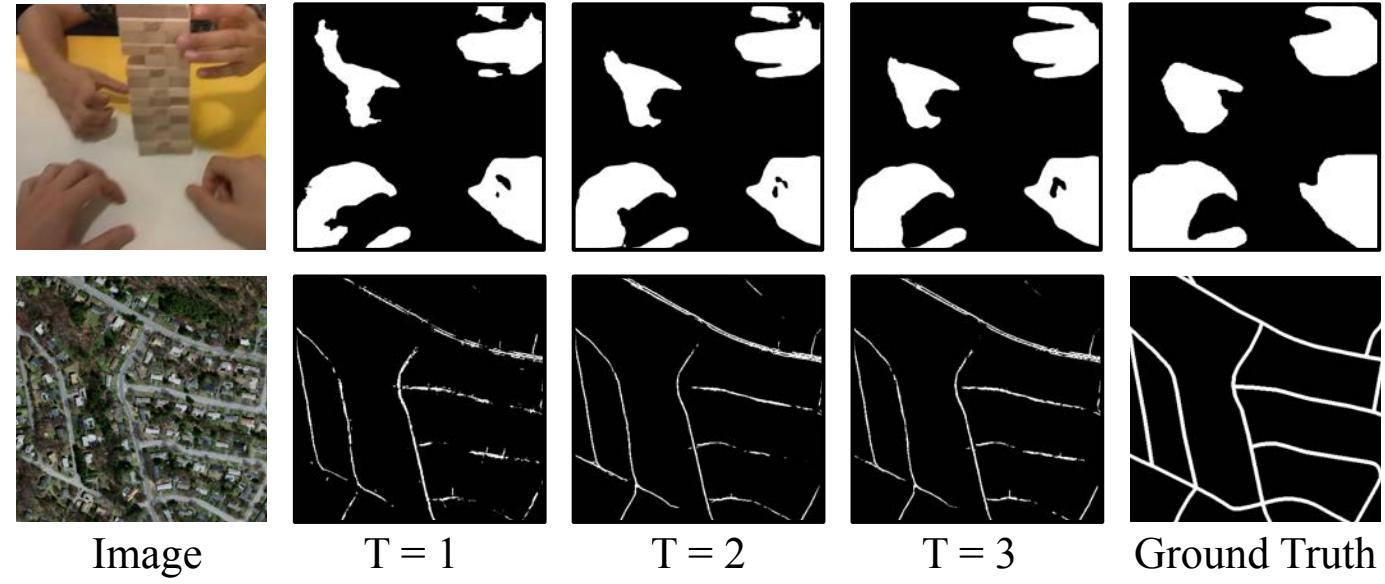
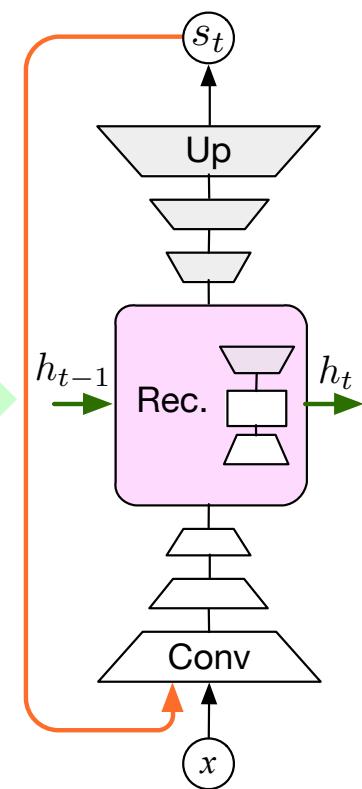
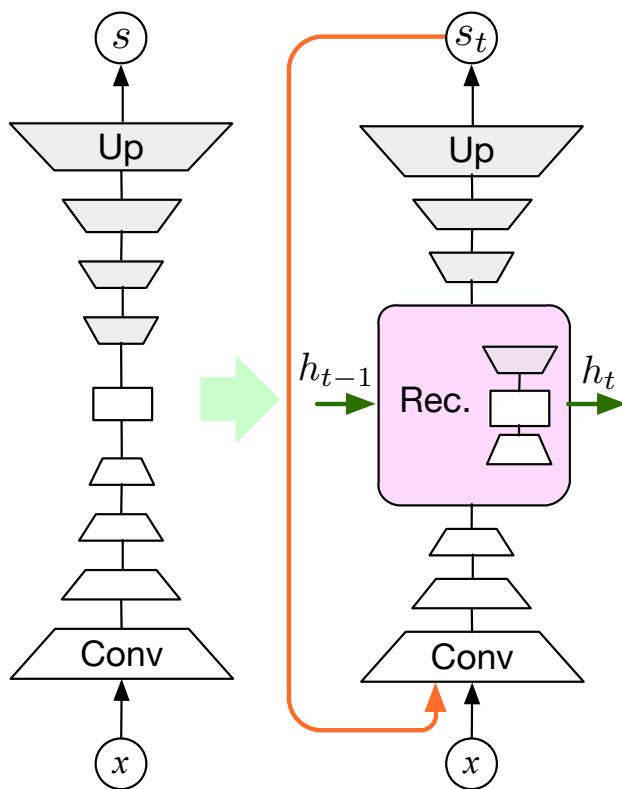


- Can be understood as generating for every output pixel a feature vector containing the output of all the intermediate layers.



Will come back to that when we talk about texture.

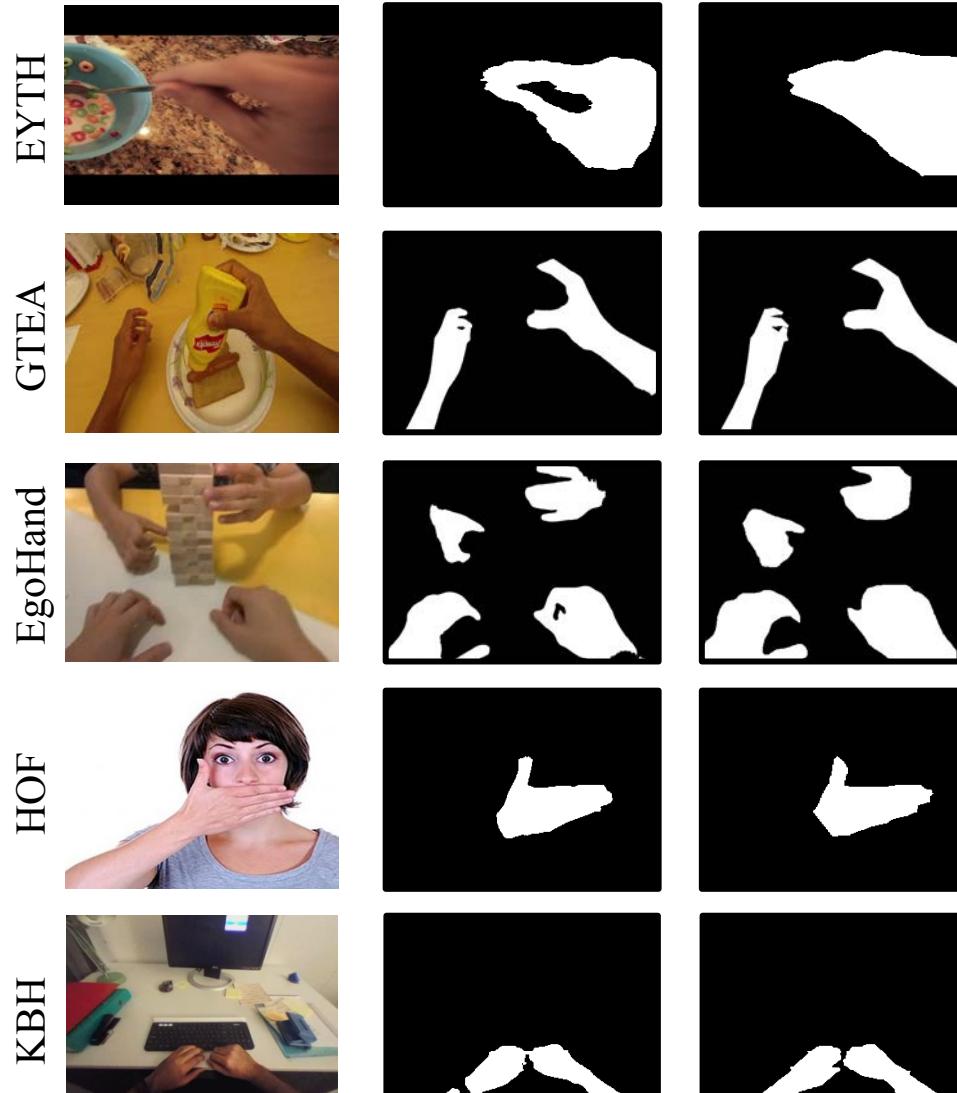
RECURSIVE HAND SEGMENTATION



U-Net

Rec. U-Net

RECURSIVE HAND SEGMENTATION



Image

Ours

Ground Truth

IN SHORT



- Low-level methods can provide valuable data but are inherently limited.
- Domain knowledge, user interaction, and training data can be used to turn this data into usable results.
- Same philosophy as for delineation.

WHAT ABOUT THE DOG?



SHAPE FROM X

One image:

- **Shading**
- Texture

Two images or more:

- Stereo
- Contours
- Motion



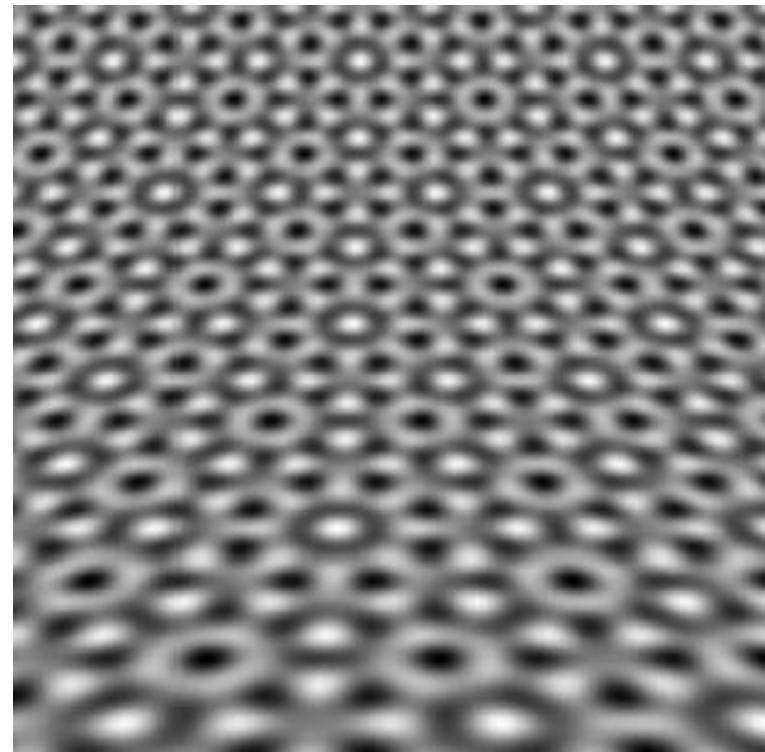
SHAPE FROM X

One image:

- Shading
- **Texture**

Two images or more:

- Stereo
- Contours
- Motion



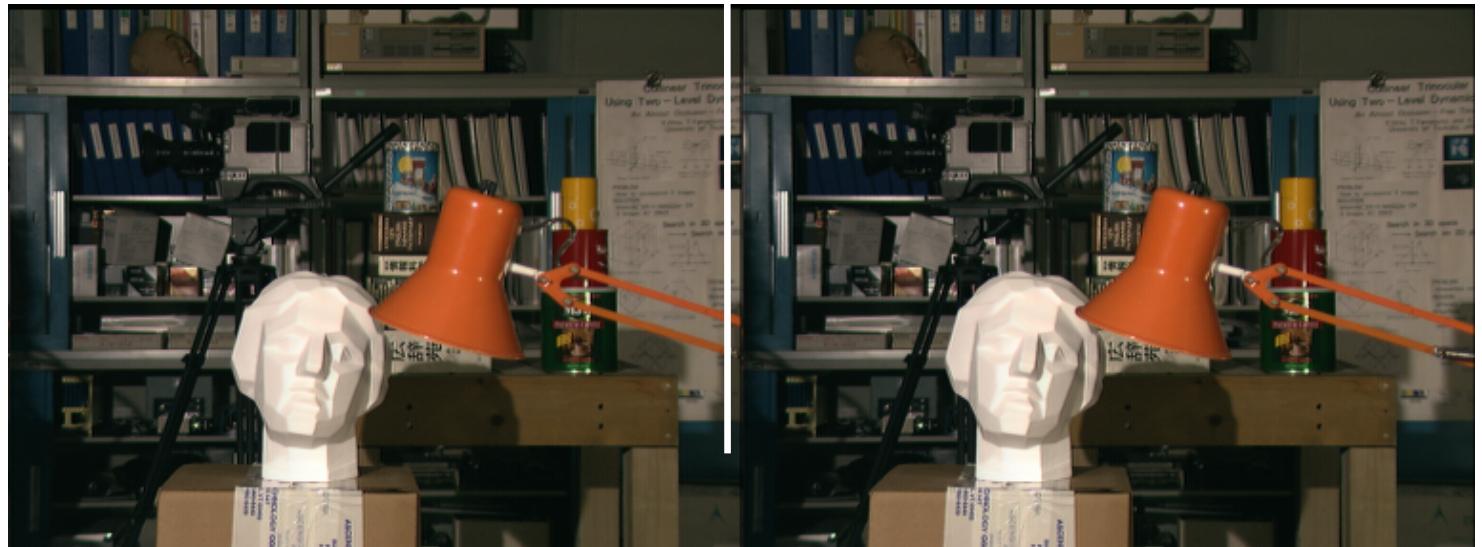
SHAPE FROM X

One image:

- Shading
- Texture

Two images or more:

- **Stereo**
- Contours
- Motion



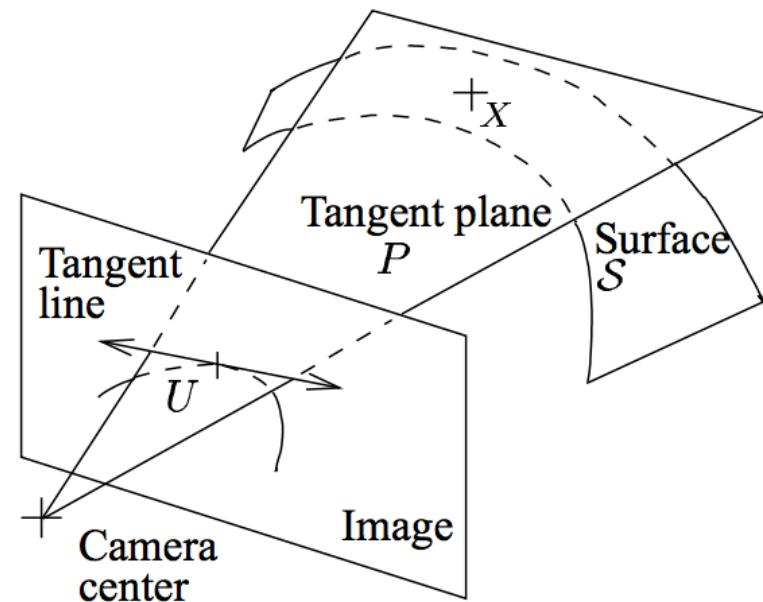
SHAPE FROM X

One image:

- Shading
- Texture

Two images or more:

- Stereo
- **Contours**
- Motion



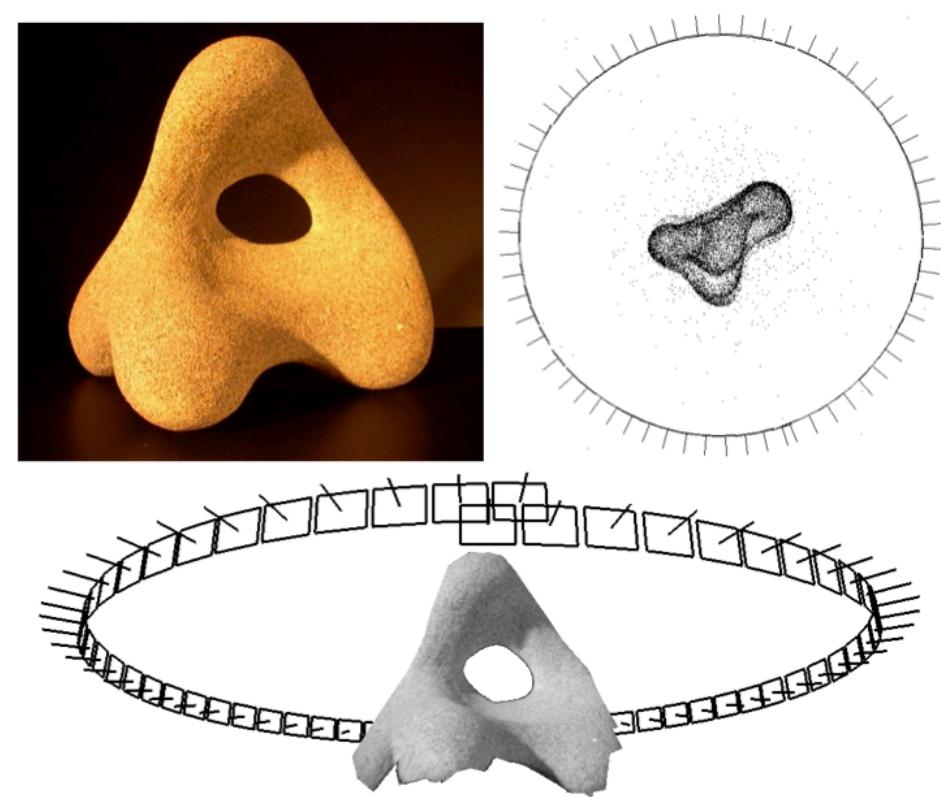
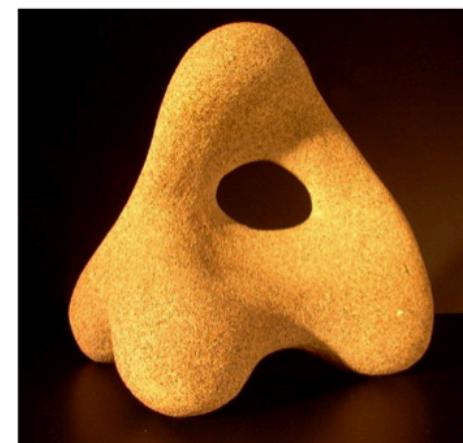
SHAPE FROM X

One image:

- Shading
- Texture

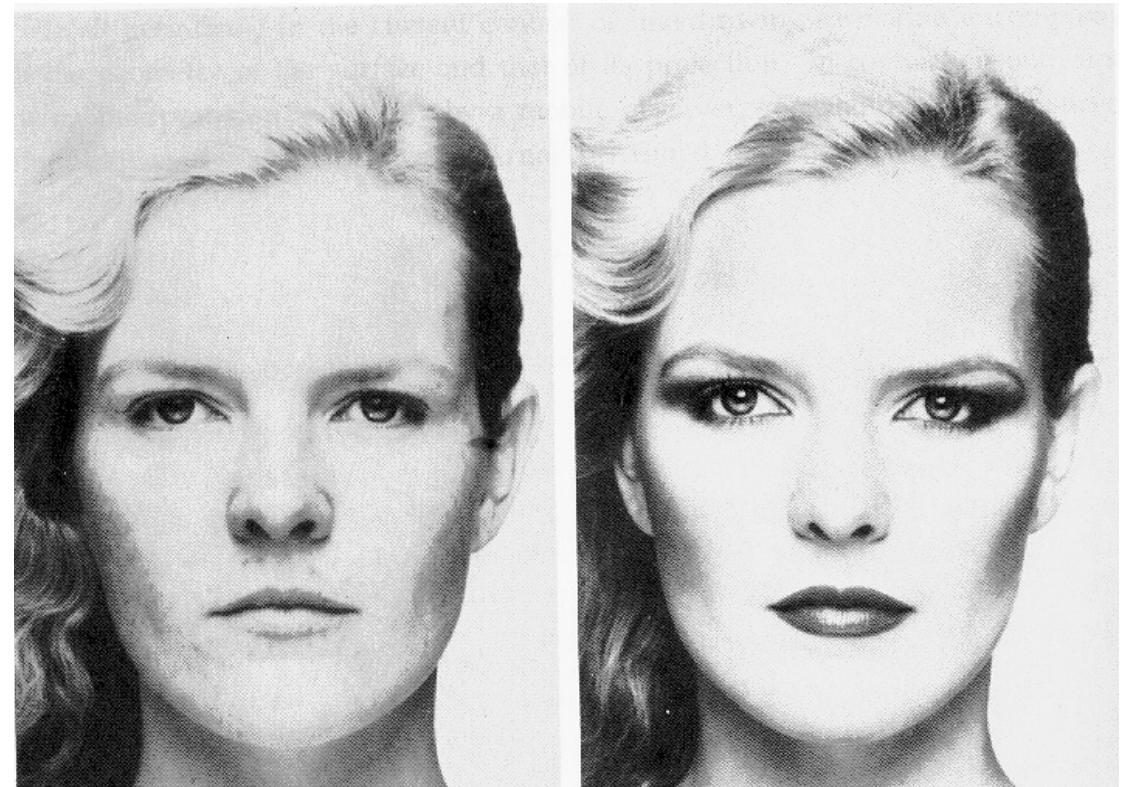
Two images or more:

- Stereo
- Contours
- **Motion**

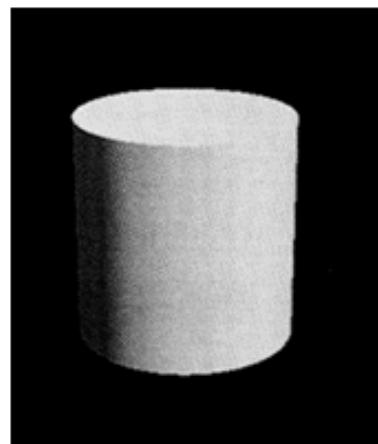
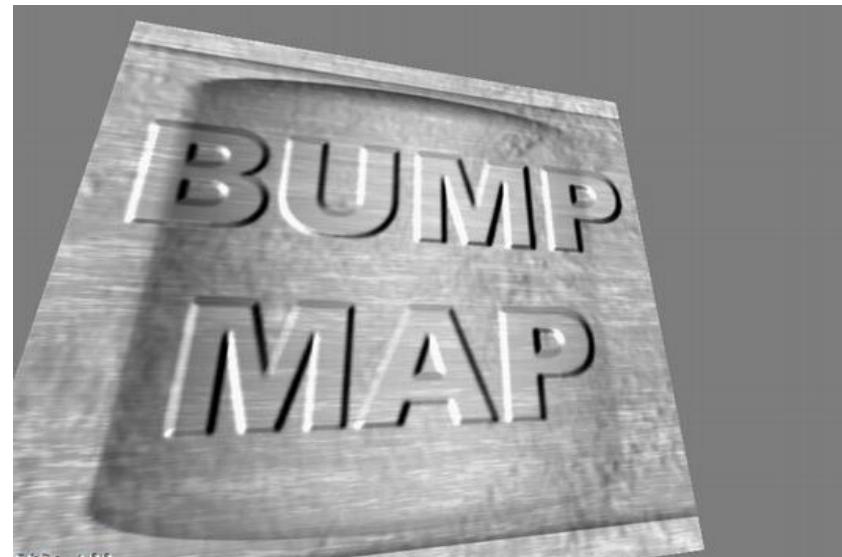
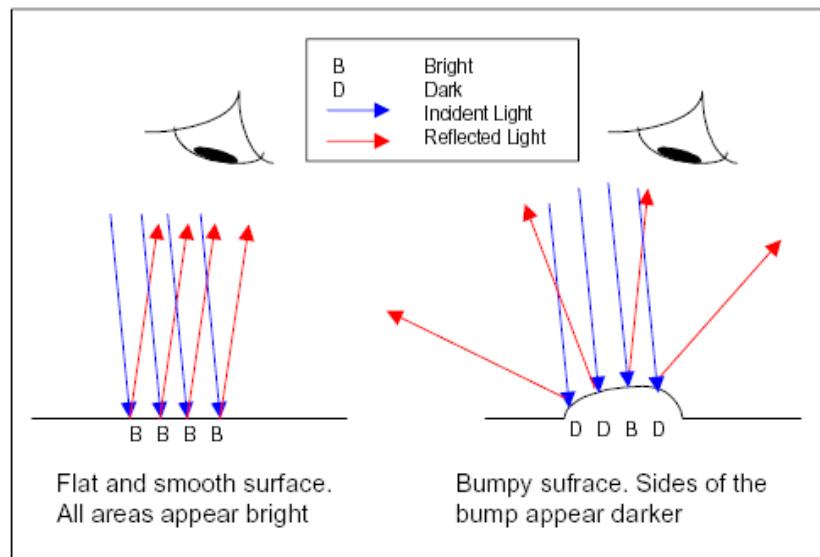


SHADING

- Shading models
- Shape from shading
 - Variational Methods
 - Photometric Stereo

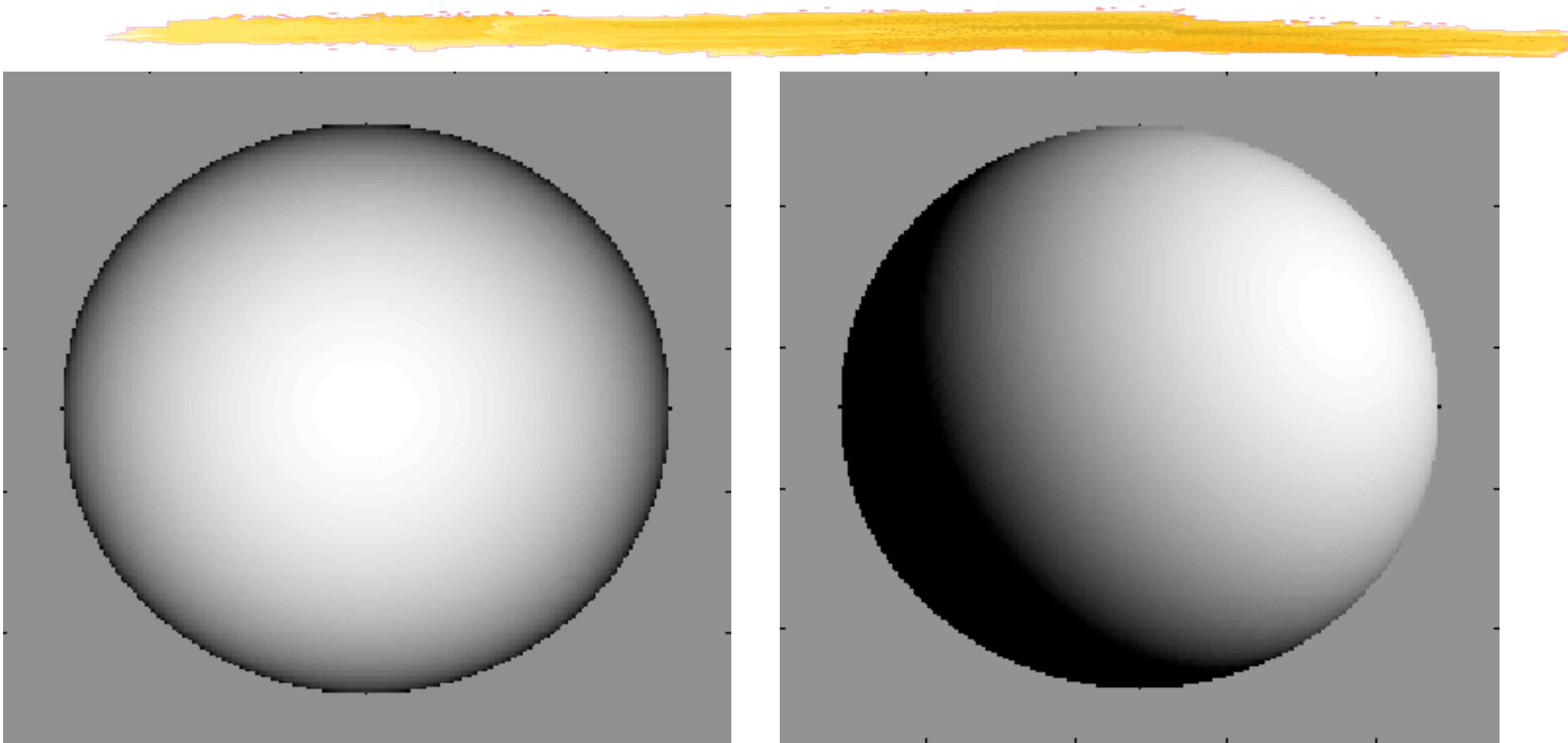


BUMP MAPPING



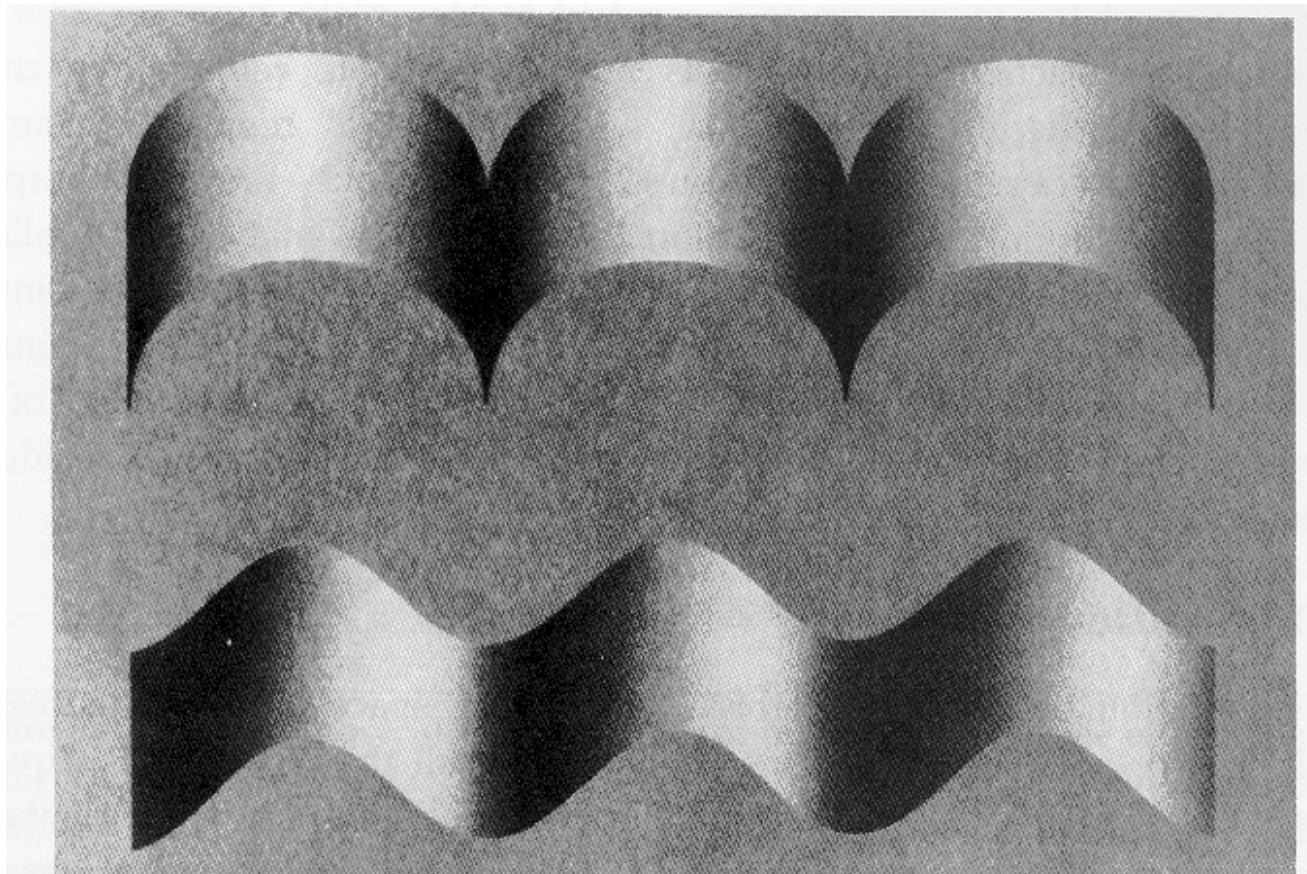
Simple mesh + 2D bump map = Complex looking object

LAMBERTIAN HALF-SPHERE



Gray level changes are interpreted as changes in the direction of the surface normal

BOUNDARY CONDITIONS



Shading gives information about surface normals
→ Boundary conditions required for a complete interpretation.

BEETHOVEN

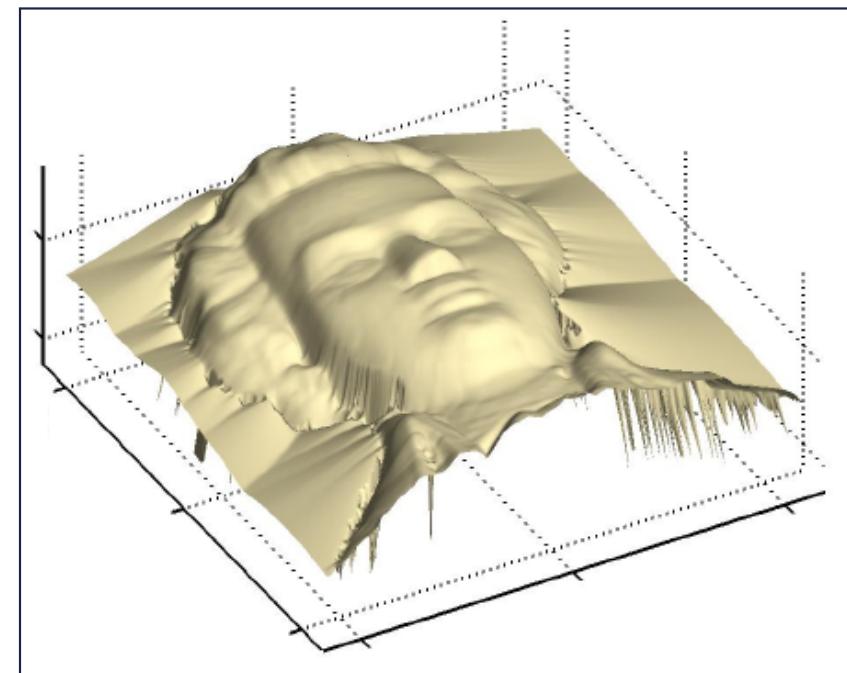
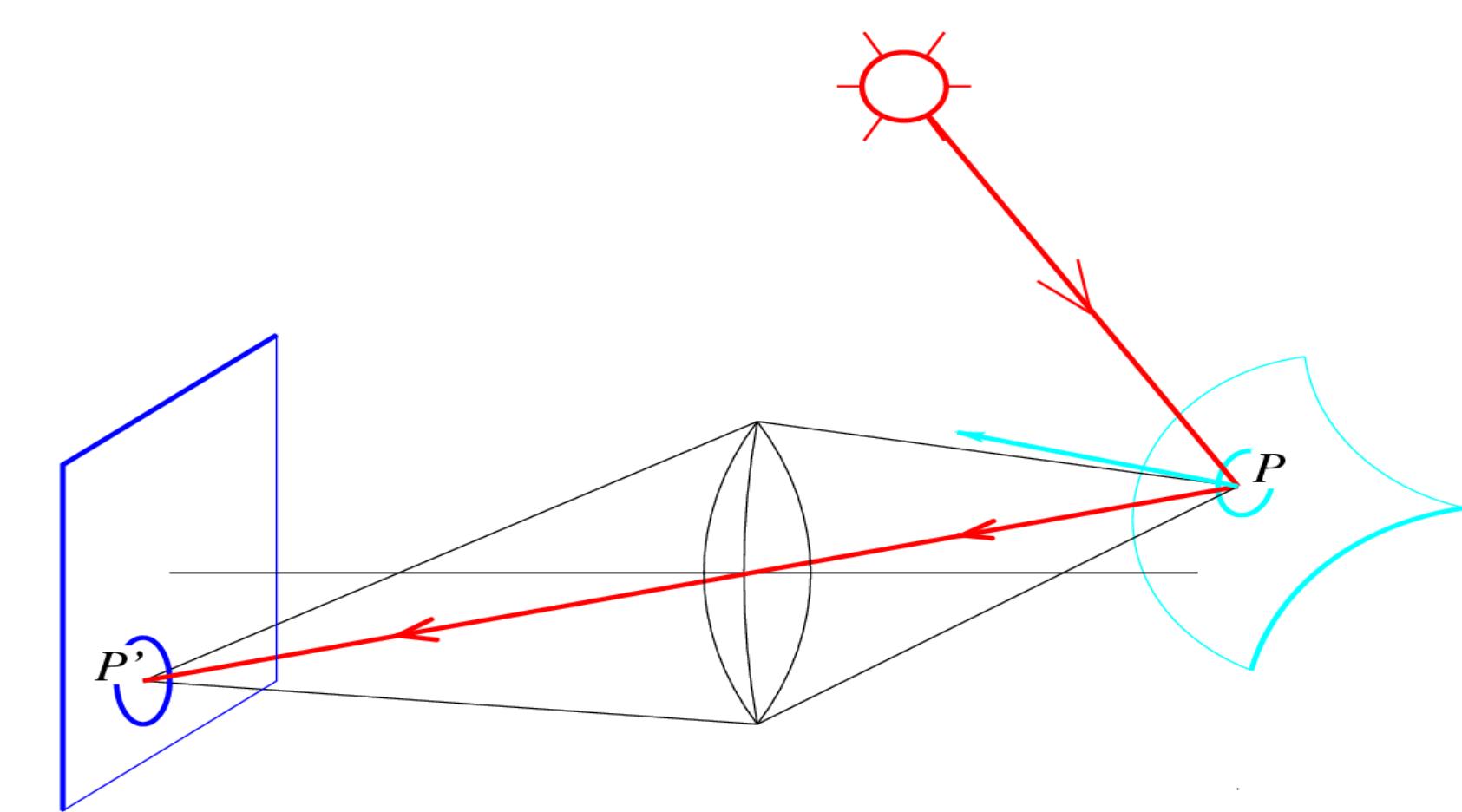
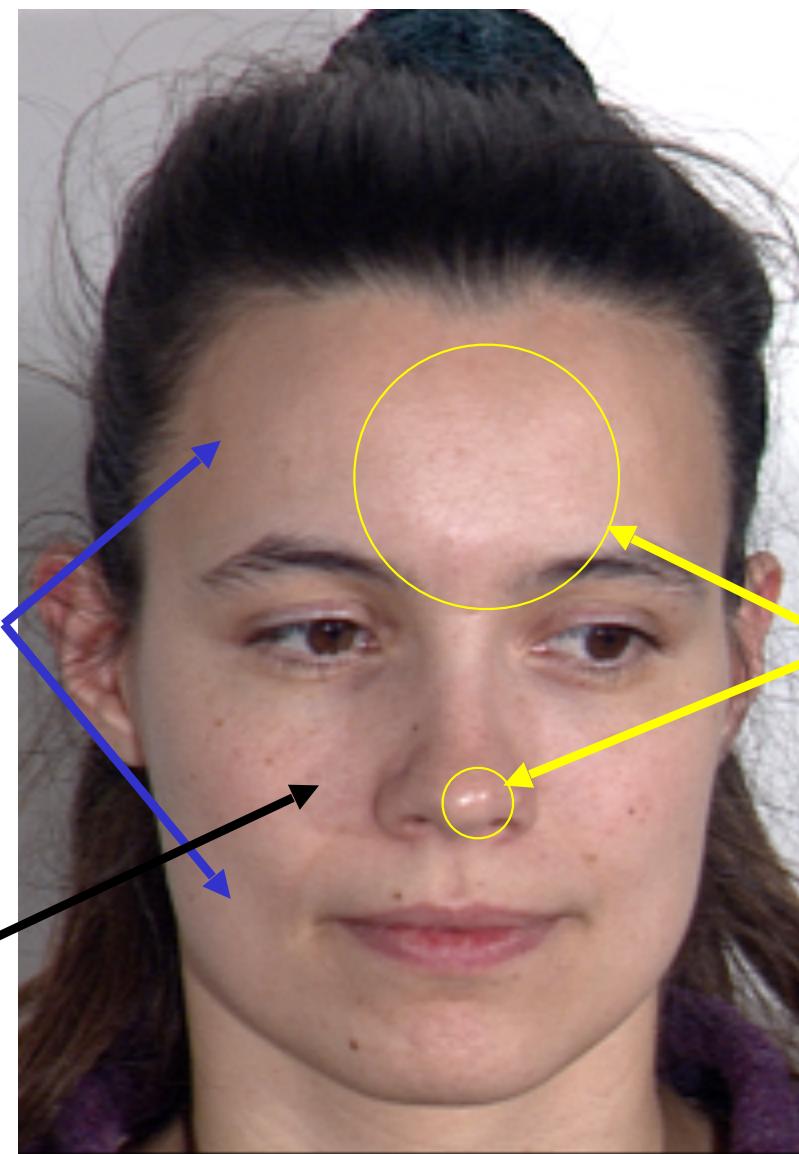


IMAGE FORMATION



DIRECT LIGHTING



Diffuse reflection

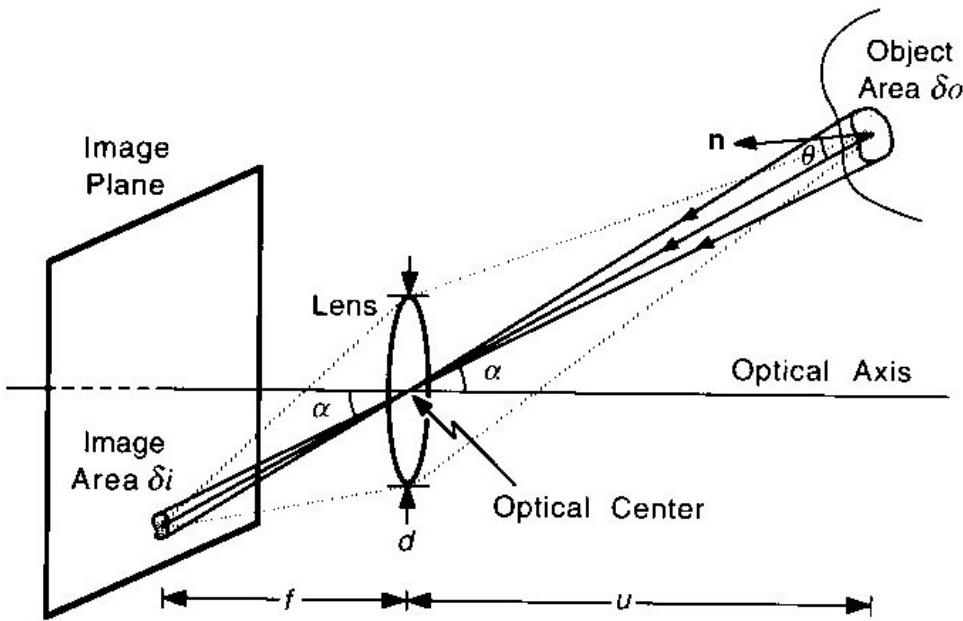
Shadow

Specularities

INDIRECT LIGHTING



FUNDAMENTAL RADIOMETRIC EQUATION



Radiance: amount of light emitted in a given direction

Irradiance: amount of light received on a surface

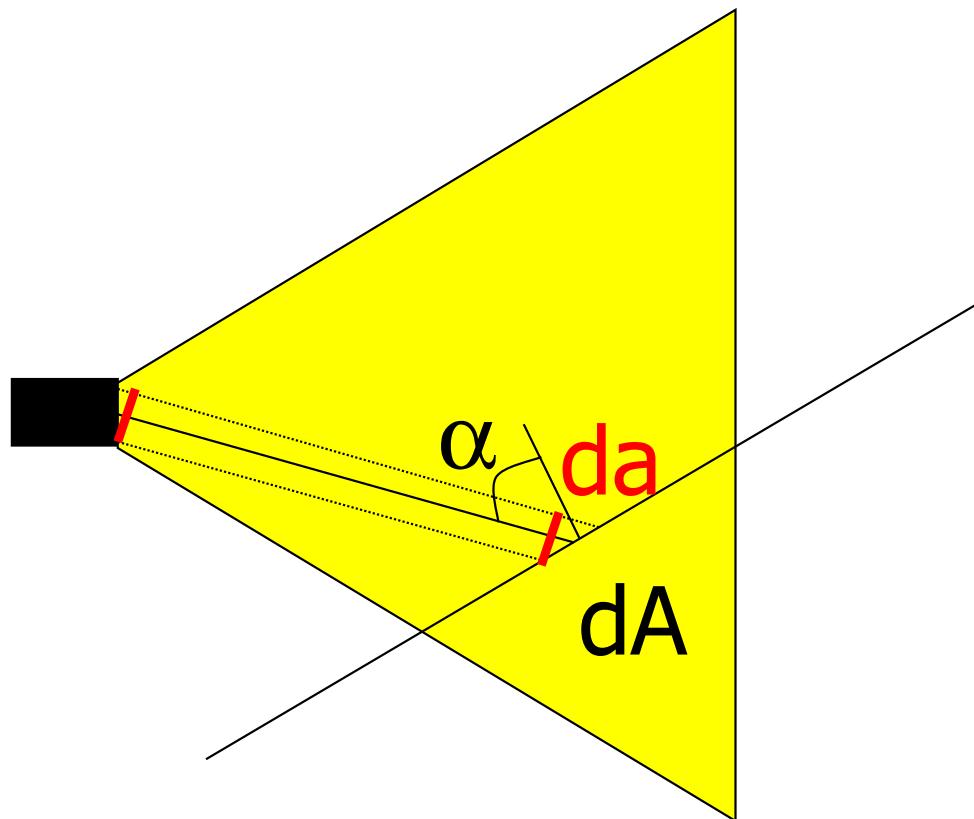
$$Irr = \frac{p}{4} \left(\frac{d}{f} \right)^2 \cos^4(a) Rad$$

$$\Rightarrow I \propto Irr \propto Rad$$

Image intensity

when the camera is photometrically calibrated.

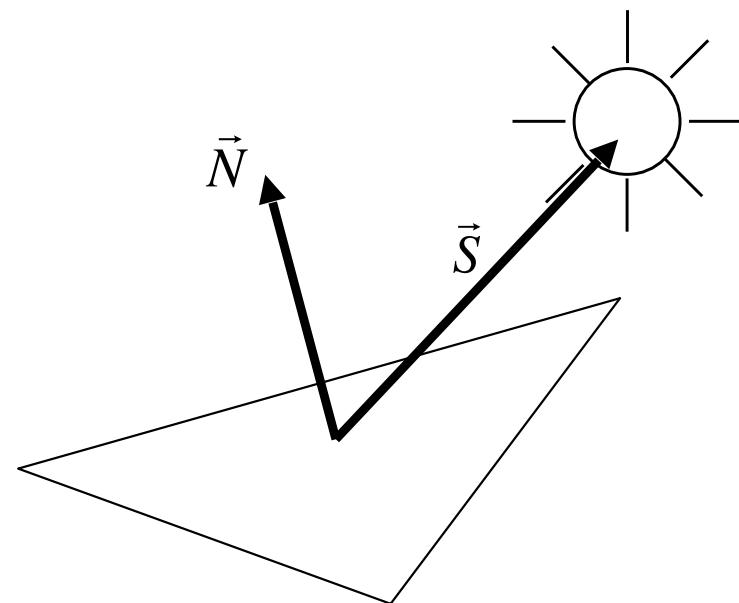
LOCAL SHADING MODEL



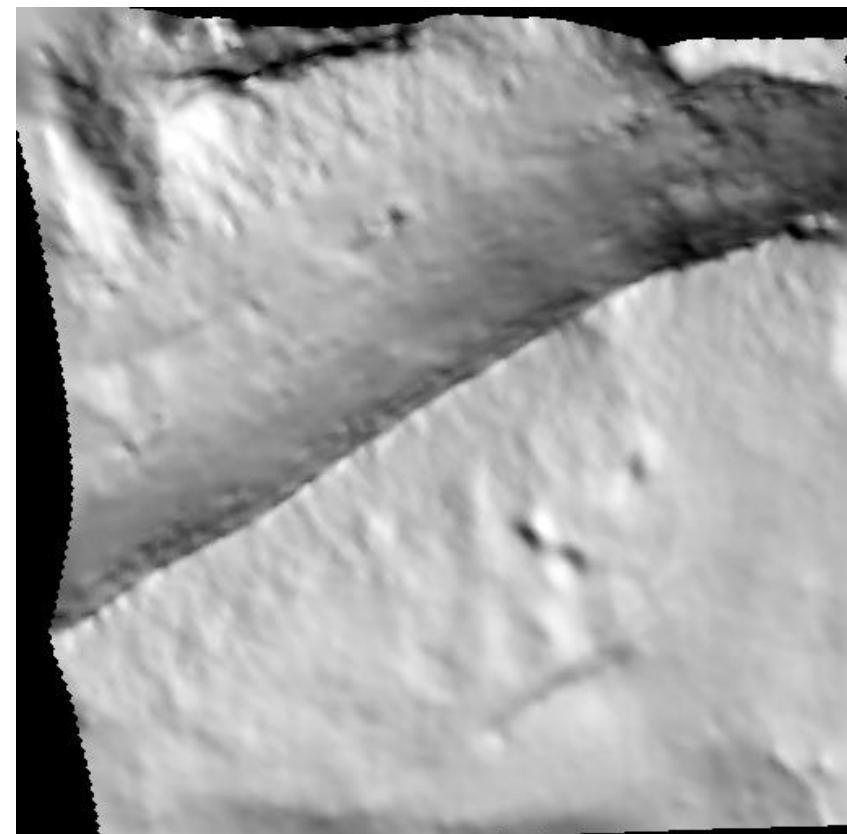
Foreshortening:
 $da = \cos(\alpha)dA$

The effect of a distant source on a surface patch depends on its **apparent** surface.

LAMBERTIAN SURFACE



$$I = \max(Albedo \cdot (\mathbf{N} \cdot \mathbf{S}), 0)$$

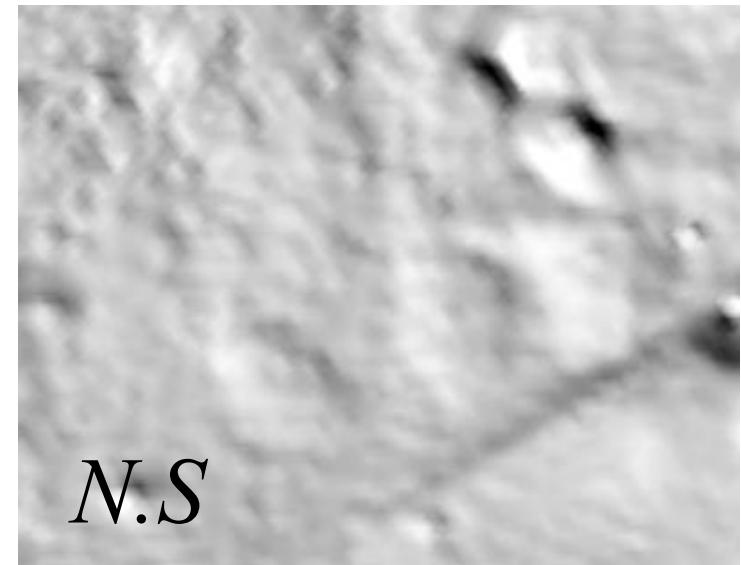


Perfectly matte surface: The radiance depends only on angle of incidence and not on viewing direction.

ESTIMATED ALBEDO



=



*

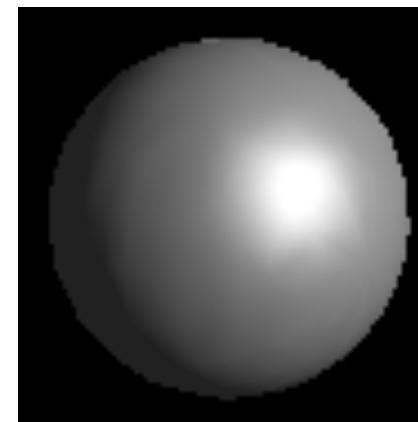


SECONDARY ILLUMINATION

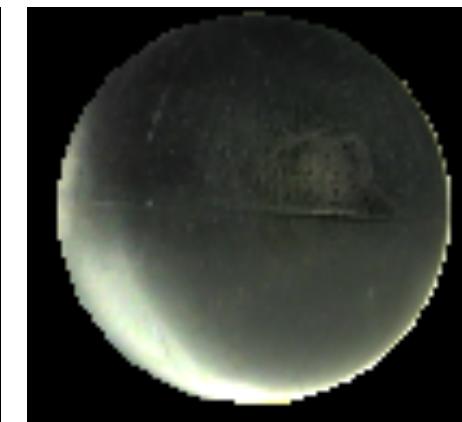
Reflections produce indirect lighting.



Unique light source assumption does not allow correct albedo recovery.

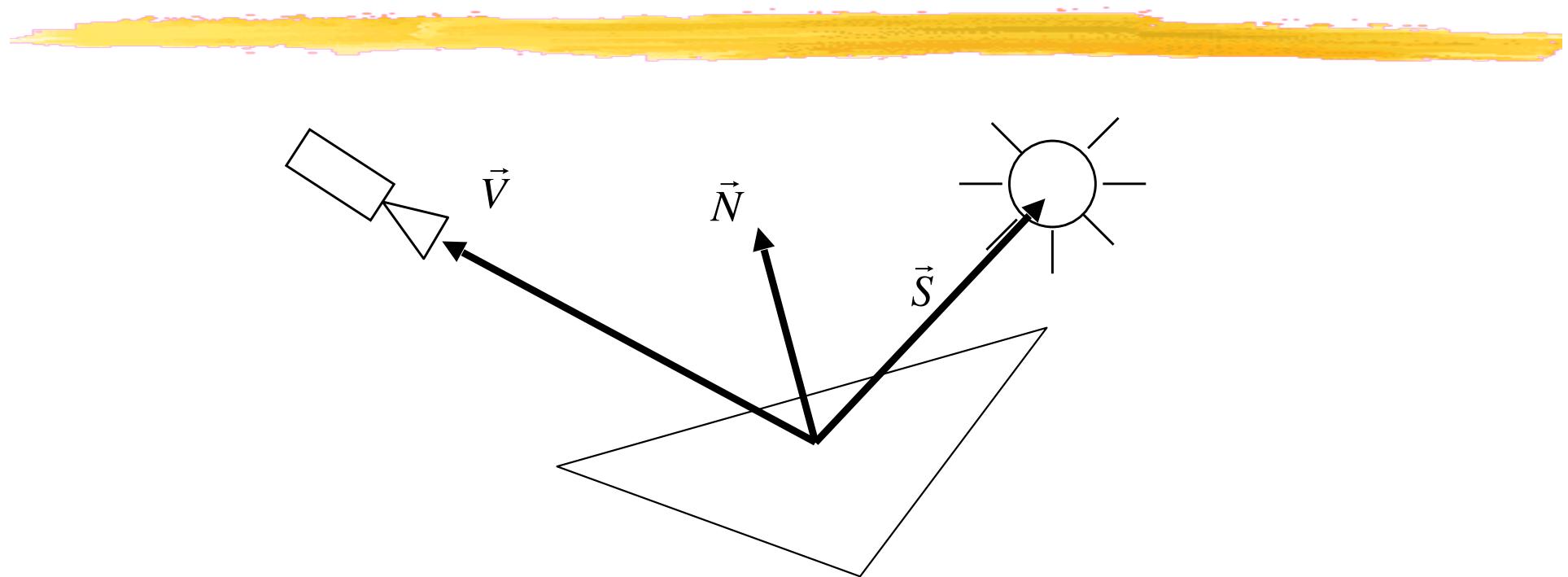


Recovered shading



Recovered albedoes

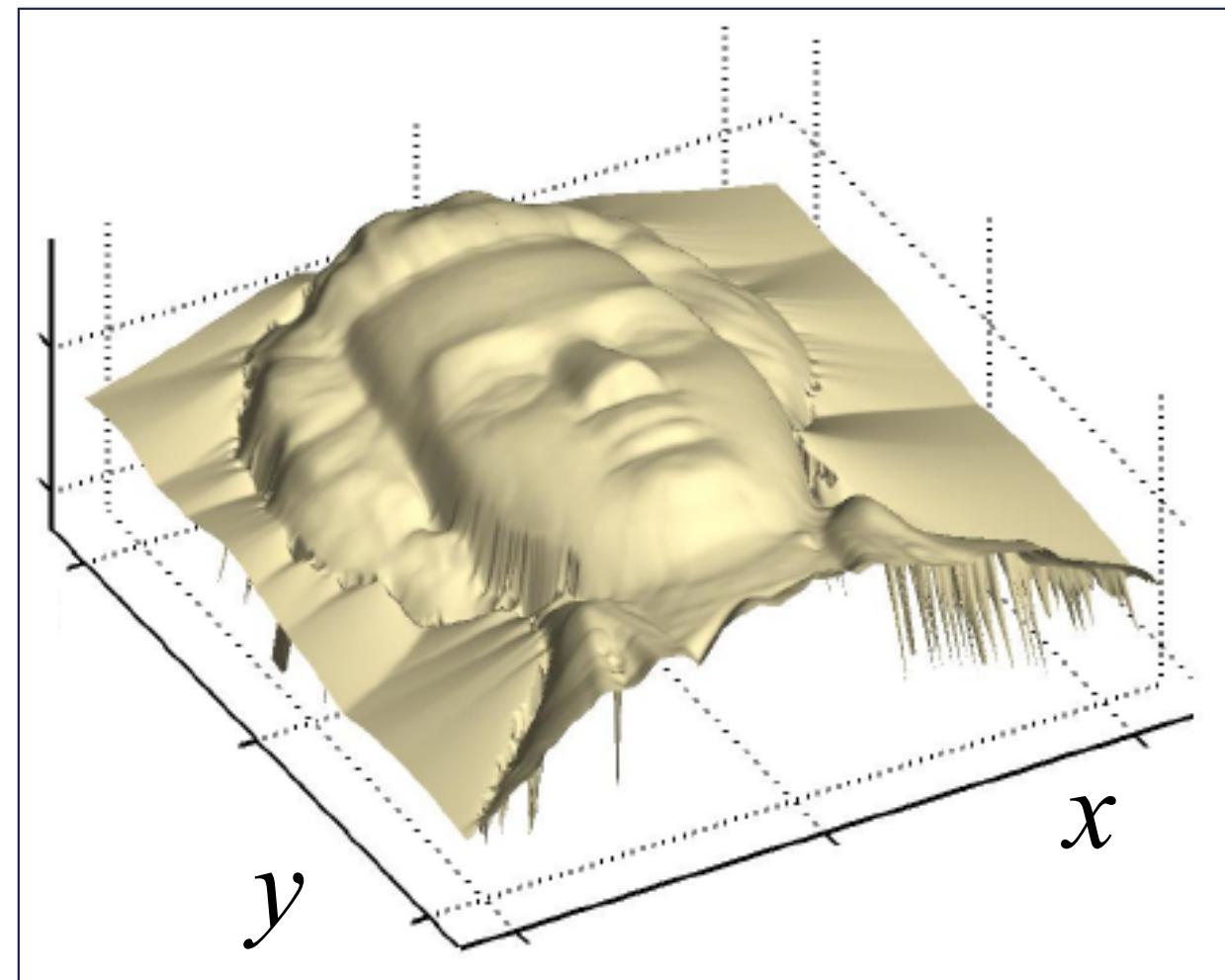
SIMPLIFYING ASSUMPTIONS



- The illumination sources are distant from the imaged surfaces
- Secondary illumination is not significant
- There are no cast shadows

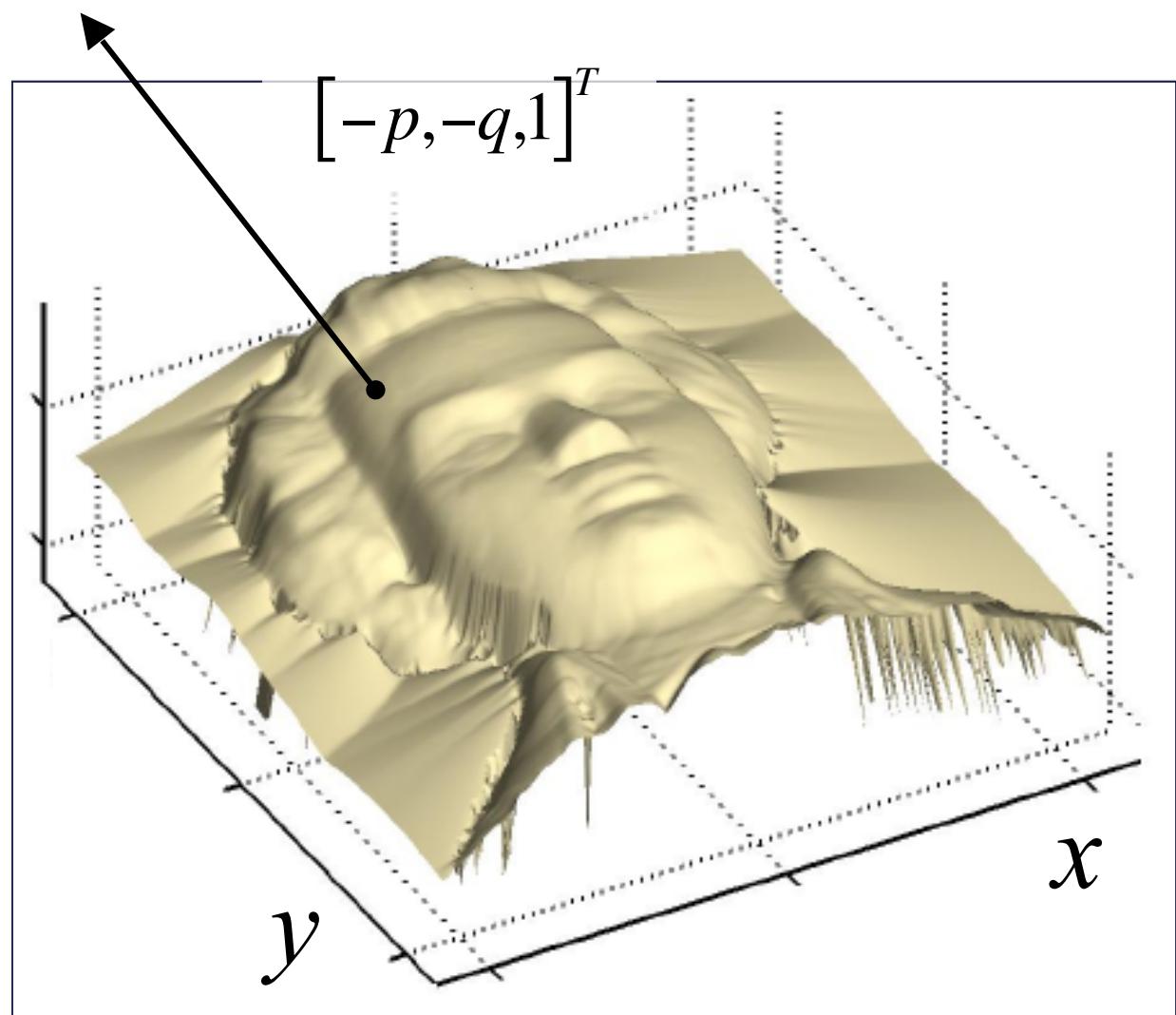
MONGE SURFACE

$$z = f(x, y)$$



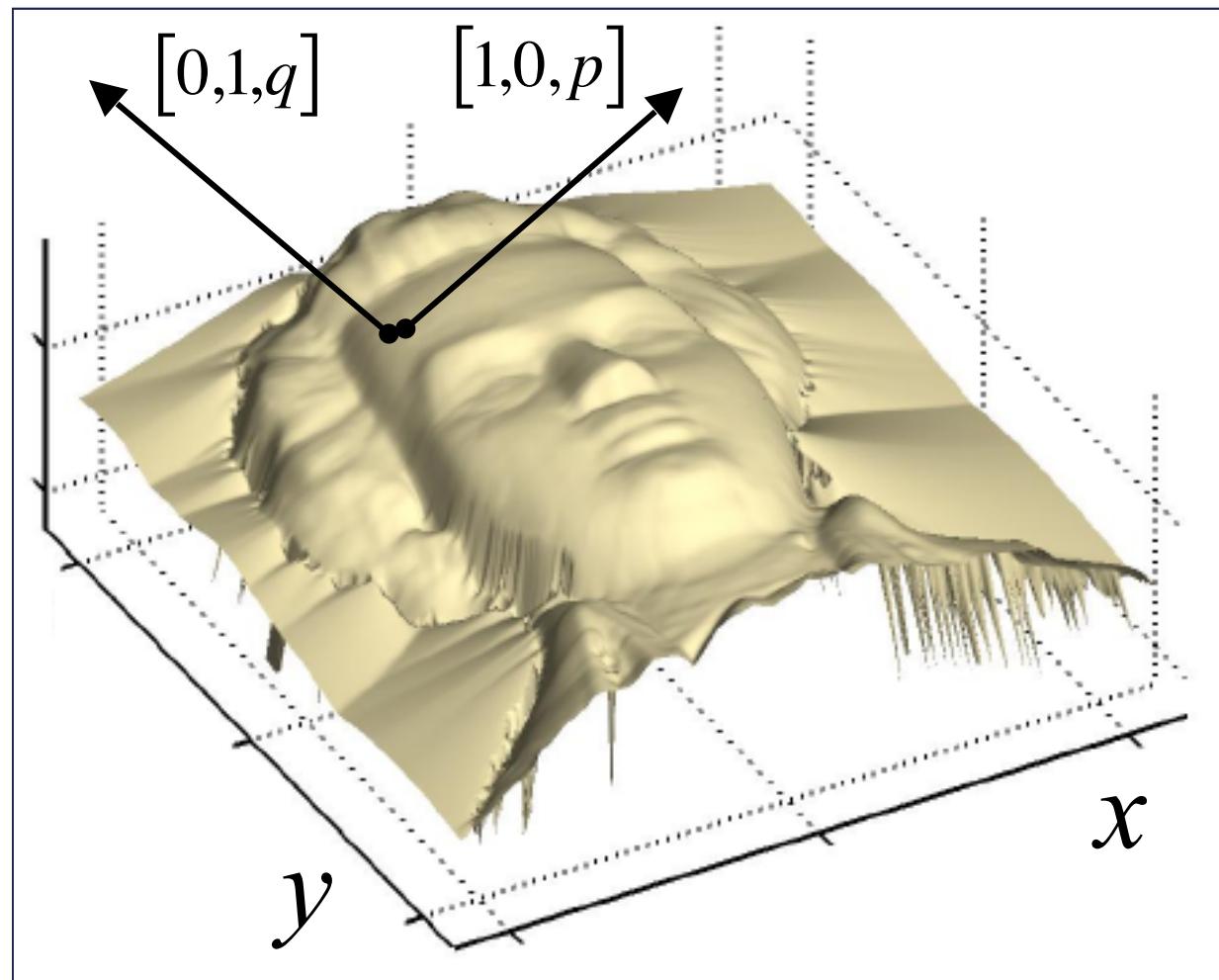
SURFACE NORMALS

$$z = f(x, y)$$
$$p = \frac{\delta z}{\delta x}$$
$$q = \frac{\delta z}{\delta y}$$



TANGENT VECTORS

$$\begin{aligned}z &= f(x, y) \\p &= \frac{\delta z}{\delta x} \\q &= \frac{\delta z}{\delta y}\end{aligned}$$



PROJECTION

Elevation and Normal :

$$z = f(x, y)$$

$$\mathbf{N}(x, y) = \frac{1}{\sqrt{1 + f_x^2 + f_y^2}} \begin{bmatrix} -f_x \\ -f_y \\ 1 \end{bmatrix}$$

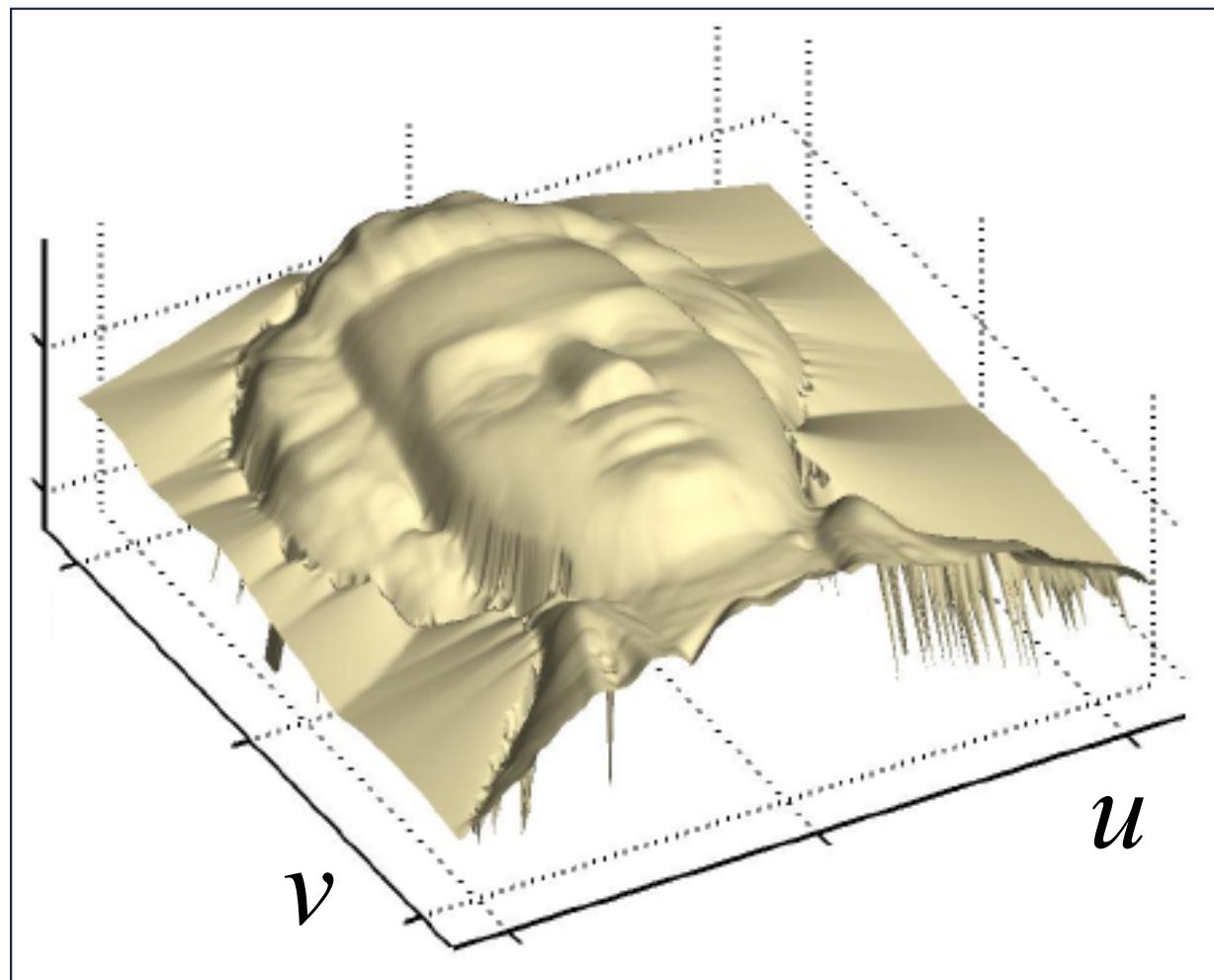
Orthographic Projection :

$$u = sx$$

$$v = sy$$

RE-PARAMETRIZATION

$$z = f(u, v)$$



SHAPE FROM NORMALS

$$\begin{aligned} N &= \frac{1}{\sqrt{1 + \frac{\delta z}{\delta x}^2 + \frac{\delta z}{\delta y}^2}} \begin{bmatrix} -\frac{\delta z}{\delta x} \\ -\frac{\delta z}{\delta y} \\ 1 \end{bmatrix} \propto \begin{bmatrix} -\frac{1}{s} \frac{\delta z}{\delta u} \\ -\frac{1}{s} \frac{\delta z}{\delta v} \\ 1 \end{bmatrix} \\ \Rightarrow \frac{\delta z}{\delta u} &= -s \frac{n_x}{n_z} \text{ and } \frac{\delta z}{\delta v} = -s \frac{n_y}{n_z} \\ \Rightarrow \frac{\delta \bar{z}}{\delta u} &= -\frac{n_x}{n_z} = n_1 \text{ and } \frac{\delta \bar{z}}{\delta v} = -\frac{n_y}{n_z} = n_2, \text{ with } \bar{z} = \frac{z}{s} \end{aligned}$$

FINITE DIFFERENCES

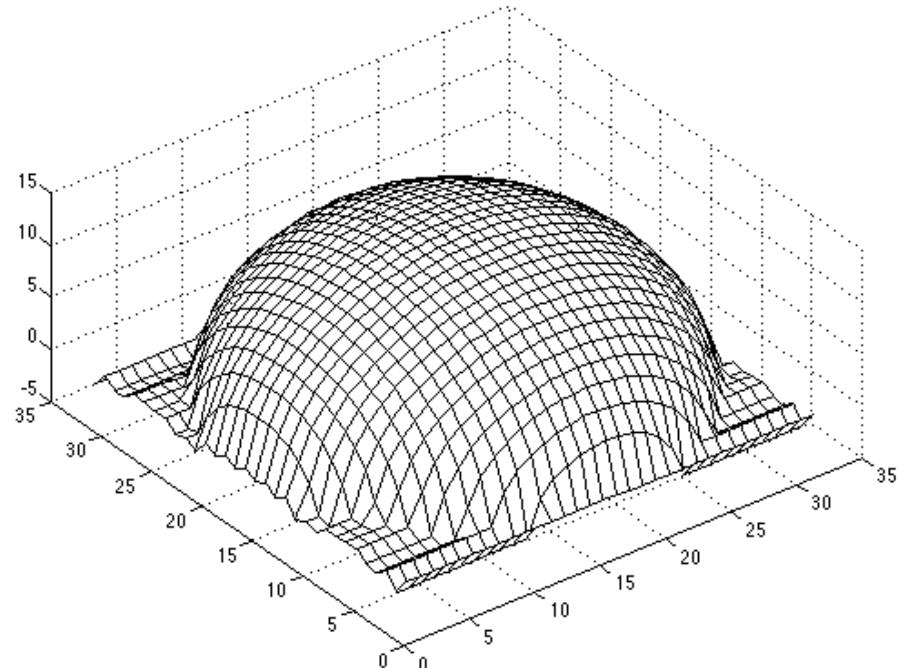
$\forall u, v$

$$\bar{z}(u+1, v) - \bar{z}(u, v) = n_1(u, v)$$

$$\bar{z}(u, v+1) - \bar{z}(u, v) = n_2(u, v)$$

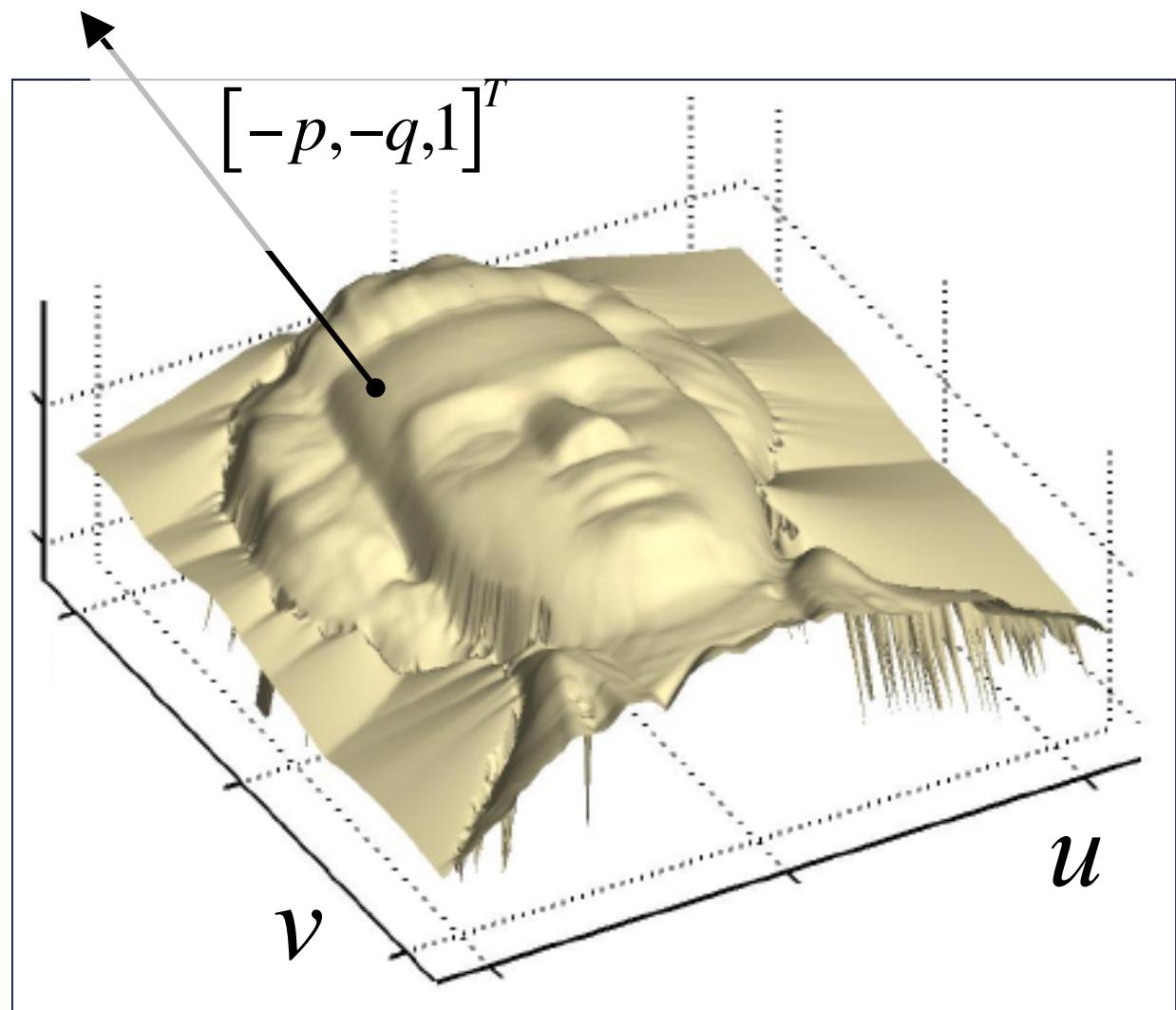
\Rightarrow Twice as many equations as there are unknown.

\Rightarrow Least square solution up to a scale factor.



GRADIENT SPACE

$$\begin{aligned}z &= f(u, v) \\p &= \frac{\delta z}{\delta u} \\q &= \frac{\delta z}{\delta v}\end{aligned}$$



REFLECTANCE MAP



Reflectance:

Amount of light reflected towards the camera.

Albedo:

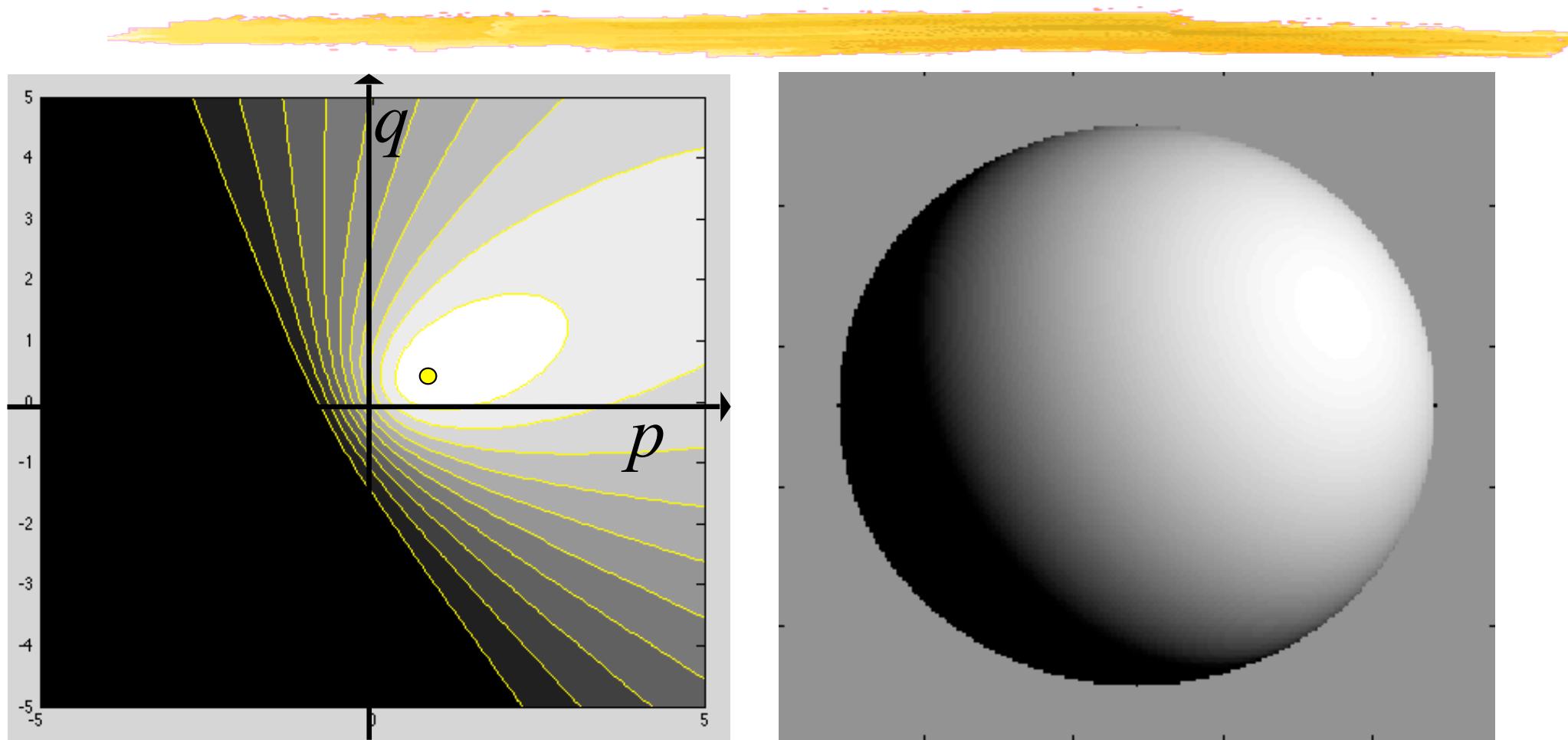
Fraction of the light incident on the surface that is reflected over all directions.

- In the Lambertian case and for a constant albedo

$$I(u,v) \propto Ref(p(u,v),q(u,v))$$

$$\propto [p(u,v), q(u,v), -1] \cdot S$$

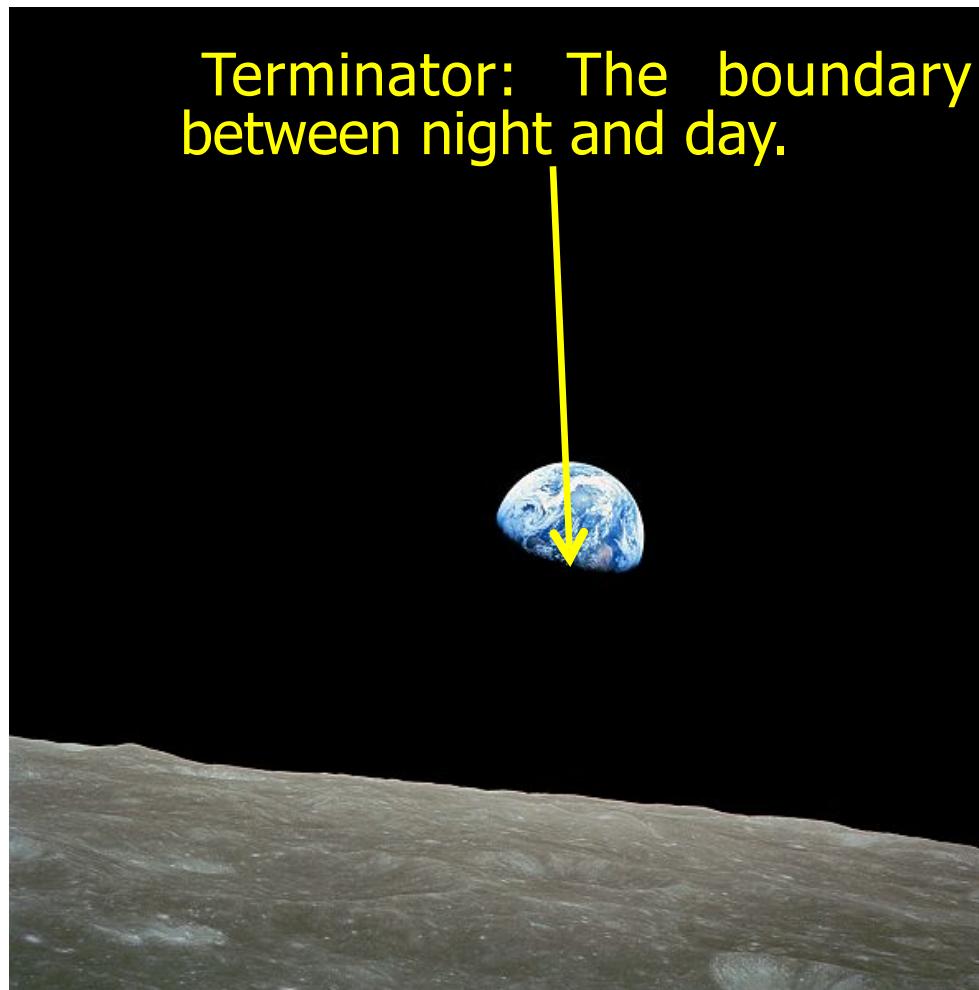
LAMBERTIAN REFLECTANCE MAP



Reflectance map and shaded surface for Lambertian surface illuminated in the direction $[-1 \ -0.5 \ -1]$.

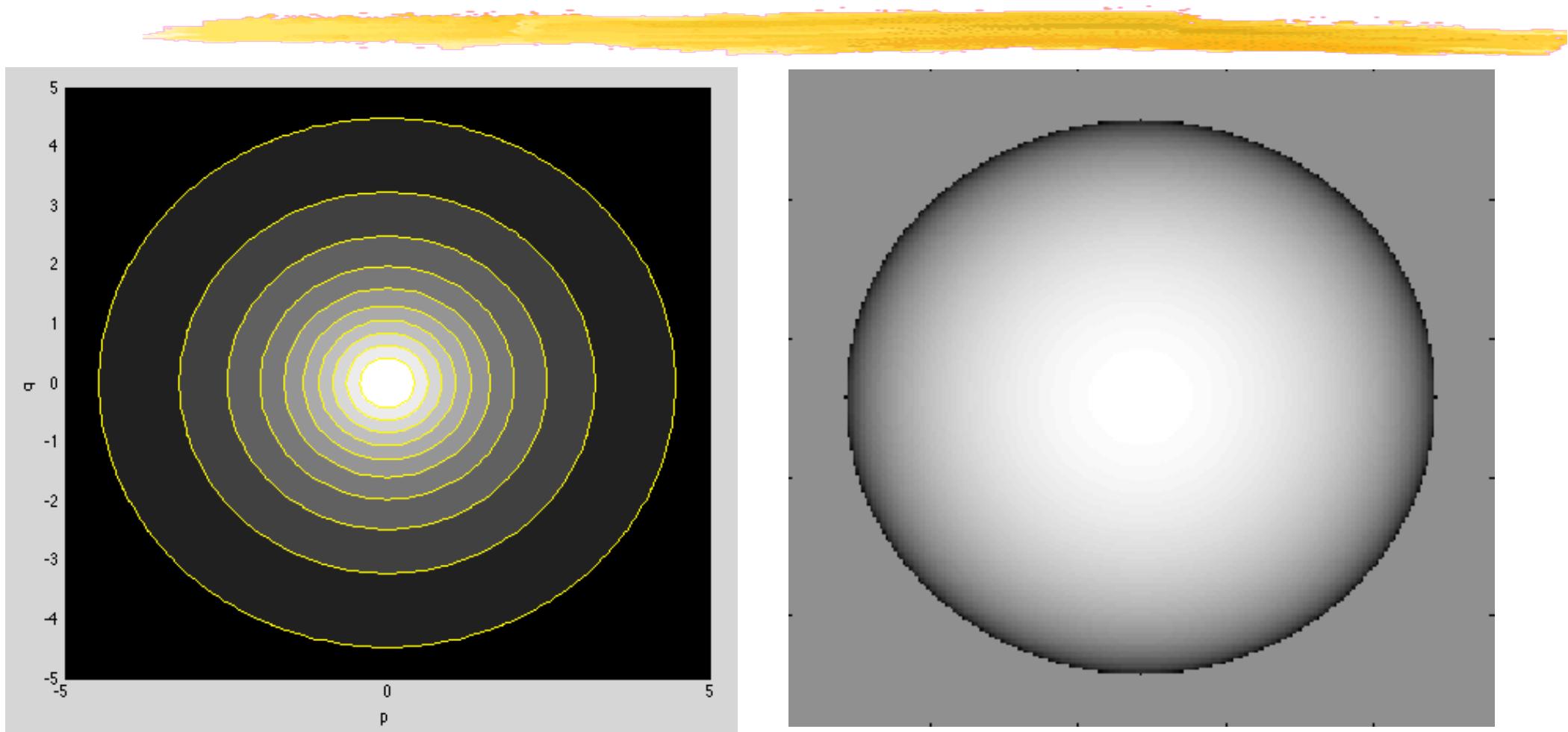
EARTH RISE

Terminator: The boundary between night and day.



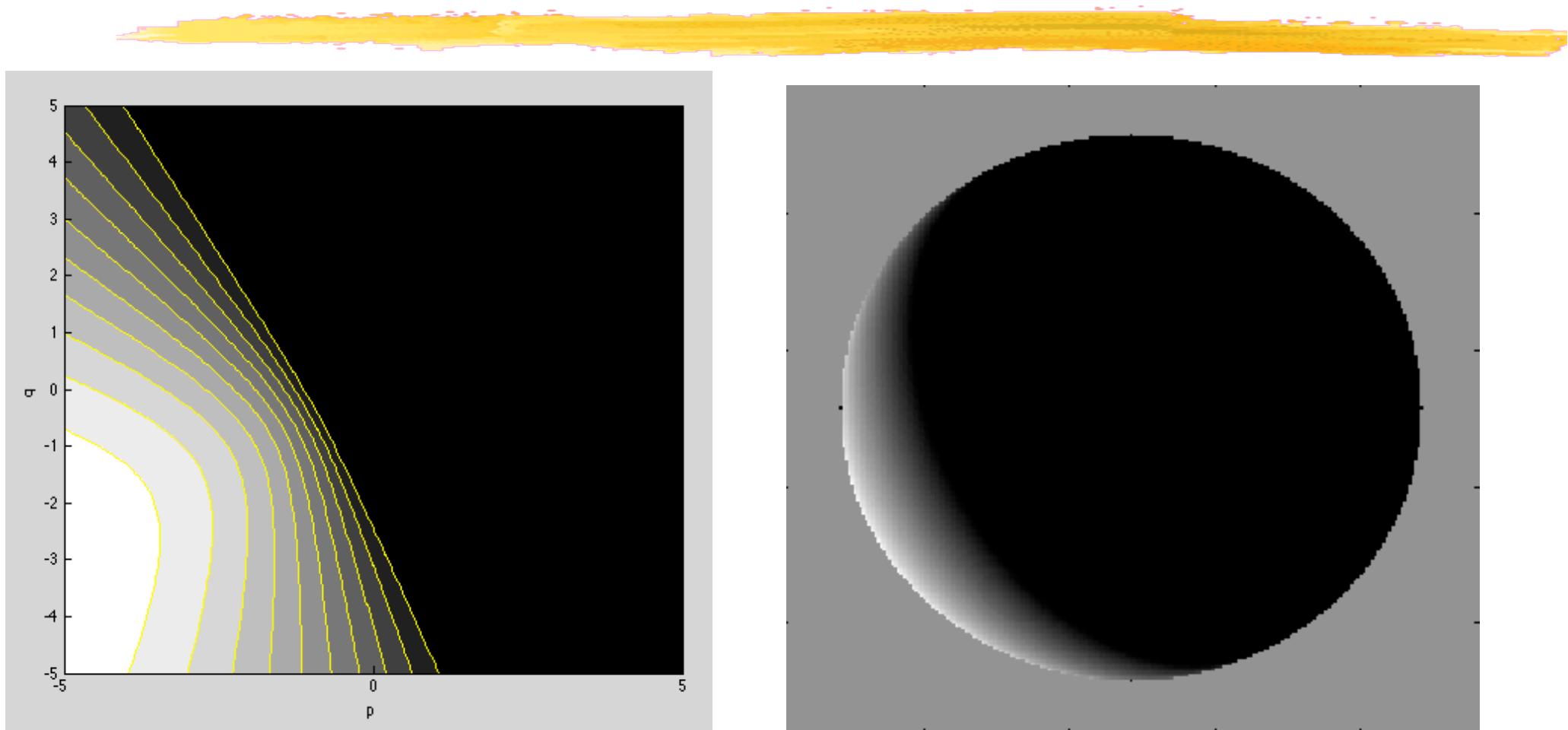
The earth seen from the moon. Apollo 8, 1968.

LAMBERTIAN REFLECTANCE MAP



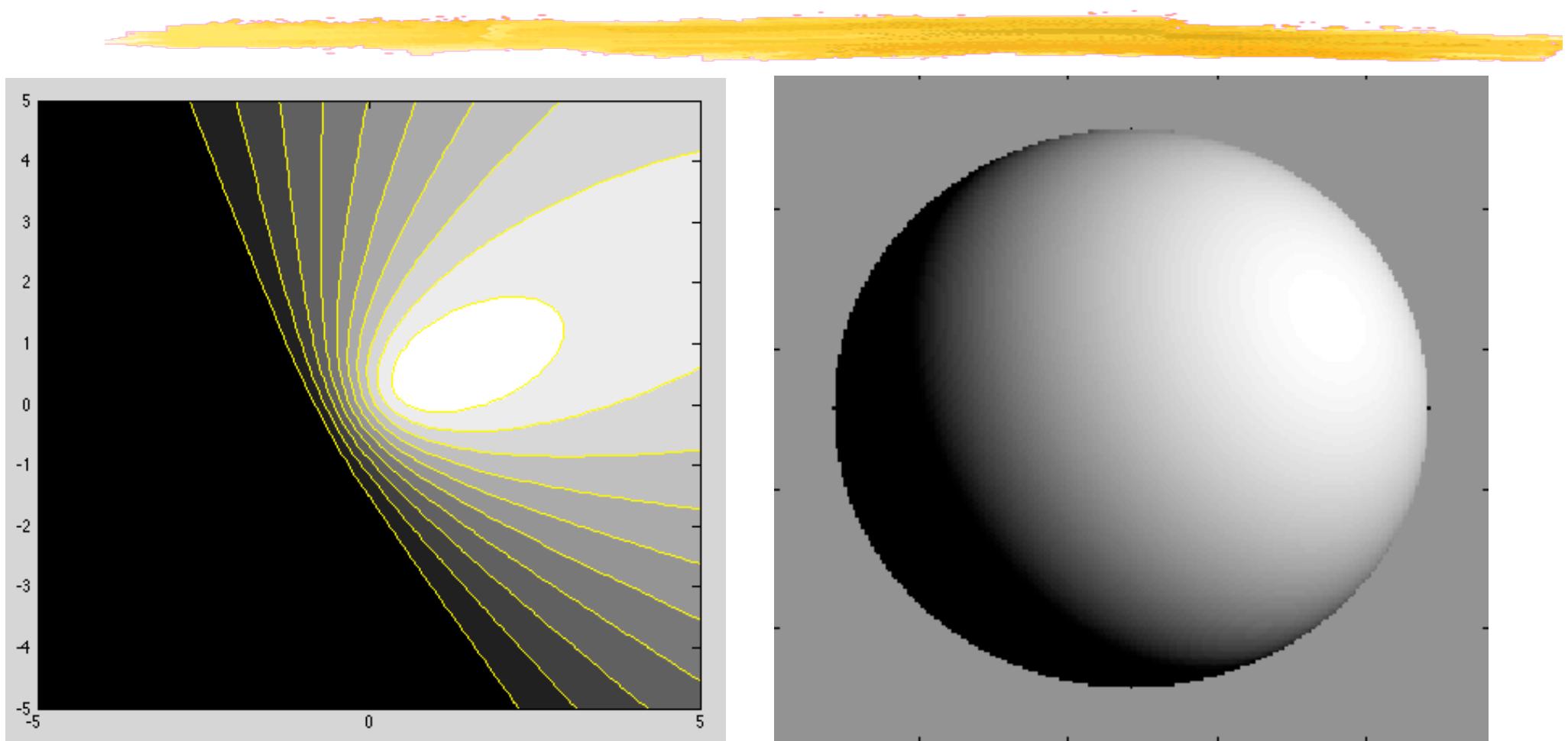
Reflectance map and shaded surface for Lambertian surface illuminated in the direction [0 0 -1].

LAMBERTIAN REFLECTANCE MAP



Reflectance map and shaded surface for Lambertian surface illuminated in the direction [1 0.5 -1].

CAN WE DETERMINE (P, Q) UNIQUELY FOR EACH IMAGE POINT INDEPENDENTLY?



NO -> Global optimization required.

VARIATIONAL METHODS

Minimize:

$$\int \int \left(\left[I(u, v) - Ref\left(\frac{\delta z}{\delta u}, \frac{\delta z}{\delta v}\right) \right]^2 + \lambda \left[\left(\frac{\delta^2 z}{\delta u^2} \right)^2 + \left(\frac{\delta^2 z}{\delta u \delta v} \right)^2 + \left(\frac{\delta^2 z}{\delta v^2} \right)^2 \right] \right) dudv$$



or:

Brightness
constraint

Smoothness
term

$$\int \int \left([I(u, v) - Ref(p, q)]^2 + \lambda \left[\left(\frac{\delta p}{\delta u} \right)^2 + \left(\frac{\delta p}{\delta v} \right)^2 + \left(\frac{\delta q}{\delta u} \right)^2 + \left(\frac{\delta q}{\delta v} \right)^2 \right] + \mu \left[\frac{\delta p}{\delta v} - \frac{\delta q}{\delta u} \right]^2 \right) dudv$$



Integrability
constraint

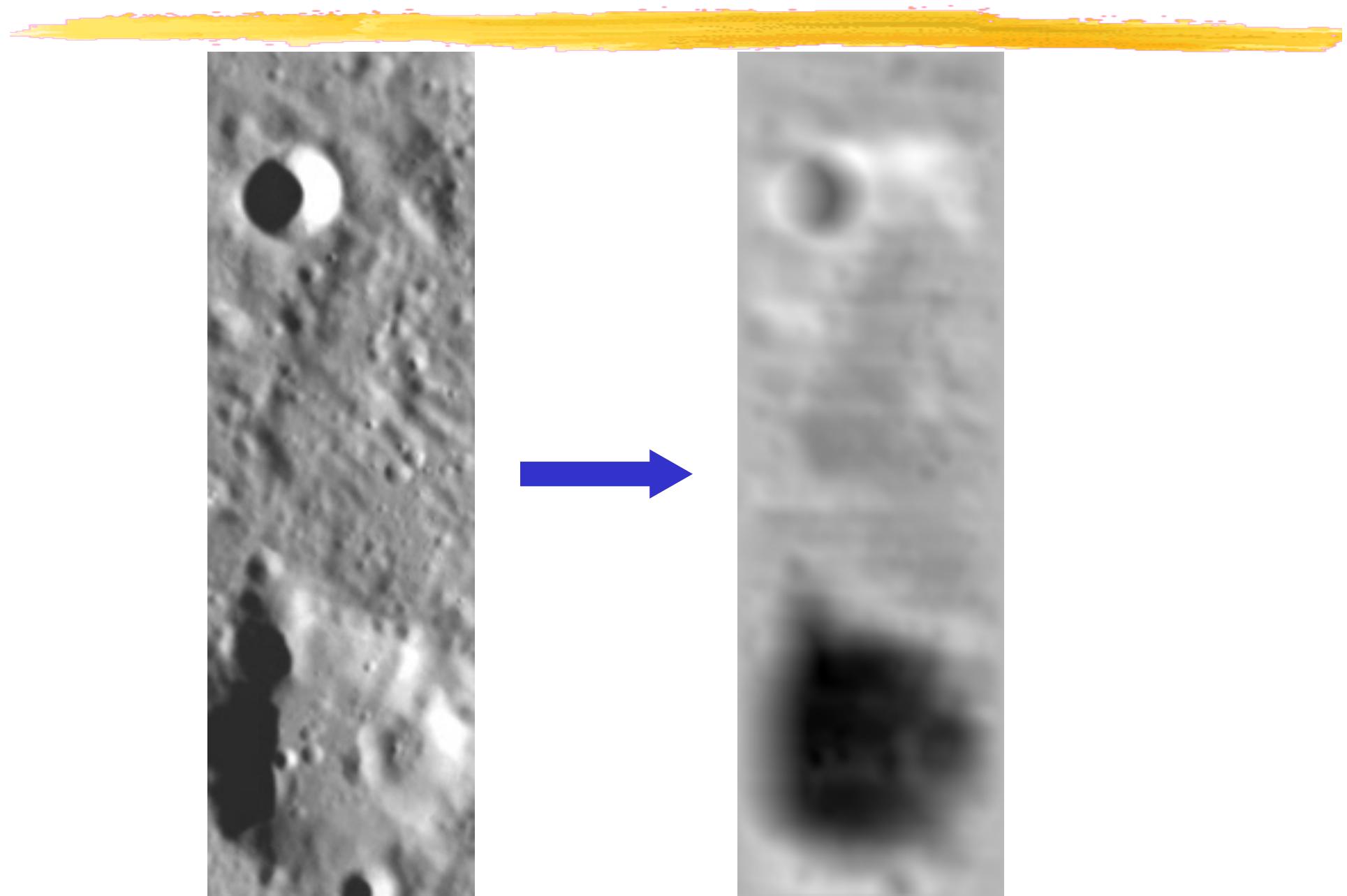
VARIATIONAL METHOD

$$\int \int \left([I(u, v) - R e f(p, q)]^2 + \lambda \left[\left(\frac{\delta p}{\delta u} \right)^2 + \left(\frac{\delta p}{\delta v} \right)^2 + \left(\frac{\delta q}{\delta u} \right)^2 + \left(\frac{\delta q}{\delta v} \right)^2 \right] + \mu \left[\frac{\delta p}{\delta v} - \frac{\delta q}{\delta u} \right]^2 \right) dudv$$

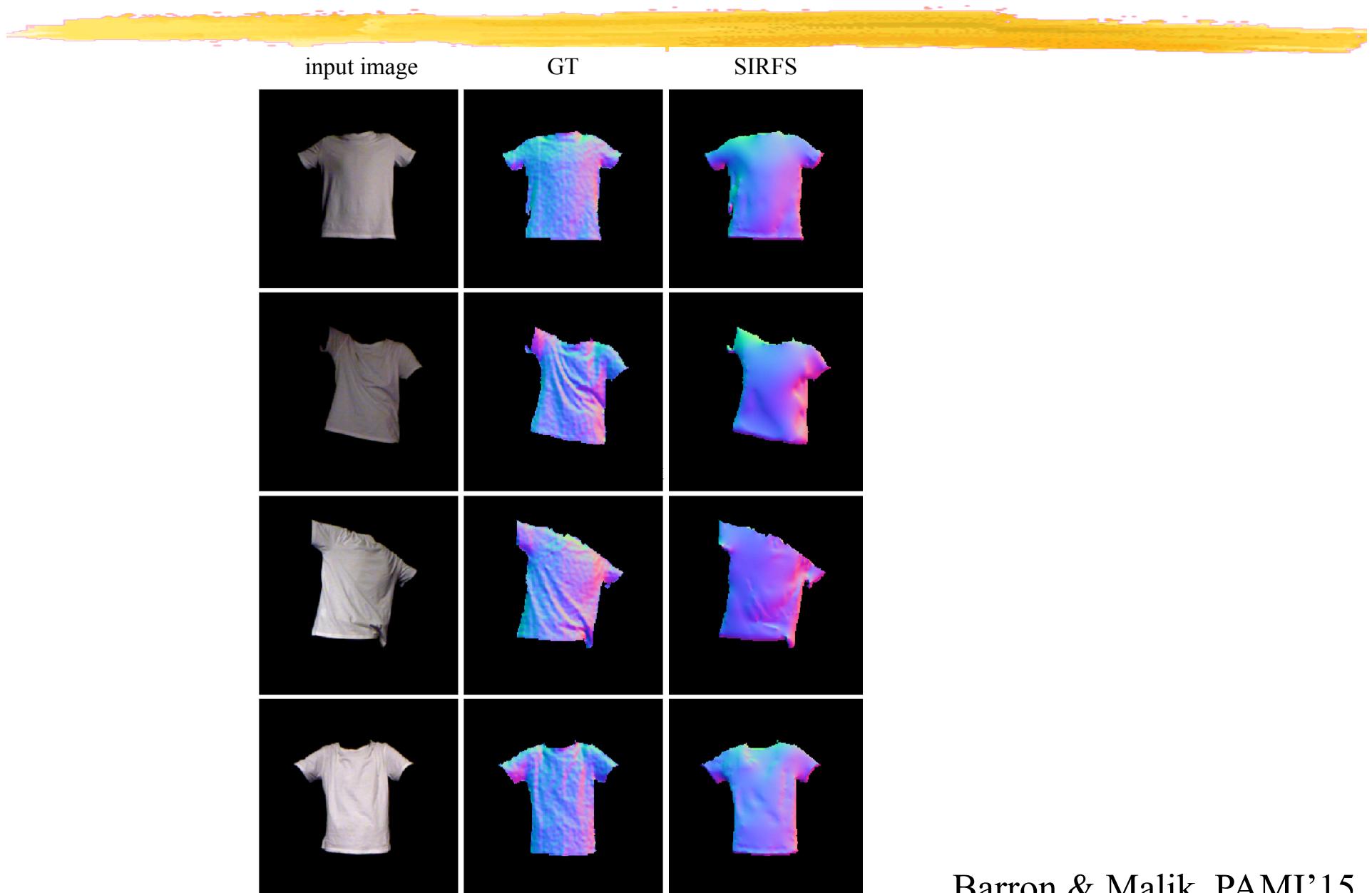
Once p and q have been estimated, integrate to recover f .

→ Need to know the boundary conditions.

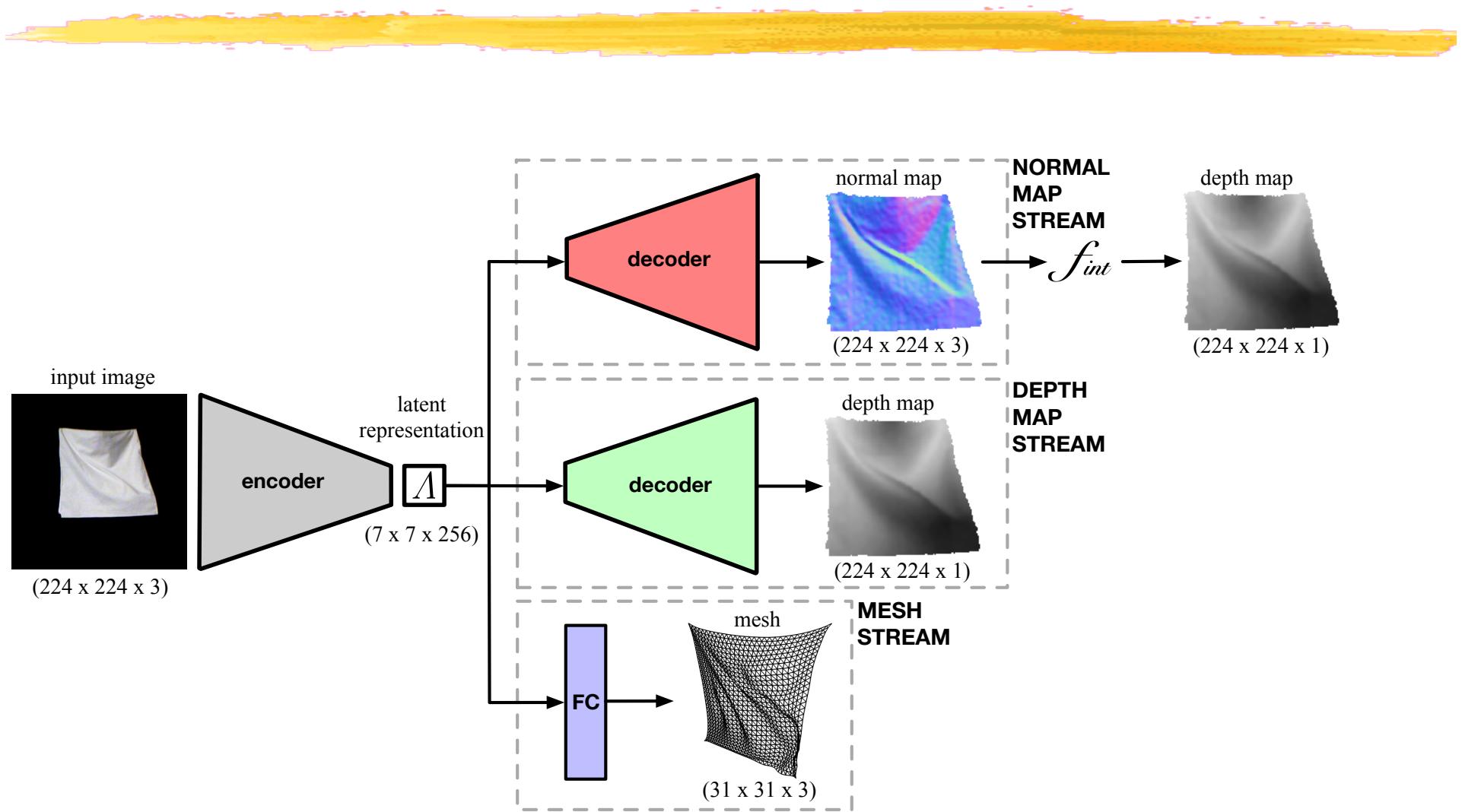
MOONSCAPE



CLOTH



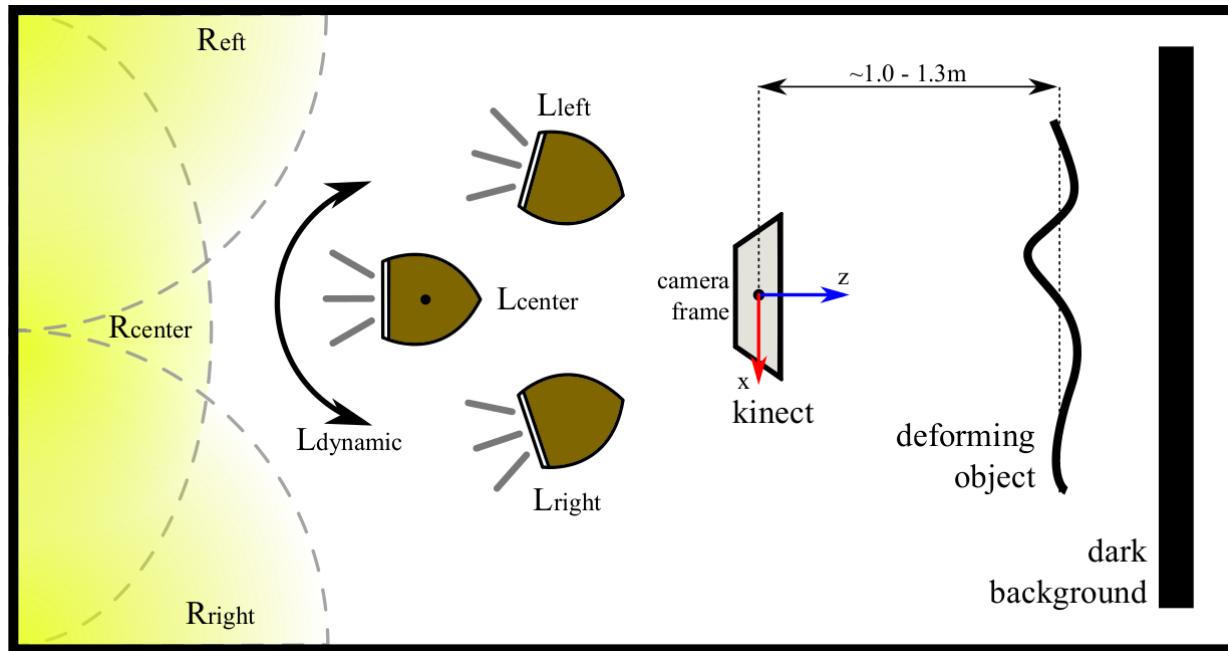
DEEP NETS



DEEP NETS

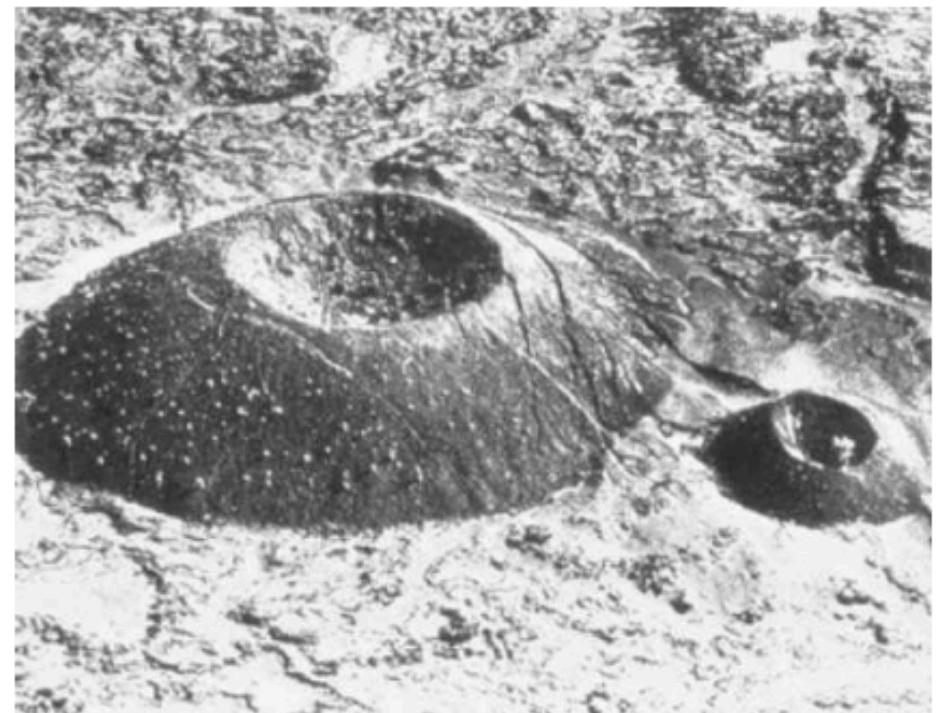


DATA ACQUISITION SETUP



- 3 fixed light sources.
- 1 mobile one.
→ Still a constrained environment.

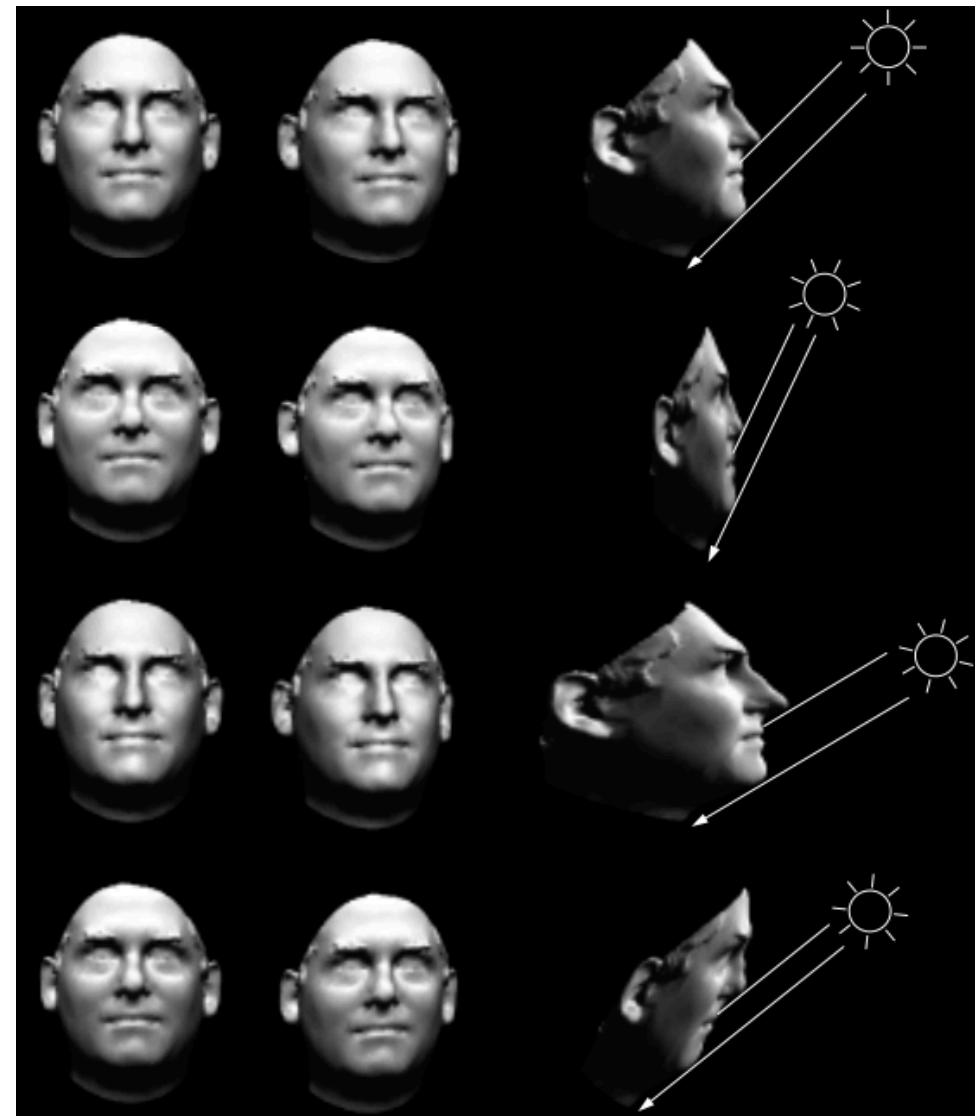
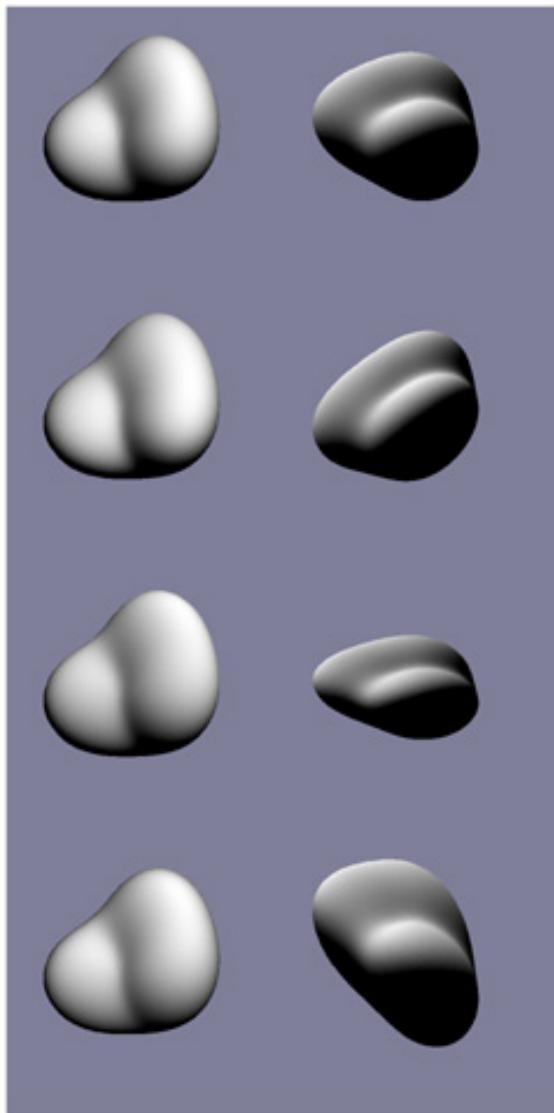
AMBIGUITIES



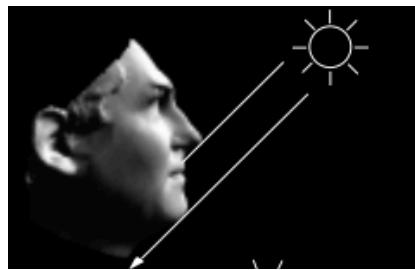
THE BAS-RELIEF AMBIGUITY



MORE GENERALLY

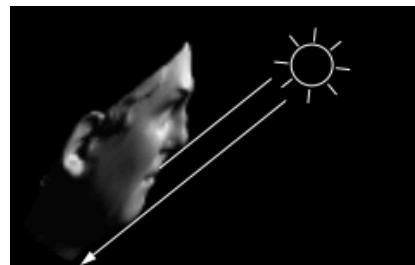


BAS-RELIEF AMBIGUITY



$$Ref = \mathbf{N} \cdot \mathbf{S}$$

For any invertible 3×3 linear transformation A :



$$\mathbf{N} \cdot \mathbf{S} = N^T S = (A\mathbf{N})^T A^{-T} \mathbf{S}$$

But for a valid surface $z=f(u,v)$, we should have:

$$\left. \begin{array}{lcl} \frac{\delta z}{\delta u} & = & -\frac{n_x^*}{n_z^*} \\ \frac{\delta z}{\delta v} & = & -\frac{n_y^*}{n_z^*} \end{array} \right\} \Rightarrow \frac{\delta \frac{n_x^*}{n_z^*}}{\delta v} = \frac{\delta \frac{n_y^*}{n_z^*}}{\delta u} \text{ with } \begin{bmatrix} n_x^* \\ n_y^* \\ n_z^* \end{bmatrix} = \mathbf{A} \begin{bmatrix} n_x \\ n_y \\ n_z \end{bmatrix}$$

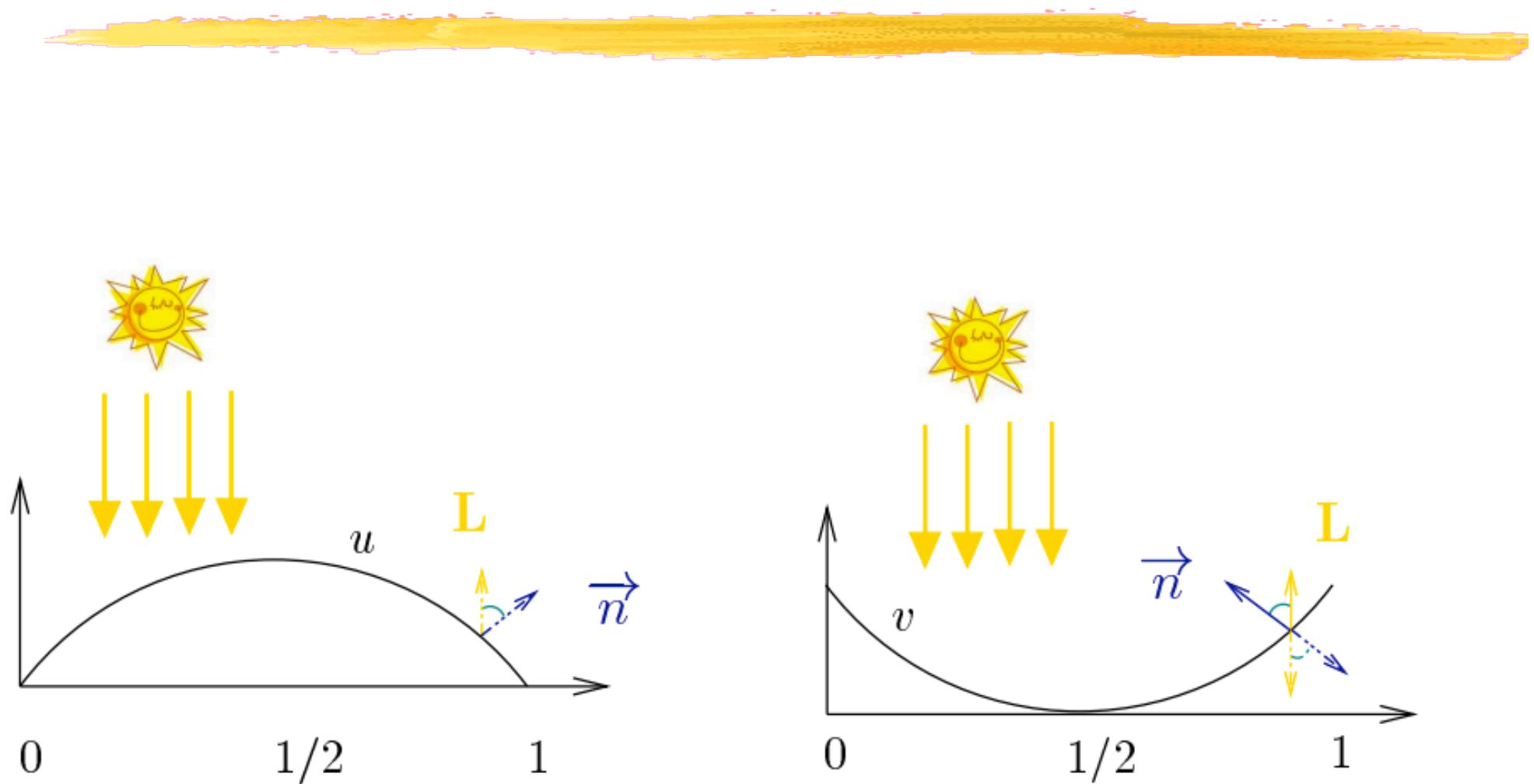
BAS-RELIEF AMBIGUITY

$$\left. \begin{array}{l} \frac{\delta z}{\delta u} = -\frac{n_x^*}{n_z^*} \\ \frac{\delta z}{\delta v} = -\frac{n_y^*}{n_z^*} \end{array} \right\} \Rightarrow \frac{\delta \frac{n_x^*}{n_z^*}}{\delta v} = \frac{\delta \frac{n_y^*}{n_z^*}}{\delta u} \text{ with } \begin{bmatrix} n_x^* \\ n_y^* \\ n_z^* \end{bmatrix} = \mathbf{A} \begin{bmatrix} n_x \\ n_y \\ n_z \end{bmatrix}$$

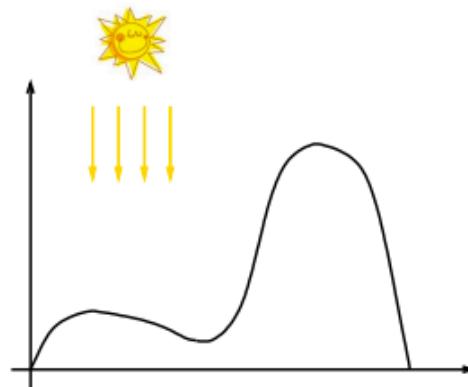
$$\Rightarrow \mathbf{A} \text{ restricted to } \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ \mu & \nu & 1 \end{bmatrix}$$

➤ The surface $f(u, v)$ can be changed into $\lambda f(u, v) + \mu u + \nu v$ and still produce the same image.

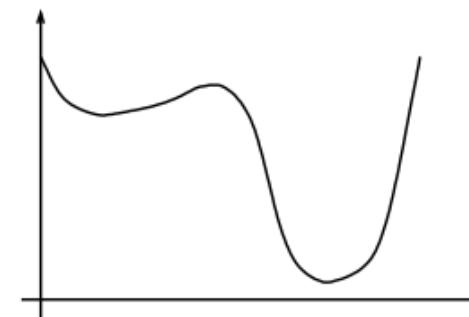
MORE AMBIGUITIES EVEN WHEN THE LIGHT SOURCE IS KNOWN...



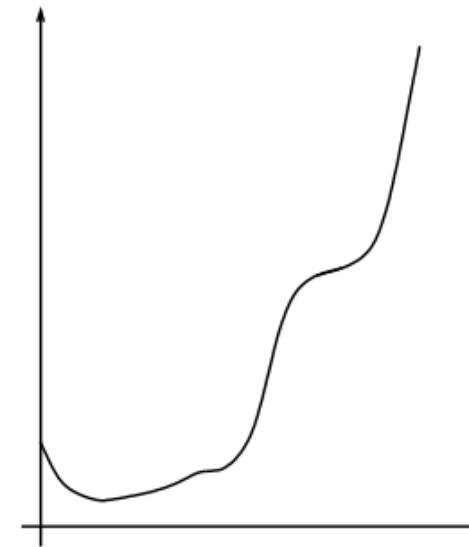
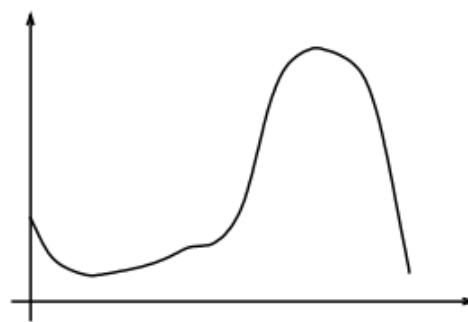
MORE AMBIGUITIES EVEN WHEN THE LIGHT SOURCE IS KNOWN...



a)



b)



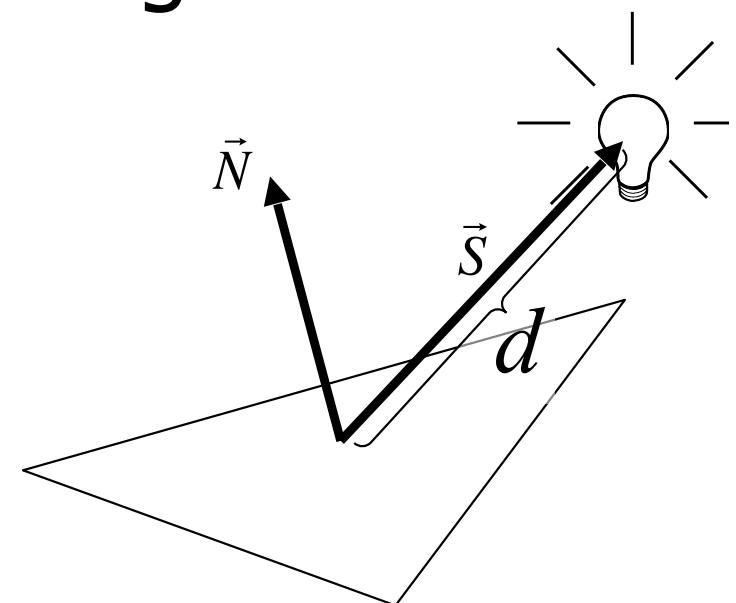
MAKING THE PROBLEM WELL-POSED

- Use perspective projection model;
- Radiance depends on distance to light source:

$$I = \frac{\text{Albedo} \cdot (\mathbf{N} \cdot \mathbf{S})}{d^2}$$

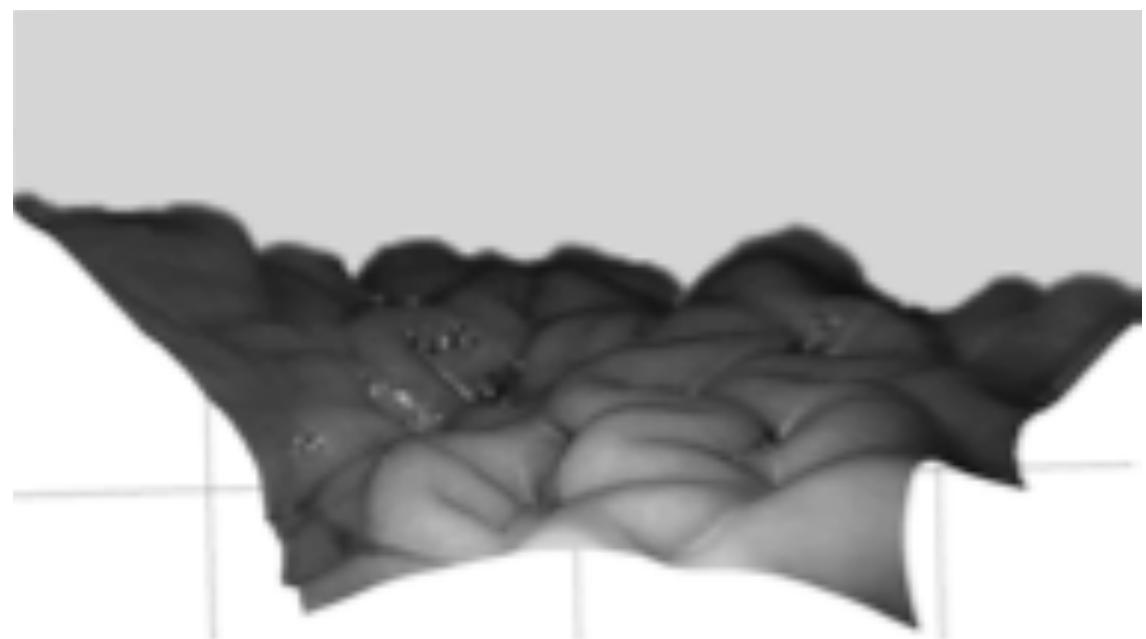
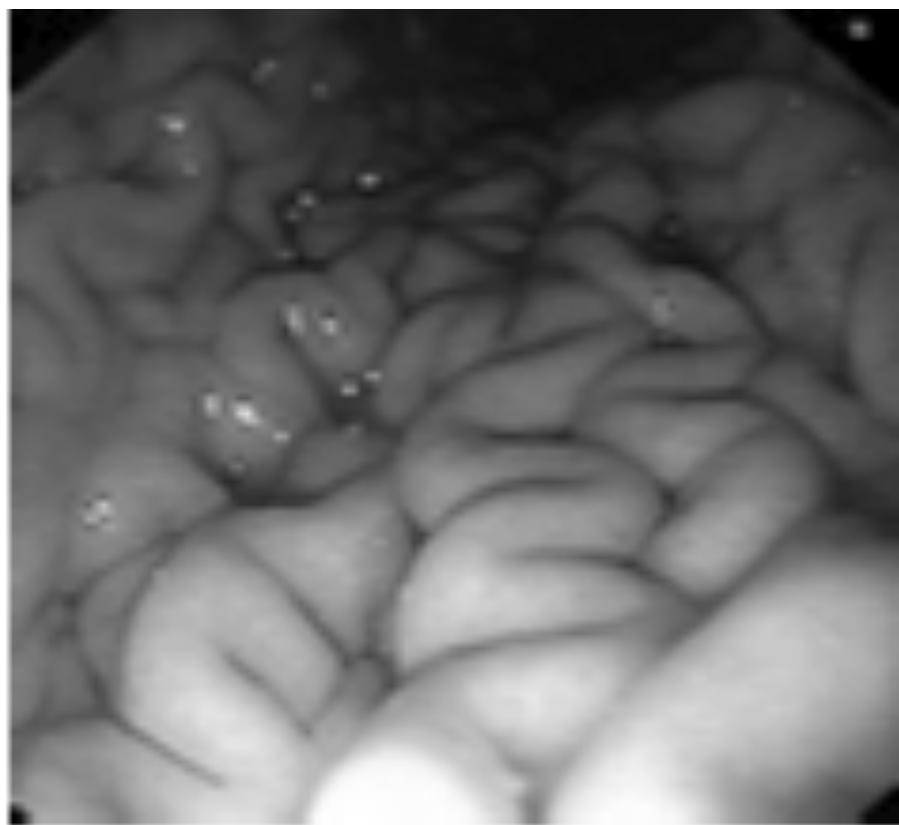
instead of

$$I = \text{Albedo} \cdot (\mathbf{N} \cdot \mathbf{S})$$

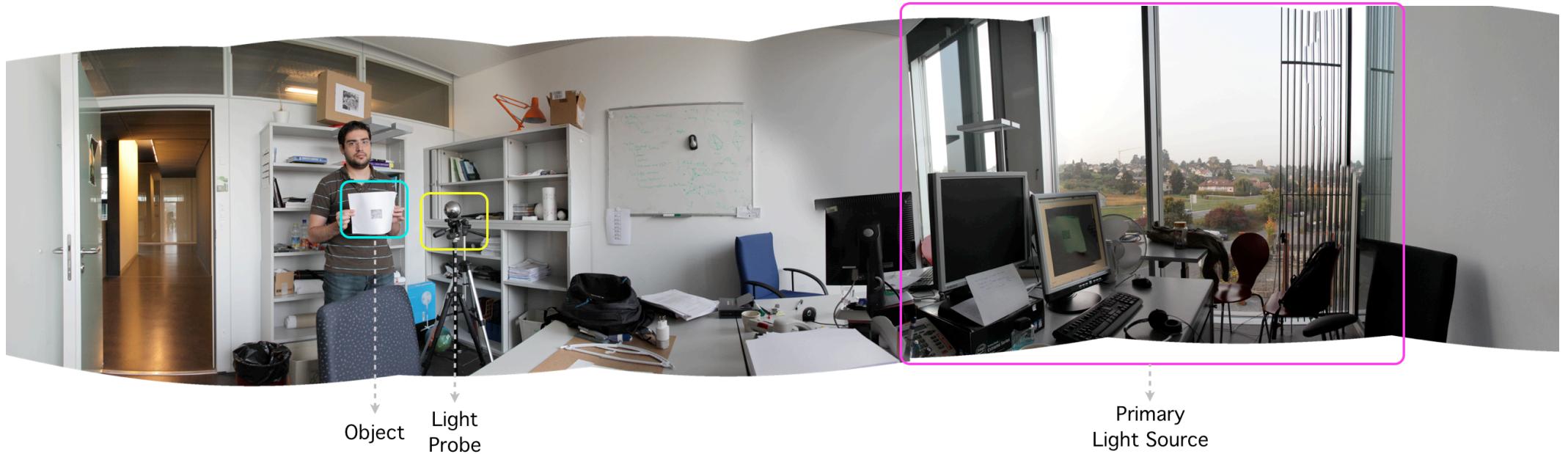


- Light source located at the optical center.
-> Unique solution.

ENDOSCOPY



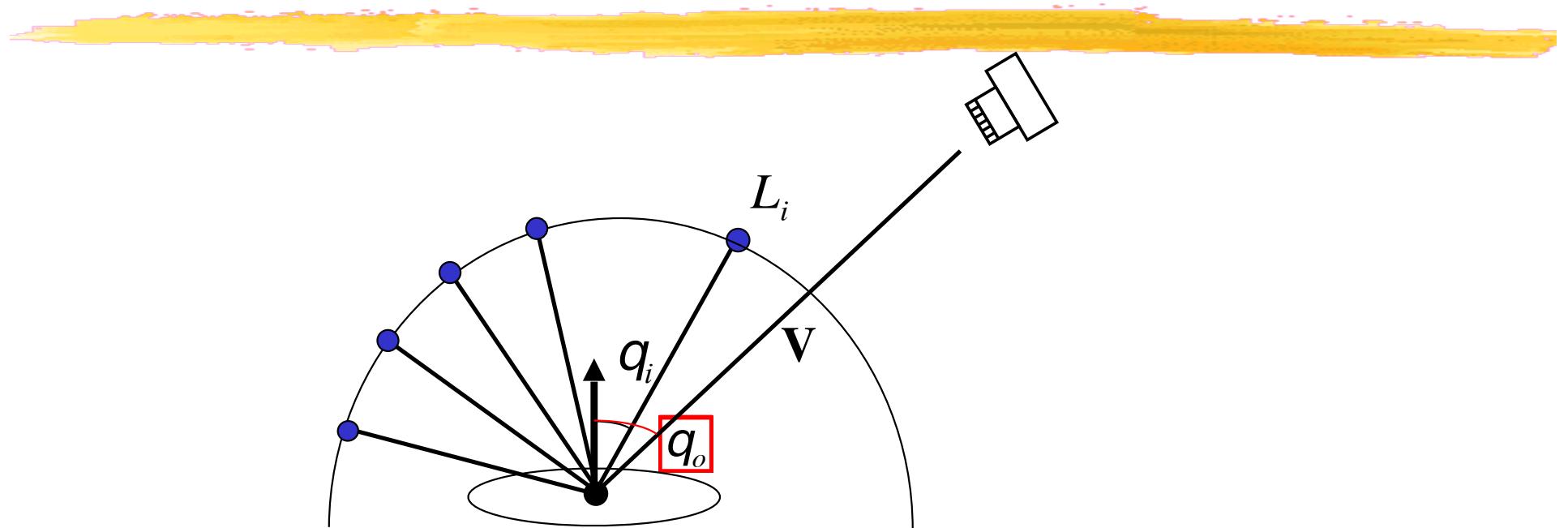
EVERYDAY SETTING



Multiple extended light sources:

- Illumination modeled as a weighted sum of spherical harmonics.
- Illumination parameters estimated using the light probe.

ILLUMINATION HEMISPHERE



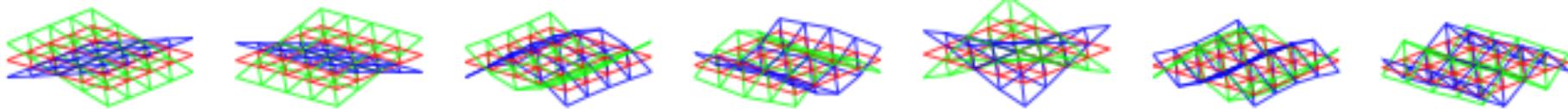
$$\begin{aligned}
 L_o(\mathbf{P}, q_o, f_o) &= \int_{\Omega_i} r_{bd}(q_o, f_o, q_i, f_i) L_i(\mathbf{P}, q_i, f_i) \cos(q_i) d\omega_i \\
 &= \int_{\Omega_i} L_i(q_i, f_i) r_{bd}(q_o, f_o, q_i, f_i) \max(0, \cos(q_i)) d\omega_i \\
 &= \int_{\Omega_i} L_i(q_i, f_i) r^*(q_i, f_i) d\omega_i
 \end{aligned}$$

with $r^*(q_i, f_i) = r_{bd}(q_o, f_o, q_i, f_i) \max(0, \cos(q_i))$ is the BRDF product function.

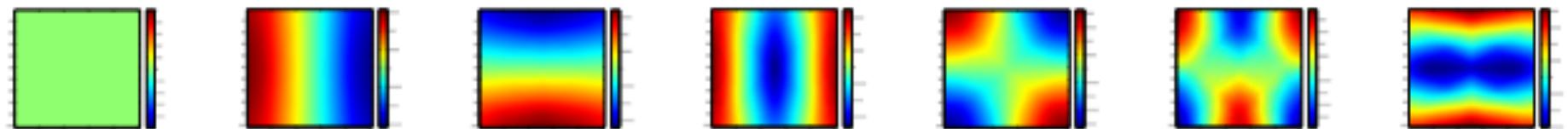
PATCH INTENSITY AND DEFORMATION MODES



Synthesize a training database containing deformed patches and the corresponding intensity patterns. Compute:

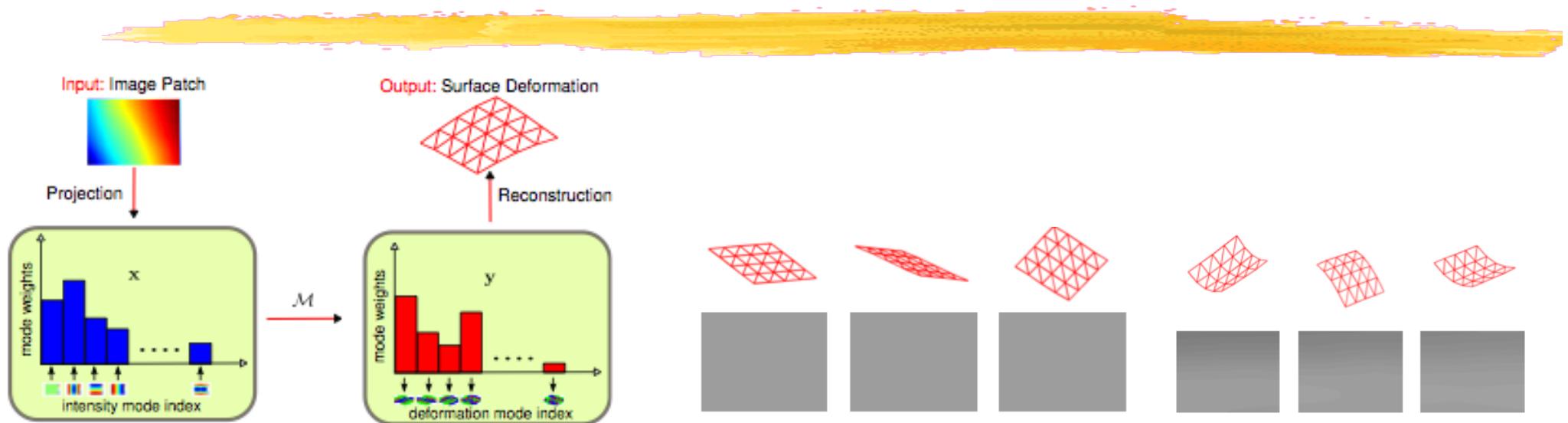


$$\text{Deformation modes: } D = D_0 + \sum_i y_i D_i$$



$$\text{Intensity modes: } I = I_0 + \sum_i x_i I_i$$

AMBIGUOUS MAPPING



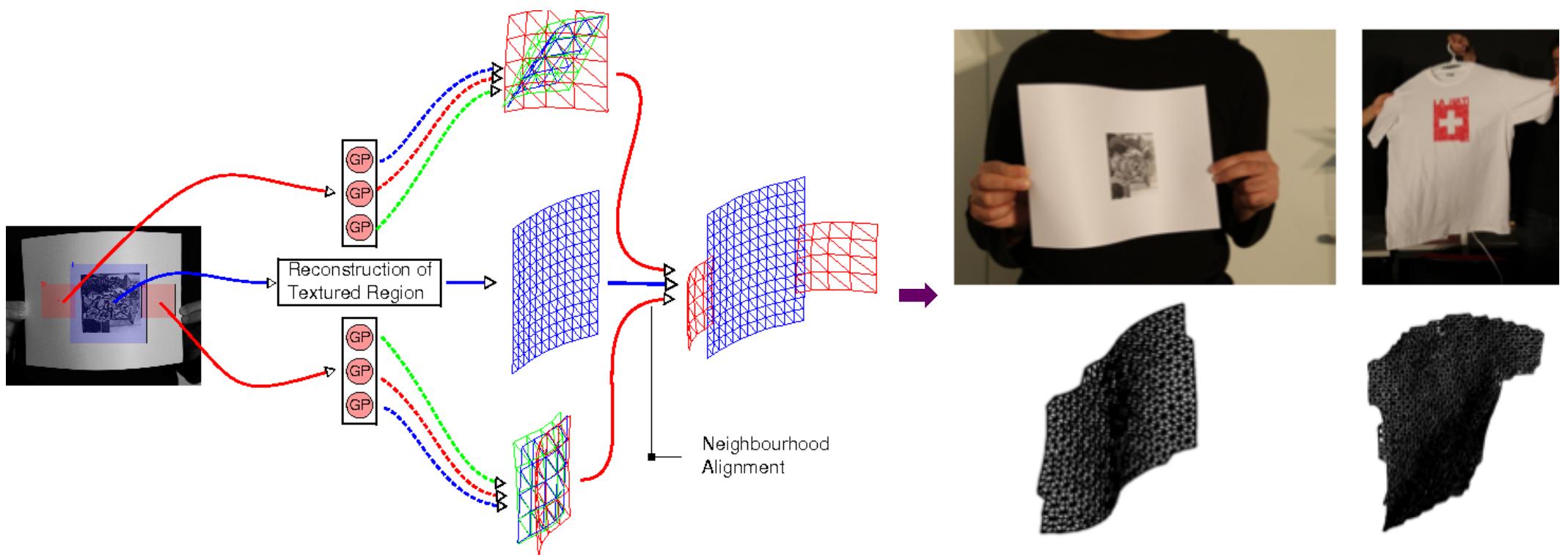
Mapping from intensity to surface deformation.

Unfortunately, the mapping is not one to one.

Algorithm:

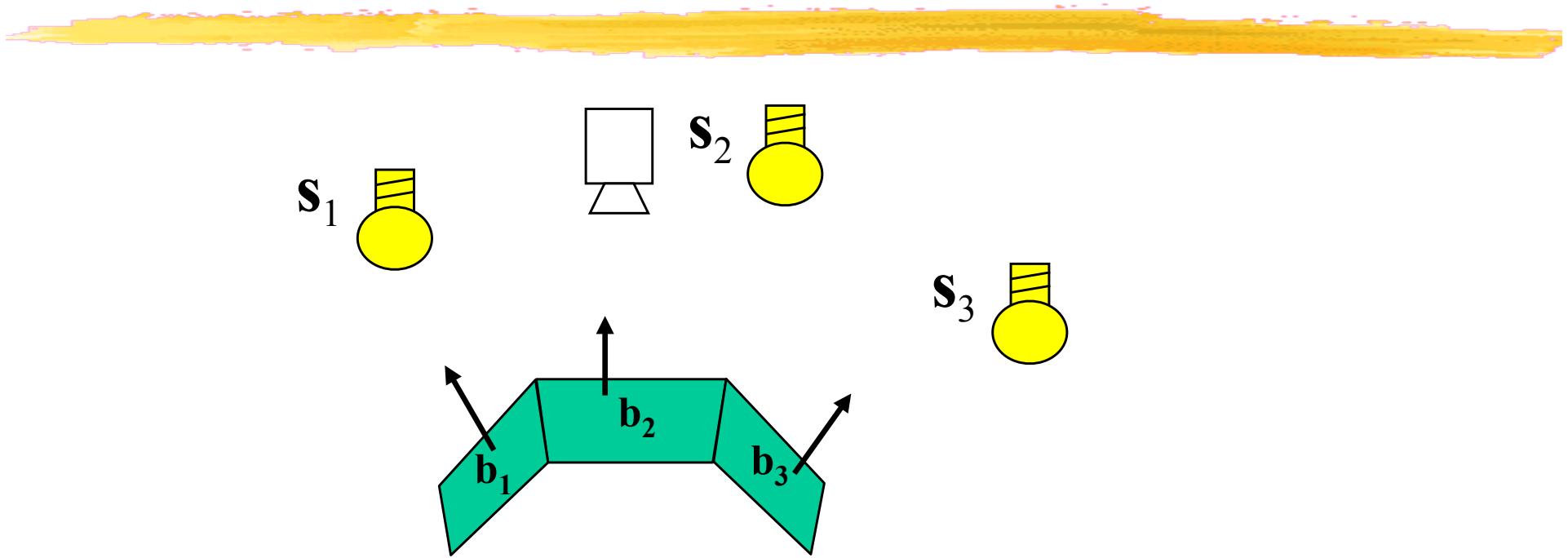
- Partition the training database according to average normal to learn an unambiguous mapping.
- At run-time, predict several potential shapes for each image-patch and use a Markov-Random field to pick a set of consistent interpretations.

ALGORITHMIC FLOW



→ The light environment and the camera need to be very carefully calibrated but there is hope ...

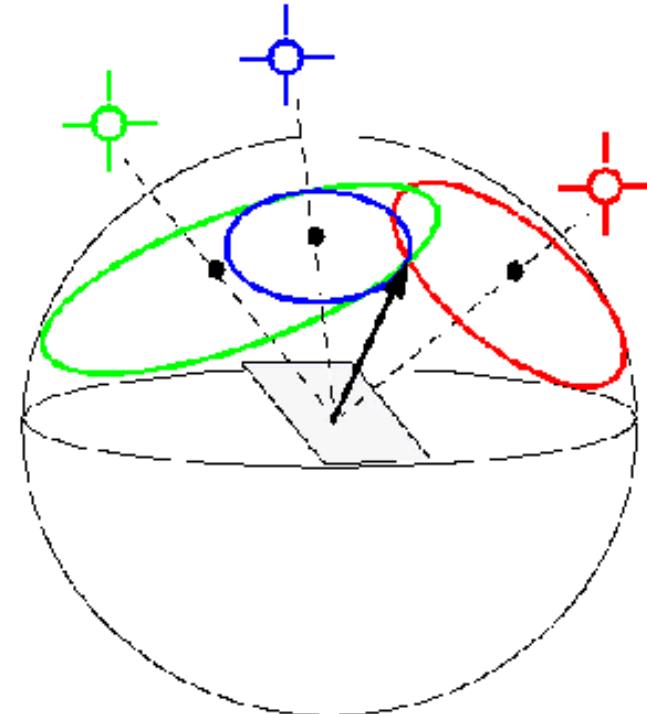
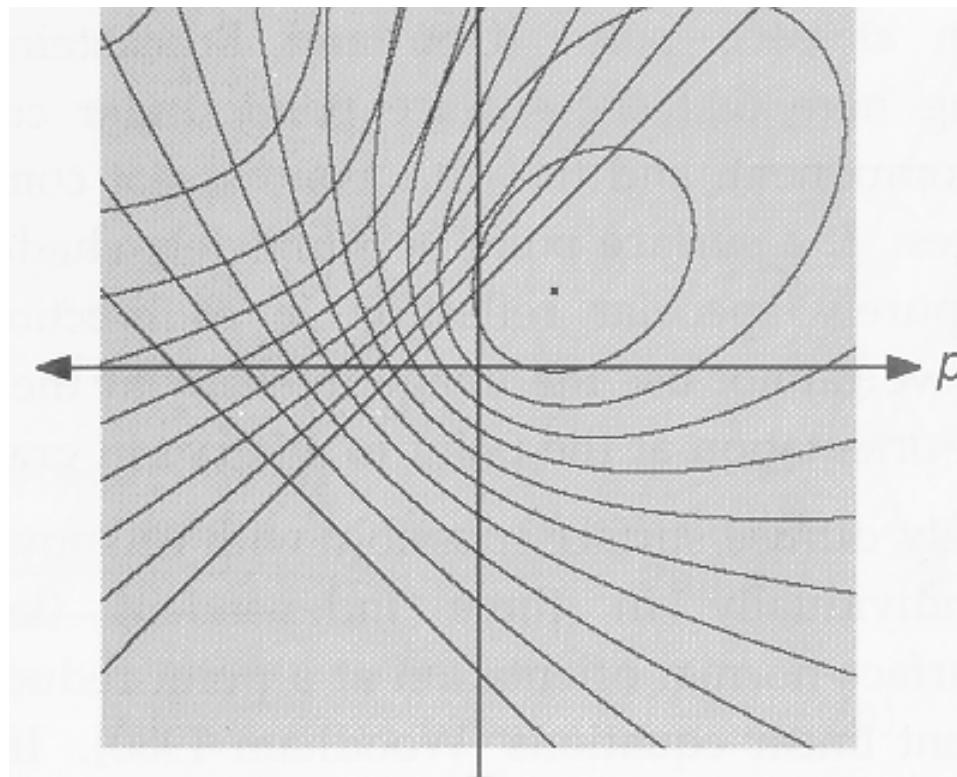
PHOTOMETRIC STEREO



Given multiple images of the same surface under different known lighting conditions, can we recover the surface shape?

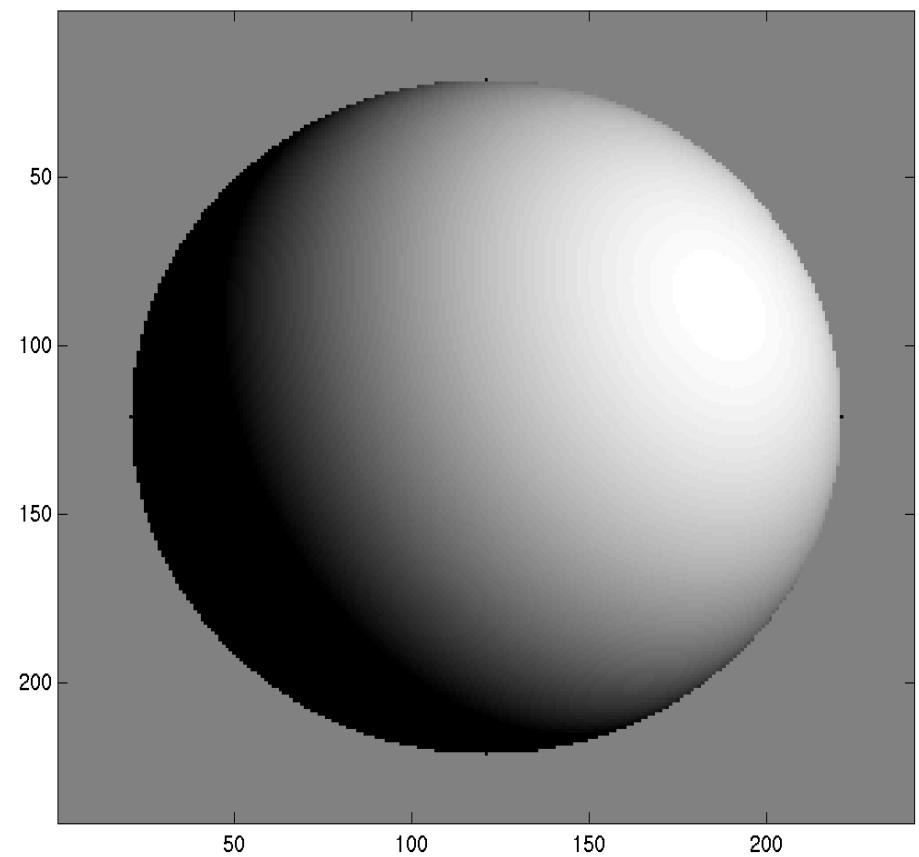
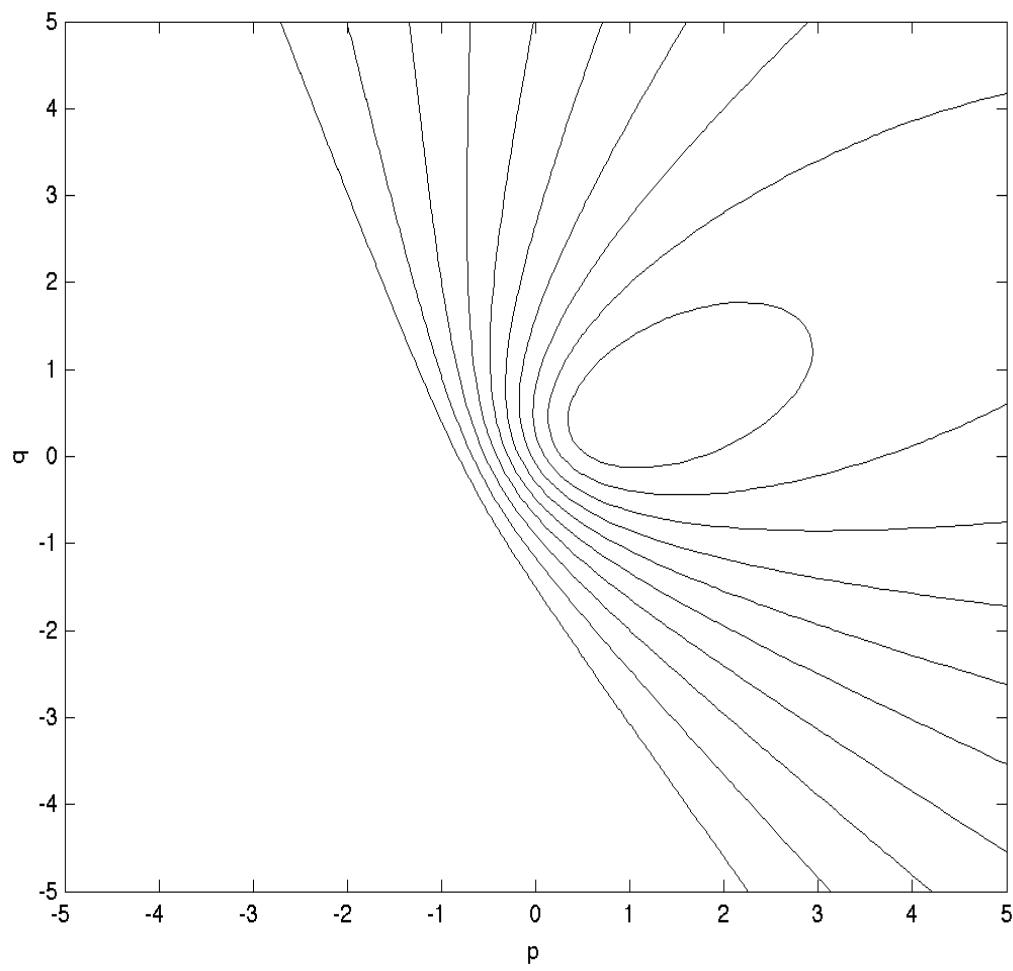
Yes! (Woodham, 1978)

PHOTOMETRIC STEREO

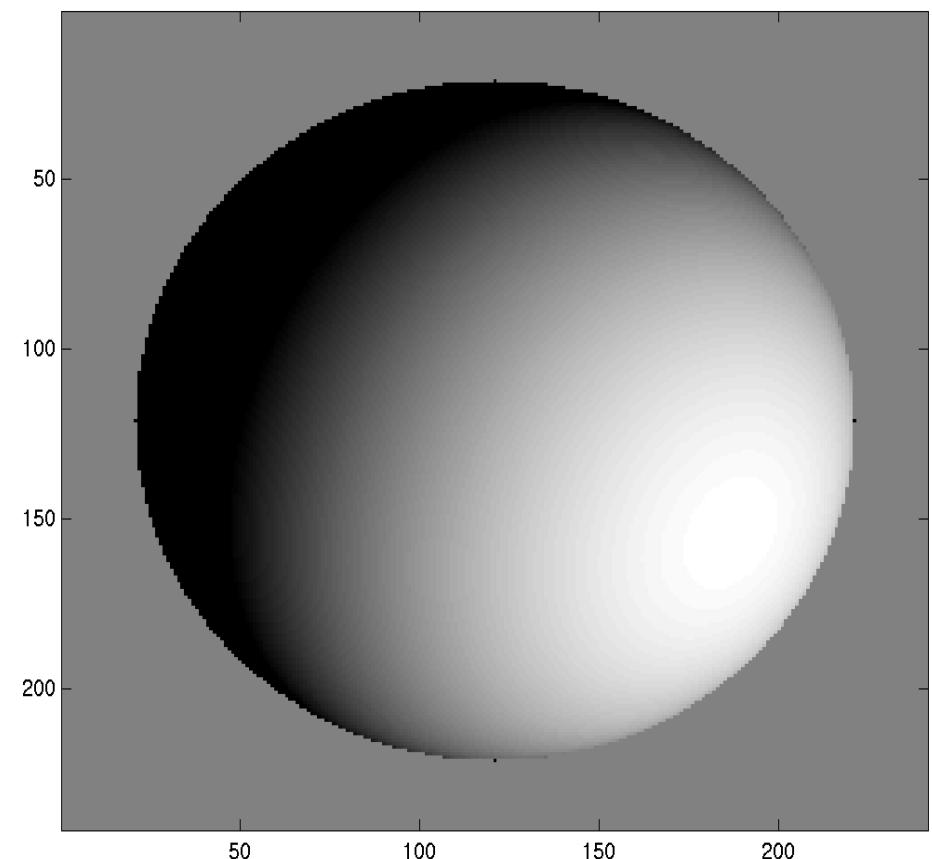
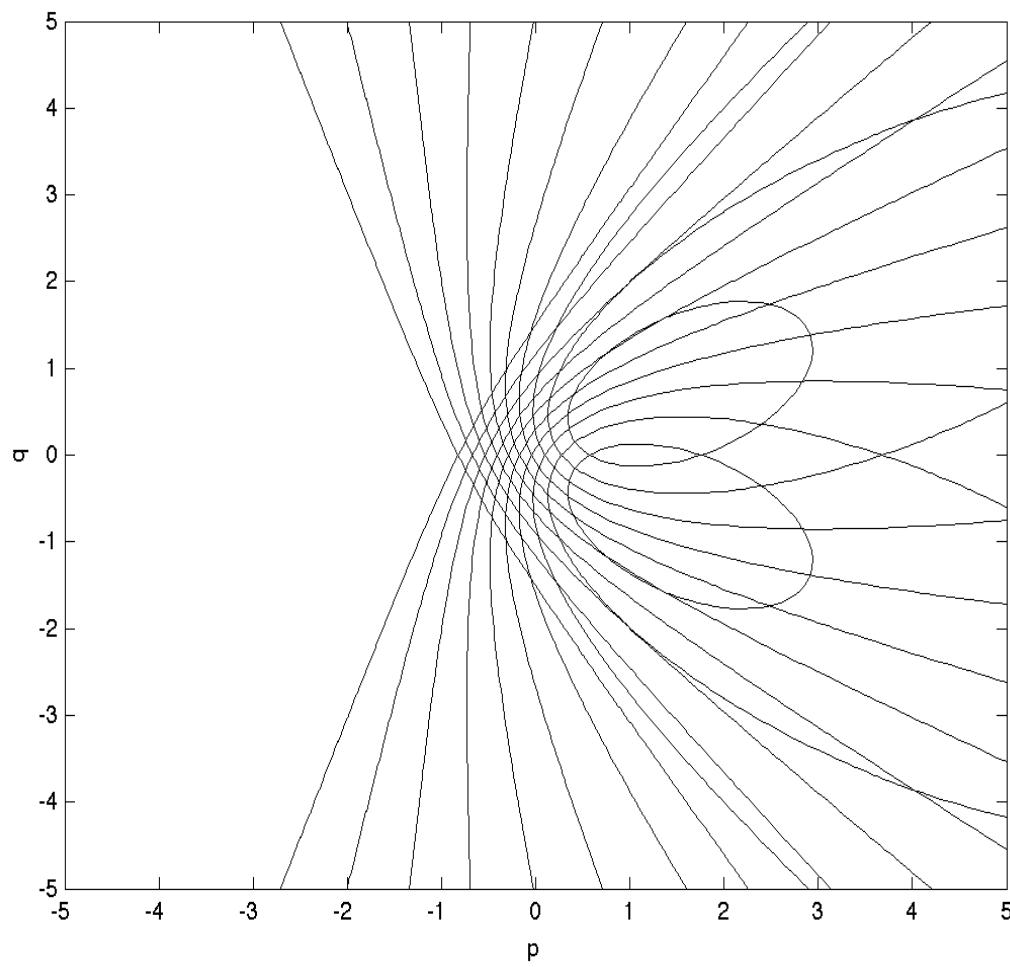


- Take several images under different lighting conditions.
- Infer the orientation from the changes in illumination.

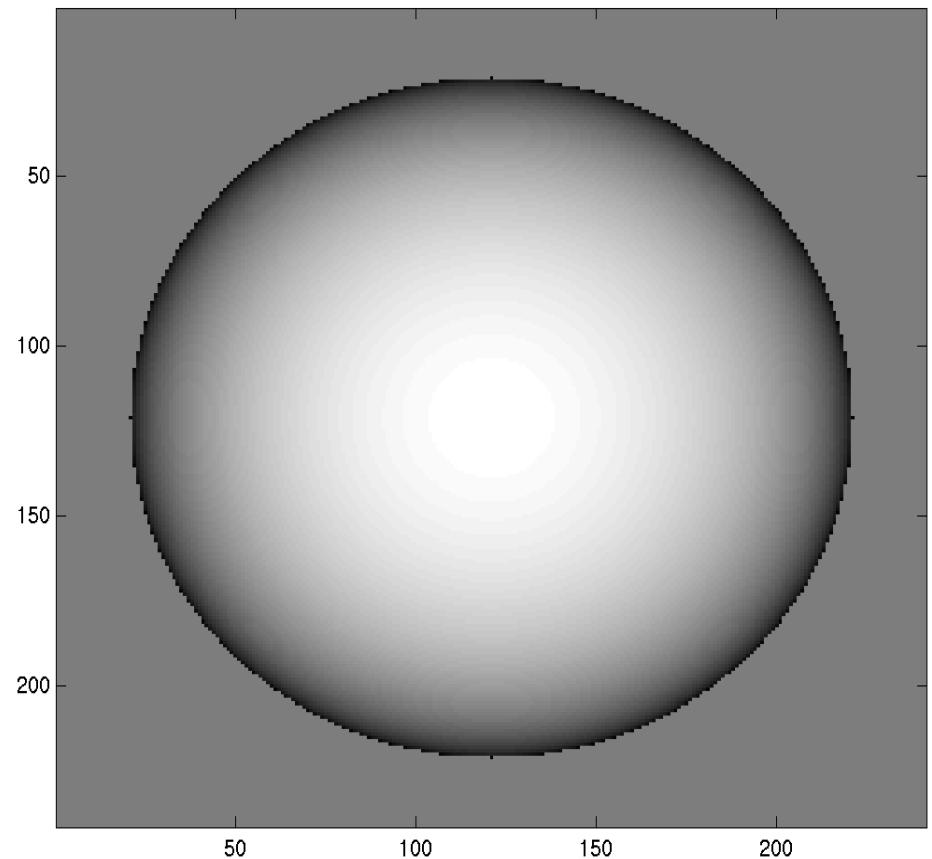
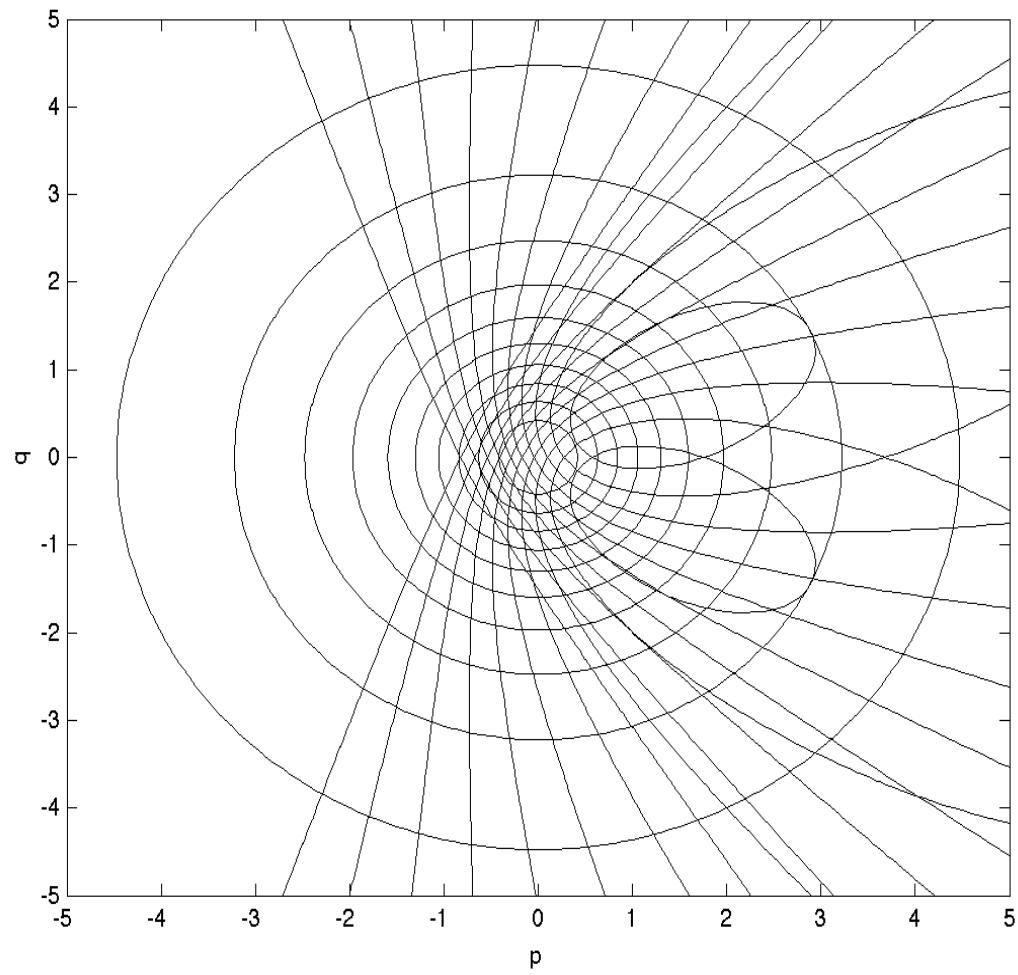
PHOTOMETRIC STEREO



PHOTOMETRIC STEREO



PHOTOMETRIC STEREO



ALGEBRAIC FORMULATION

Lambertian model: $I = \alpha(\mathbf{L} \cdot \mathbf{N}) = (\mathbf{L} \cdot \mathbf{M})$

Three light sources:

$$\begin{bmatrix} I_1 \\ I_2 \\ I_3 \end{bmatrix} = \begin{bmatrix} \mathbf{L}_1 \\ \mathbf{L}_2 \\ \mathbf{L}_3 \end{bmatrix} \mathbf{M}$$

$$\mathbf{N} = \frac{\mathbf{M}}{||\mathbf{M}||}$$

$$\alpha = ||\mathbf{M}||$$

ADDITIONAL LIGHTS

Over-constrained problem:

$$\mathbf{I} = \mathbf{LM}, \text{ with } \mathbf{I} = \begin{bmatrix} I_1 \\ \vdots \\ I_n \end{bmatrix} \text{ and } \mathbf{L} = \begin{bmatrix} \mathbf{L}_1 \\ \vdots \\ \mathbf{L}_n \end{bmatrix}$$

$$\Rightarrow \mathbf{LL^tM} = \mathbf{L^tI} \text{ (Least - squares solution)}$$

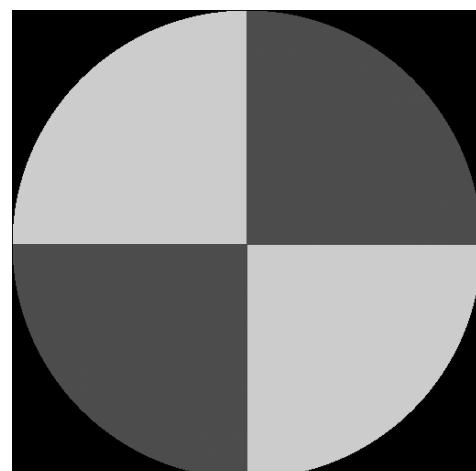
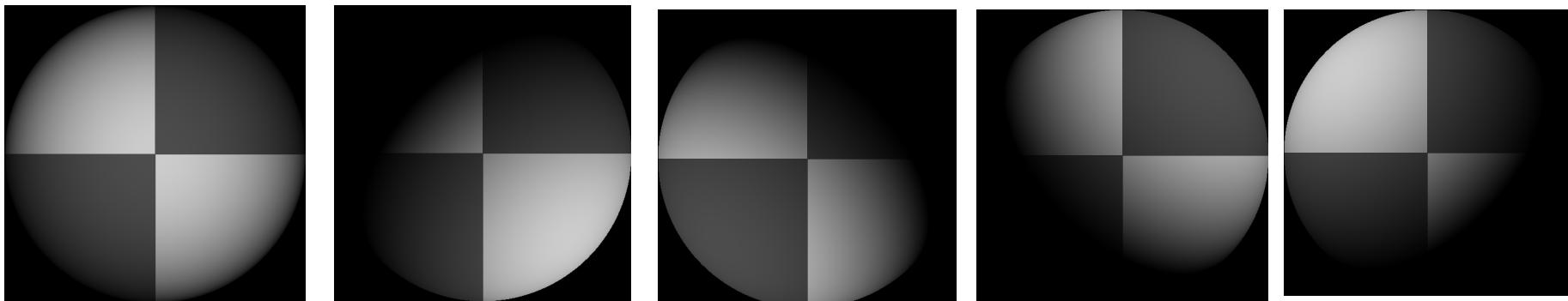
SHADOWS

- Shadowed pixels for a given light source position are outliers.
- Premultiplying by a thresholded weight matrix eliminates their contributions.

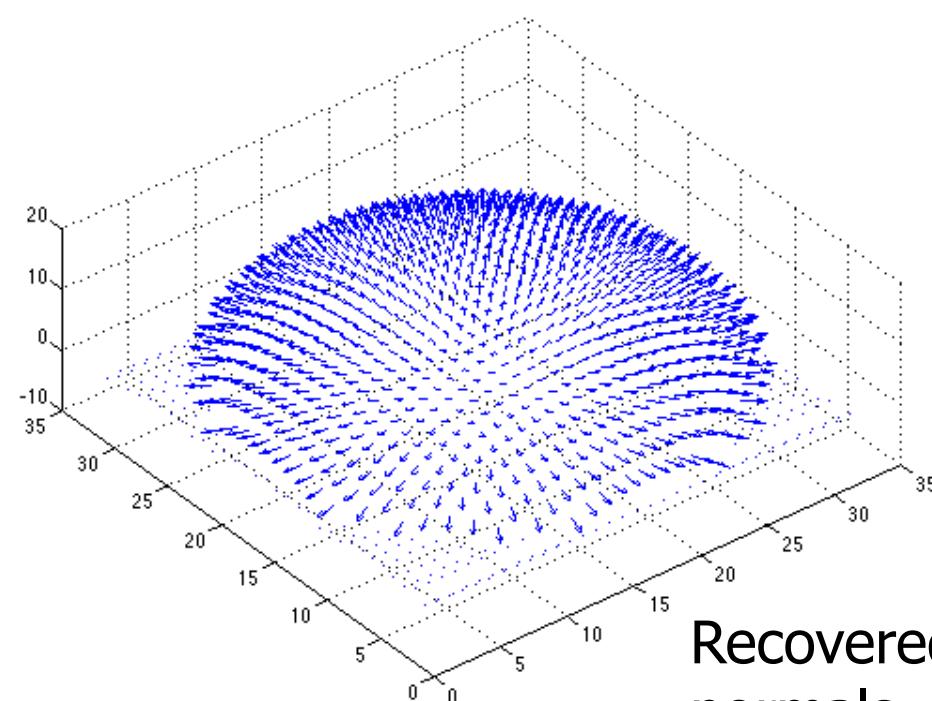
$$\begin{bmatrix} I_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & I_n \end{bmatrix} \mathbf{I} = \begin{bmatrix} I_1 & 0 & 0 \\ 0 & \ddots & 0 \\ 0 & 0 & I_n \end{bmatrix} \mathbf{LM}$$

SYNTHETIC SPHERE IMAGES

Five different lighting conditions:

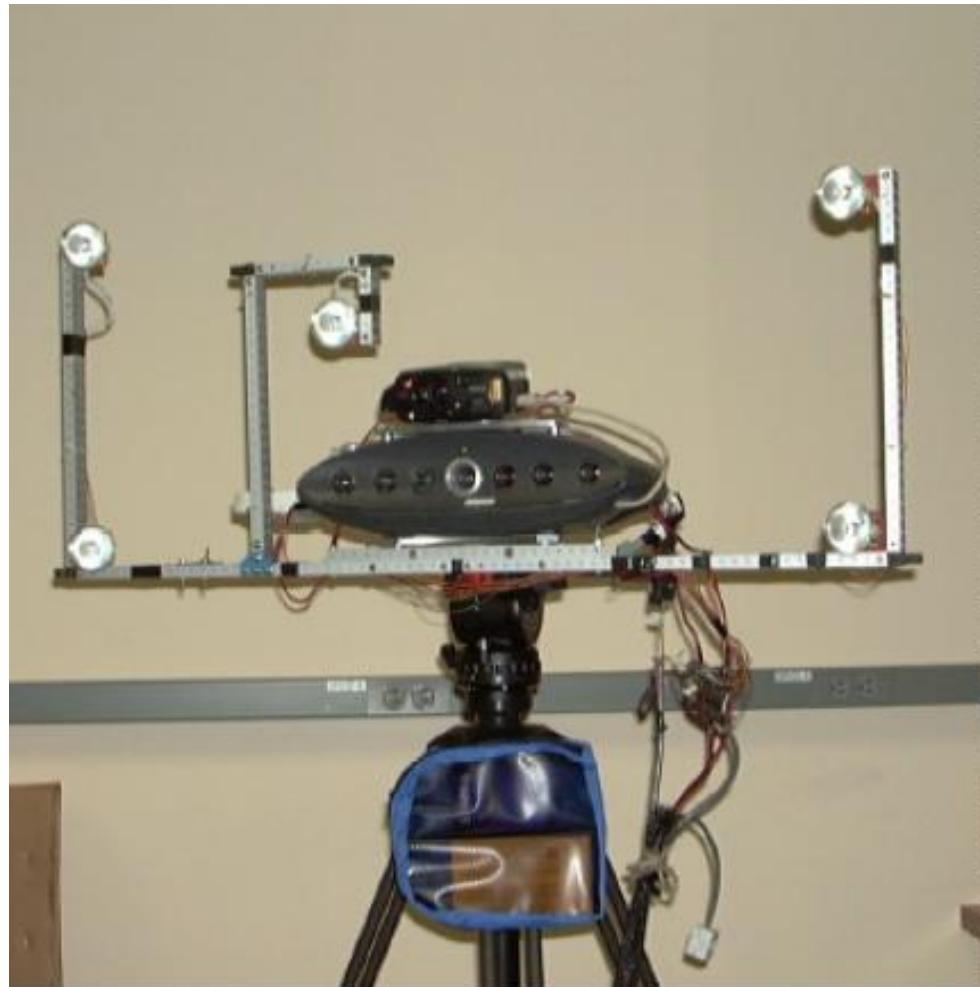


Recovered albedo



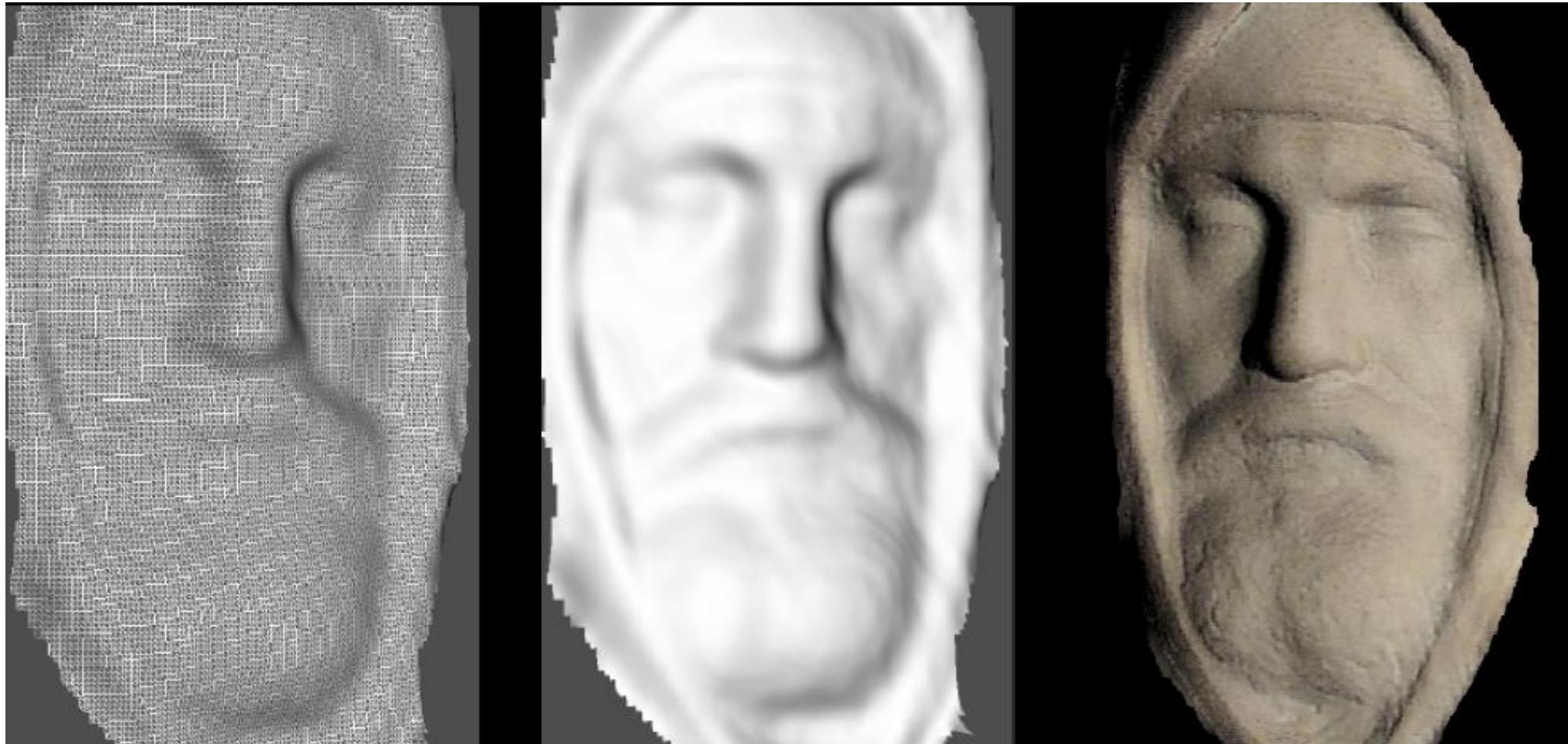
Recovered surface
normals

VIRTUOSO

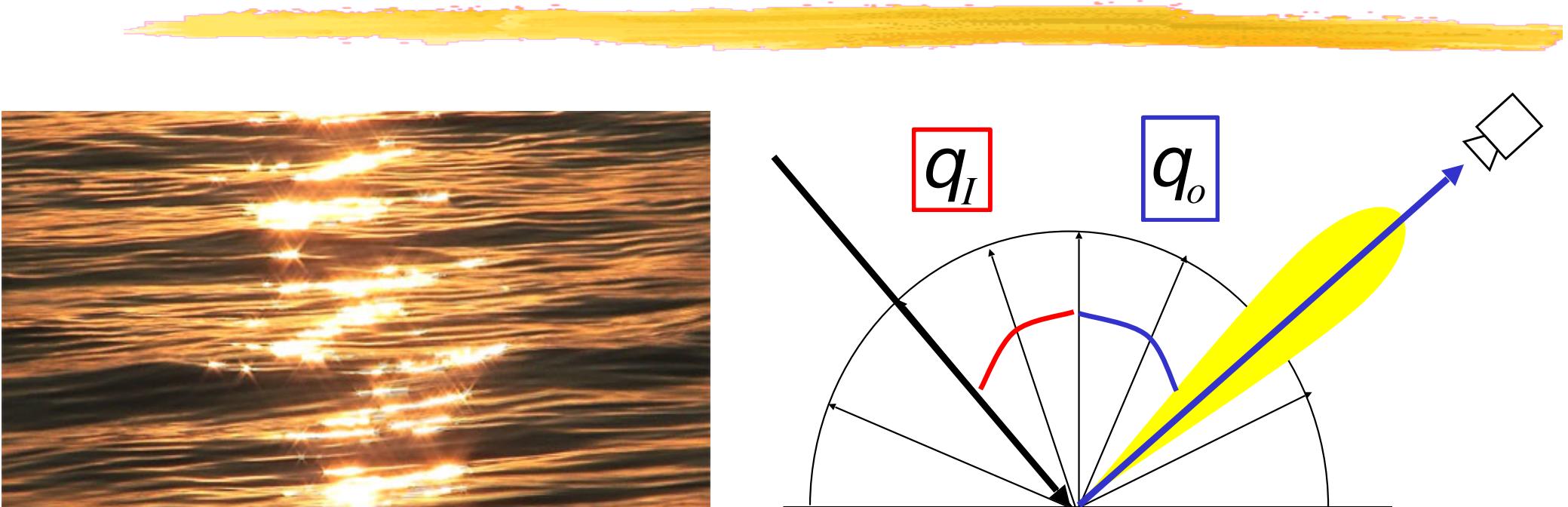


One camera and five light sources

DELIGHTED TEXTURE MAPS

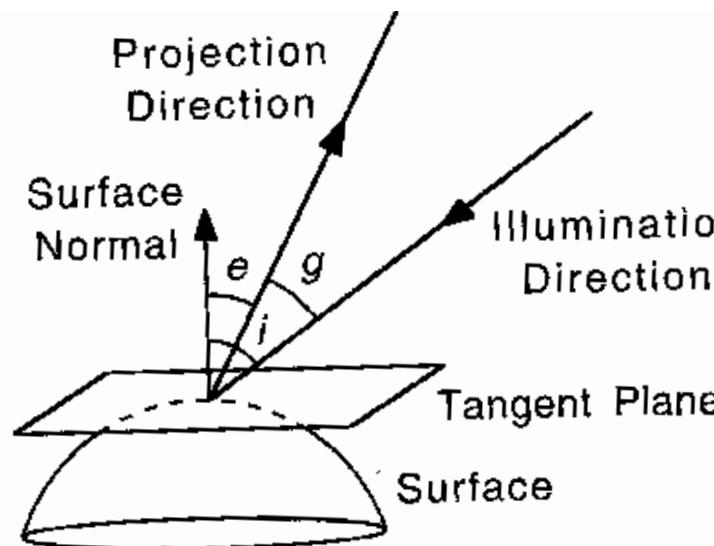


SPECULARITIES



- At specular points Lambertian assumptions are violated.
- However, they can be used to infer normal information.

ANGLES

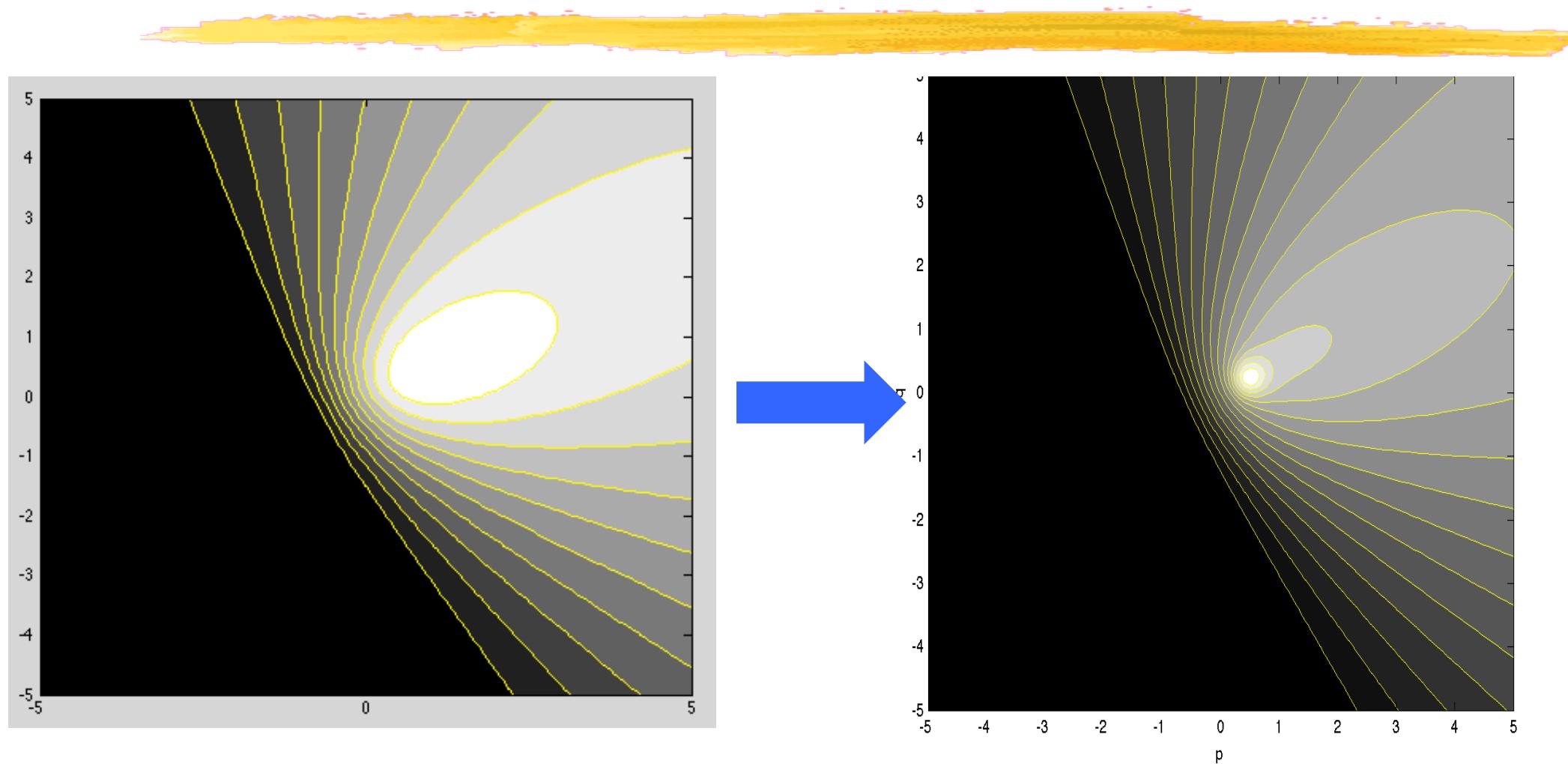


Angle of incidence (i): angle between the surface normal and the direction of the incident light ray.

Angle of emittance (e): angle between emitted light ray and surface normal.

Phase angle (g): angle between incident and emitted light ray.

LAMBERTIAN+SPECULAR REFLECTANCE MAP



Weighted average of the individual diffuse- and specular-component of glossy white paint.

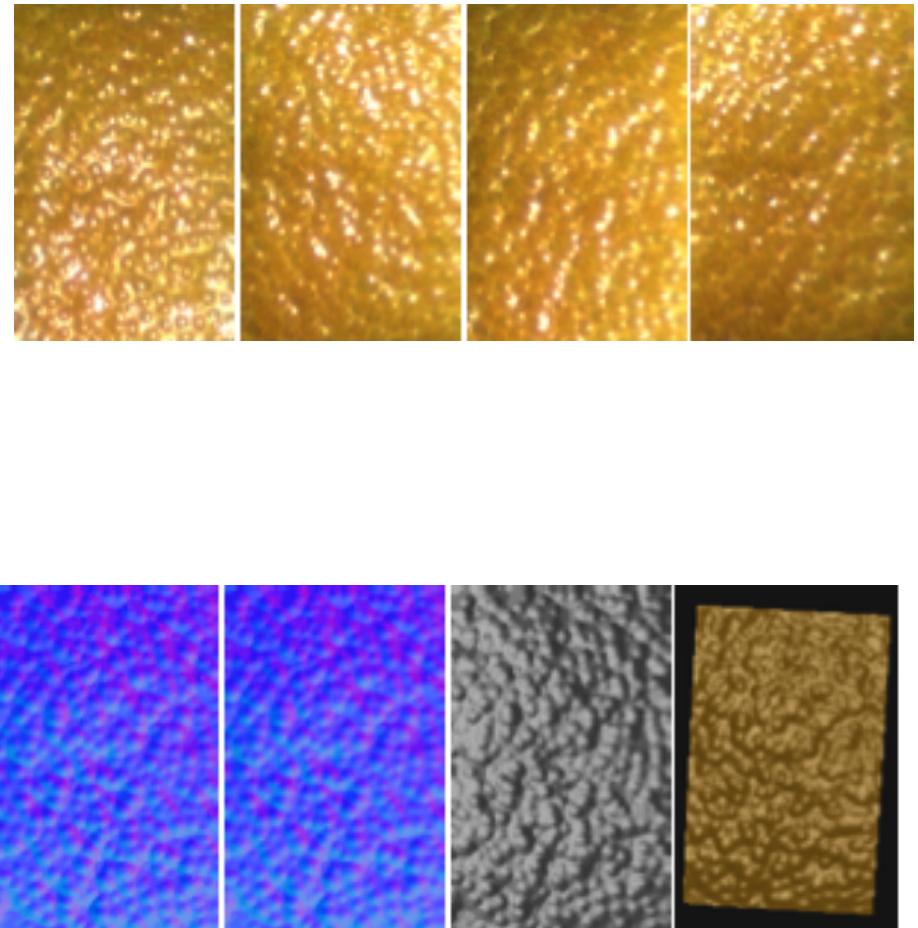
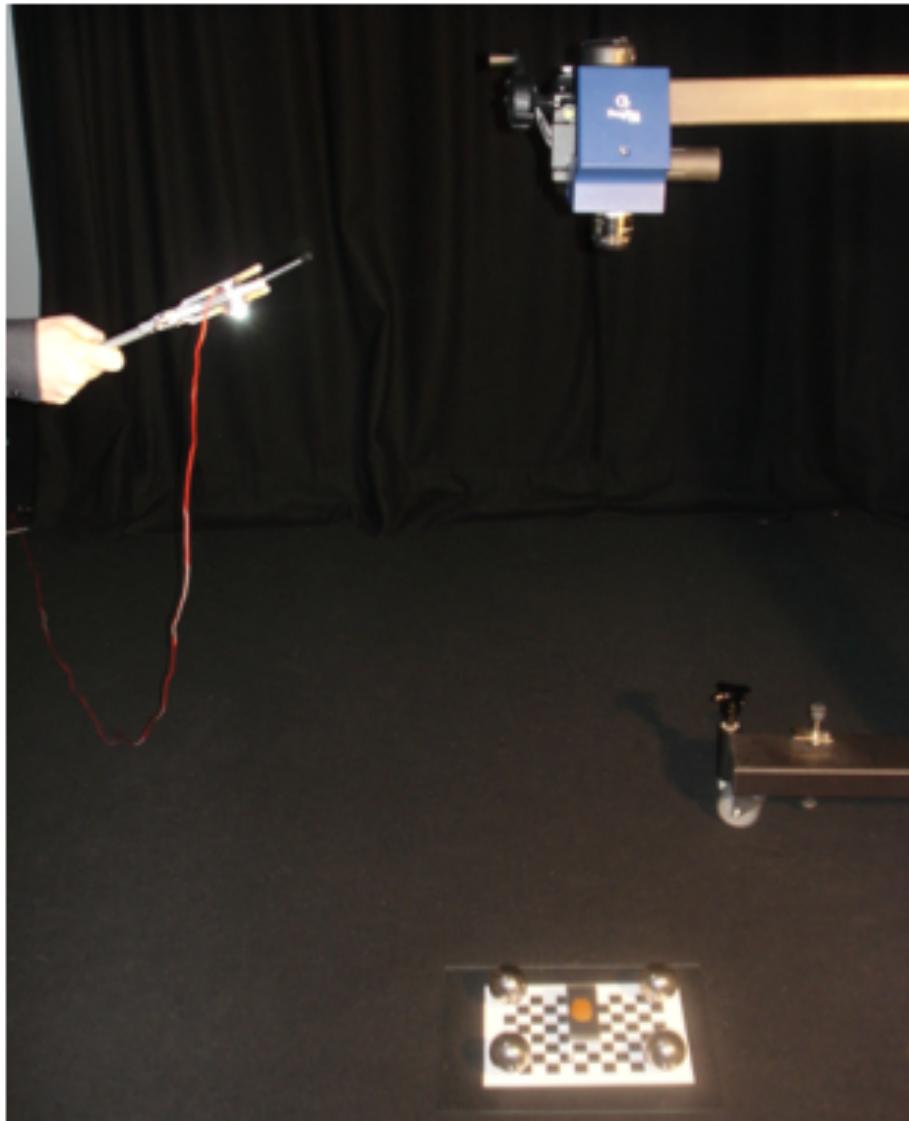
SPECULAR SURFACE



Perfect mirror: Reflects light only when $i = e$ and $i + e = g$.

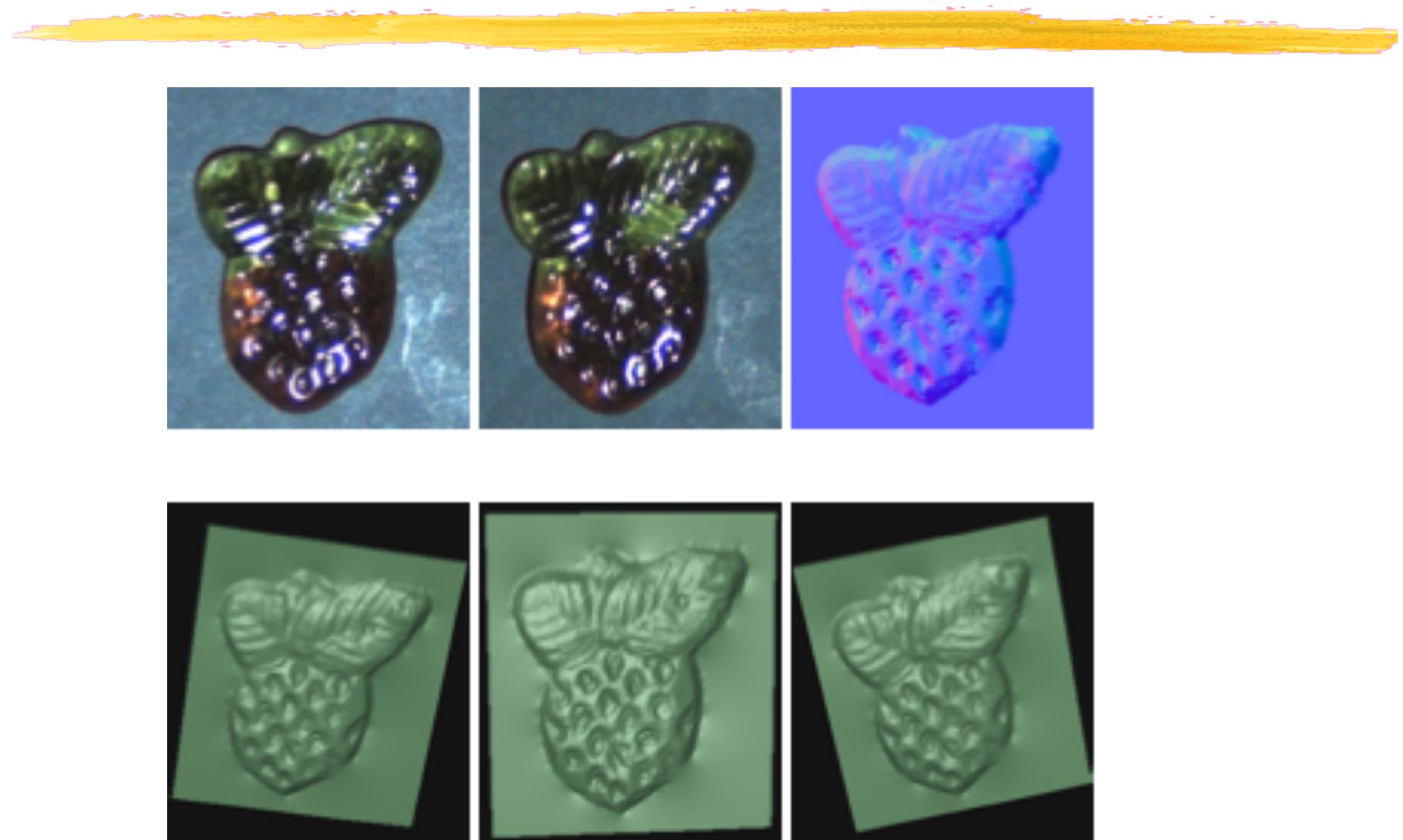
→ In practice, most surfaces are a mixture of specular and Lambertian.

SHAPE FROM SPECULARITIES

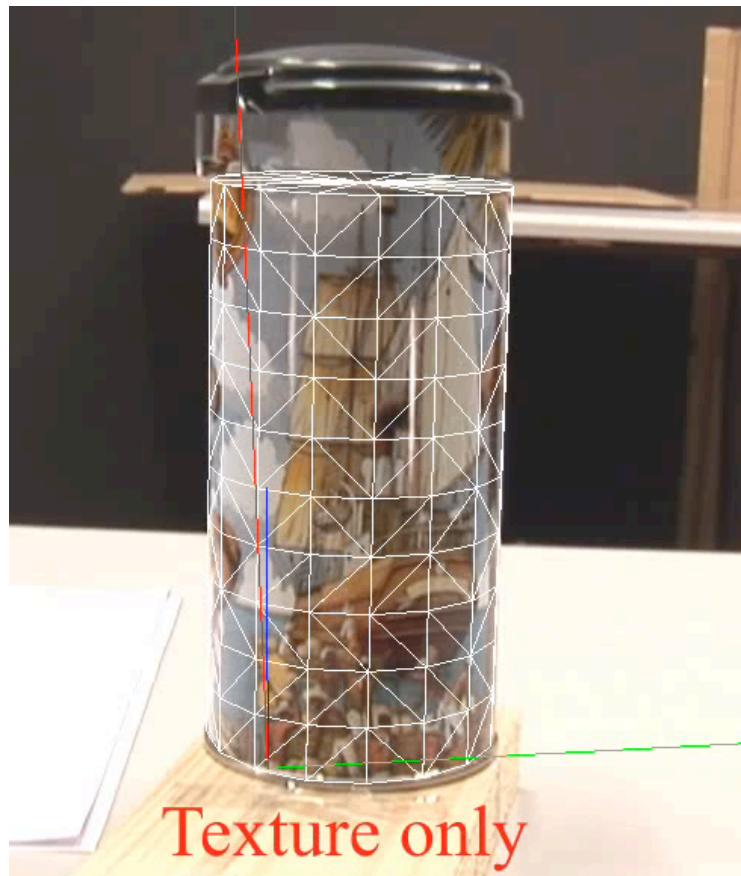


Chen et al., CVPR 2006

SHAPE FROM SPECULARITIES



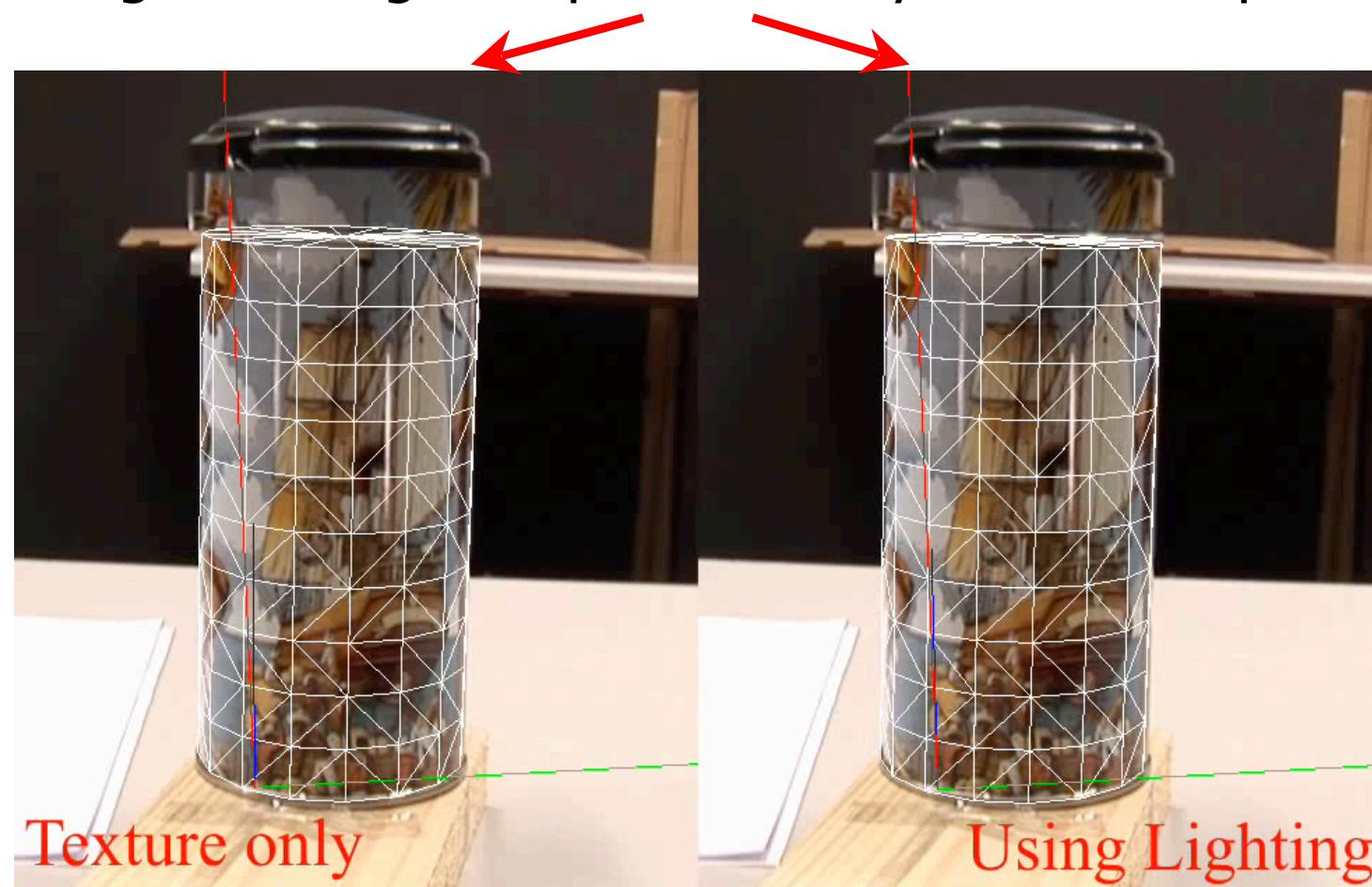
POSE FROM TEXTURE



- ❖ Texture tracking yields correct reprojection of frontal facets.
- ❖ But the jittering top shows that the tracking is nevertheless inaccurate.

POSE FROM SPECULARITIES

Taking advantage of specularities yields better poses



→ No more jittering top!

TEXTURE AND SPECULARITIES

Specularities are:

- Very sensitive to motion
- Affected differently than texture by the same motion

Specularities are not:

- Capable of constraining all the degrees of freedom

Therefore texture and specularities must be combined



IN SHORT



Traditional Shape-from-Shading requires making strong assumptions:

- Constant or piece-wise constant albedo
 - No inter-reflections
 - No shadows
 - No specularities
- In a single image context, it is most useful in conjunction with other information sources.

SHAPE FROM X



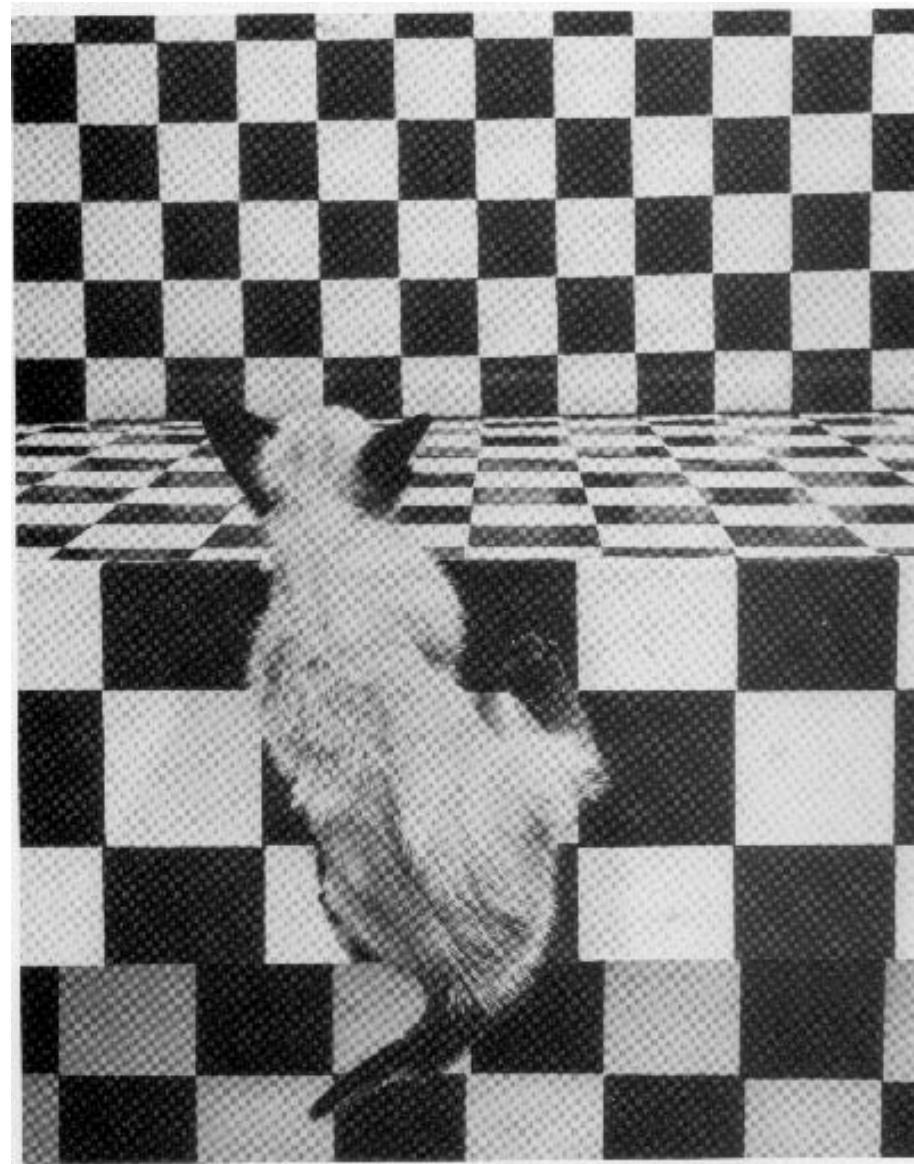
One image:

- Shading
- **Texture**

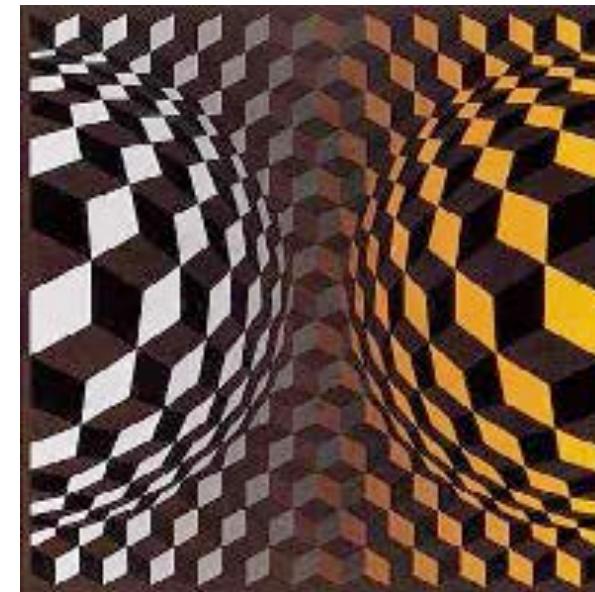
Two images or more:

- Stereo
- Contours
- Motion

SHAPE FROM TEXTURE



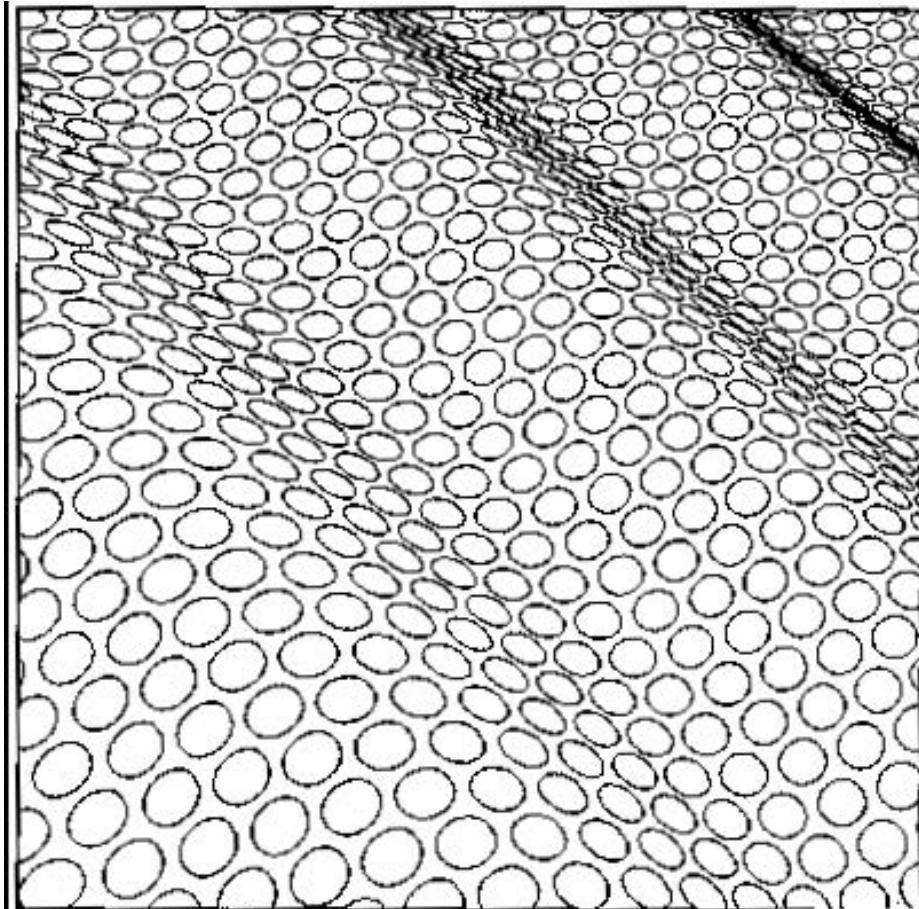
SHAPE FROM TEXTURE



Recover surface orientation or surface shape from image texture.

- Assume texture 'looks the same' at different points on the surface
- This means that the deformation of the texture is due to the surface curvature

STRUCTURAL SHAPE RECOVERY

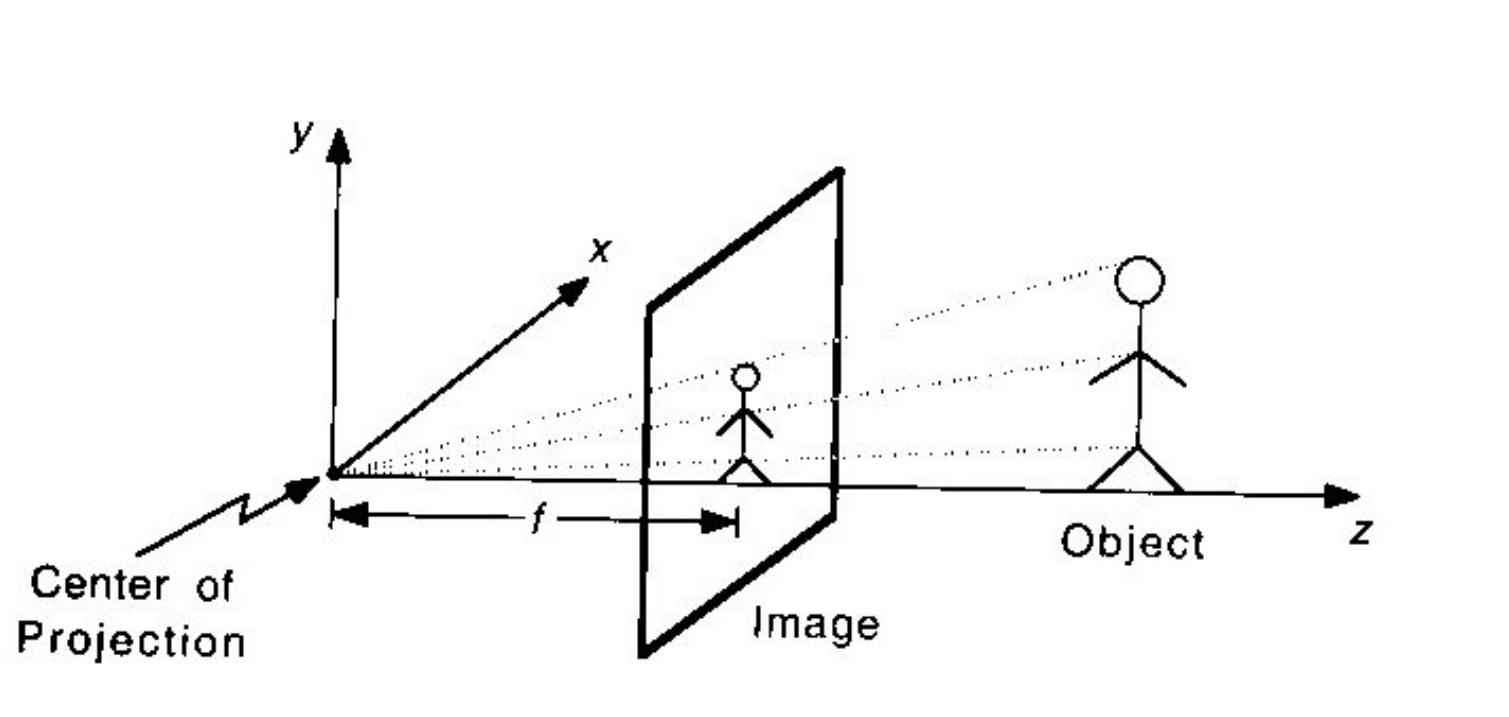


Basic hypothesis: Texture resides on the surface and has no thickness.

--> Computation under:

- Perspective projection
- Paraperspective projection
- Orthographic projection

PERSPECTIVE PROJECTION

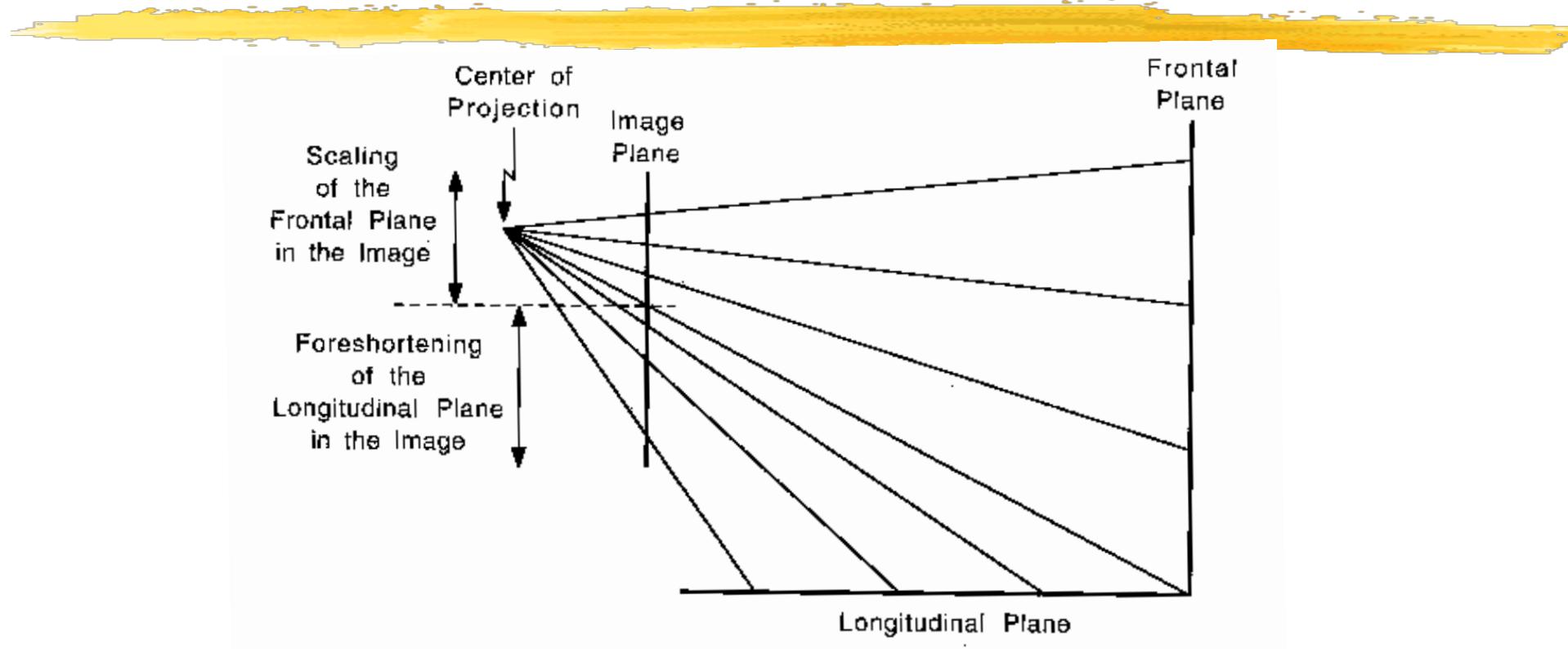


$$u = f \frac{x}{z}$$

$$v = f \frac{y}{z}$$

Pinhole geometry without image reversal

PERSPECTIVE DISTORTION



Perspective projection distortion of the texture

- depends on both depth and surface orientation,
- is anisotropic.

FORESHORTENING

Depth vs Orientation:

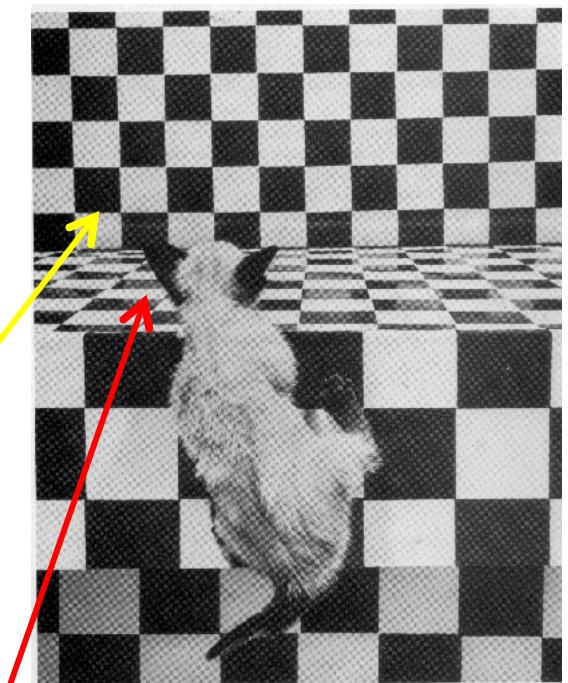
Infinitesimal vector $[\Delta x, \Delta y, \Delta z]$ at location $[x, y, z]$. The image of this vector is

$$\frac{f}{z} [\Delta x - \frac{x}{z} \Delta z, \Delta y - \frac{y}{z} \Delta z]$$

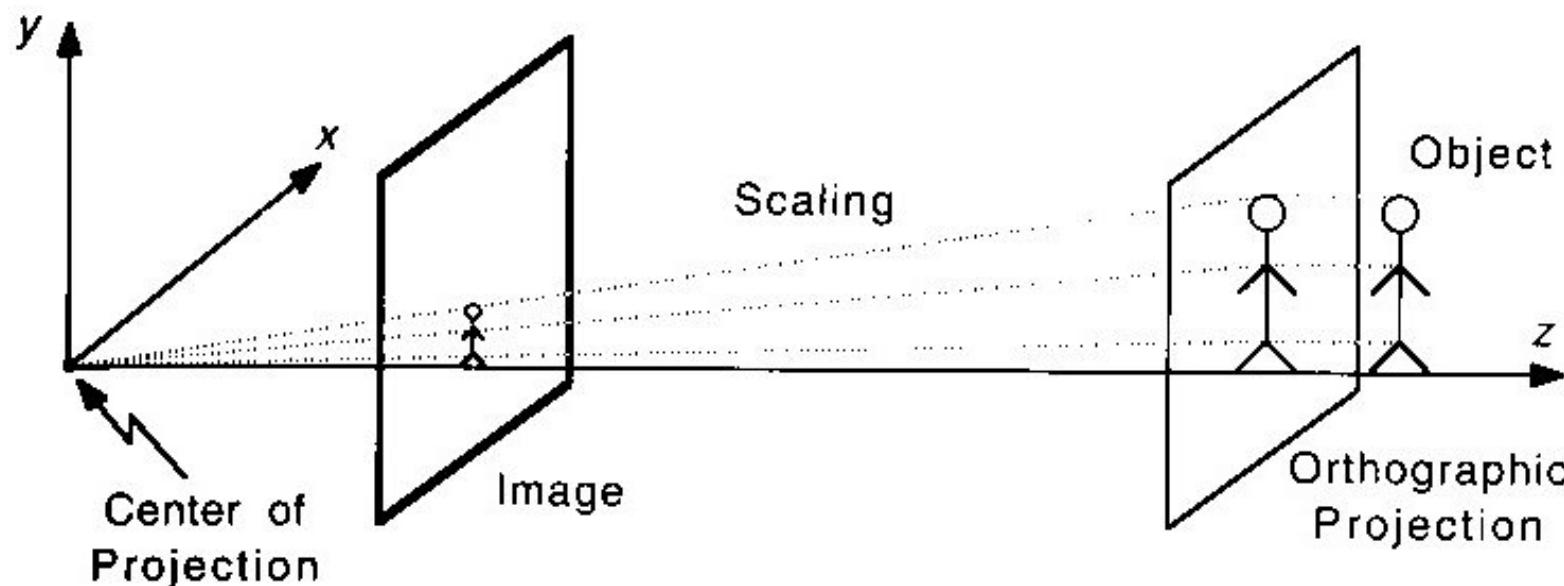
Two special cases:

$\Delta z = 0$: The object is scaled

$\Delta x = \Delta y = 0$: The object is foreshortened



ORTHOGRAPHIC PROJECTION



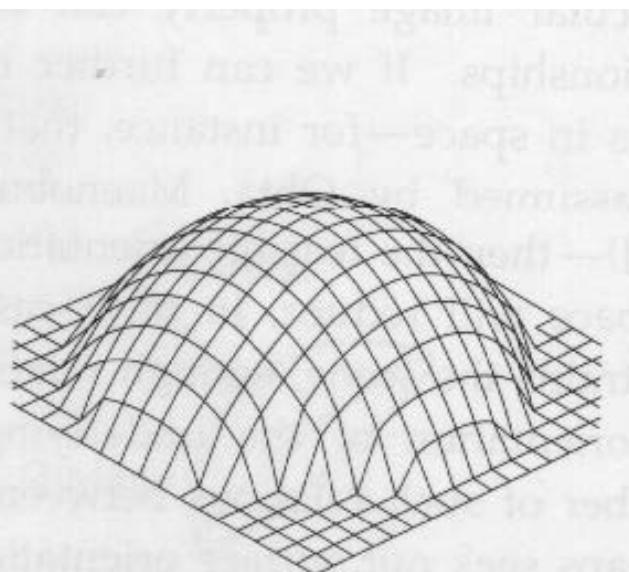
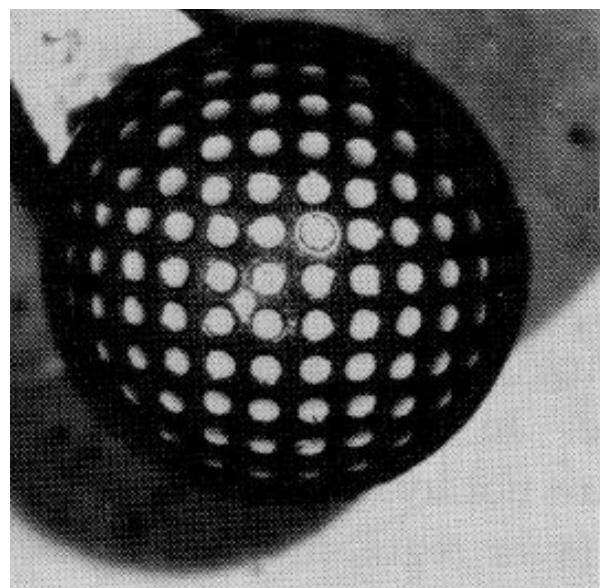
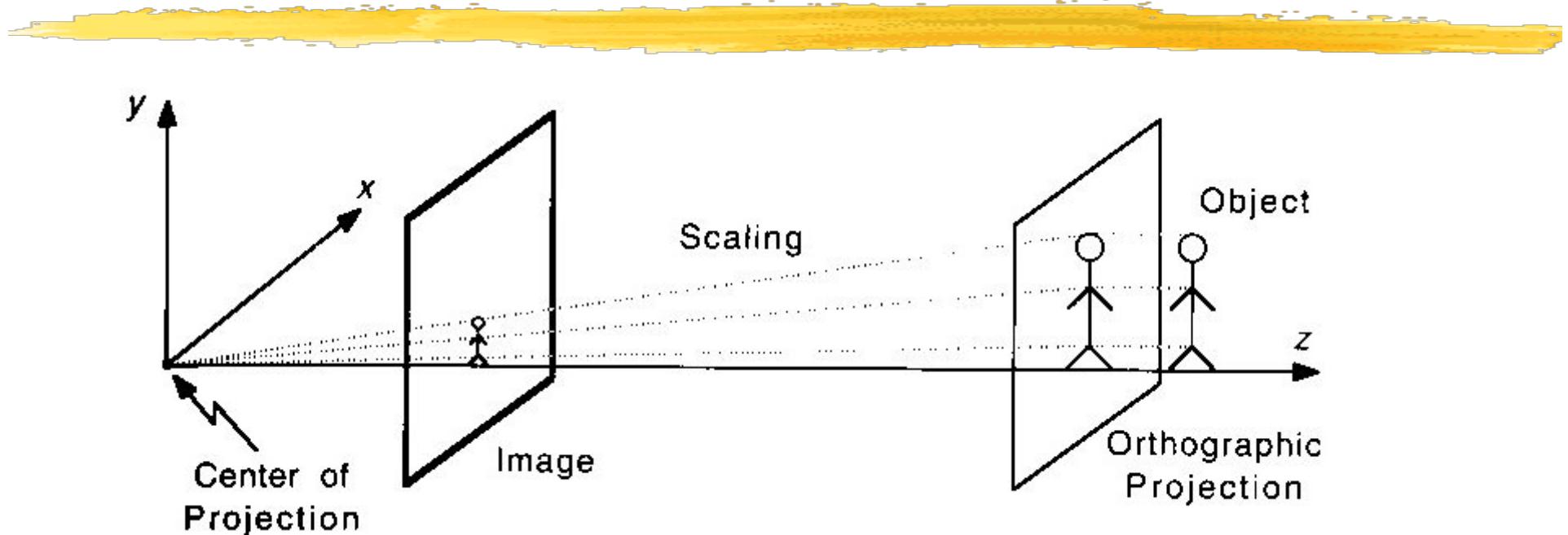
$$u = sx$$

$$v = sy$$

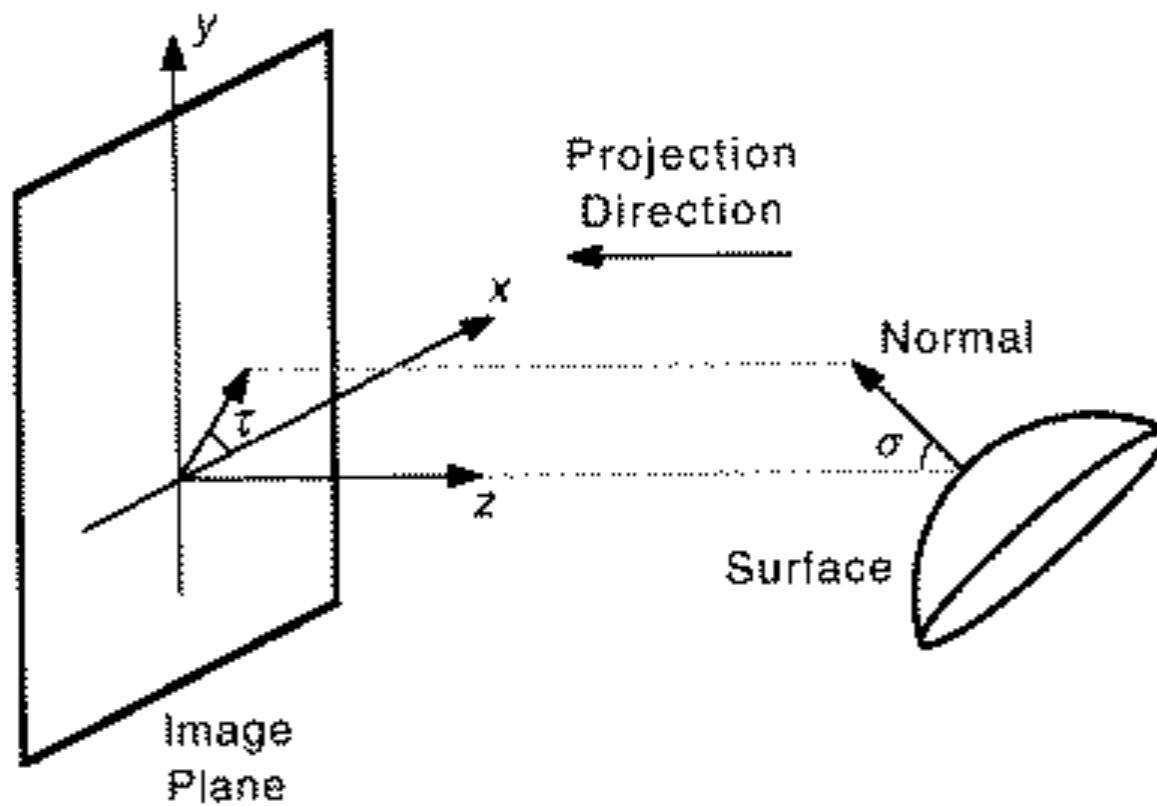
Special case of perspective projection:

- Large f
 - Objects close to the optical axis
- Parallel lines mapped into parallel lines.

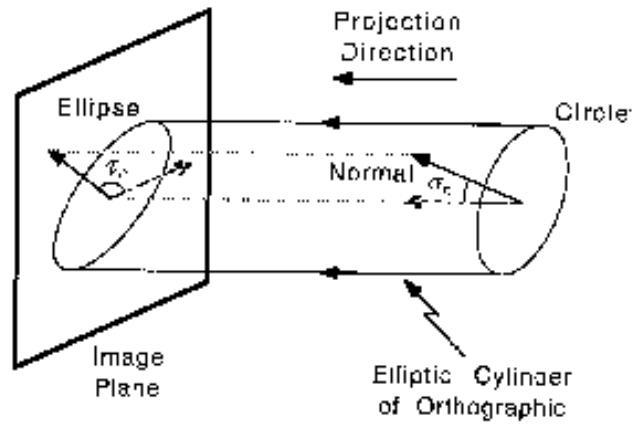
ORTHOGRAPHIC PROJECTION



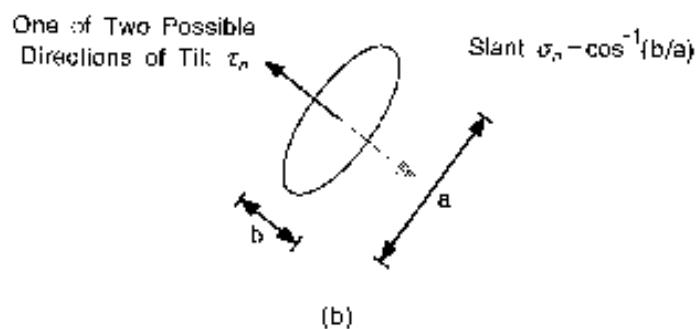
TILT AND SLANT



ORTHOGRAPHIC PROJECTION



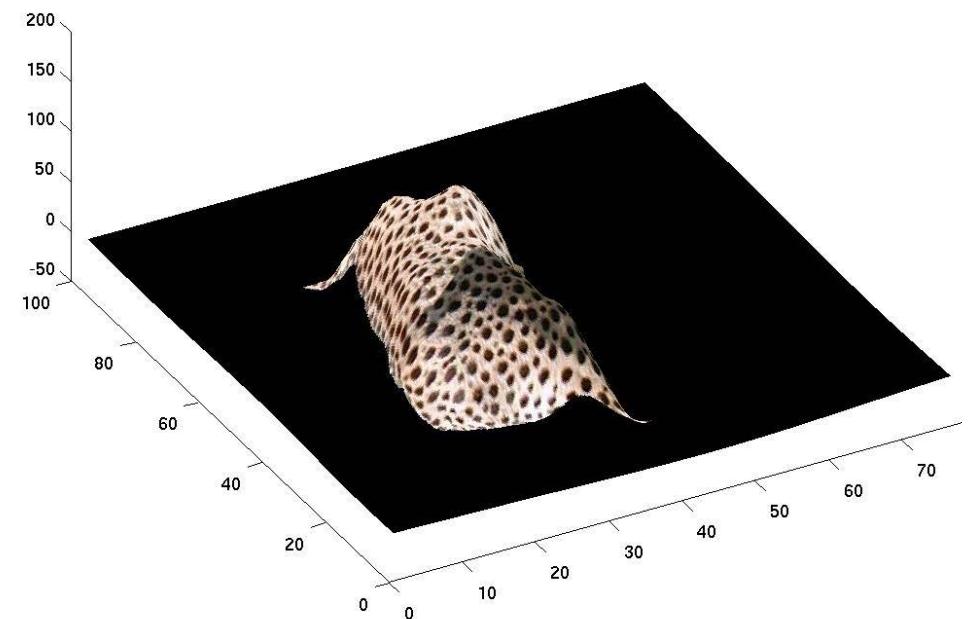
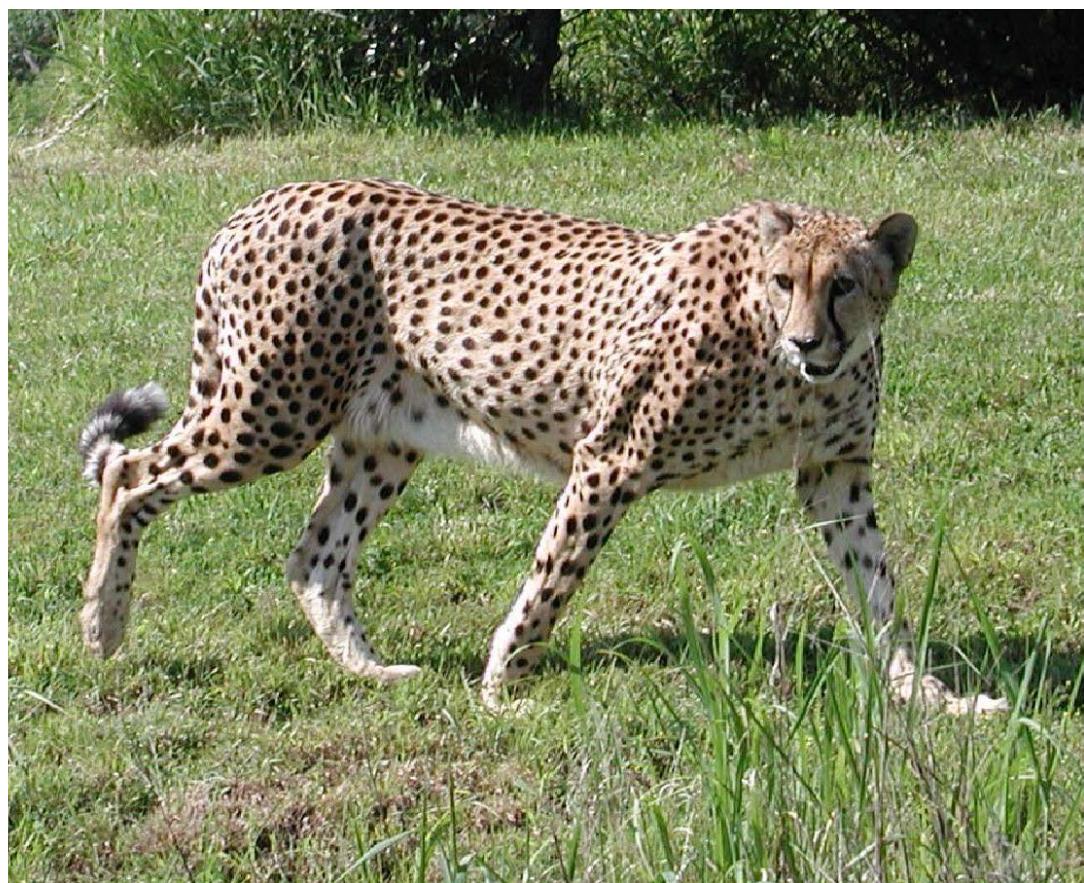
(a)



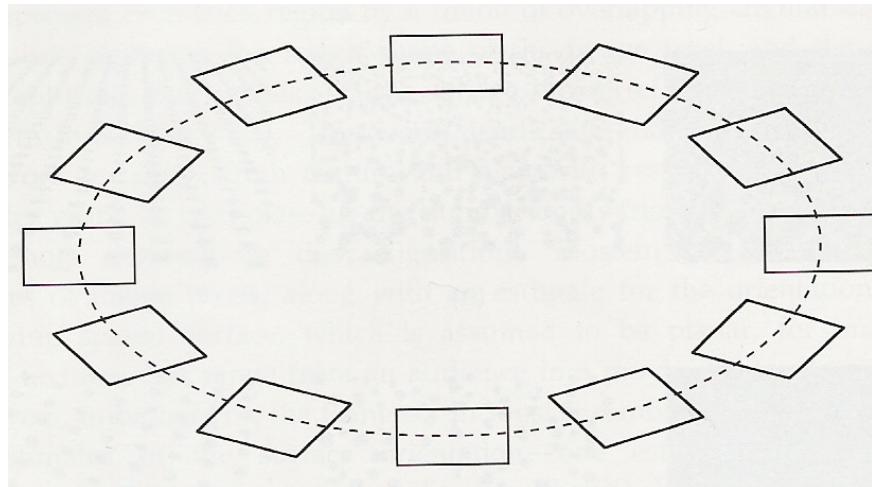
Tilt: Derived from the image direction in which the surface element undergoes maximum compression.

Slant: Derived from the extent of this compression.

CHEETAH



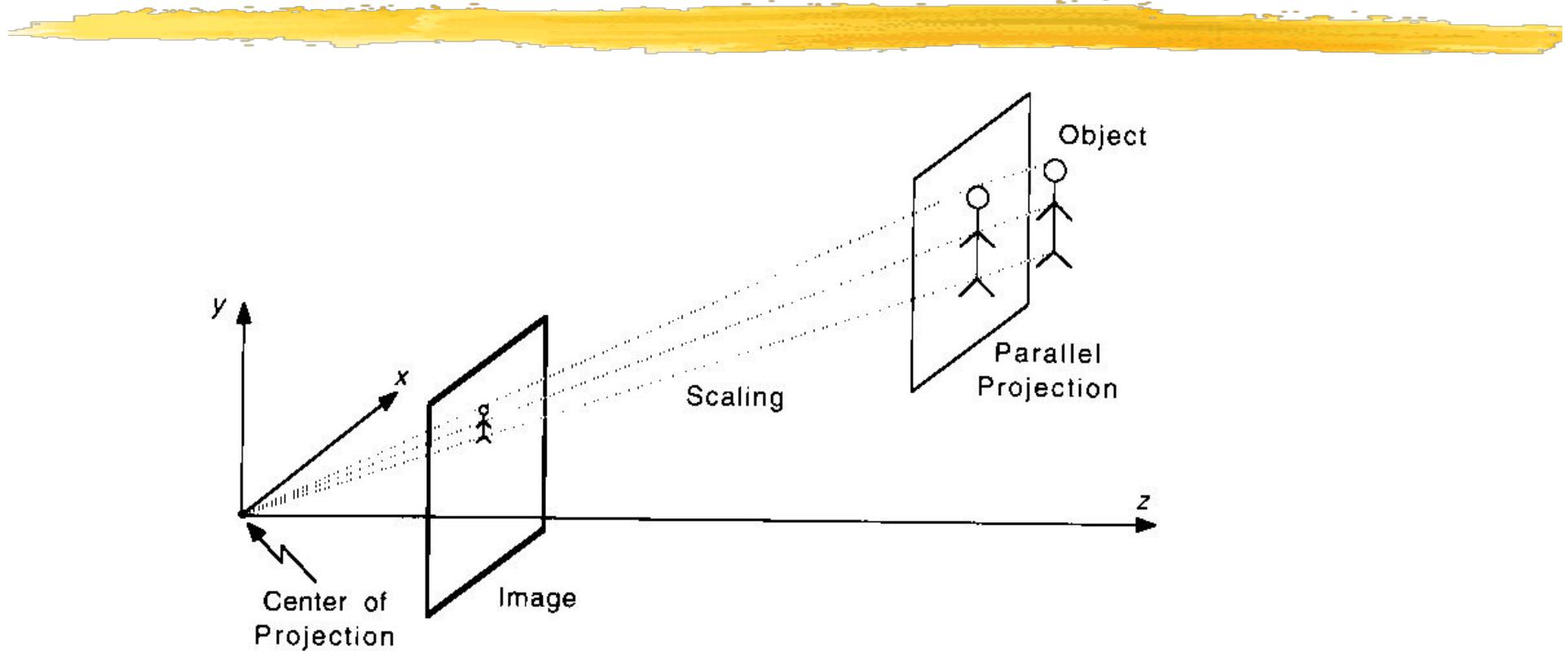
PERPENDICULAR LINES



Orthographic projections of squares that are rotated with respect to each other in a plane inclined at $\omega=60^\circ$ to the image plane.

$$\frac{\|(\mathbf{p}_1/l_1) \times (\mathbf{p}_2/l_2)\|}{\|\mathbf{p}_1/l_1\|^2 + \|\mathbf{p}_2/l_2\|^2} = \frac{\cos(\omega)}{1 + \cos^2(\omega)}$$

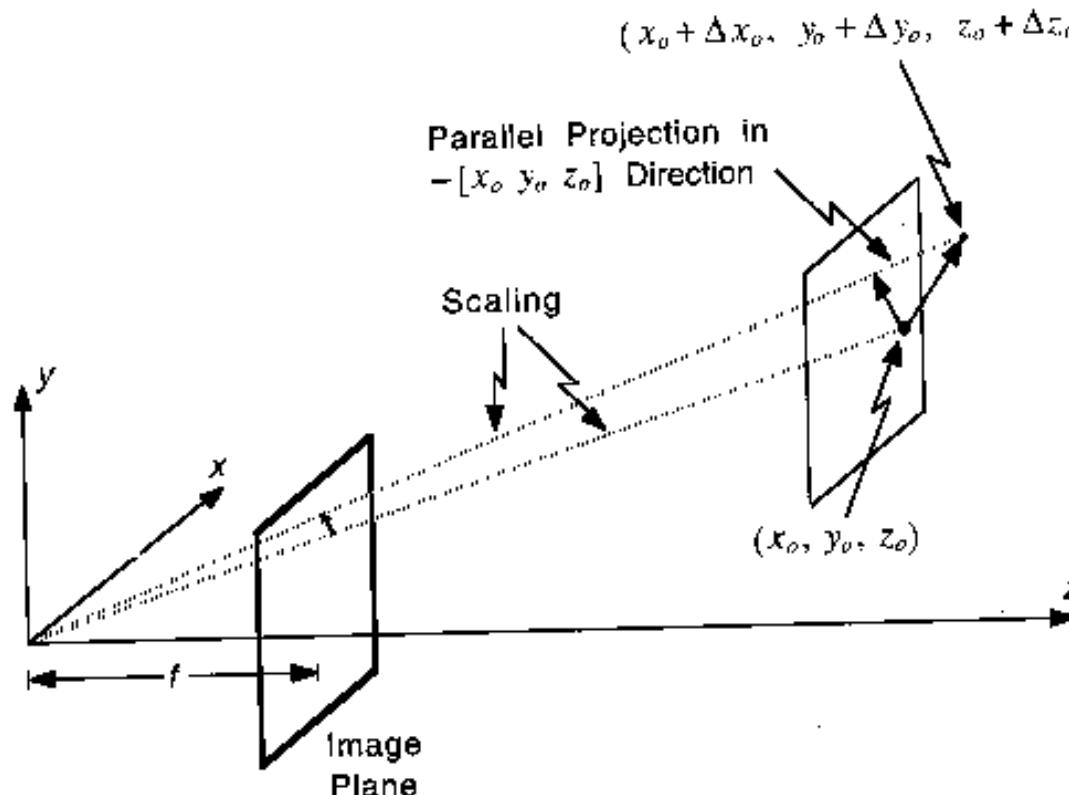
PARAPERSPECTIVE PROJECTION



Generalization of the orthographic projection:

- Object dimensions small wrt distance to the center of projection.
→ Parallel projection followed by scaling

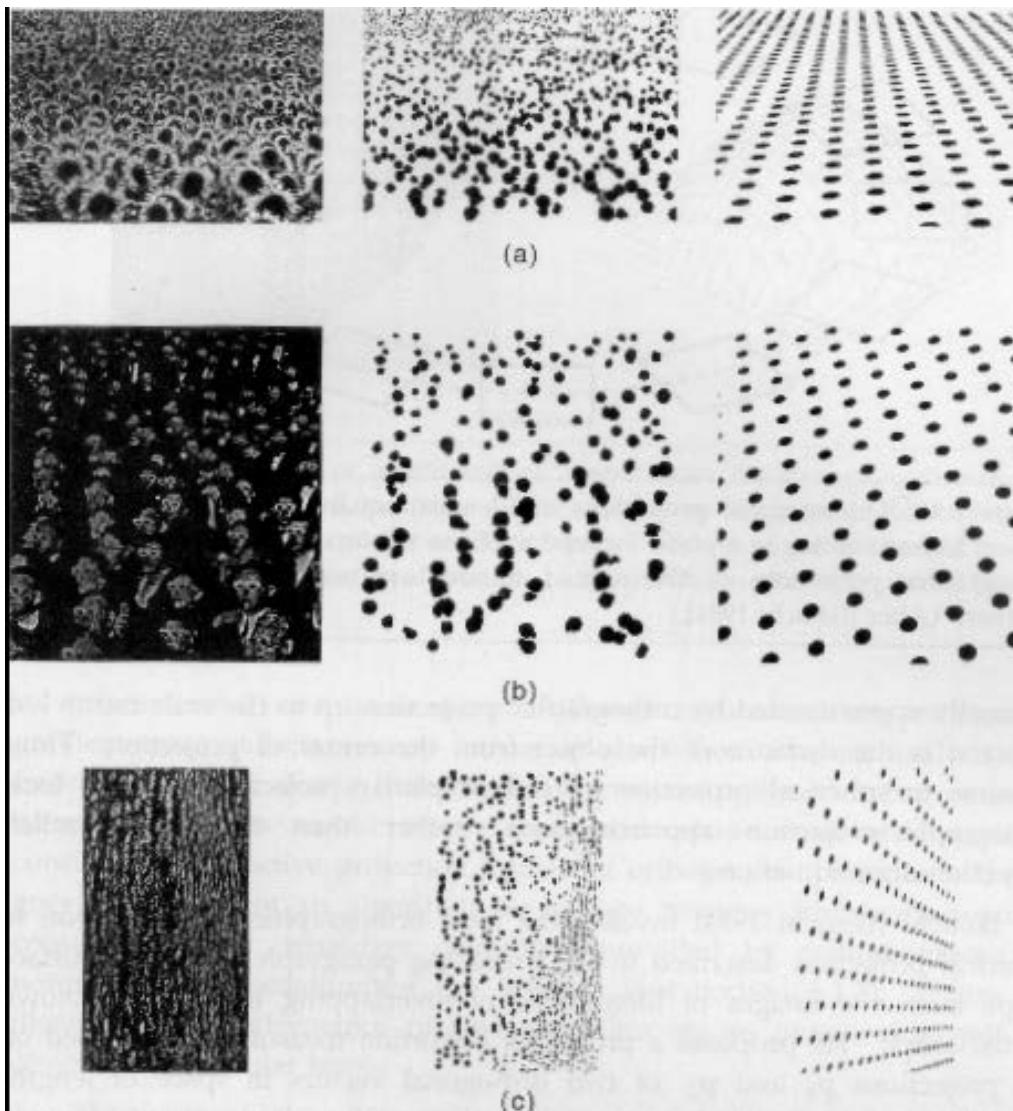
PARAPERSPECTIVE PROJECTION



For planar texels:

$$A' = -\frac{f^2}{z_0^3} \mathbf{n} \cdot [x_o \ y_o \ z_o] A$$

PARAPERSPECTIVE PROJECTION



Texels:

- Image regions that are brighter or darker than their surroundings.
 - Assumed to have the same area in space.
- Given enough texels, it becomes possible to estimate the normal.

TEXTURE GRADIENT



STATISTICAL SHAPE RECOVERY



Measure texture density as opposed to texel area, that is, the number of textural primitives per unit surface.

Assuming the texture to be homogeneous, we have: $\psi \mathbf{n} \propto \mathbf{b}$

$$\begin{aligned}\psi &= \begin{bmatrix} u_1 & v_1 & 1 \\ \dots & \dots & \dots \\ u_n & v_n & 1 \end{bmatrix}^t \\ \mathbf{b} &= [b_1, \dots, b_n]^t \quad \text{Image coordinates.} \\ \Rightarrow \mathbf{n} &= \frac{\psi \mathbf{n}}{\|\psi \mathbf{n}\|} \quad \text{Function of density.}\end{aligned}$$

STRENGTHS AND LIMITATIONS



Strengths:

- Emulates an important human ability.

Limitations:

- Requires regular texture.
- Involves very strong assumptions.
- Deep learning might weaken them.

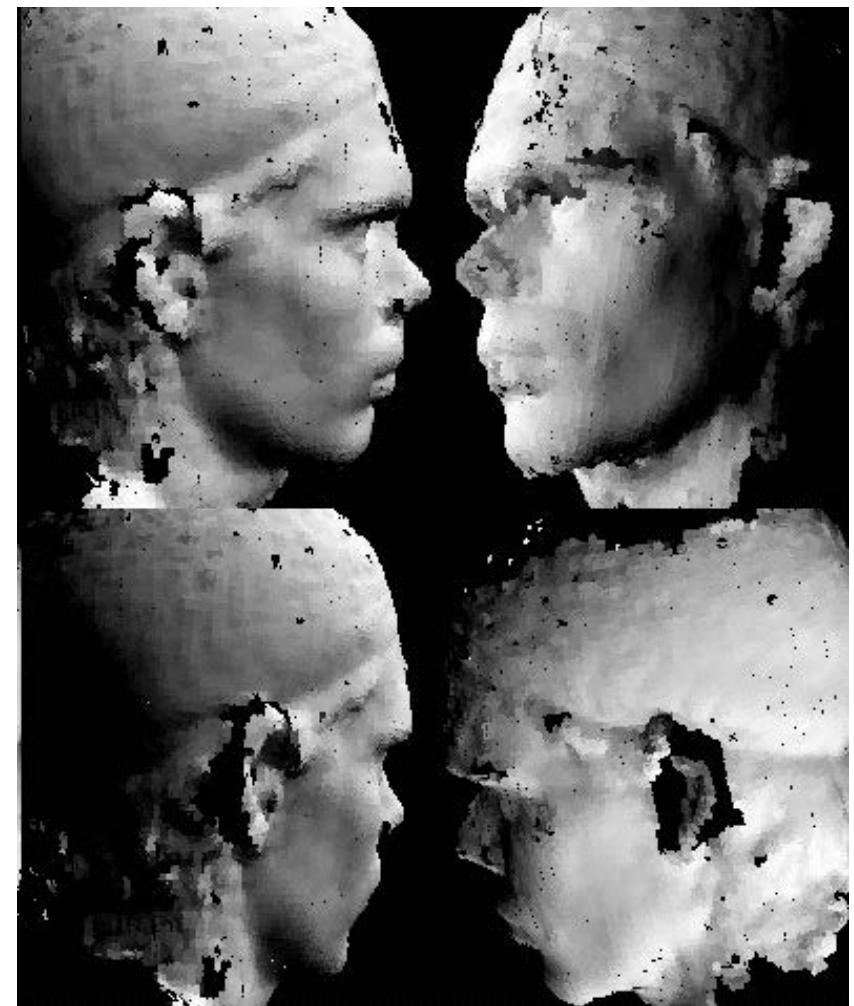
SHAPE FROM X

One image:

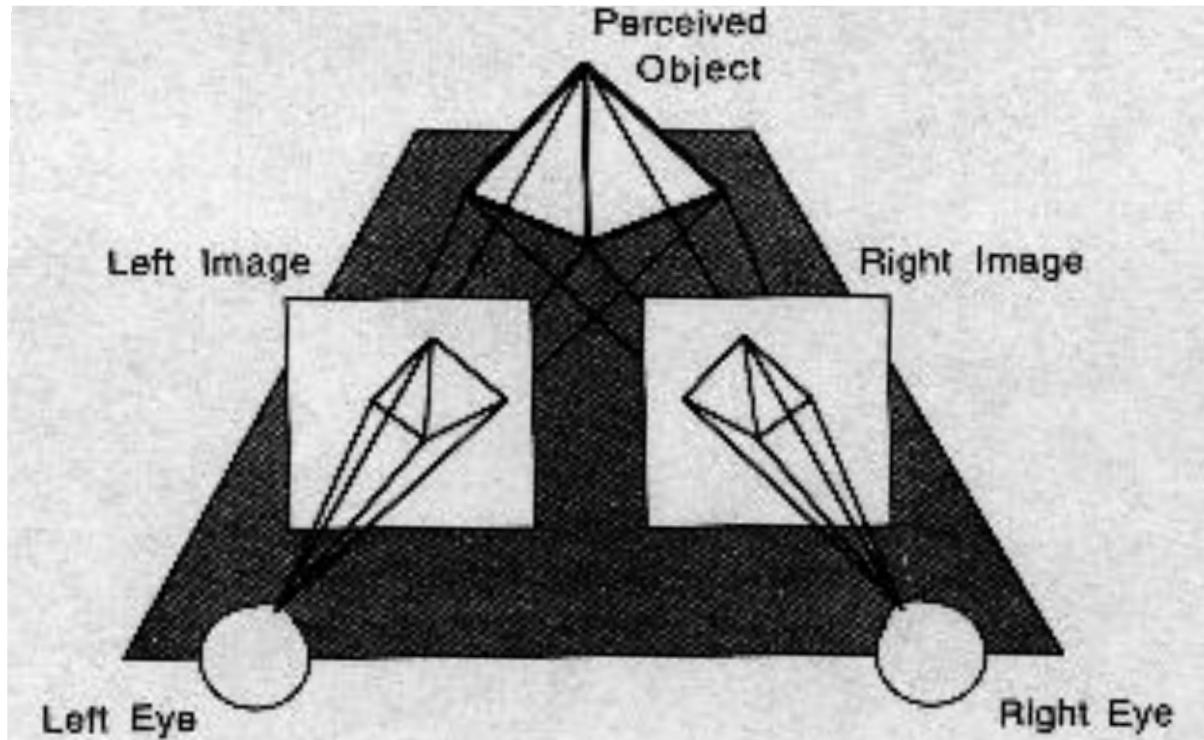
- Texture
- Shading

Two images or more:

- **Stereo**
- Contours
- Motion

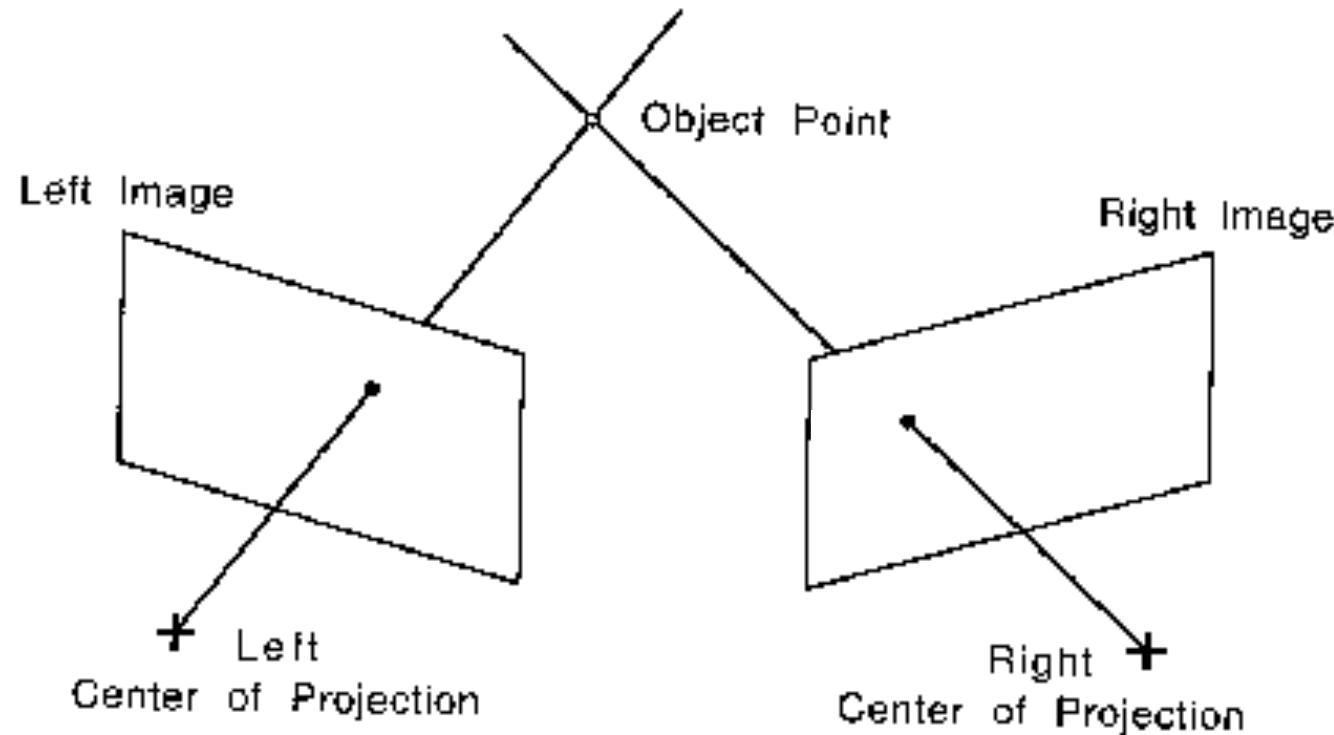


GEOMETRIC STEREO



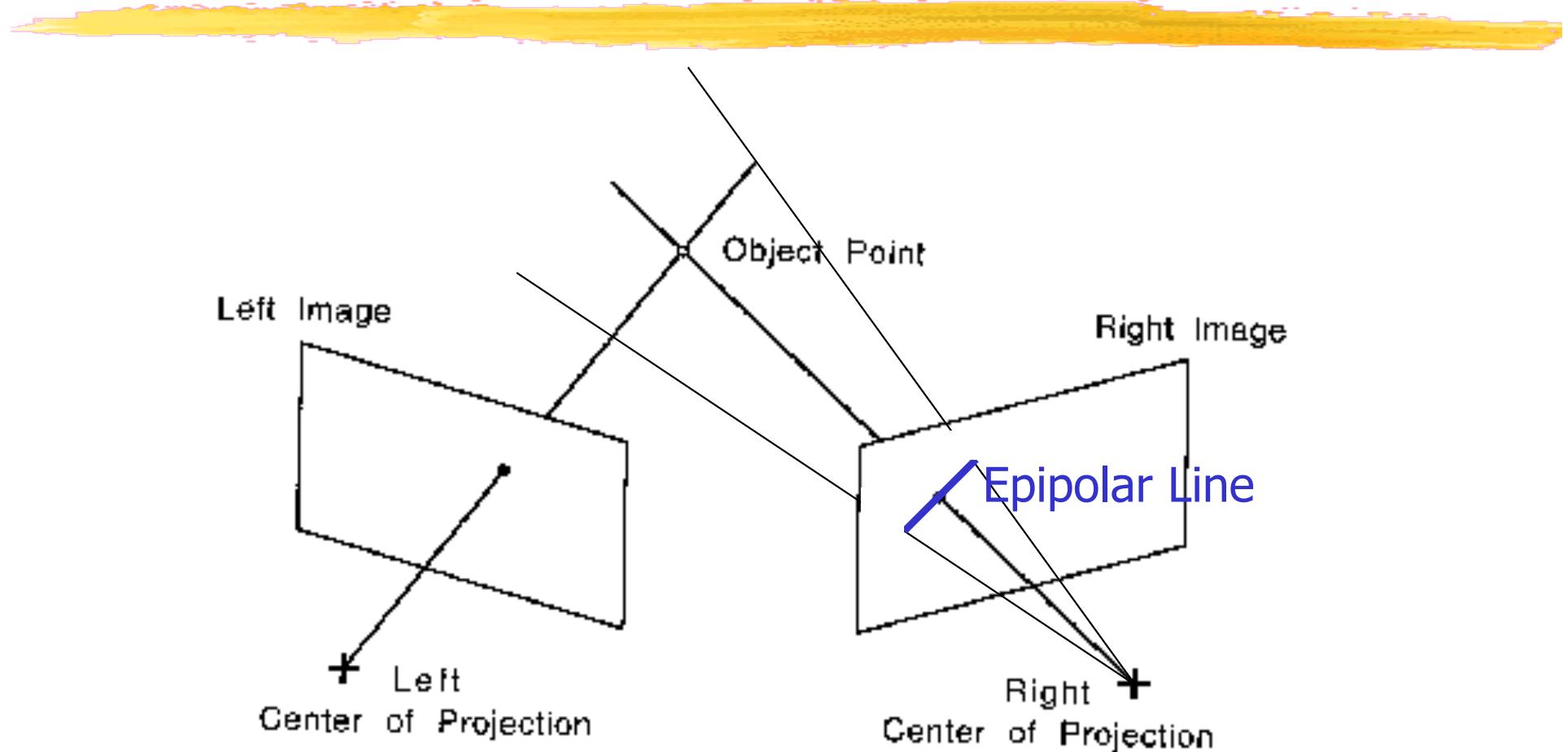
- Depth from two or more images:
- Geometry of image pairs
 - Establishing correspondences

TRIANGULATION



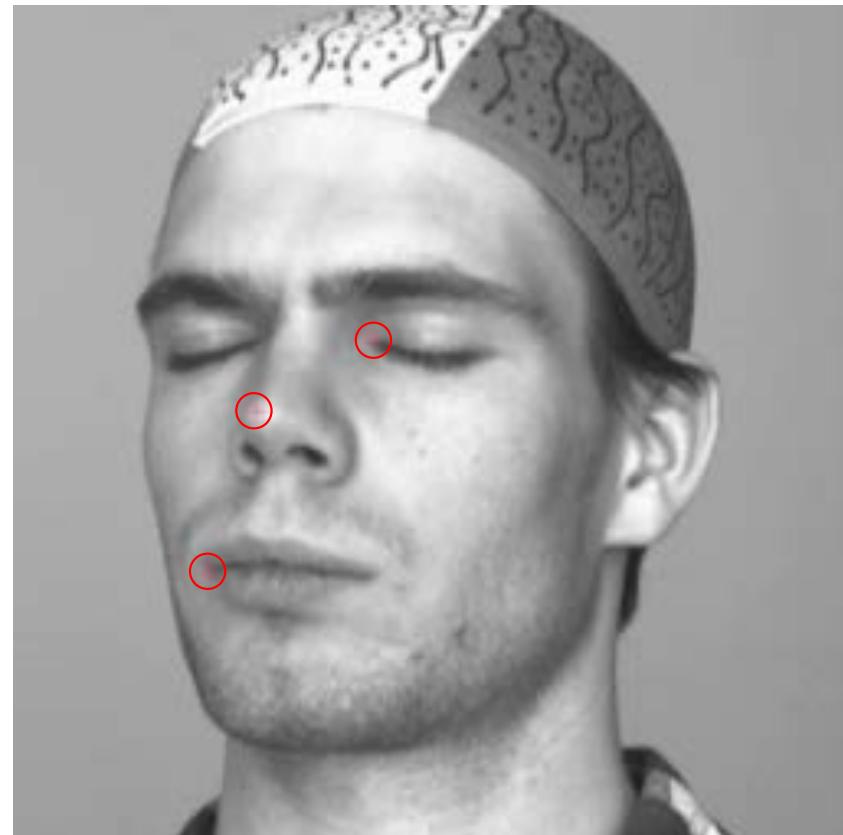
Geometric Stereo: Depth from two images

EPIPOLAR LINE

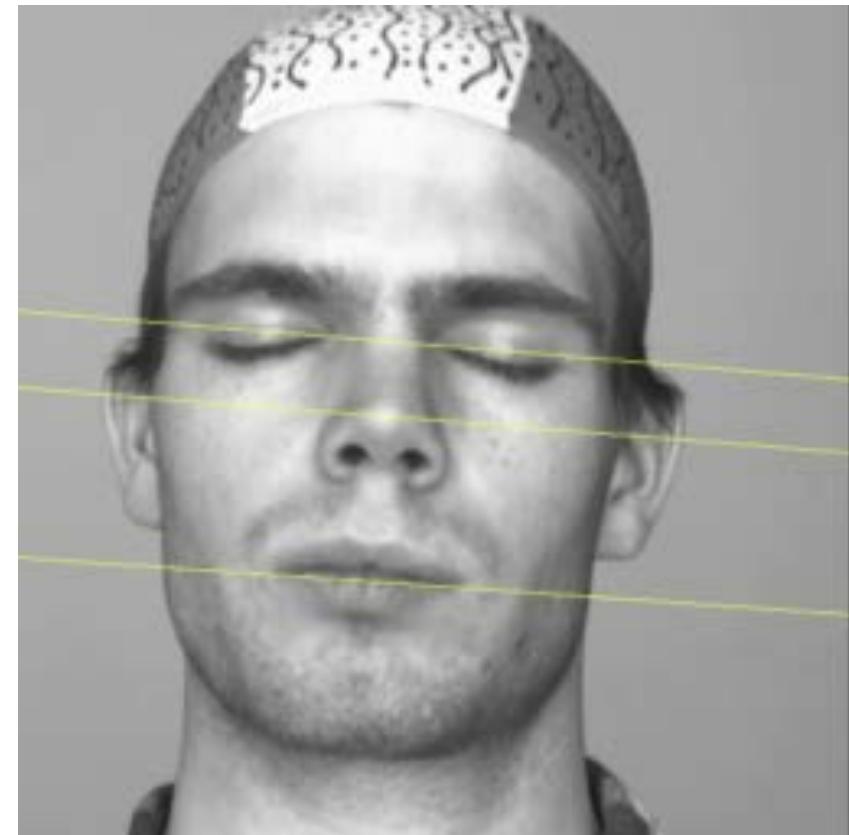


Line on which the corresponding point must lie.

EPIPOLAR LINES

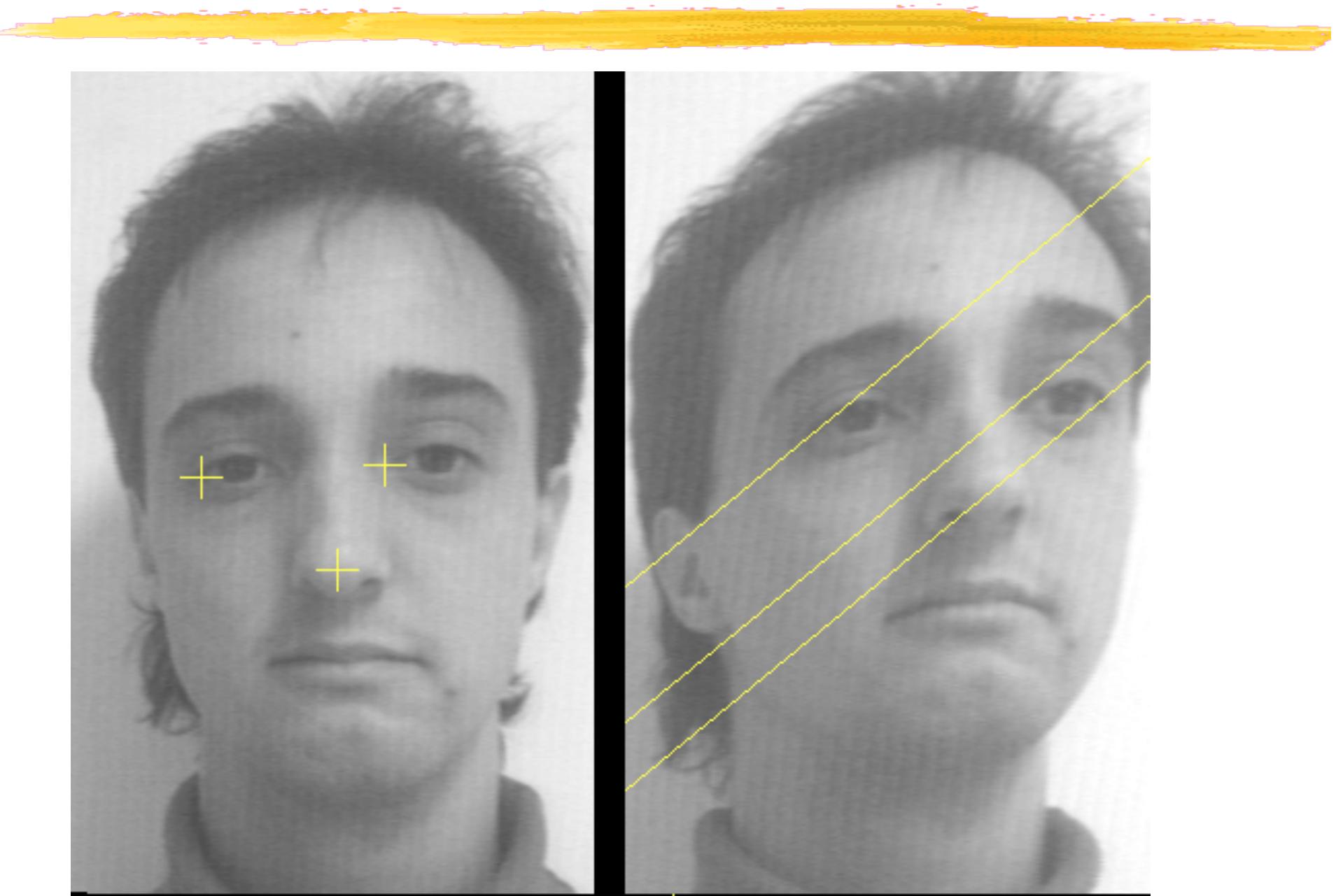


Three points shown as
red crosses.

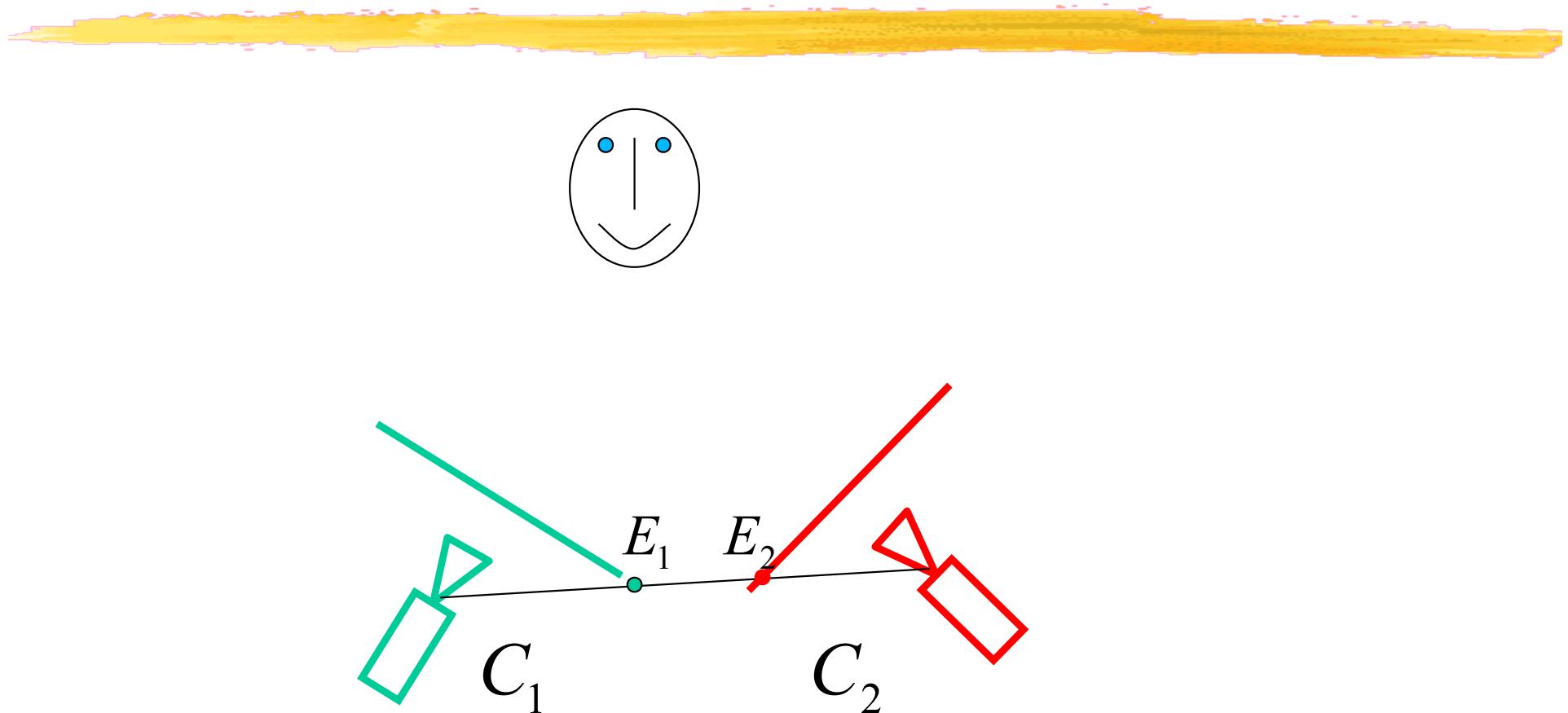


Corresponding epipolar
lines.

EPIPOLAR LINES



EPIPOLE



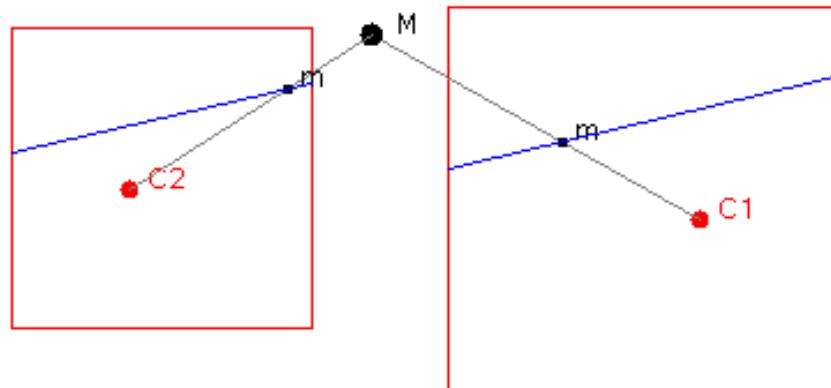
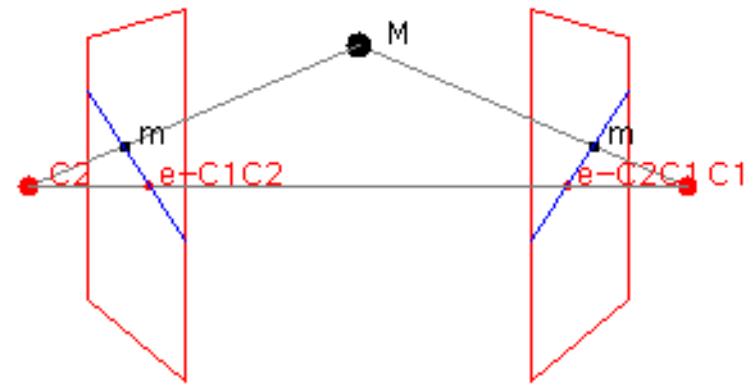
Point at which **all** epipolar lines intersect:

- ▶ Located at the intersection of line joining optical centers and image plane.

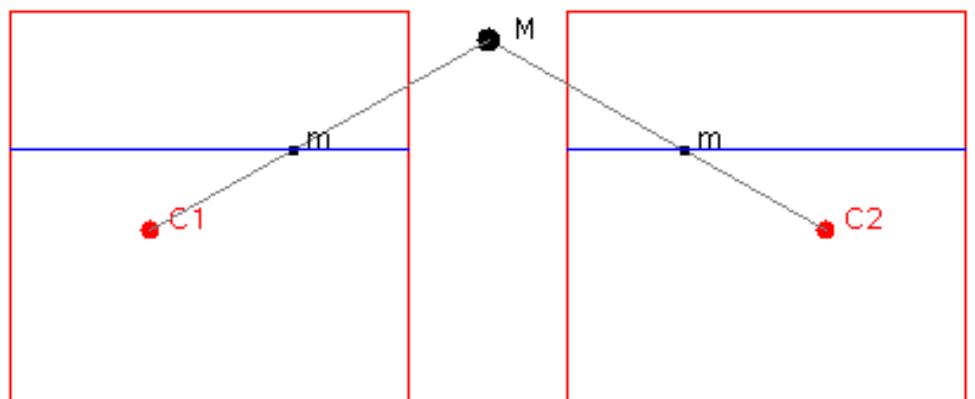
EPIPOLAR GEOMETRY



In general:

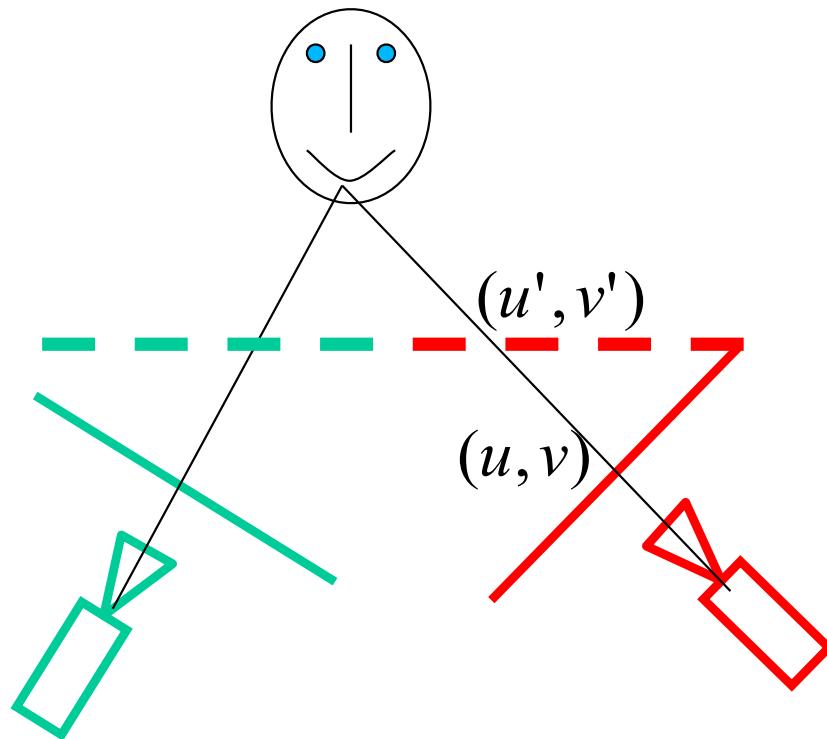


Parallel image planes



Horizontal baseline

RECTIFICATION



$$\begin{bmatrix} U' \\ V' \\ W' \end{bmatrix} = \begin{bmatrix} r_{11} & r_{12} & r_{13} \\ r_{21} & r_{22} & r_{23} \\ r_{31} & r_{32} & 1 \end{bmatrix} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix}$$

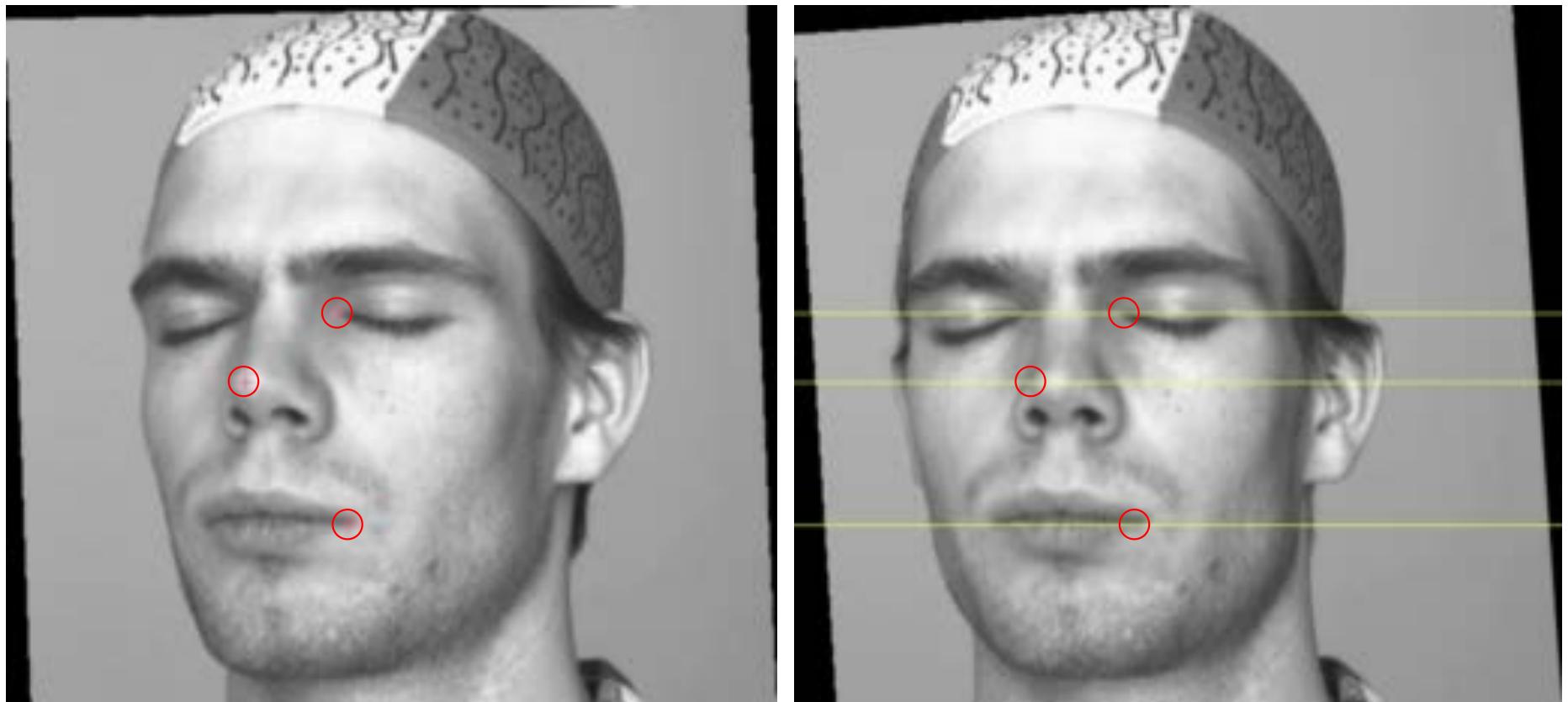
$$u' = U' / W'$$

$$v' = V' / W'$$

Reprojection into parallel virtual image planes:

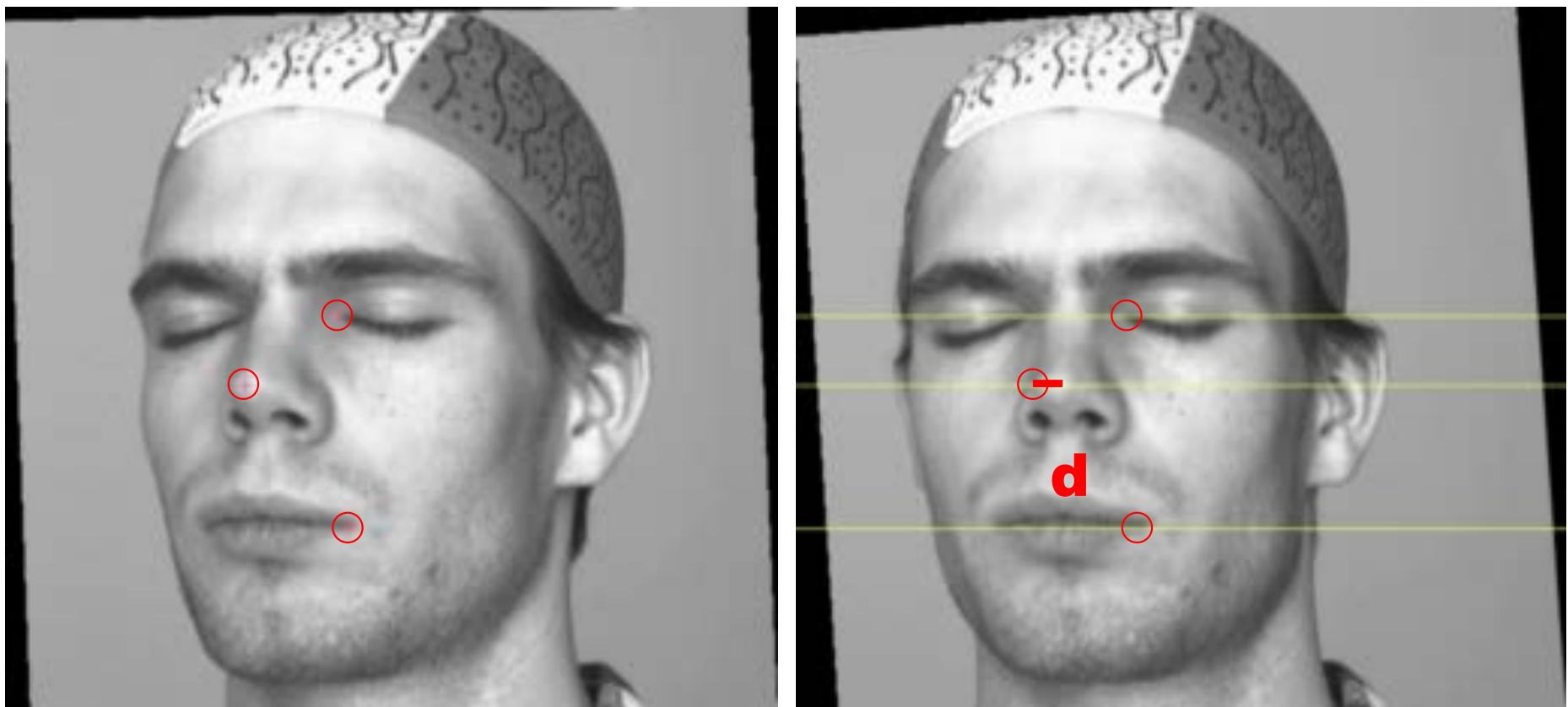
- Linear operation in projective coordinates
- Real-time implementation possible

RECTIFICATION



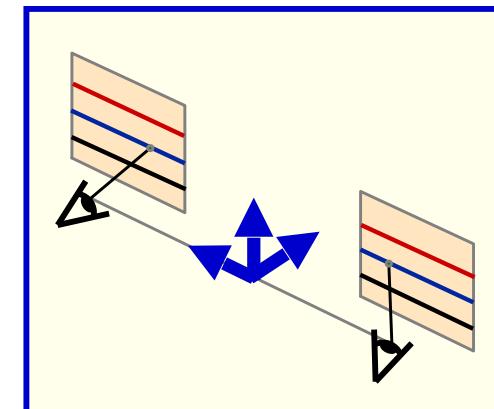
Intersecting epipolar lines
→ Parallel epipolar lines

DISPARITY



Horizontal shift along epipolar line, inversely proportional to distance.

DISPARITY VS DEPTH



$$u_l = \frac{f(X - b/2)}{Z}, v_l = \frac{fY}{Z}$$

$$u_r = \frac{f(X + b/2)}{Z}, v_l = \frac{fY}{Z}$$

$$d = f \frac{b}{Z}$$

→ Disparity is inversely proportional to depth.

WINDOW BASED APPROACH



- Compute a cost for each C_n location.
- Pick the lowest cost one.

FINDING A PATTERN IN AN IMAGE

Straightforward Approach



Pattern

Move pattern everywhere and compare with image.

But how?

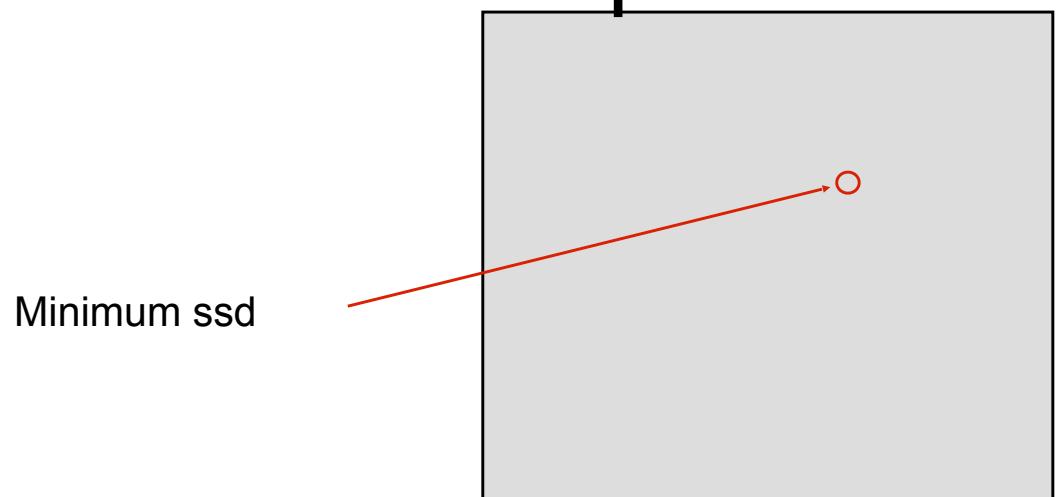
SUM OF SQUARE DIFFERENCES

Subtract pattern and image pixel by pixel and add squares:

$$ssd(u,v) = \sum_{(x,y) \in N} [I(u+x, v+y) - P(x, y)]^2$$

If identical $ssd=0$, otherwise $ssd > 0$

→ Look for minimum of ssd with respect to u and v .



CORRELATION

$$\begin{aligned} ssd(u,v) &= \sum_{(x,y) \in N} [I(u+x, v+y) - P(x, y)]^2 \\ &= \sum_{(x,y) \in N} I(u+x, v+y)^2 + \sum_{(x,y) \in N} P(x, y)^2 - 2 \sum_{(x,y) \in N} I(u+x, v+y)P(x, y) \end{aligned}$$

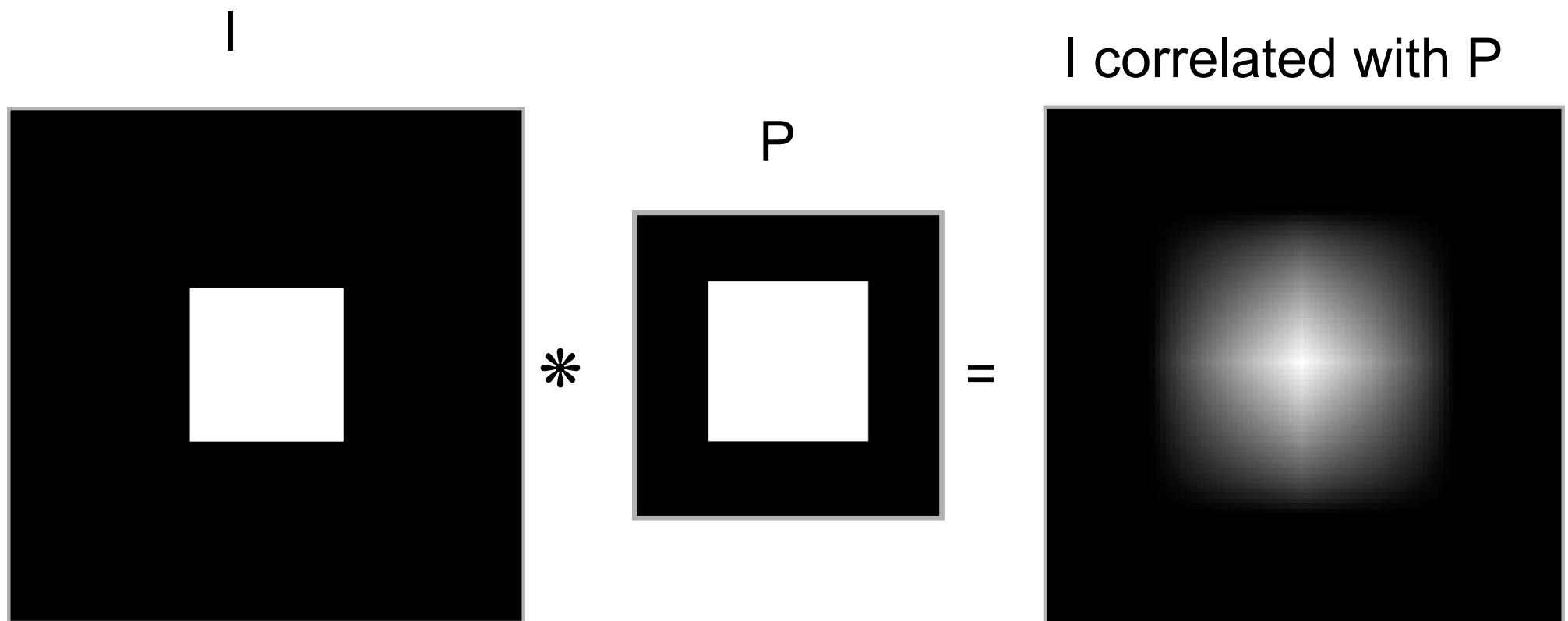
Sum of squares of
the window
(slow varying)

Sum of squares of
the pattern
(constant)

Correlation

$ssd(u,v)$ is minimized when correlation is largest
→ Correlation measures similarity

SIMPLE EXAMPLE

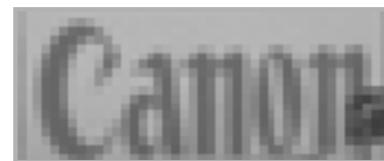


NOT SO SIMPLE EXAMPLE

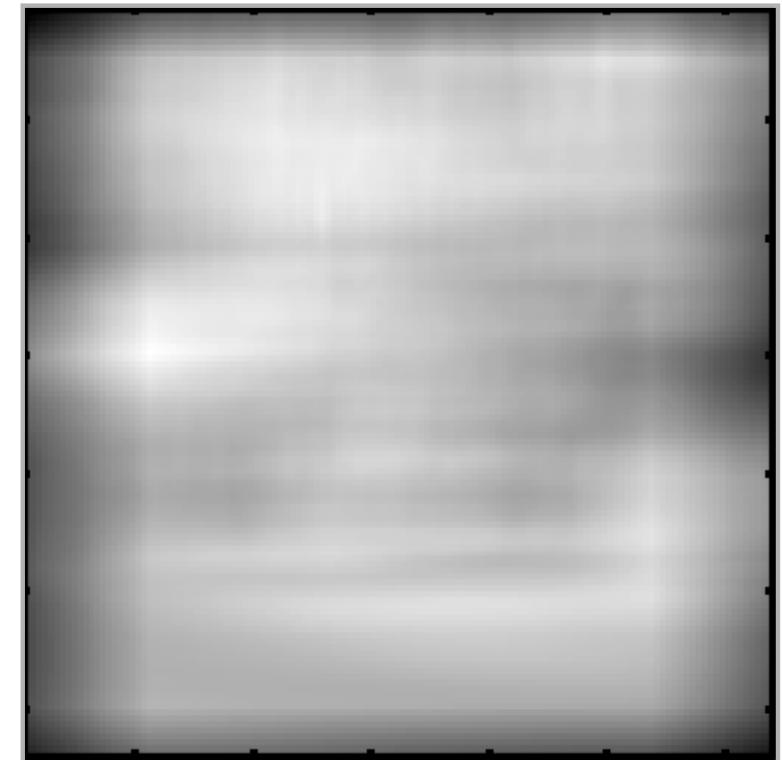
Image



Pattern



Correlation



- Correlation value depends on the local gray levels of the pattern and image window.
- Need to normalize.

NORMALIZED CROSS CORRELATION

$$ncc(u,v) = \frac{\sum_{(x,y) \in N} [I(u+x, v+y) - \bar{I}] [P(x, y) - \bar{P}]}{\sqrt{\sum_{(x,y) \in N} [I(u+x, v+y) - \bar{I}]^2 \sum_{(x,y) \in N} [P(x, y) - \bar{P}]}}$$

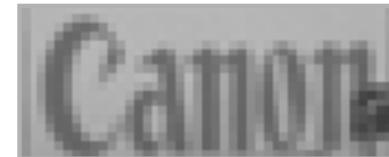
- Between -1 and 1
- Invariant to linear transforms
- Independent of the average gray levels of the pattern and the image window

NORMALIZED EXAMPLE

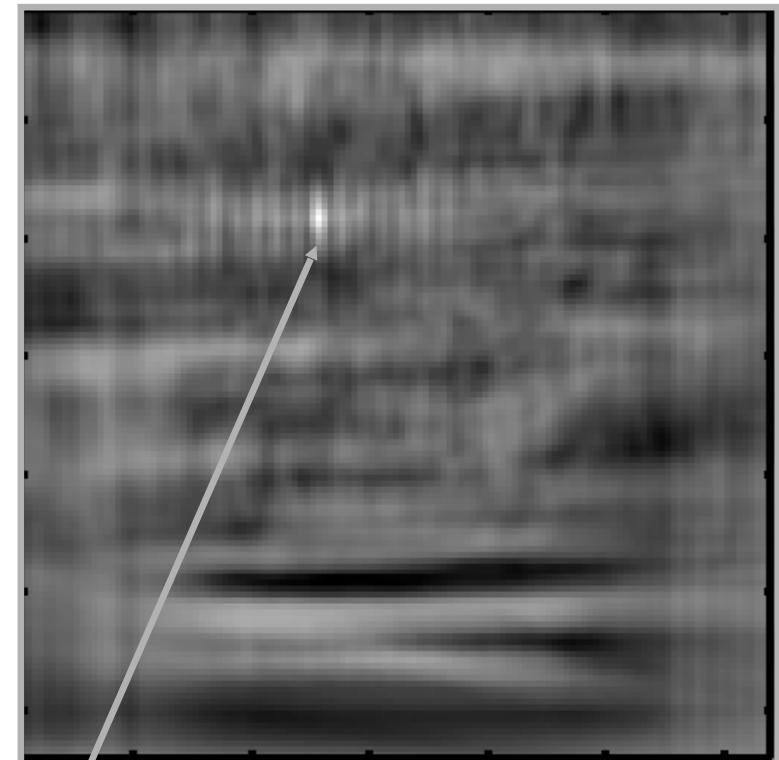
Image



Pattern

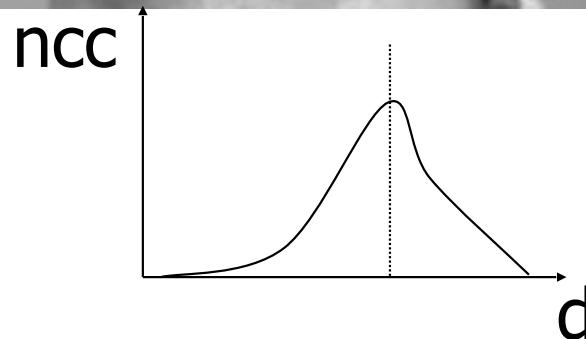
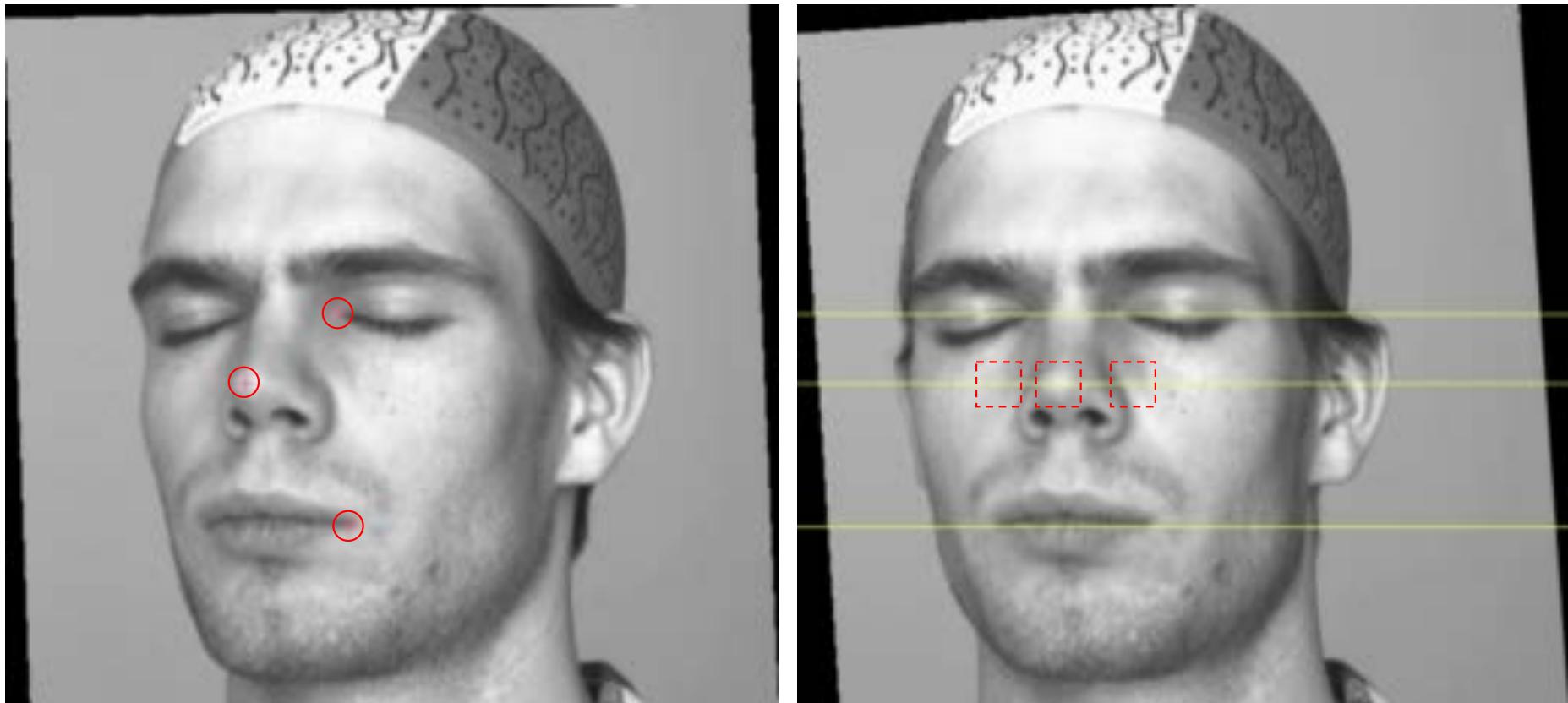


Normalized Correlation

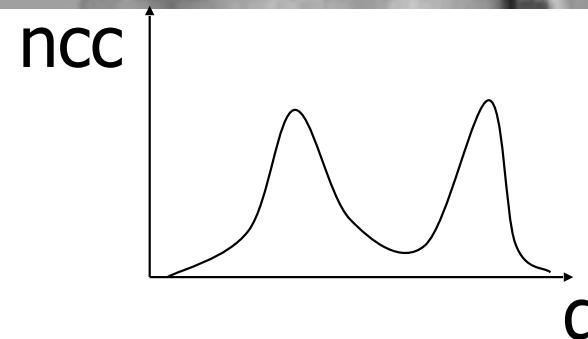


Point of maximum correlation

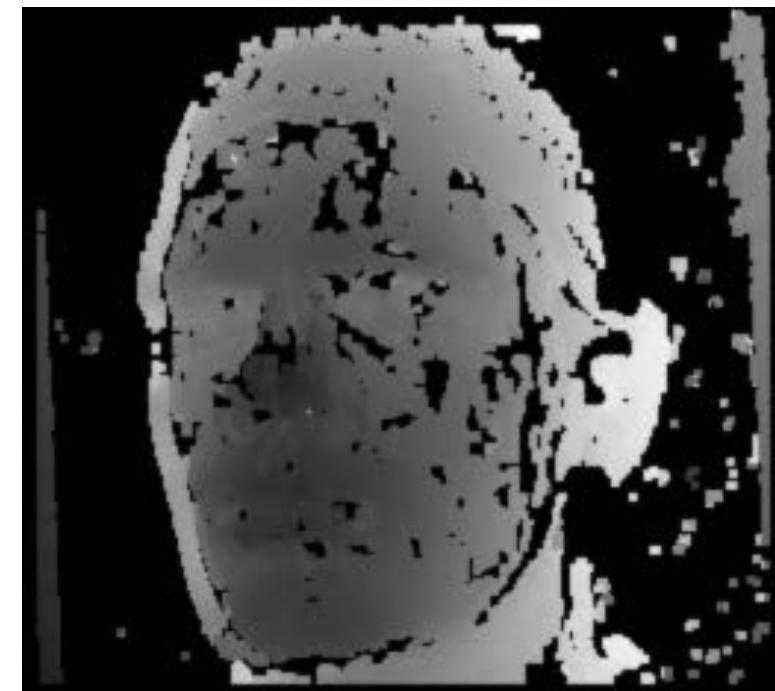
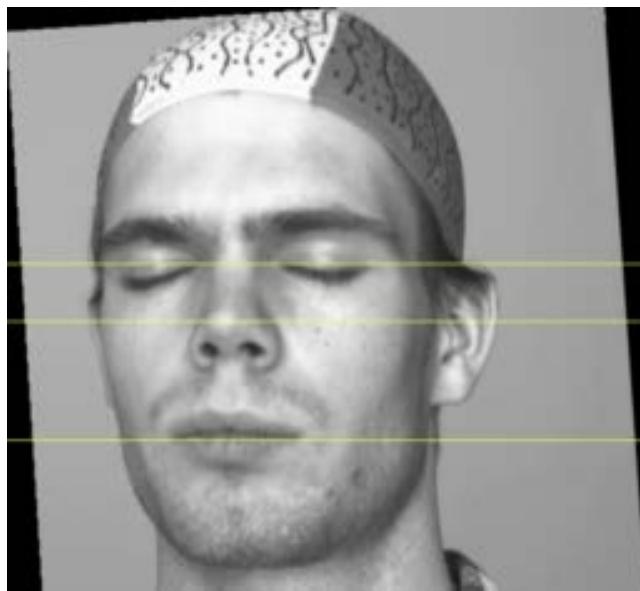
SEARCHING ALONG EPIPOLAR LINES



or

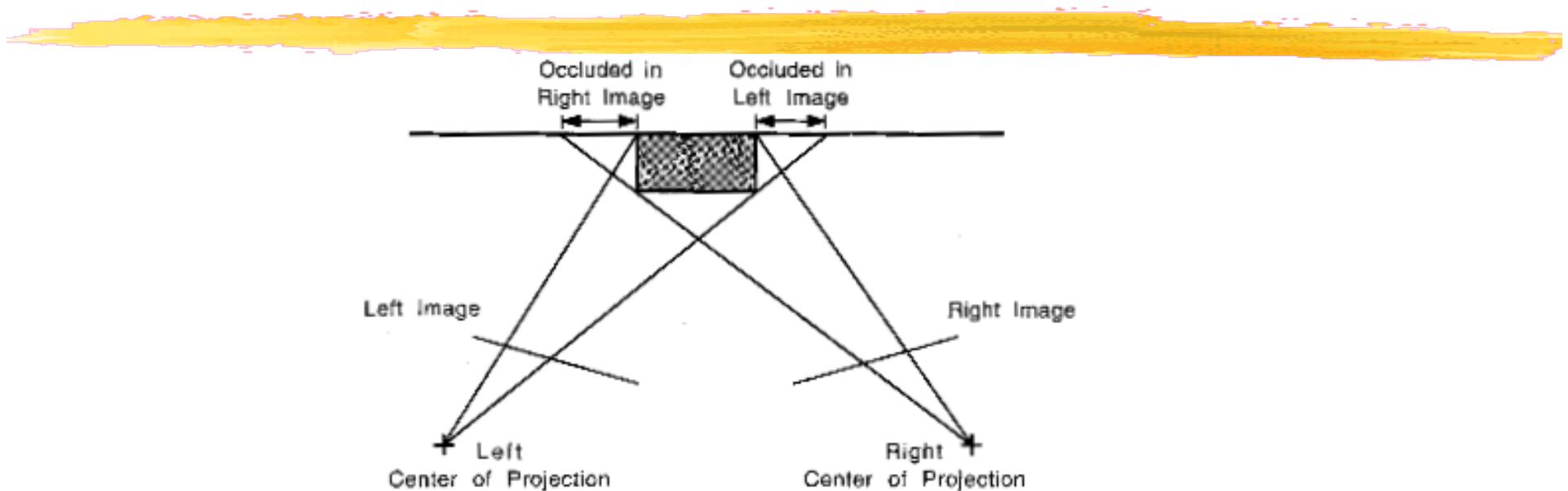


DISPARITY MAP

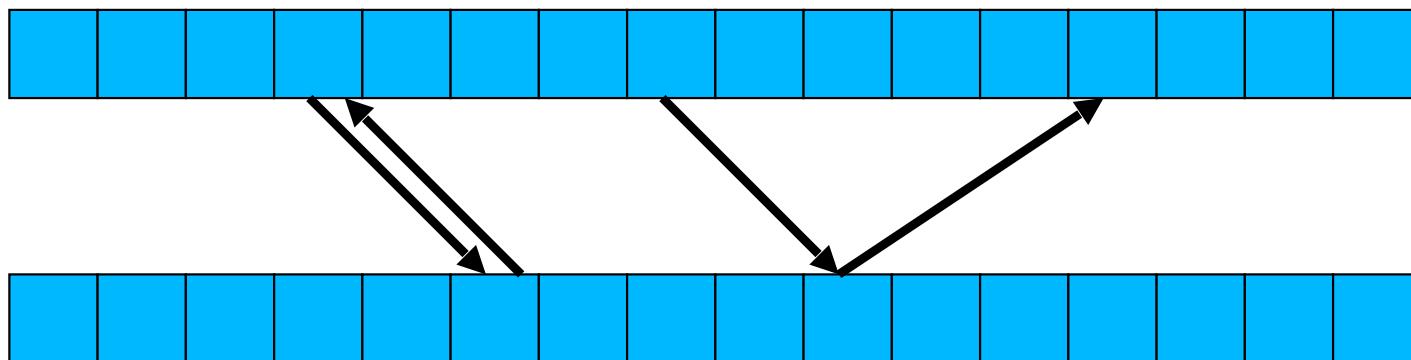


Black pixels: No disparity.

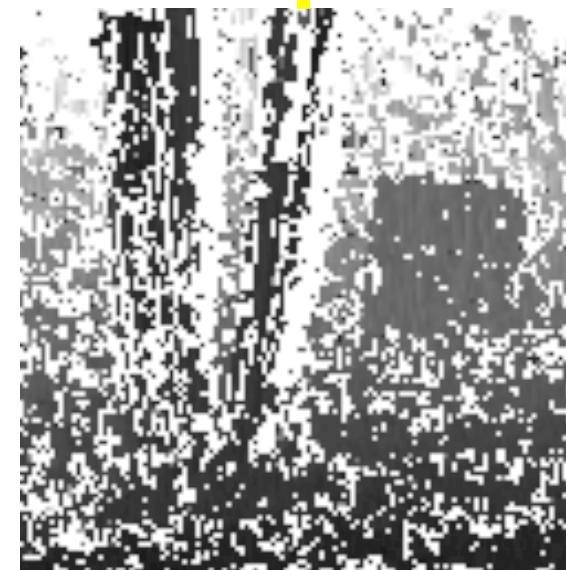
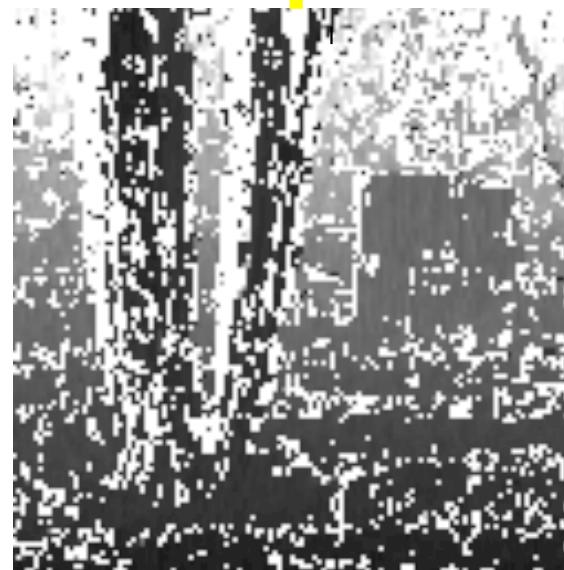
OCCLUSIONS



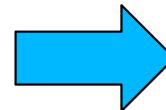
→ Consistency test



GROUND LEVEL STEREO

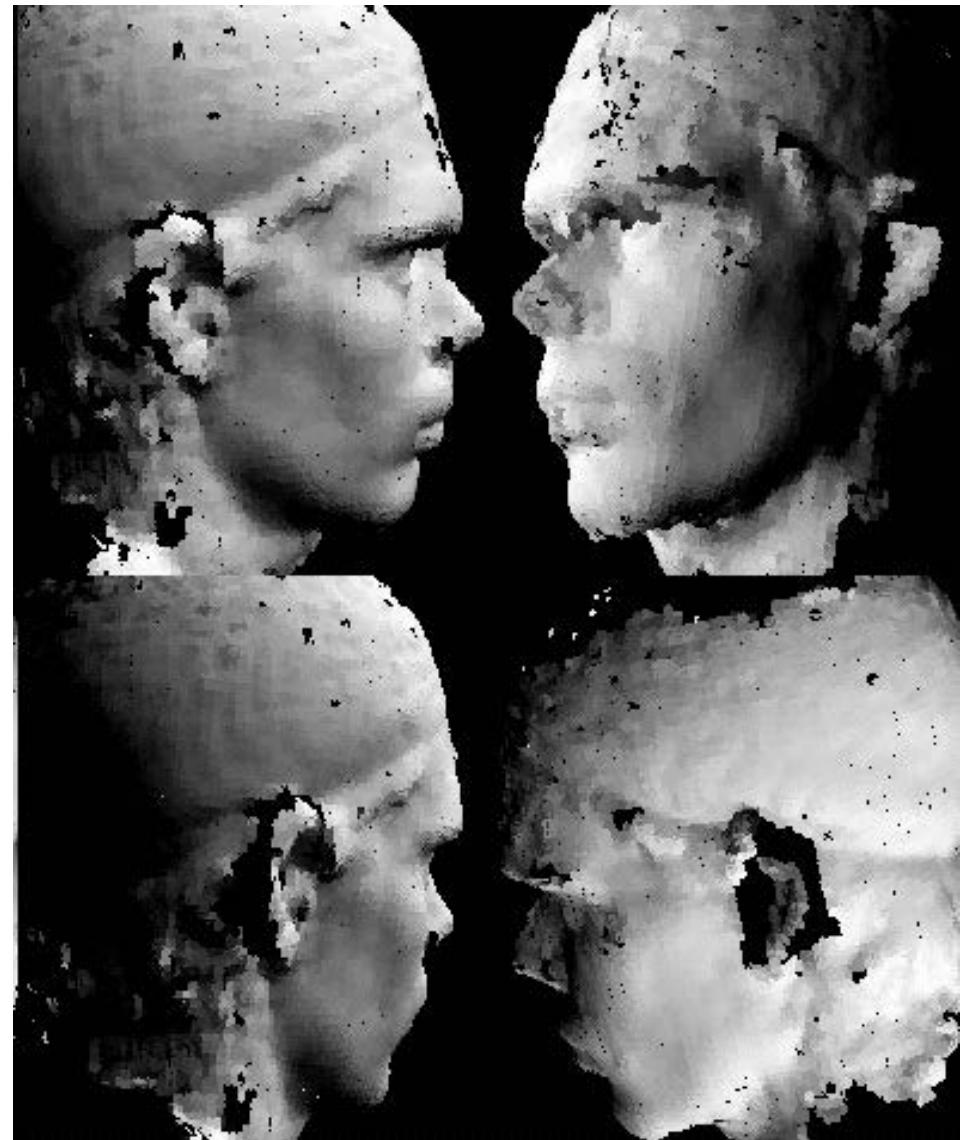
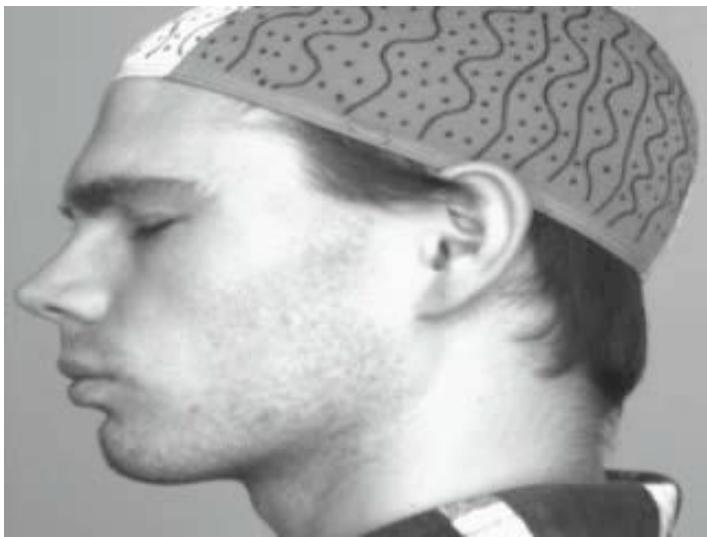


COMBINING DISPARITY MAPS



- Merging several disparity maps.
- Smoothing the resulting map.

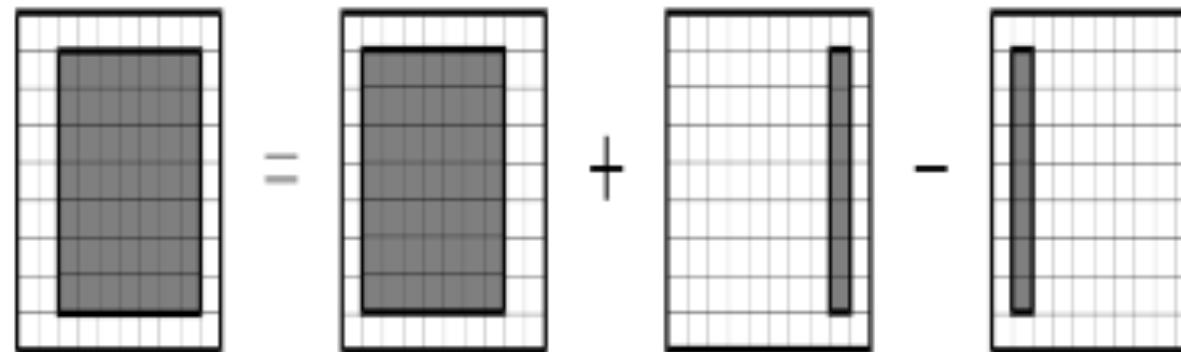
SHAPE FROM VIDEO



Treat consecutive images as stereo pairs.

1. Compute disparity maps.
2. Merge 3-D point clouds.
3. Represent as particles.

REAL-TIME IMPLEMENTATION

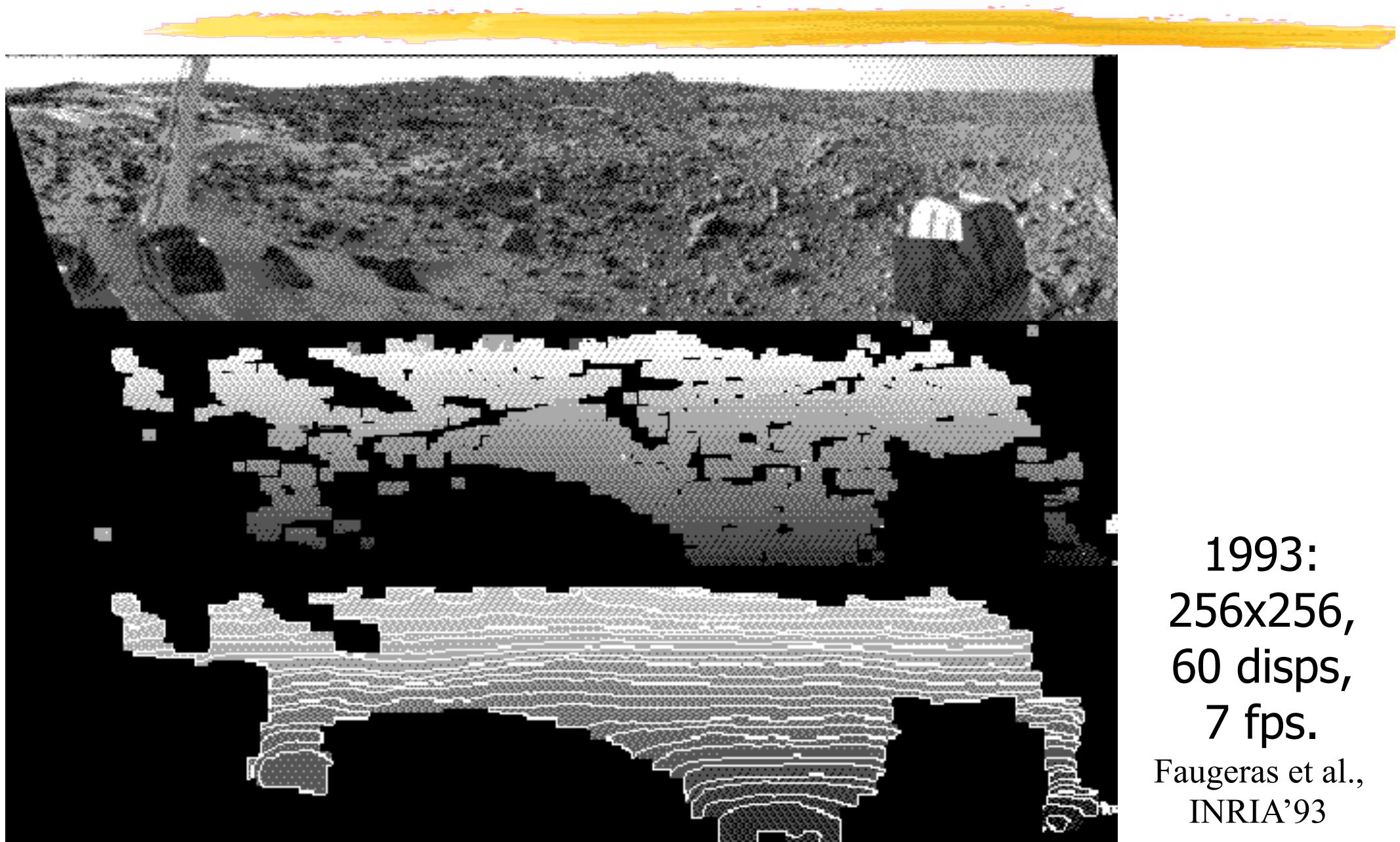


$$C(x, y, d) \propto \frac{\sum_{i,j} I_1(x+i, y+j) \times I_2(x+d+i, y+j)}{\sqrt{\sum_{i,j} I_2(x+d+i, y+j)^2}}$$

$$C(x+1, y, d) \propto \frac{\sum_{i,j} I_1(x+1+i, y+j) \times I_2(x+1+d+i, y+j)}{\sqrt{\sum_{i,j} I_2(x+1+d+i, y+j)^2}}$$

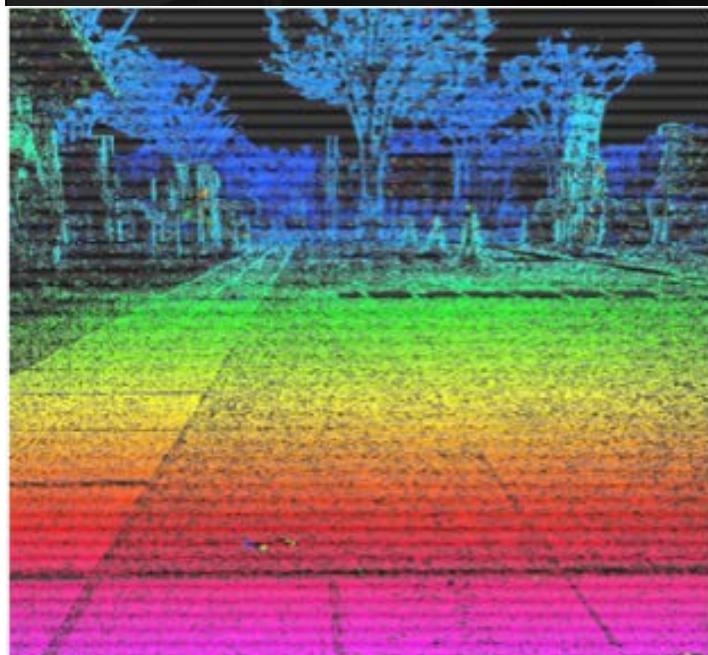
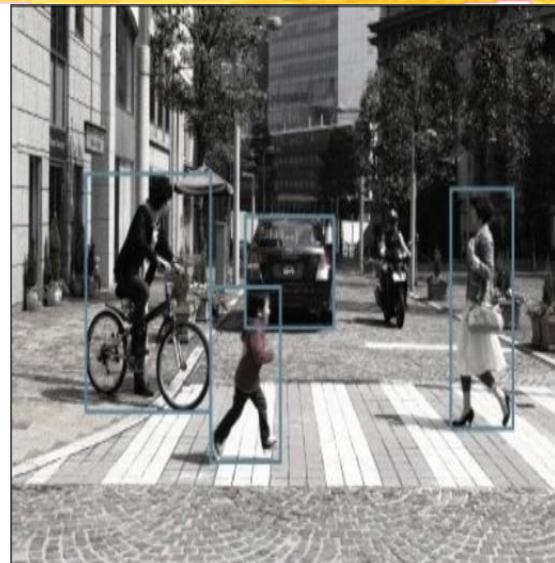
$$\propto \frac{\sum_{i',j} I_1(x+i', y+j) \times I_2(x+d+i', y+j)}{\sqrt{\sum_{i,j} I_2(x+d+i', y+j)^2}}$$

THEN



1993:
256x256,
60 disps,
7 fps.
Faugeras et al.,
INRIA'93

... AND MORE RECENTLY



Subaru's EyeSight System

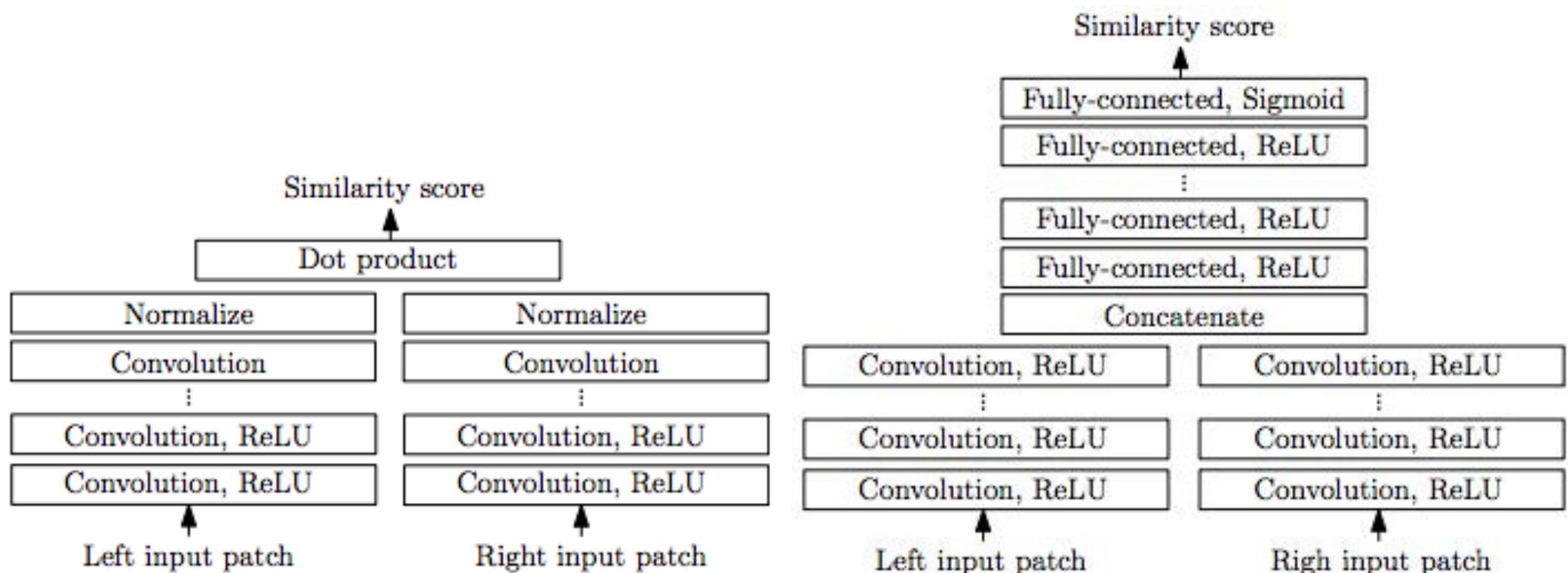
<http://www.gizmag.com/subaru-new-eyesight-stereoscopic-vision-system/14879/>

2011:
1312x688,
176 disps,
160 fps.

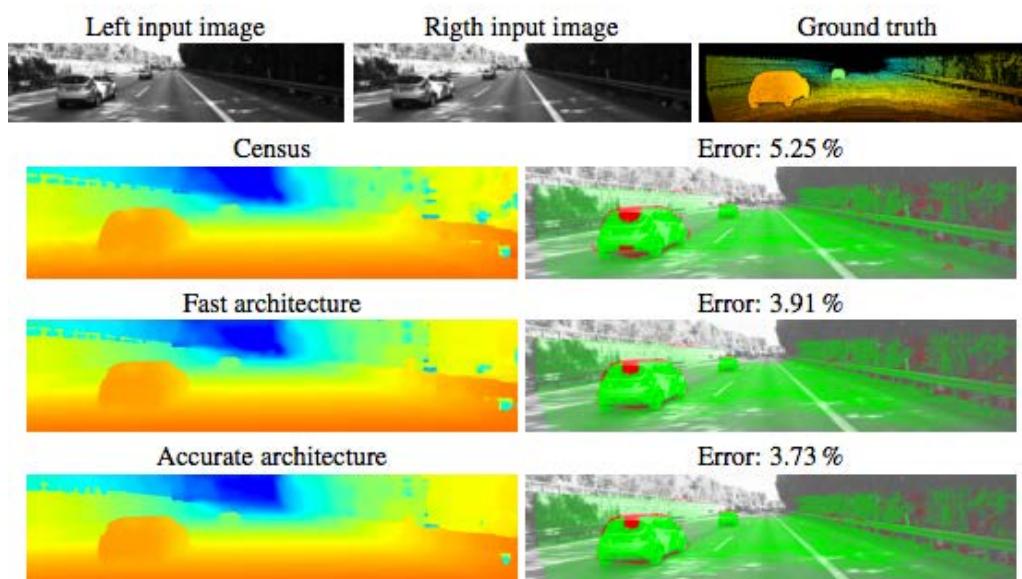
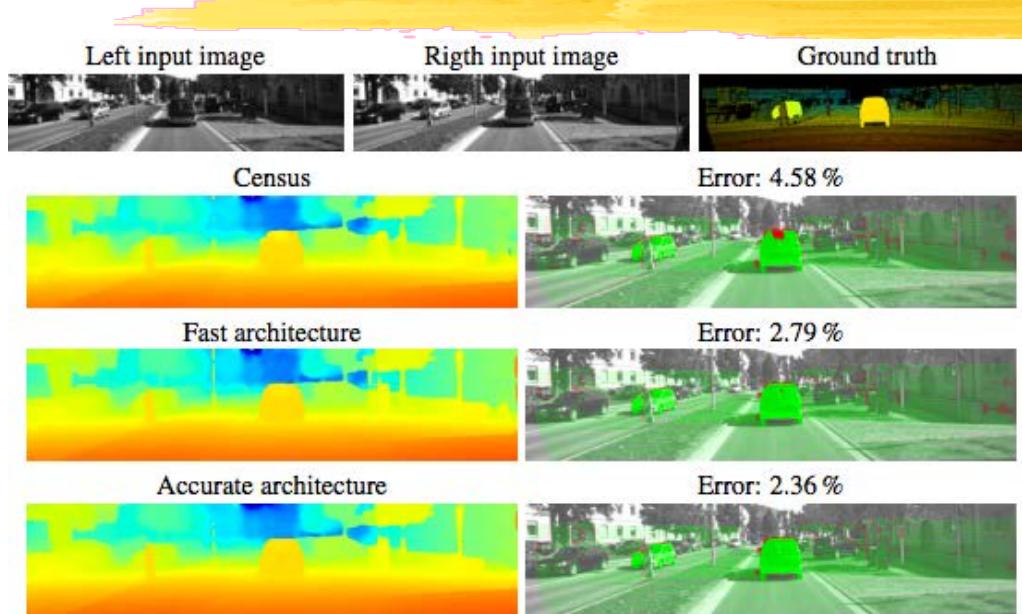
Saneyoshi, CMVA'11

... AND EVEN MORE RECENTLY

Train Siamese nets to return a similarity score.



COMPARATIVE RESULTS



Improved performance on test data but

- How well will it generalize to unseen images?

- Is it worth the much heavier computational load?

Time will tell.

WINDOW SIZE



Small windows:

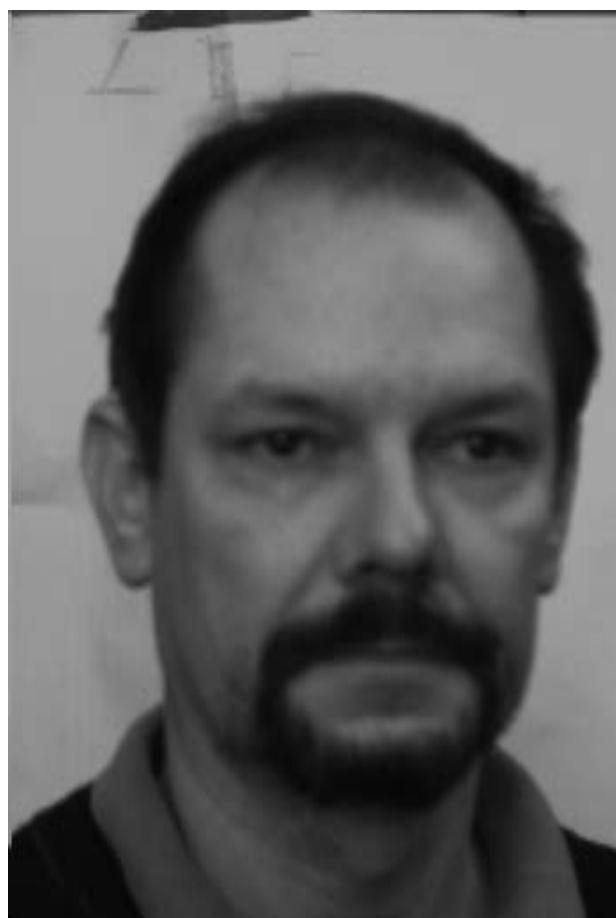
- Good precision
- Sensitive to noise

Large windows:

- Diminished precision
- Increased robustness to noise

→ Same kind of trade-off as for edge-detection.

WINDOW SIZE



15x15

7x7

SCALE-SPACE REVISITED



Gaussian pyramid

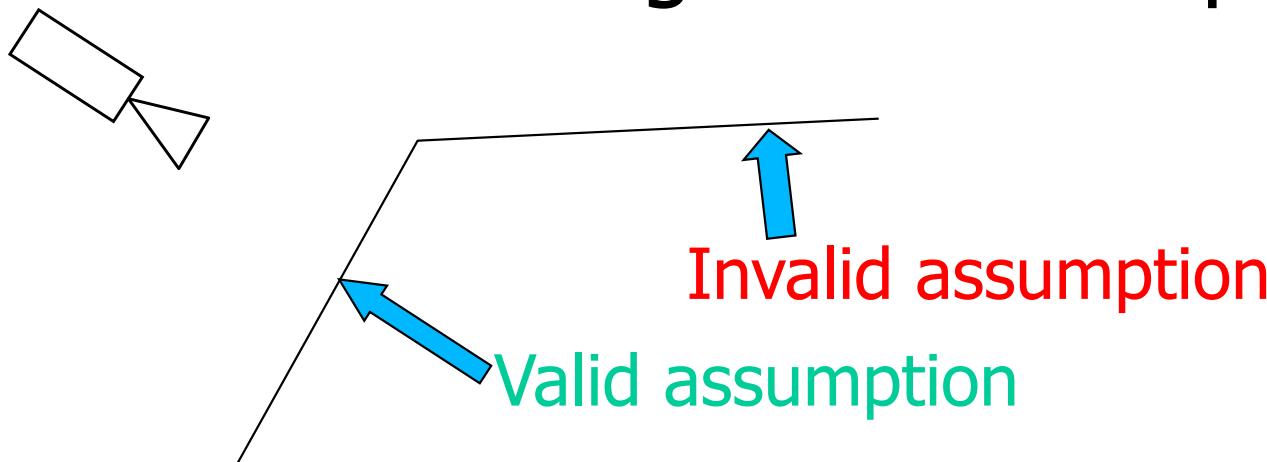


Difference of
Gaussians

- Using a small window on a reduced image is equivalent to using a large one on the original image.
- Using difference of Gaussian images is an effective way of achieving normalization.
→ It becomes natural to use results obtained using low resolution images to guide the search at higher resolution.

FRONTO-PARALLEL ASSUMPTION

The disparity is assumed to be the same in the whole correlation window, which is equivalent to assuming constant depth.



→ Ok when the surface faces the camera but breaks down otherwise.

MULTI-VIEW STEREO



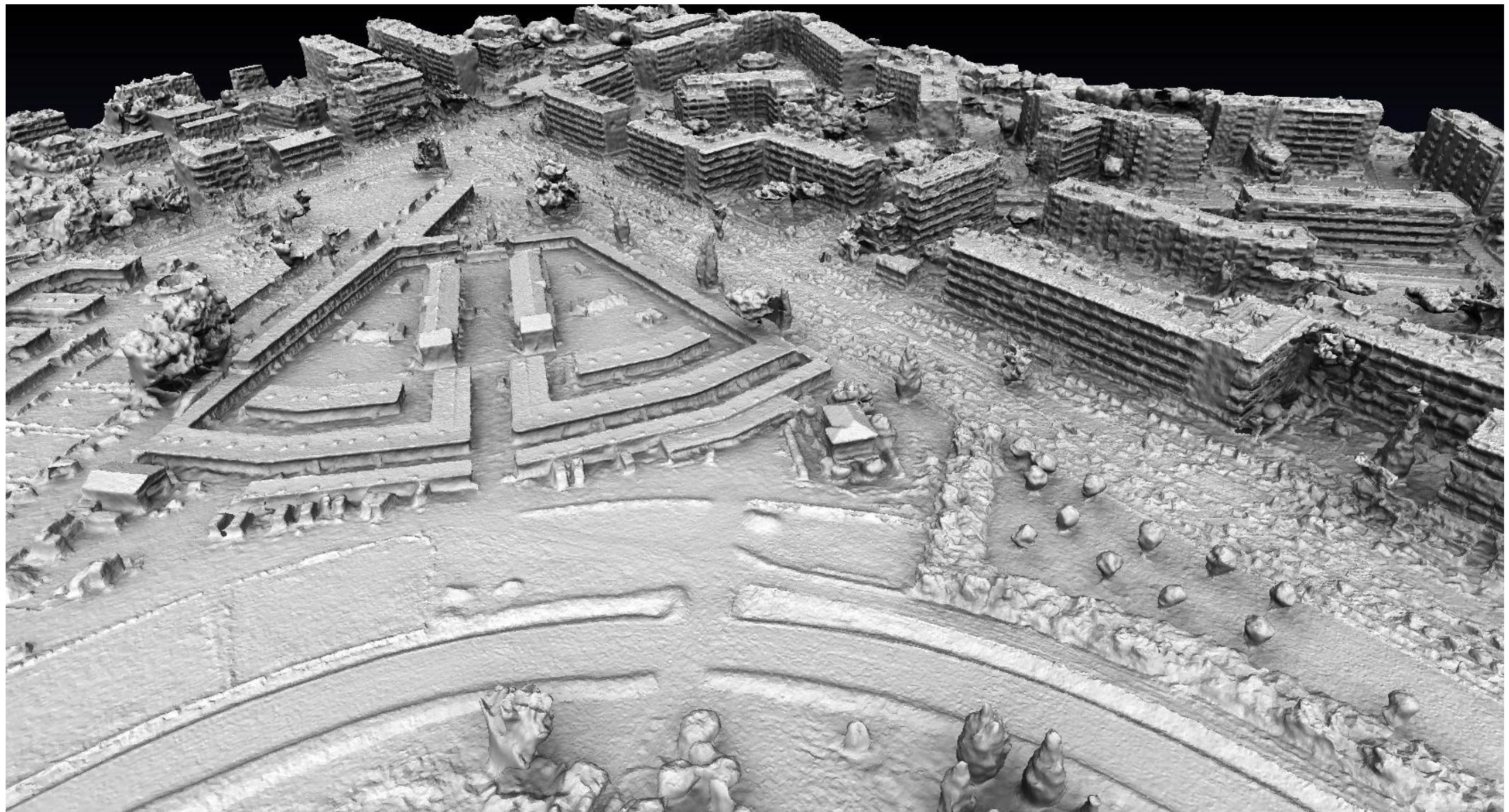
Multi-view reconstruction setup

- Adjust correlation window shapes to handle orientation.

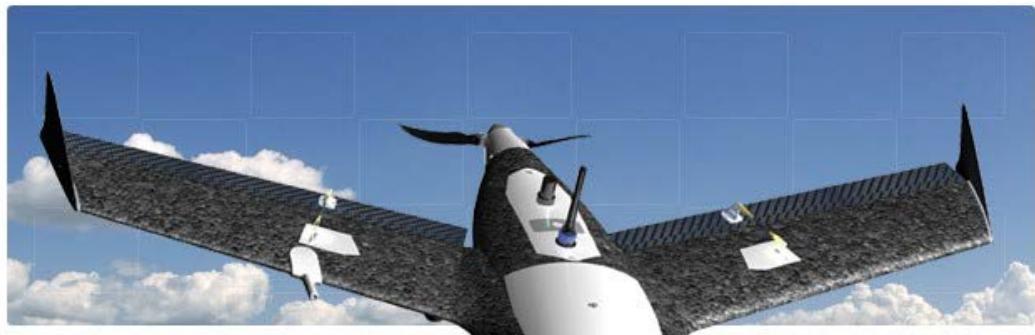


Textured
Skeletal
Mapped
3D Model

MULTI-VIEW STEREO



SMALL DRONES

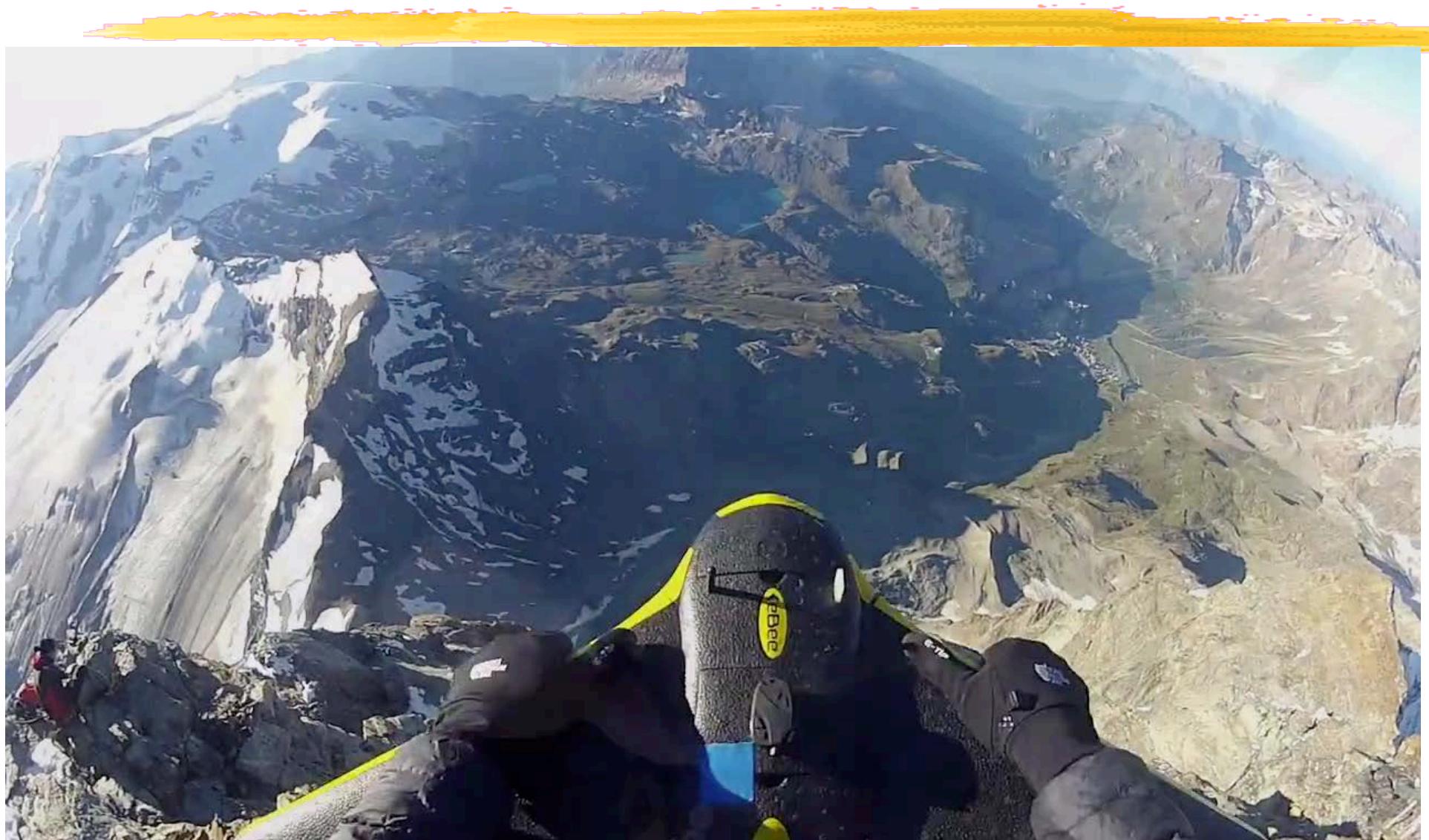


The X100
revolutionary mapping.
PATENT PENDING

SenseFly:
www.sensefly.com

Gatewing:
www.gatewing.com

MATTERHORN



Drone: www.sensefly.com

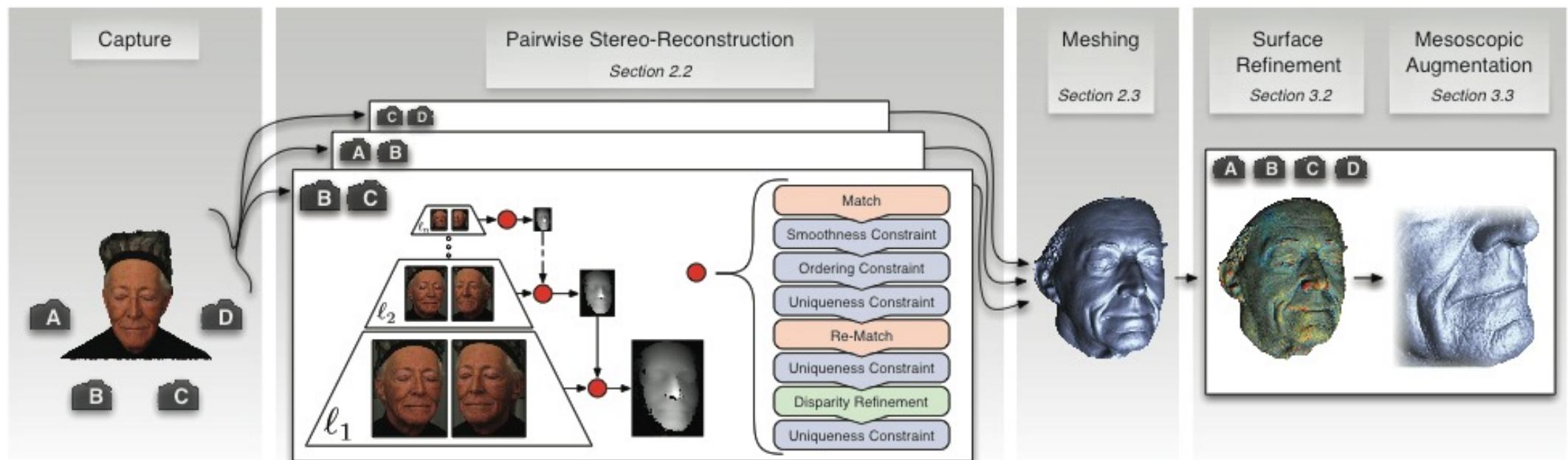
Mapping: www.pix4d.com

FACE RECONSTRUCTION



Beeler et al. SIGGRAPH'10

FACE RECONSTRUCTION



DYNAMIC SHAPE



Lightweight Binocular Facial Performance Capture under Uncontrolled Lighting

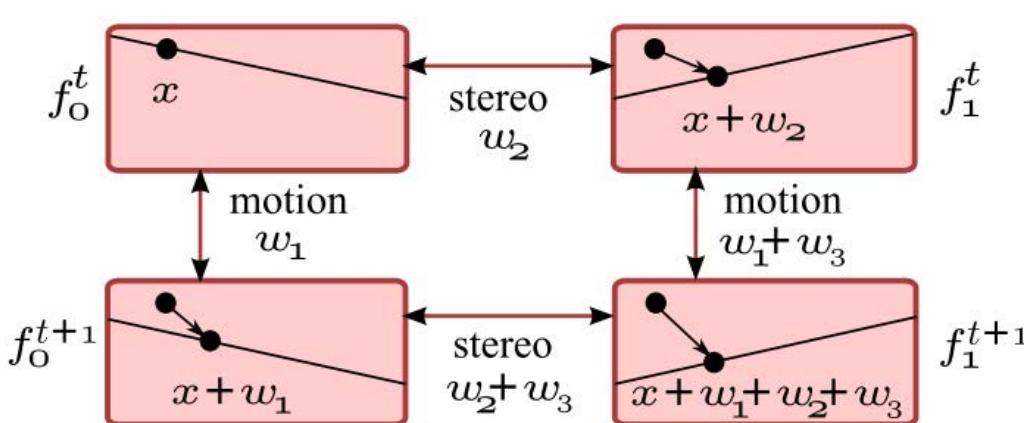
Levi Valgaerts¹ Chenglei Wu^{1,2} Andrés Bruhn³
Hans-Peter Seidel¹ Christian Theobalt¹

¹ MPI for Informatics

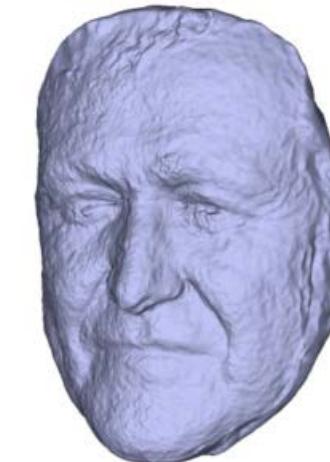
² Intel Visual Computing Institute

³ University of Stuttgart

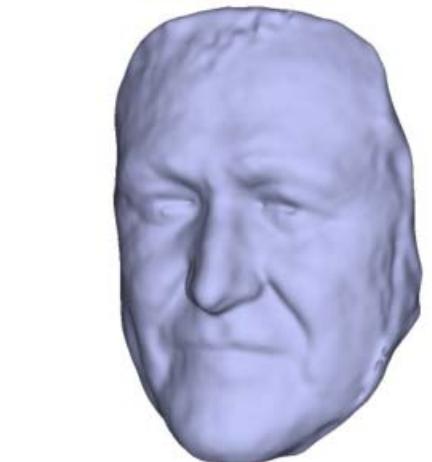
SCENE FLOW



Correspondences across cameras and across time

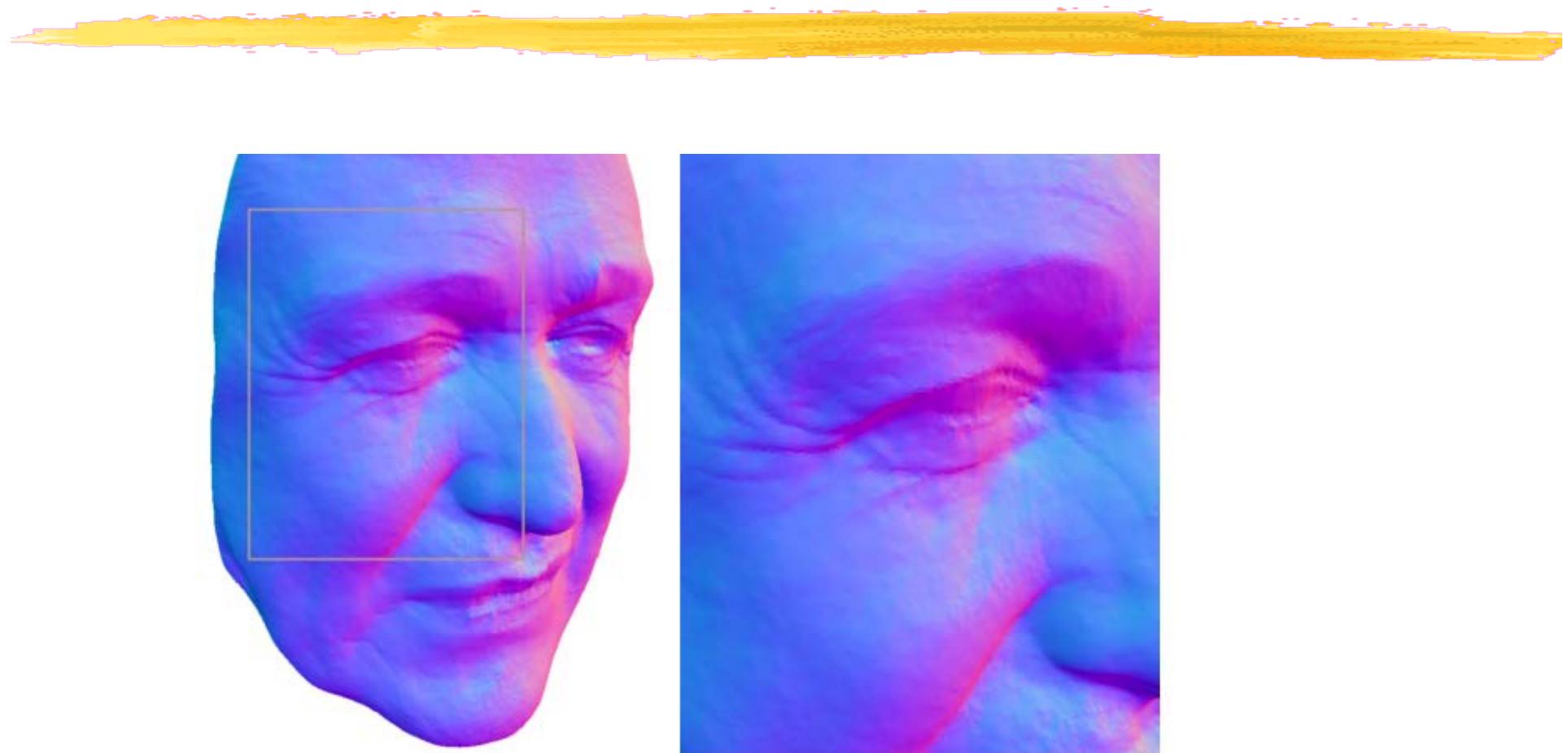


Stereo Only



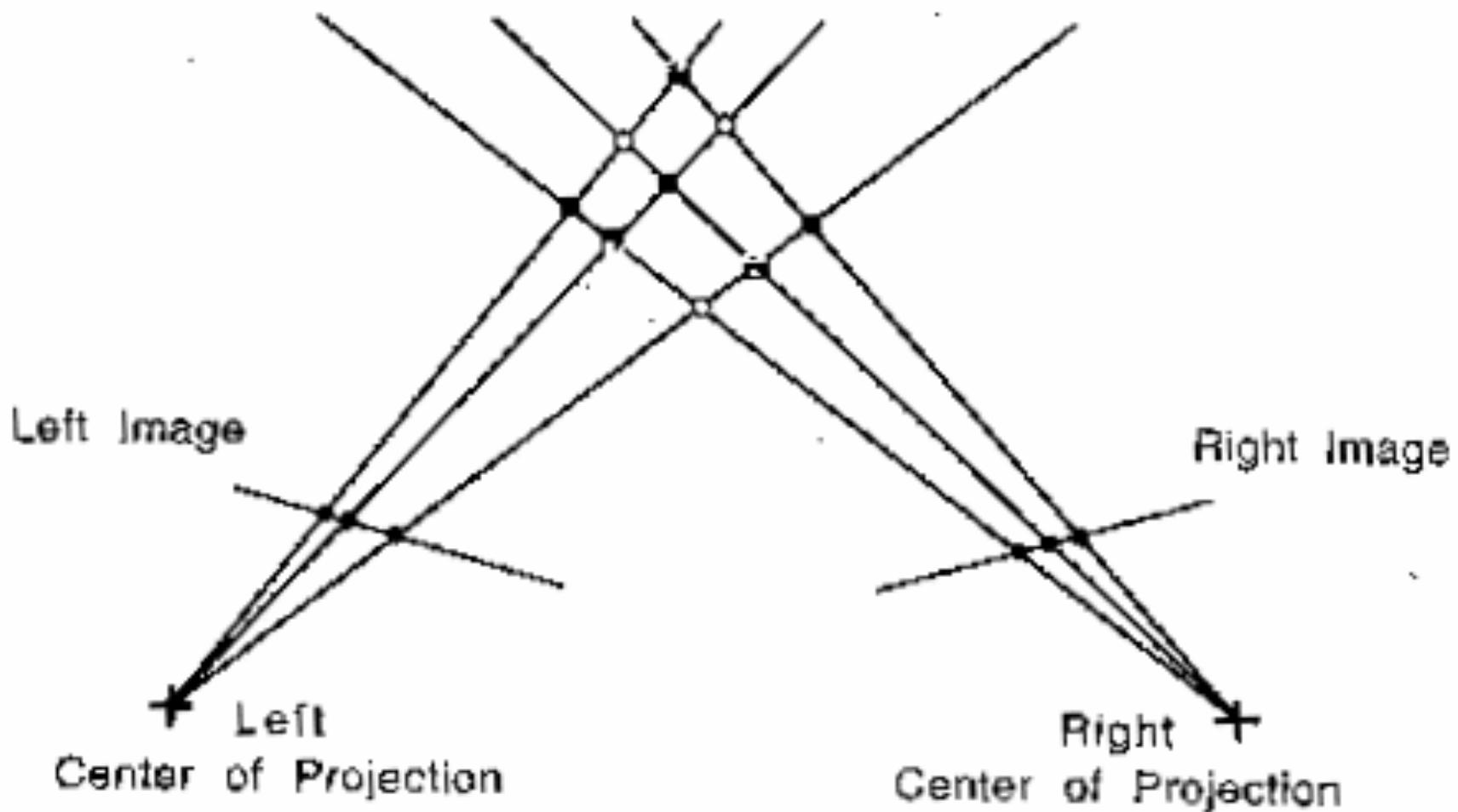
Stereo + Flow

SHAPE FROM SHADING

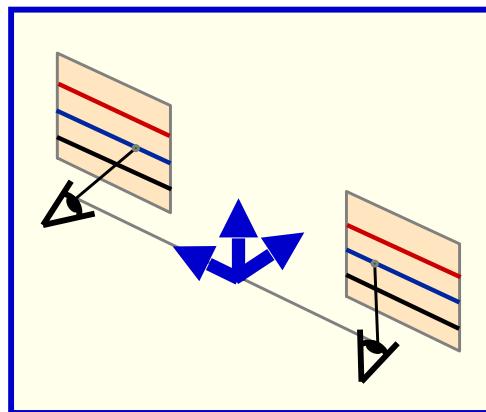


Shape-from-shading is used to refine the shape and provide high-frequency details.

UNCERTAINTY



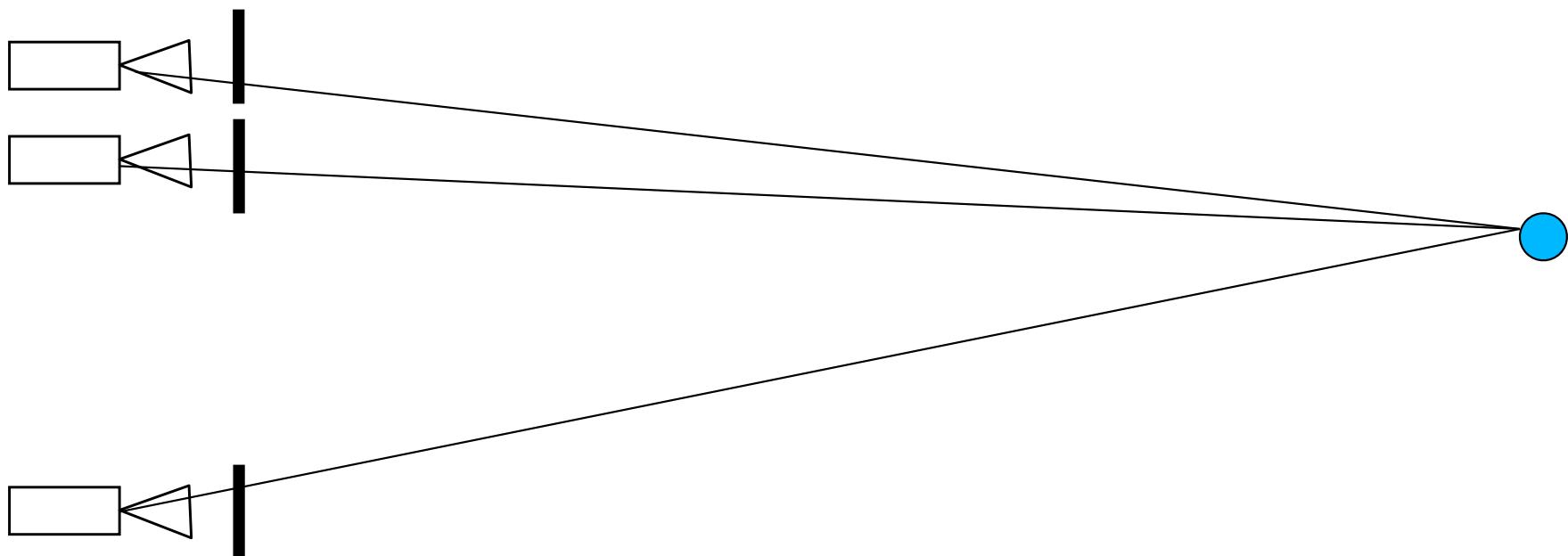
PRECISION vs BASELINE



$$d = f \frac{b}{Z}$$
$$\Rightarrow Z = f \frac{b}{d}$$
$$\Rightarrow \frac{\delta Z}{\delta d} = -f \frac{b}{d^2} = -\frac{Z^2}{fb}$$

- Beyond a certain depth stereo stops being useful.
- Precision is inversely proportional to baseline length.

SHORT vs LONG BASELINE



Long baseline:

- Harder to match
- More occlusions
- Better precision

Short baseline:

- Good matches
- Few occlusions
- Poor precision

MARS ROVER



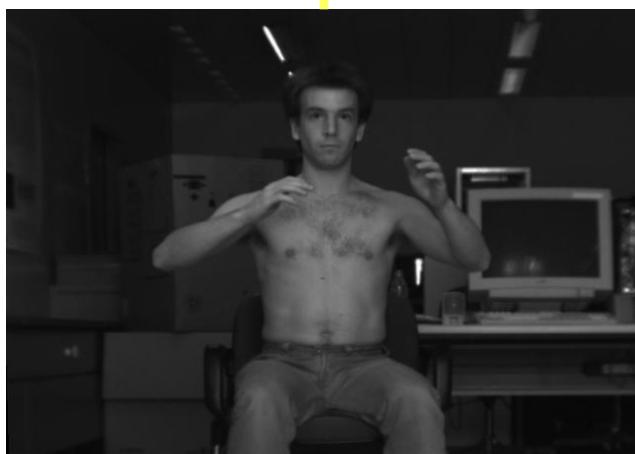
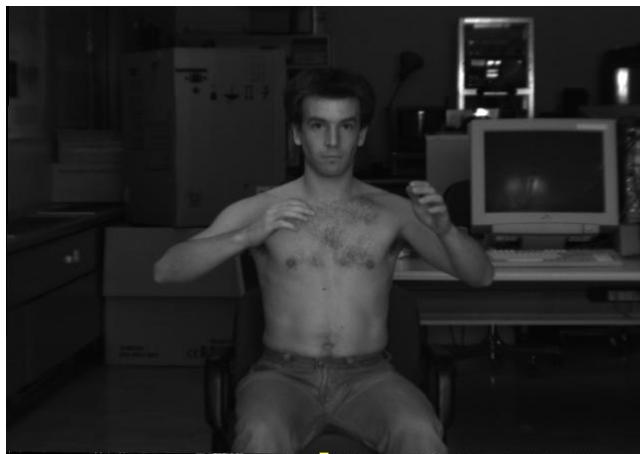
There are four cameras!

VIDEO-BASED MOTION CAPTURE

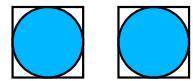


Fitting an articulated body model to stereo data.

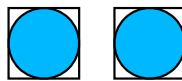
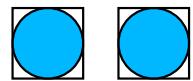
TRINOCULAR STEREO



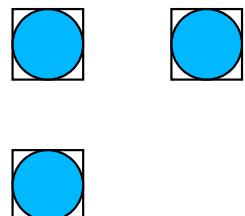
MULTI-CAMERA CONFIGURATIONS



3 cameras give both robustness and precision

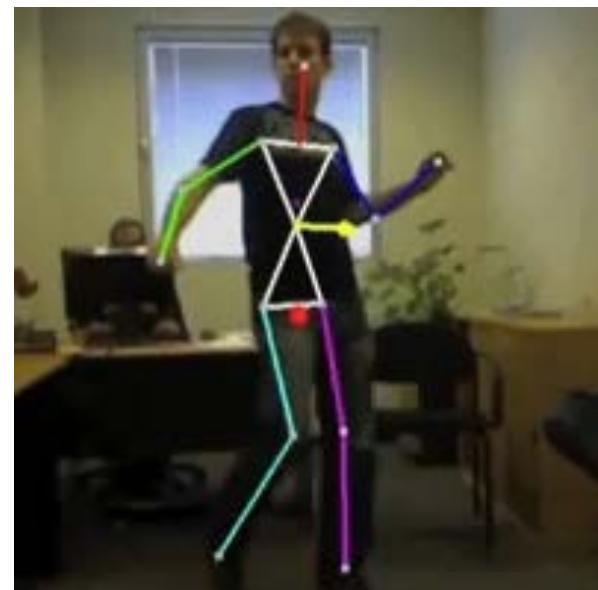


4 cameras give additional redundancy



3 cameras in a T arrangement allow the system to see vertical lines.

KINECT: STRUCTURED LIGHT



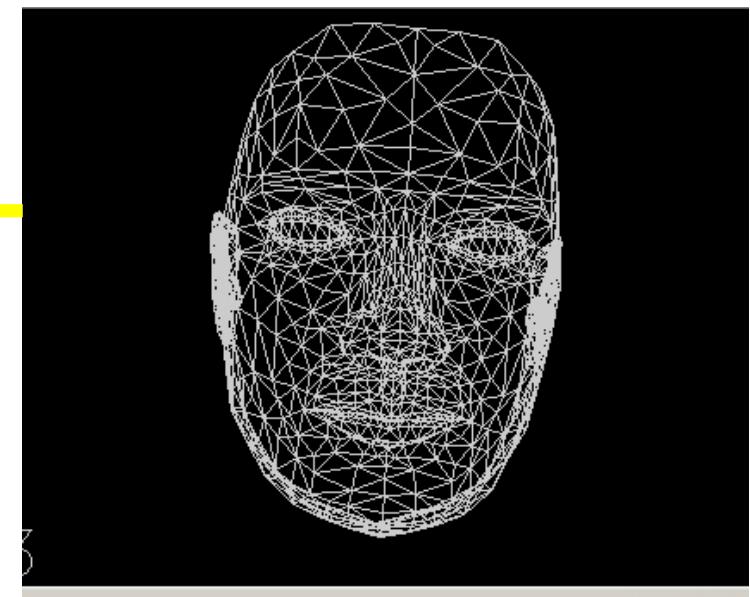
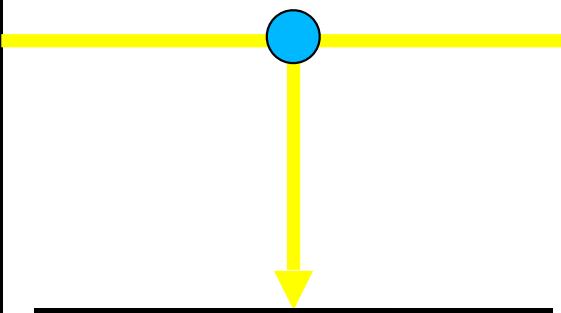
- The Kinect camera projects a IR pattern and measures depth from its distortion.
- Same principle but the second camera is replaced by the projector.

FACES FROM LOW-RESOLUTION VIDEOS

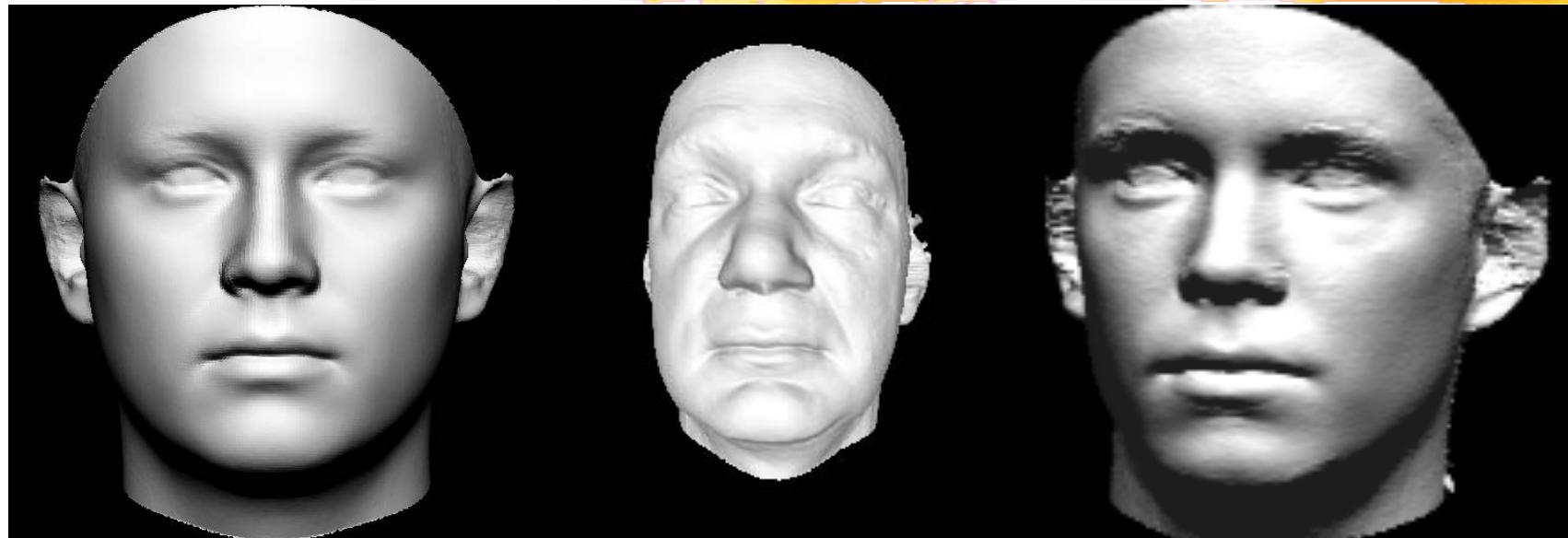


- No calibration data
- Relatively little texture
- Difficult lighting

SIMPLE FACE MODEL



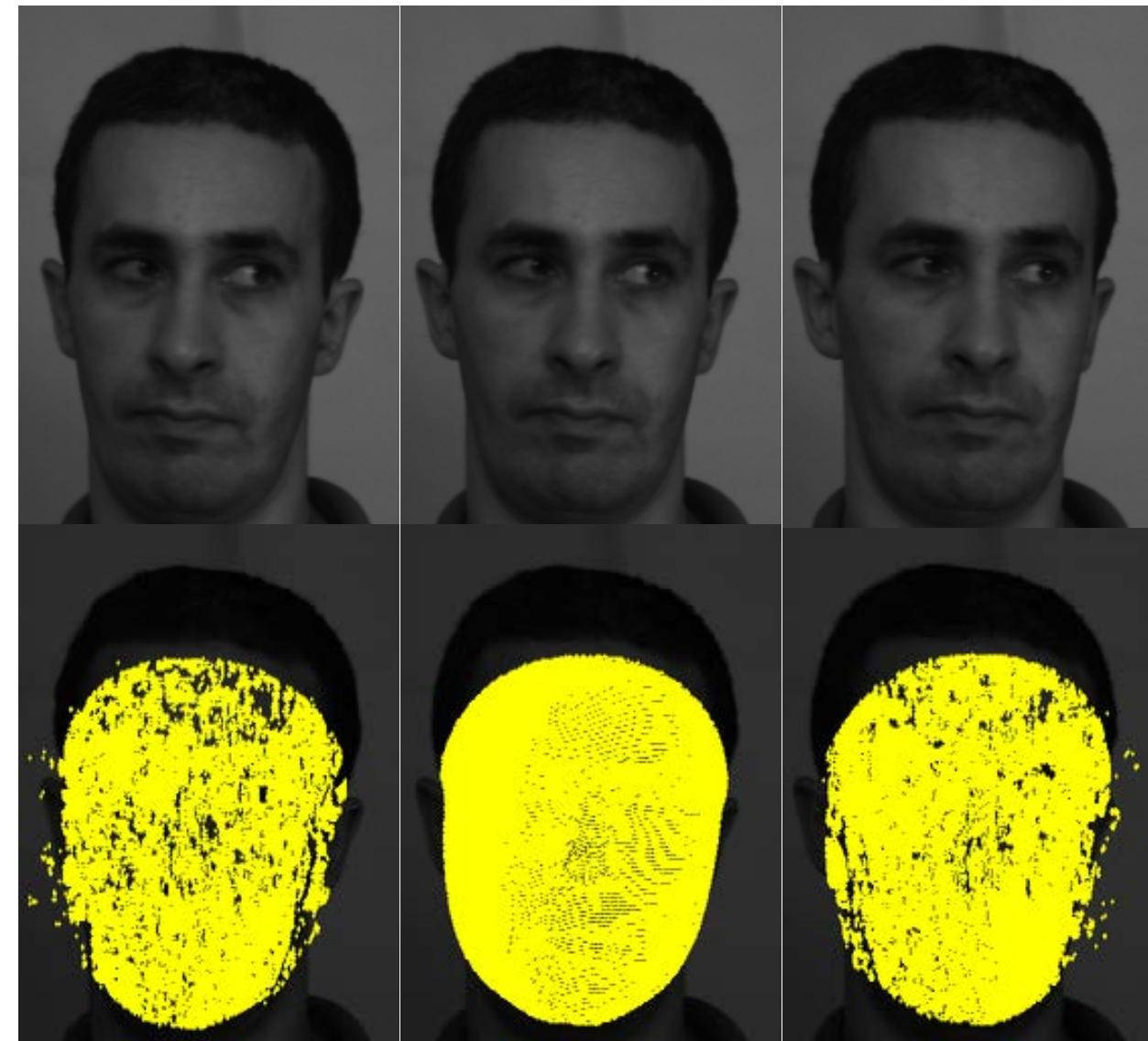
PCA FACE MODEL



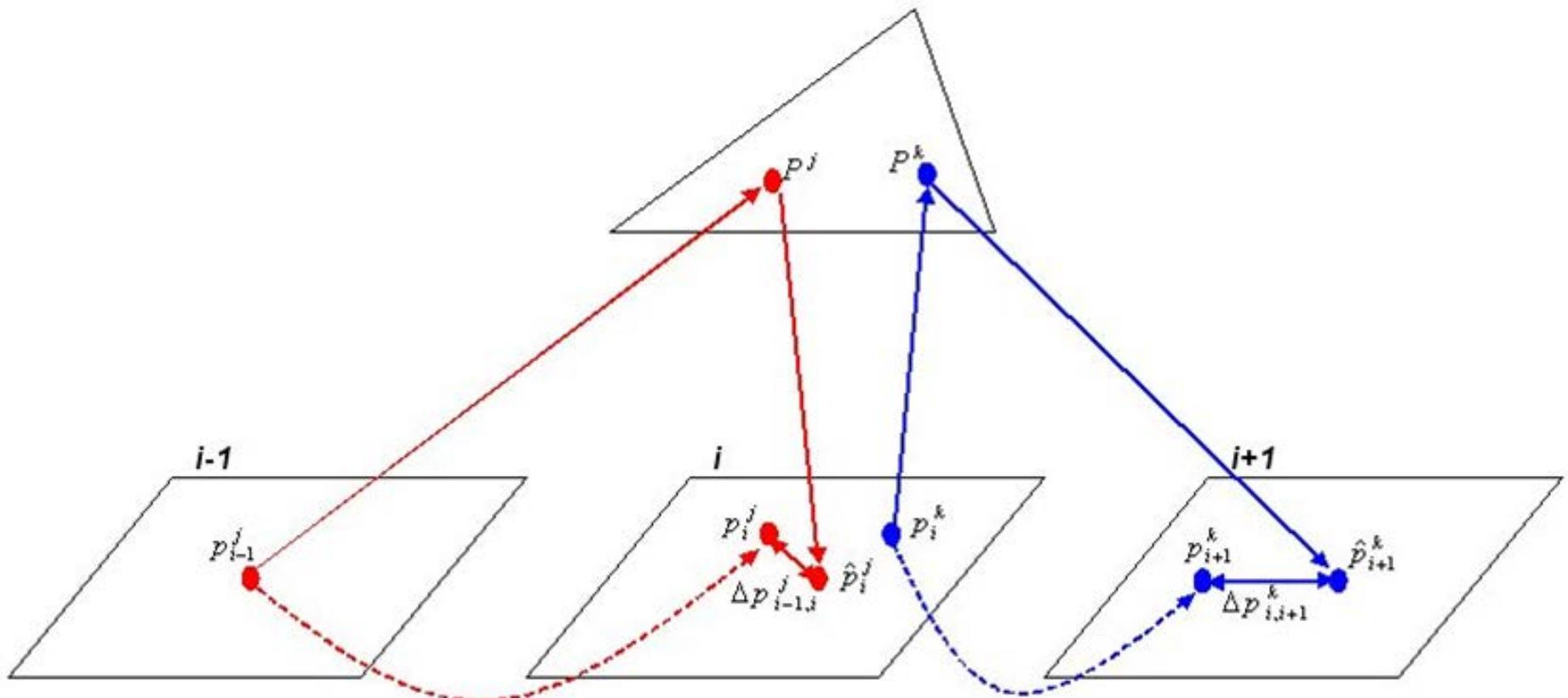
$$S = \bar{S} + \sum_{i=1}^{99} a_i S_i$$

\bar{S} : Average shape
 S_i : Shape vector
 a_i : Shape coefficients

CORRESPONDENCES

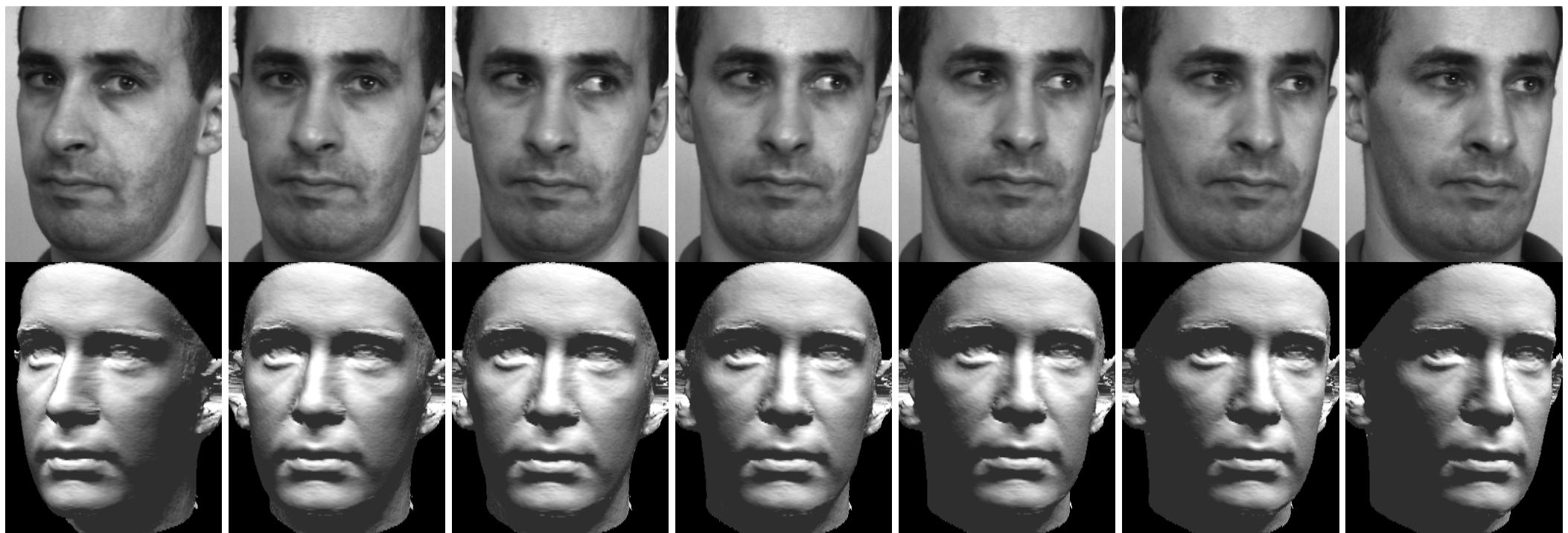


TRANSFER FUNCTION



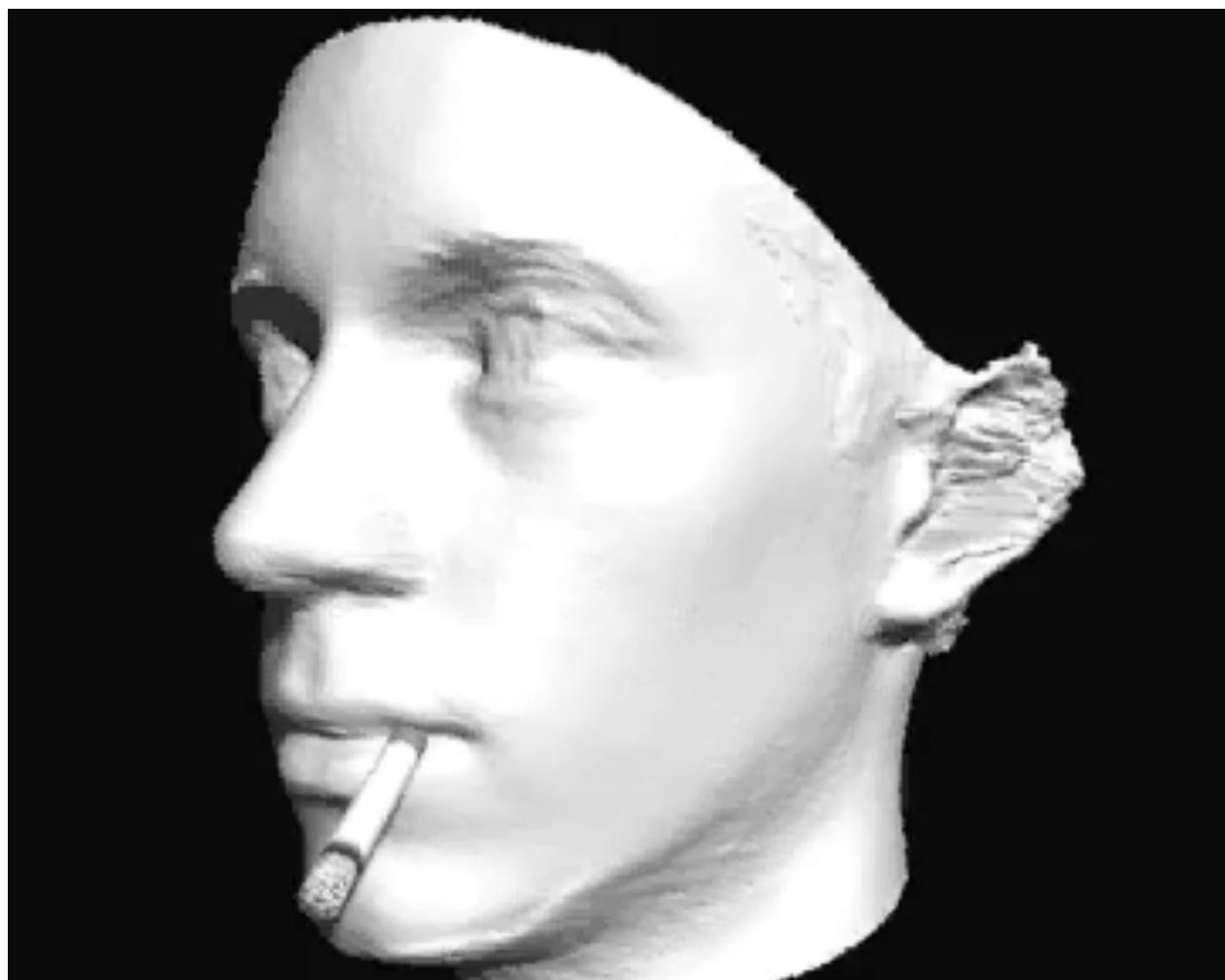
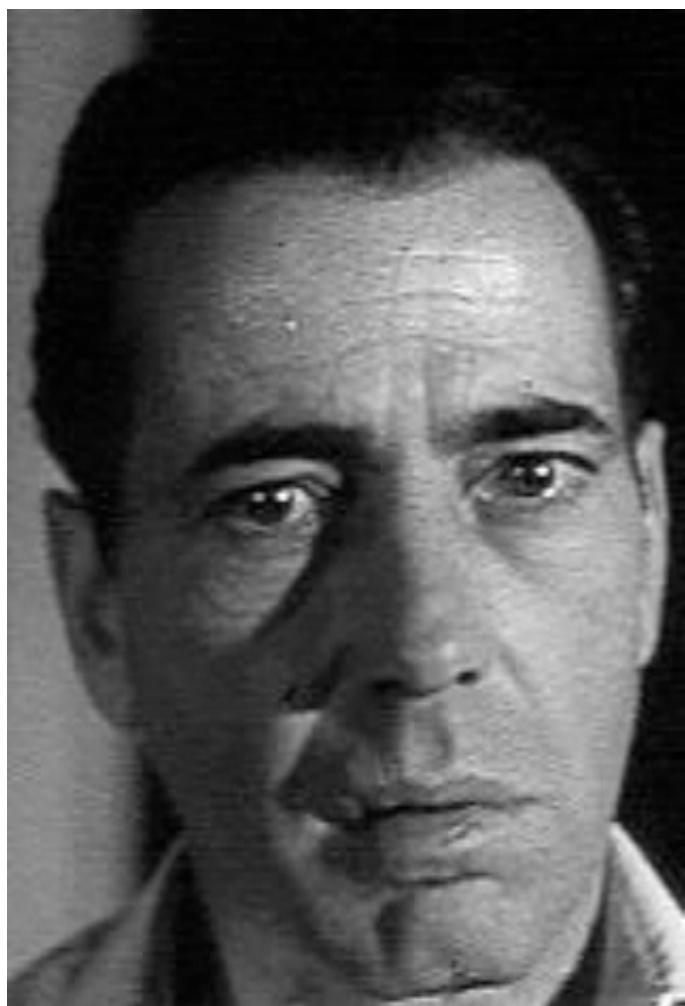
$$F_3(A, C_{i-1}, C_i, C_{i+1}) = \sum_{j \in Q_{i-1}} \left\| \Delta p_{i-1,i}^j \right\|^2 + \sum_{k \in Q_i} \left\| \Delta p_{i,i+1}^k \right\|^2$$

MODEL BASED BUNDLE ADJUSTMENT

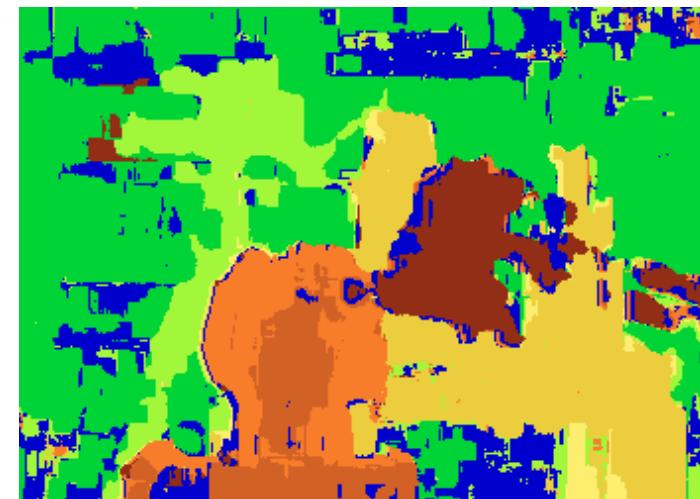


→ Median accuracy greater than 0.5mm

MODEL FROM OLD MOVIE

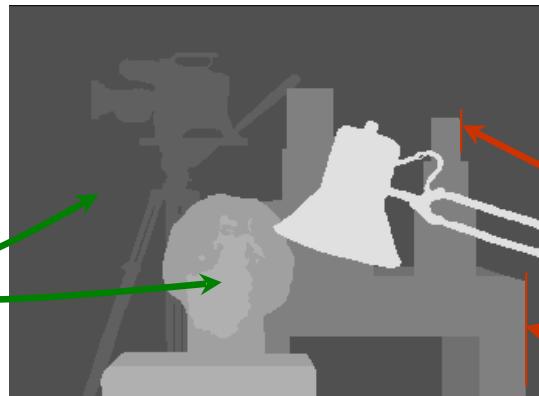


LIMITATIONS OF WINDOW BASED METHODS



ENERGY MINIMIZATION

Disparity
continuous in
most places,



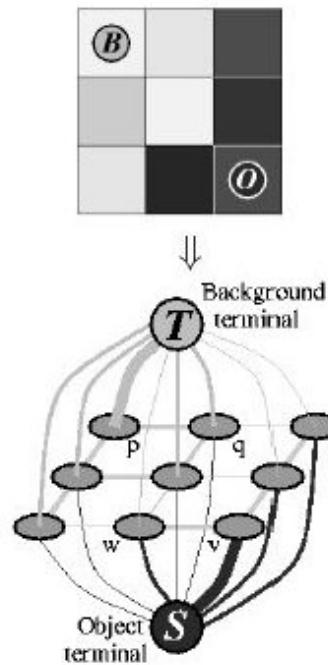
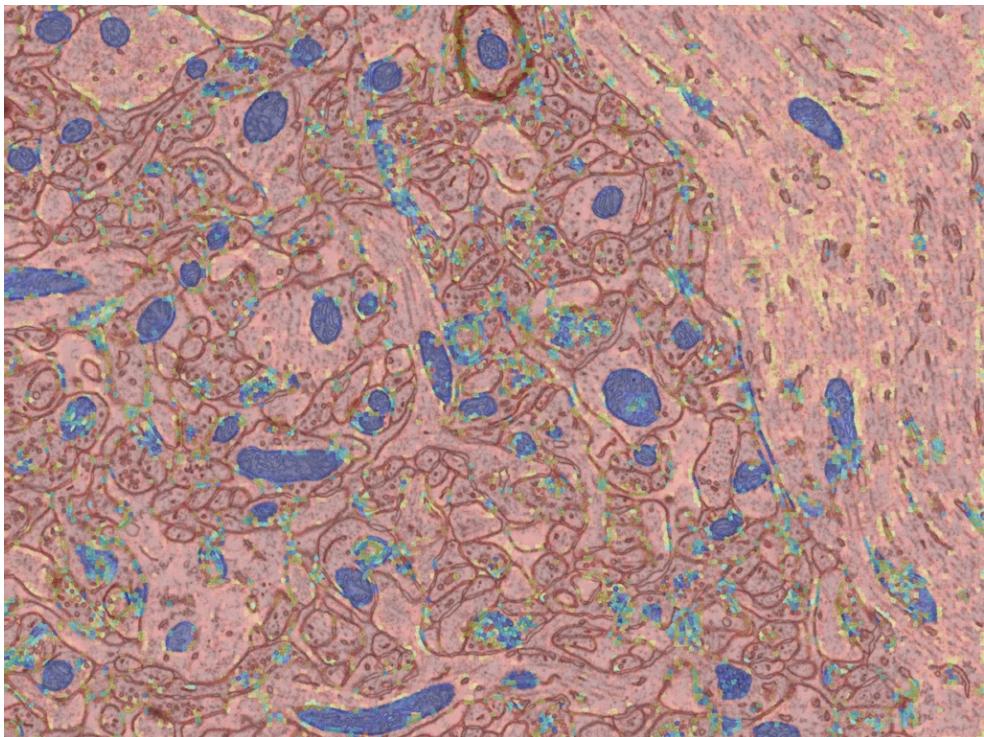
except at
depth
discontinuities

1. Matching pixels should have similar intensities.
2. Most nearby pixels should have similar disparities

→ Minimize

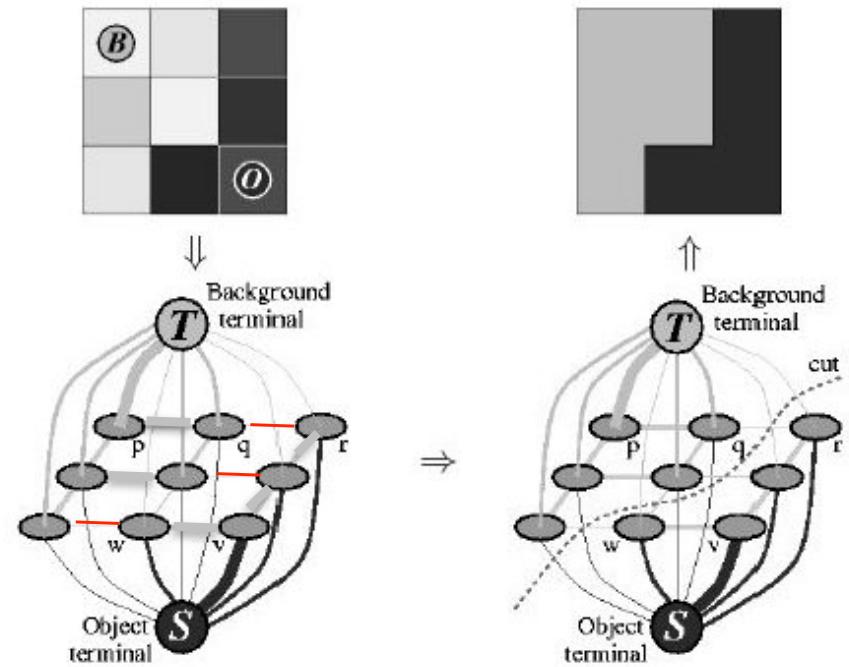
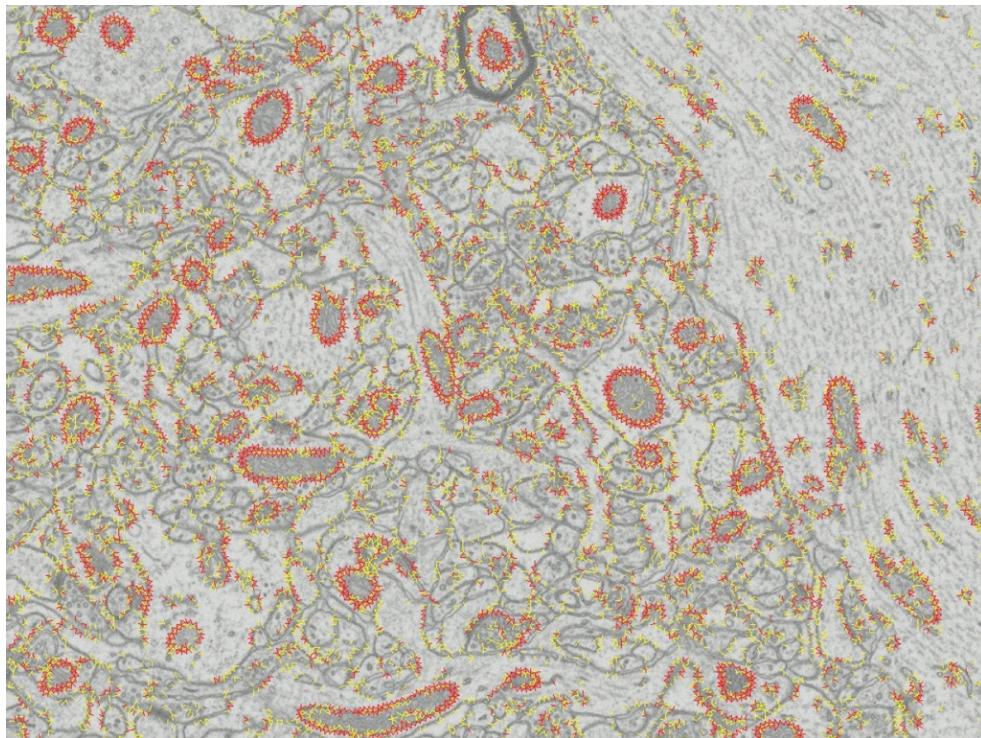
$$\sum [I_2(x+D(x,y), y) - I_1(x, y)]^2 + \lambda \sum [D(x+1, y) - D(x, y)]^2 + \mu \sum [D(x, y+1) - D(x, y)]^2$$

MITOCHONDRIA REMINDER



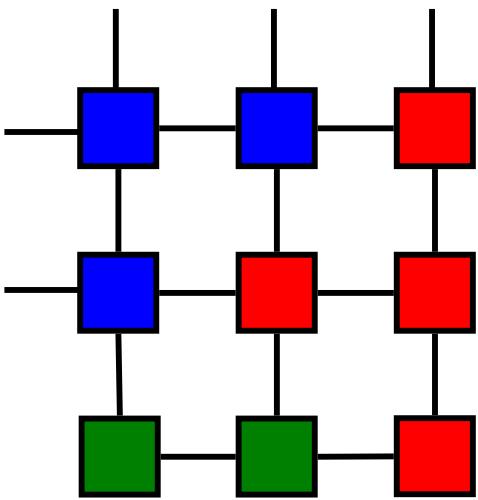
- A high probability of being a mitochondria can be represented by a strong edge connecting a supervoxel to the source and a weak one to the sink.
- And conversely for a low probability.

MITOCHONDRIA REMINDER



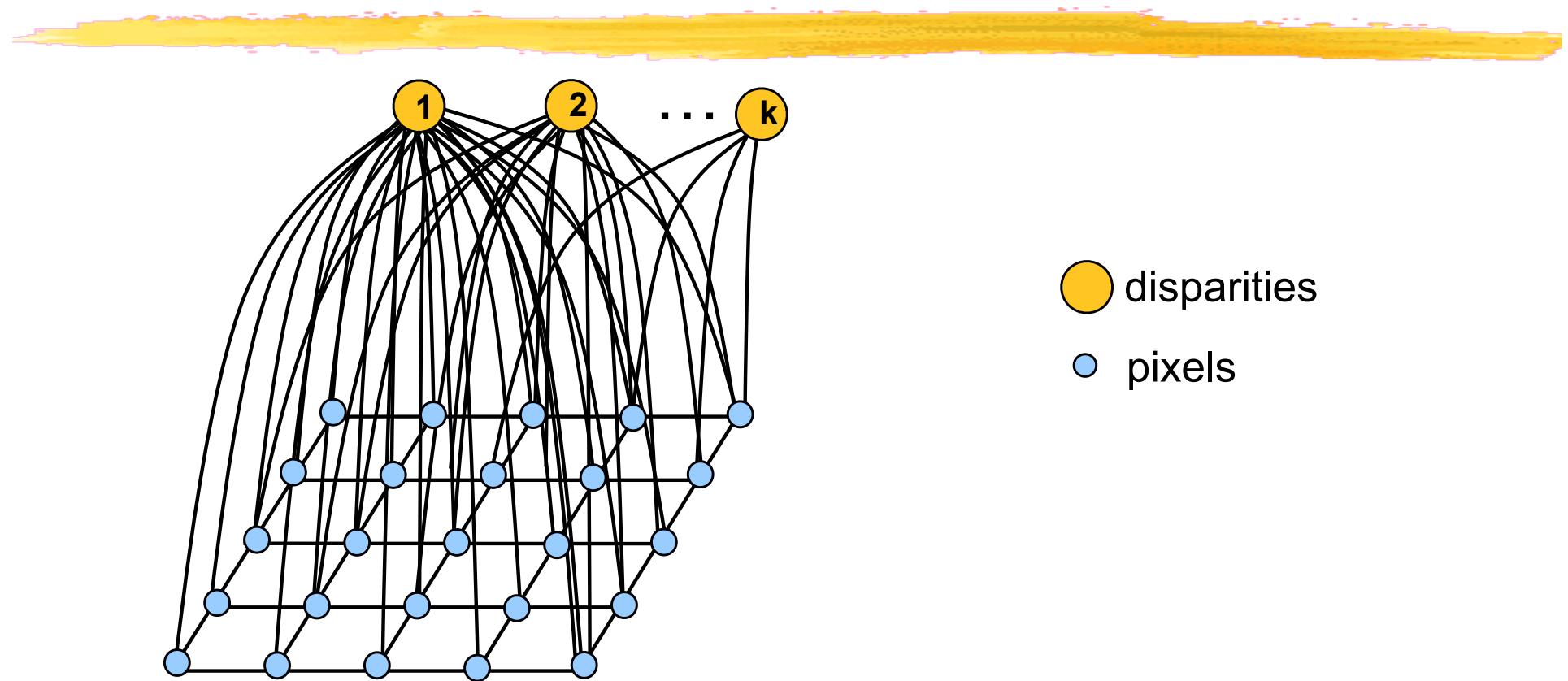
- Another classifier can be trained to assign a high-weight to edges connecting supervoxels belonging to the same class and a low one to others.

GRAPH CUTS



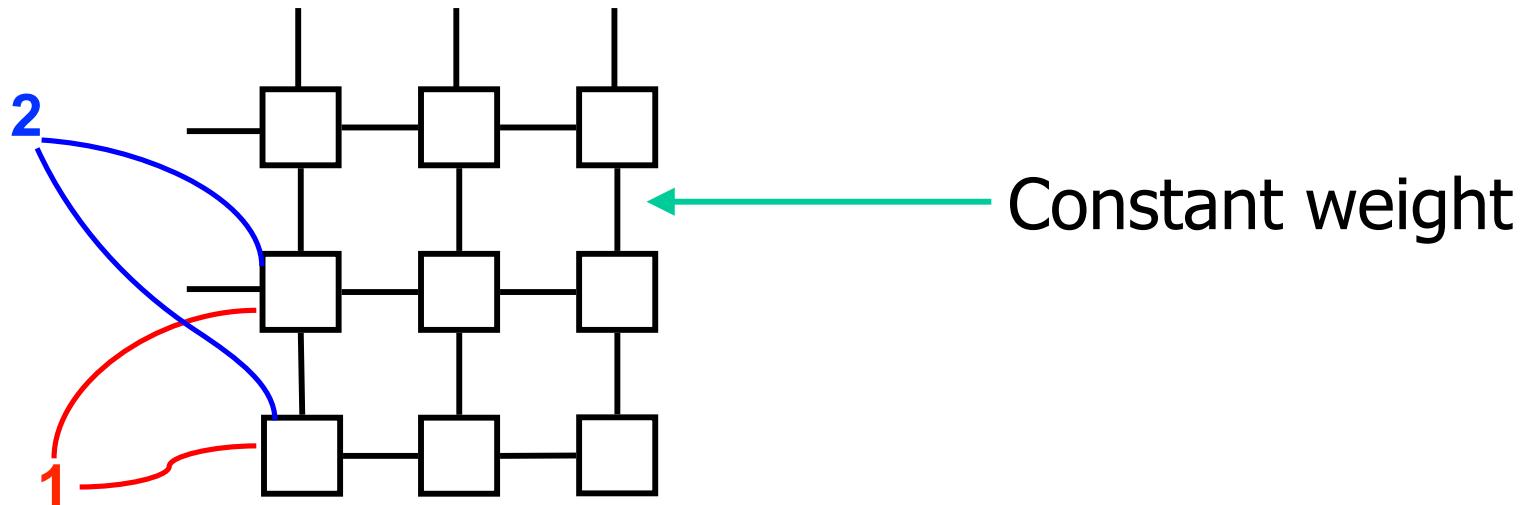
1. Stereo is a labeling problem. → Use graph cut.
2. Connect each pixel to each possible disparity value.

Building The Graph



Connect each pixel to each possible disparity value.

Assigning Edge Weights



Assign a **weight** that is inversely proportional to $|I_2(x+1,y) - I_1(x,y)|$
Assign a **weight** that is inversely proportional to $|I_2(x+2,y) - I_1(x,y)|$

.....

Minimizing the Objective Function

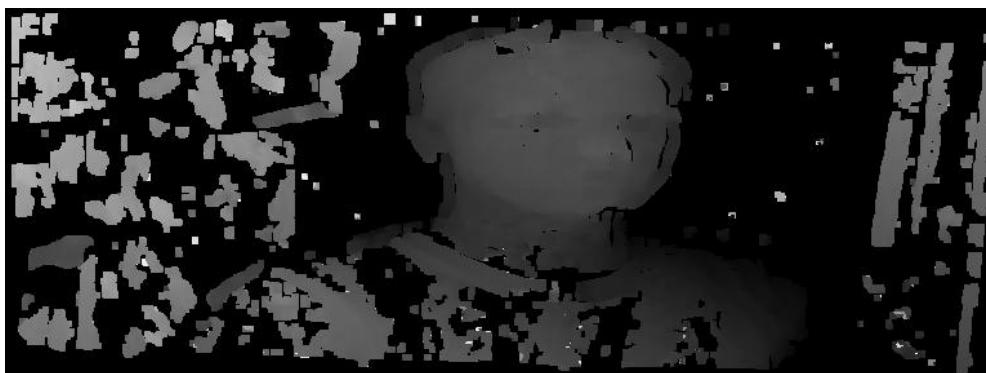
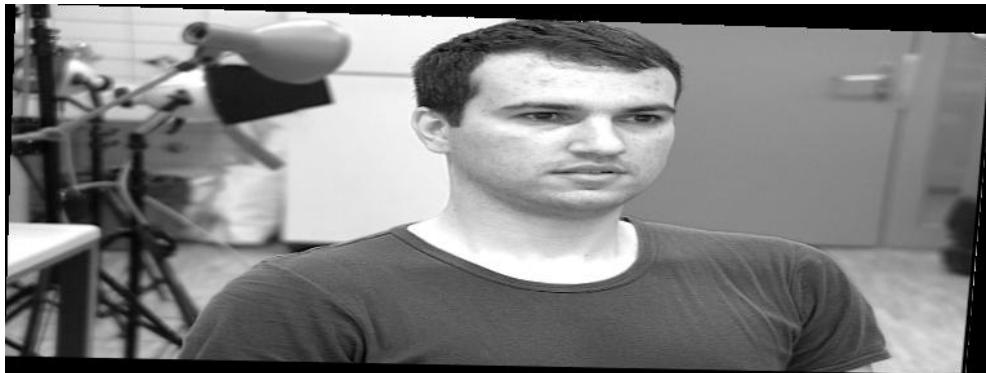
Minimize:

$$\sum [I_2(x+D(x, y), y) - I_1(x, y)]^2 + \lambda \sum [D(x+1, y) - D(x, y)]^2 + \mu \sum [D(x, y+1) - D(x, y)]^2$$

Graph cut algorithm:

- Guarantees an absolute minimum only when there are only two possible disparities.
- Effective heuristics (α -expansion, $\alpha-\beta$ swap) otherwise.

NCC vs GRAPH CUTS

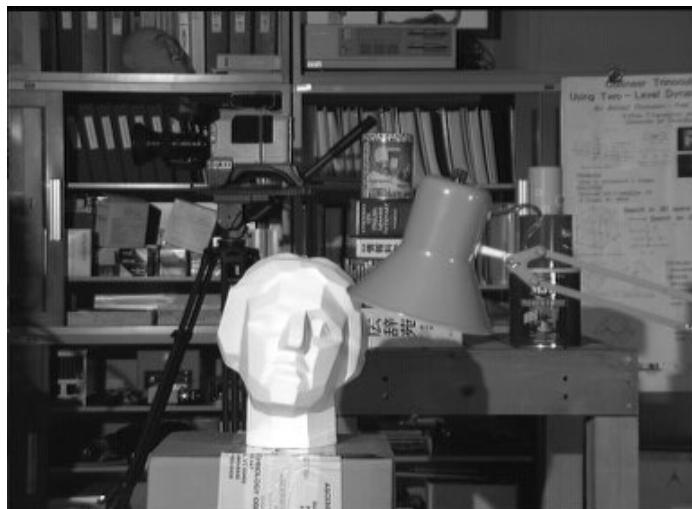


Normalized correlation

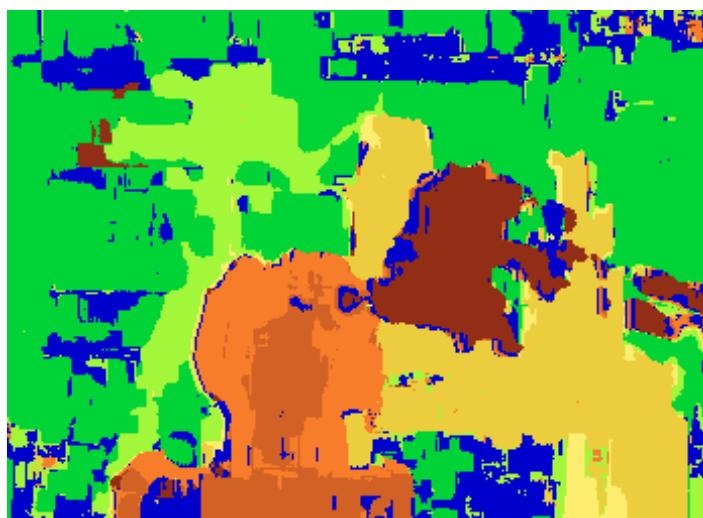
Graph Cuts

NCC vs GRAPH CUTS

left image



Normalized correlation



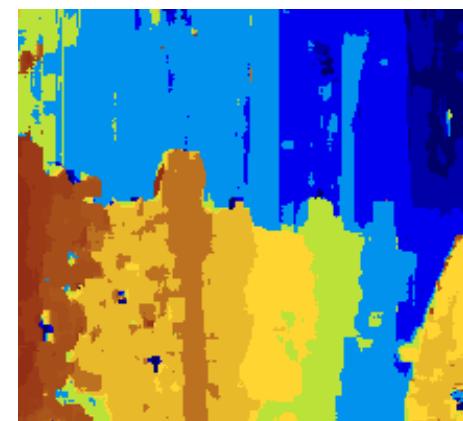
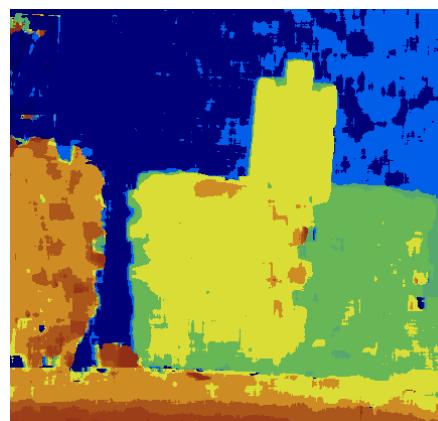
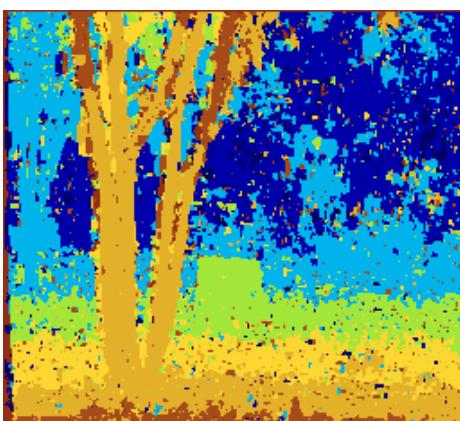
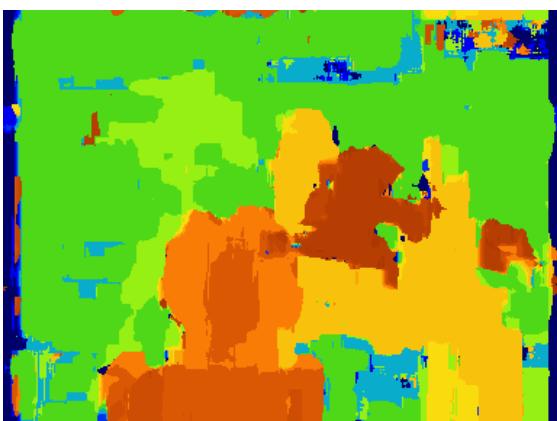
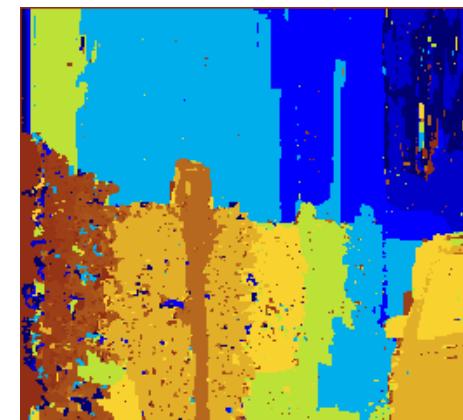
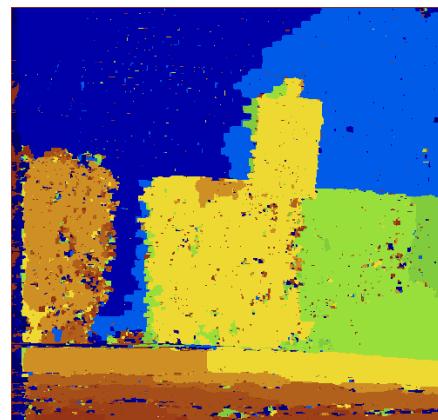
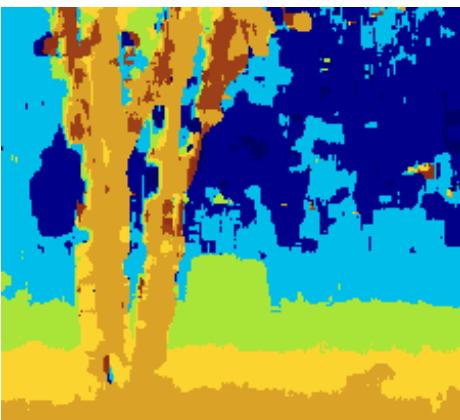
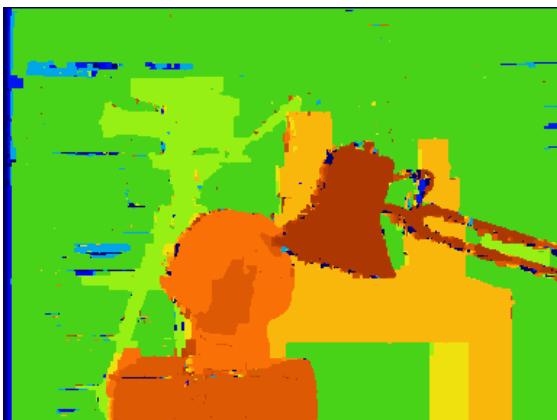
true disparities



Graph Cuts



GRAPH CUT RESULTS



STRENGTHS AND LIMITATIONS



Strengths:

- Practical method for depth recovery.
- Runs in real-time on ordinary hardware.

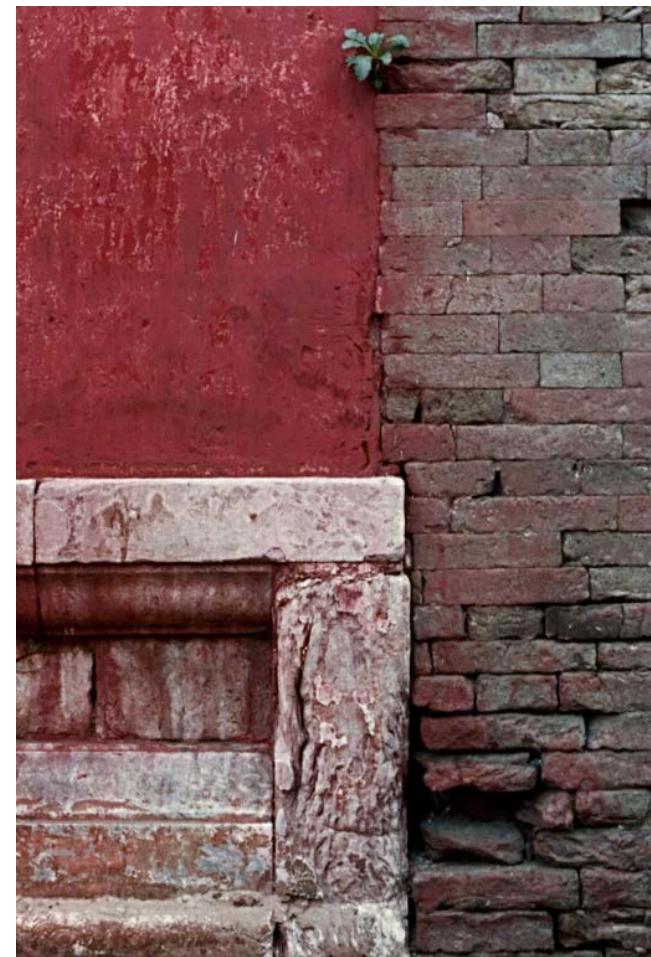
Limitations:

- Requires multiple views.
- Only applicable to reasonably textured objects.

TEXTURE



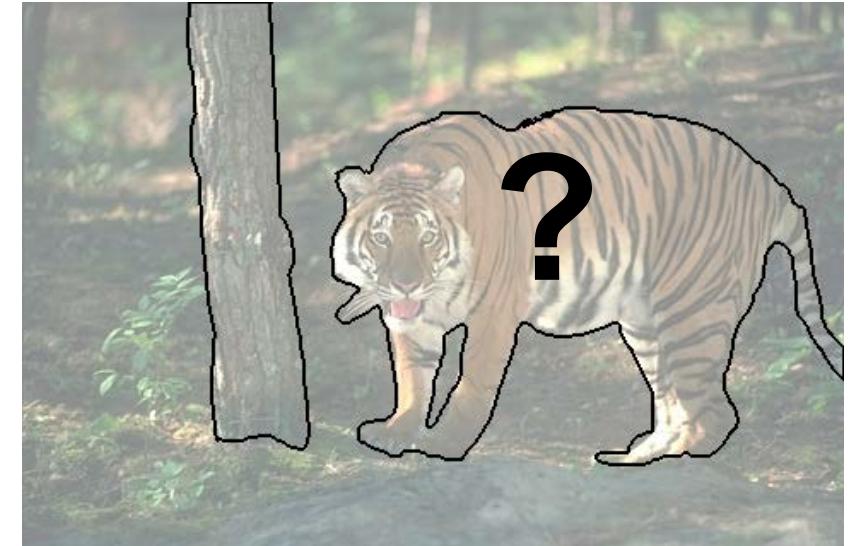
- What is texture?
- Texture analysis
- Deep Texture



HOMOGENEOUS OR NOT?

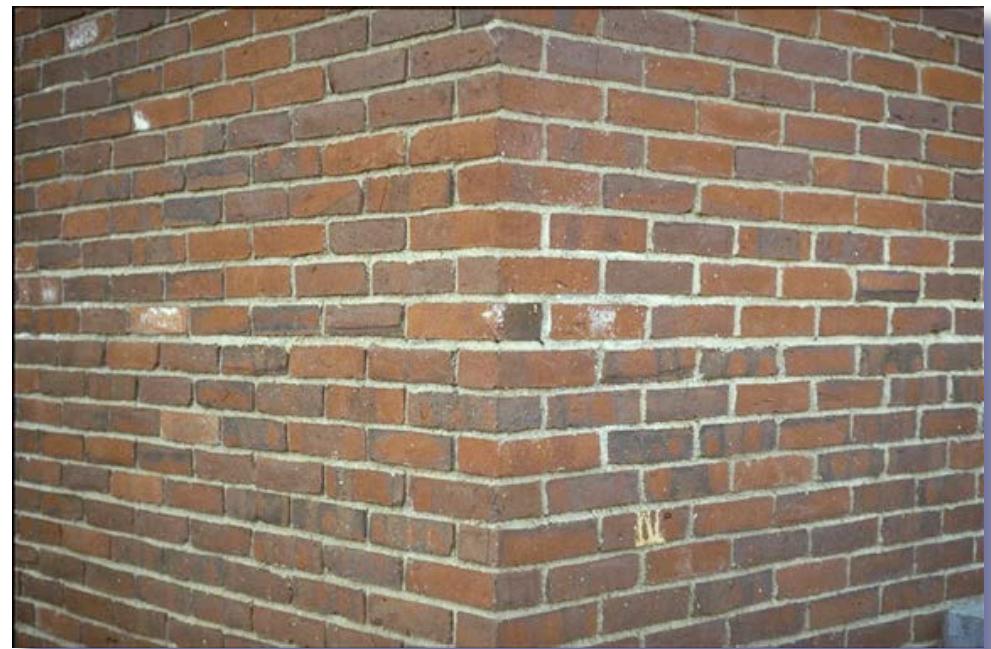


TEXTURAL IMAGES



- Assign to individual pixels whose texture is similar the same values to form a textural image.
- Evaluate homogeneity both in the original image and in the textural one.

TEXTURE BOUNDARIES



WHAT IS TEXTURE?

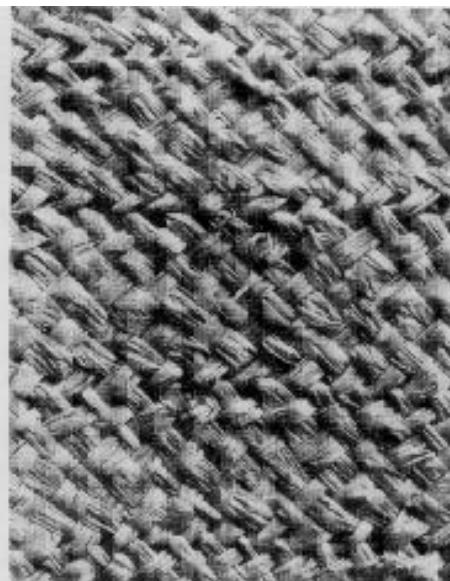
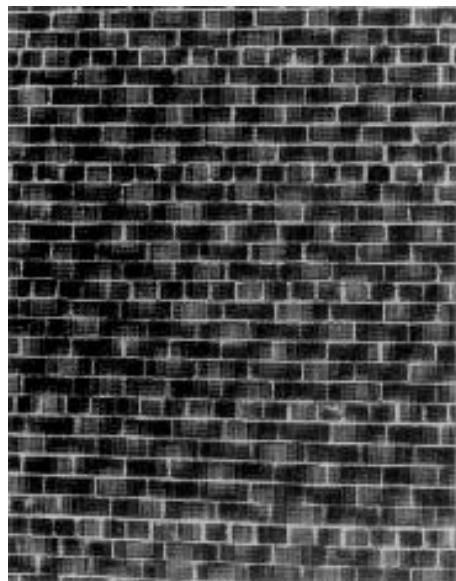


Repetition of a basic pattern:

- Structural
- Statistical

→ Non local property, subject to distortions.

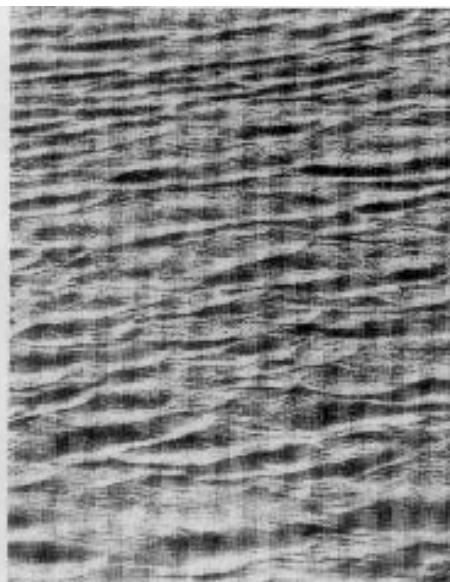
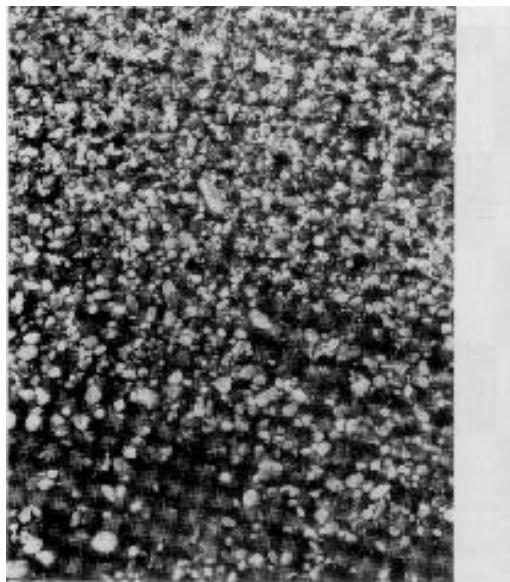
STRUCTURAL TEXTURES



Repetitive Texture Elements (Texels)

A texel represents the smallest graphical element in a two-dimensional texture that creates the impression of a textured surface.

STATISTICAL TEXTURES

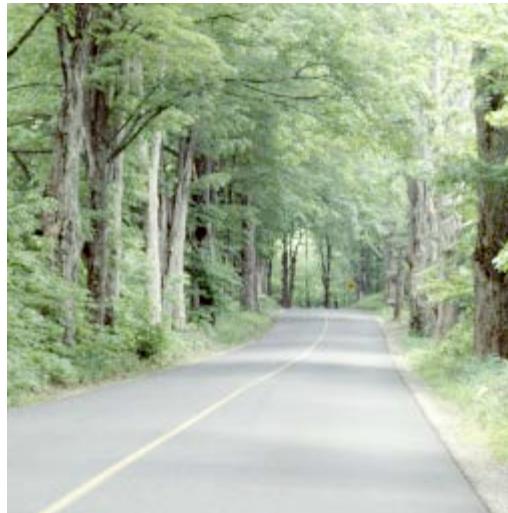


Homogeneous Statistical Properties

STRUCTURAL vs STATISTICAL



Segmenting out texels is difficult or impossible in most real images.



What are the fundamental texture primitives in this image?

Numeric quantities or statistics that describe a texture can be computed from the gray levels or colors alone.
→ This approach is less intuitive, but computationally more efficient.

TEXTURED vs SMOOTH



A “featureless” surface can be regarded as the most elementary spatial texture:

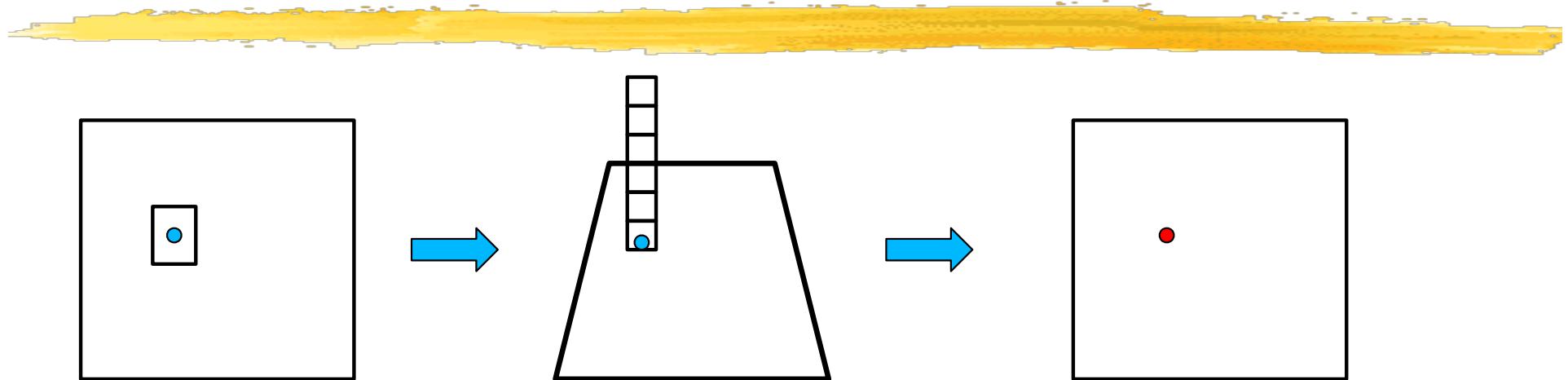
- Microstructures define reflectance properties.
- They may be uniform or smoothly varying.

→ Texture is a scale dependent phenomenon

SCALE DEPENDENCE



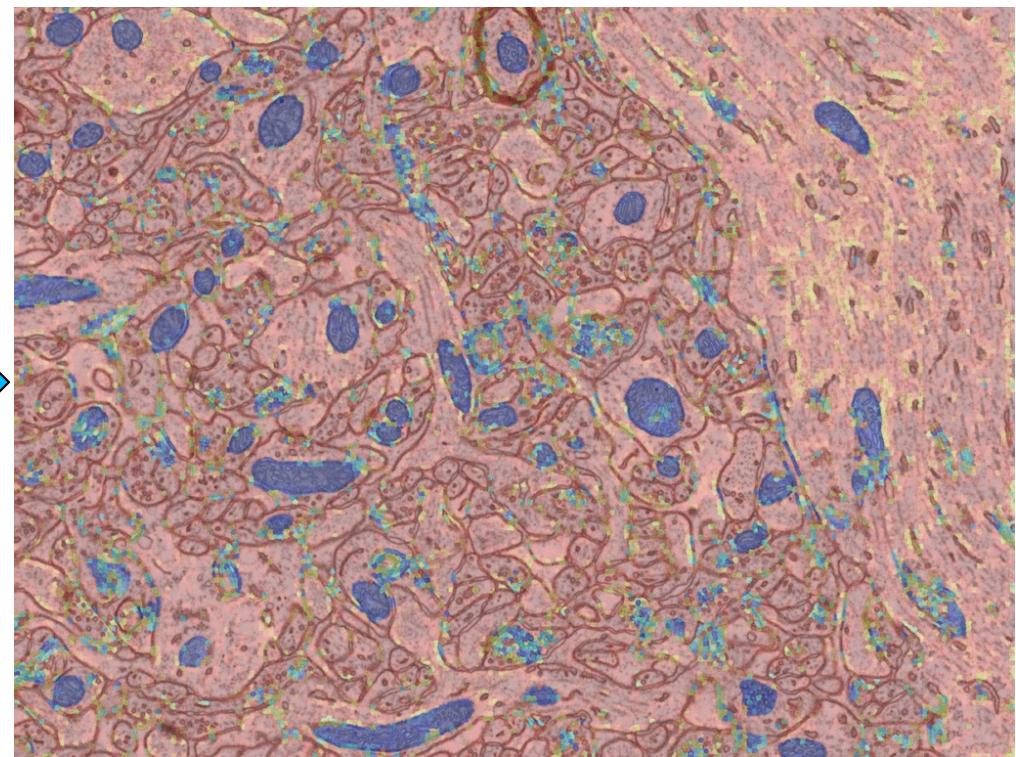
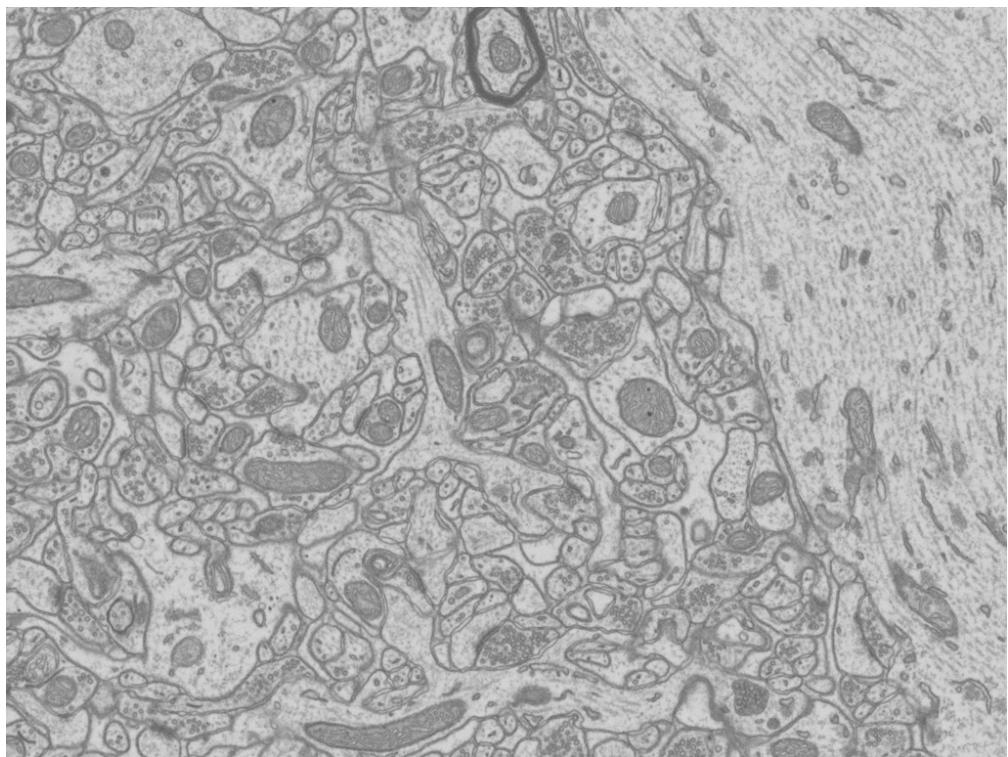
CREATING TEXTURAL IMAGES



Because texture is non-local, the texture of individual pixels must be estimated using neighborhoods surrounding them:

- For each pixel, compute a feature vector using either an image patch or a set of filters.
- Run a classification algorithm to assign a texture value to each pixel.

MITOCHONDRIA



PATCH-BASED MEASURES



Spectral metrics:

- Texture characterized by properties of its Fourier transform.

Statistical Metrics:

- Texture is a quantitative measure of the arrangement of intensities in a region.
- Can be modeled as a Markov Process.

DISCRETE FOURIER TRANSFORM

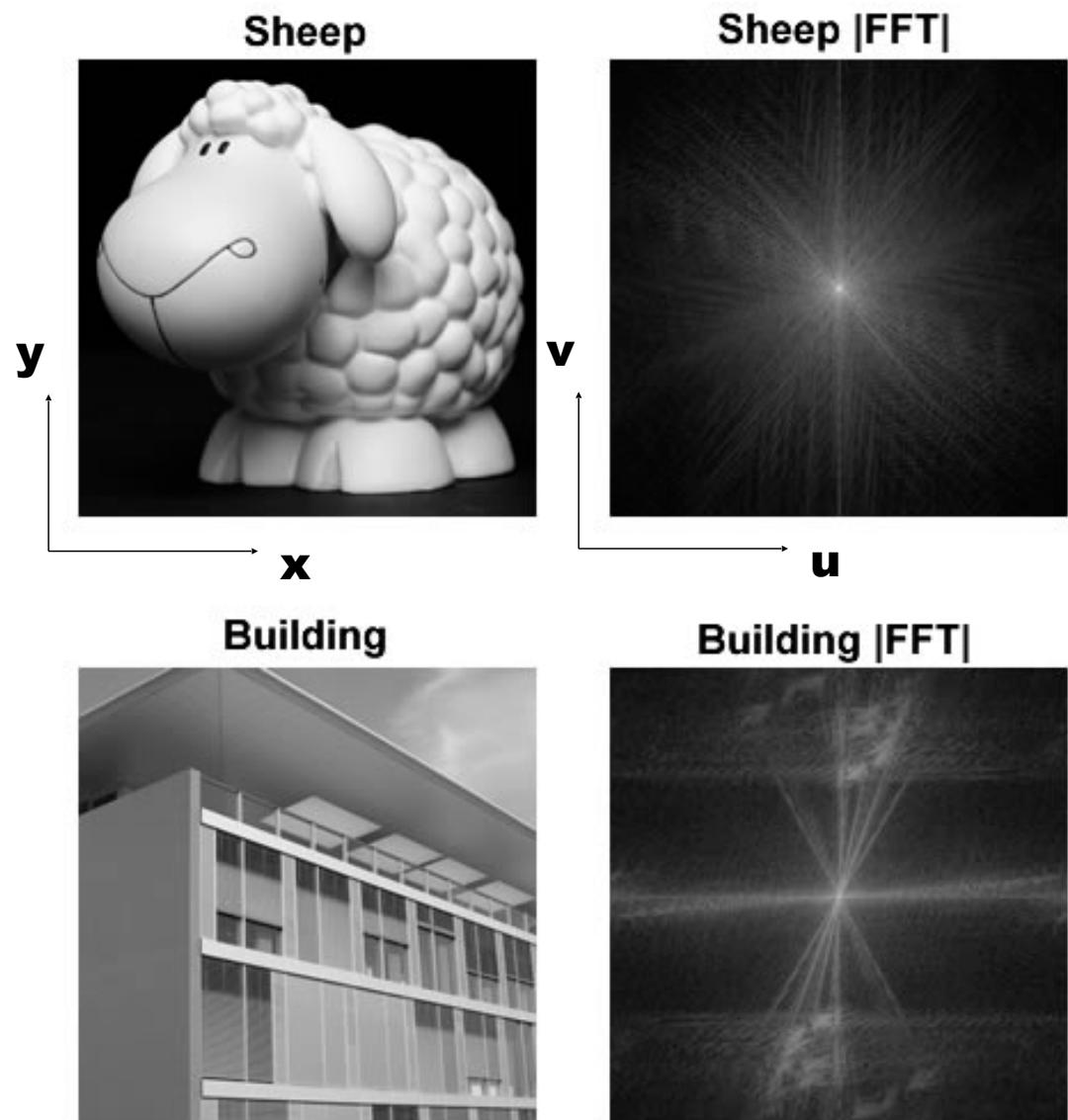
$$F(\mu, \nu) = \frac{1}{MN} \sum_{x=0}^{M-1} \sum_{y=0}^{N-1} f(x, y) \exp(-2i\pi(\mu x/M + \nu y/N))$$

$$f(x, y) = \frac{1}{MN} \sum_{\mu=0}^{M-1} \sum_{\nu=0}^{N-1} F(\mu, \nu) \exp(+2i\pi(\mu x/M + \nu y/N))$$

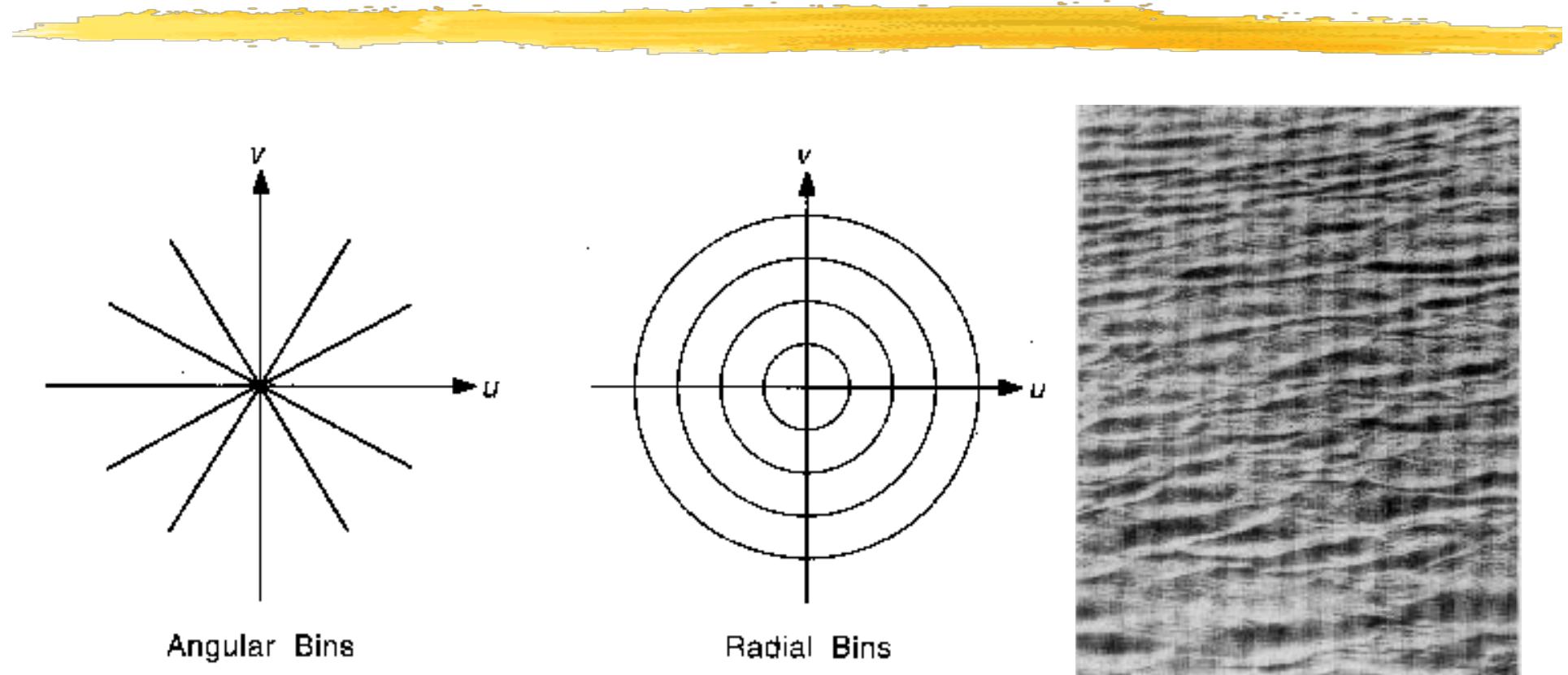
- The DFT of $f * g$ is the product of the DFT of f with the DFT of g .
- The DFT of a symmetric function is real.

SPECTRAL ANALYSIS

The magnitude of the DFT captures the main orientations in the image.

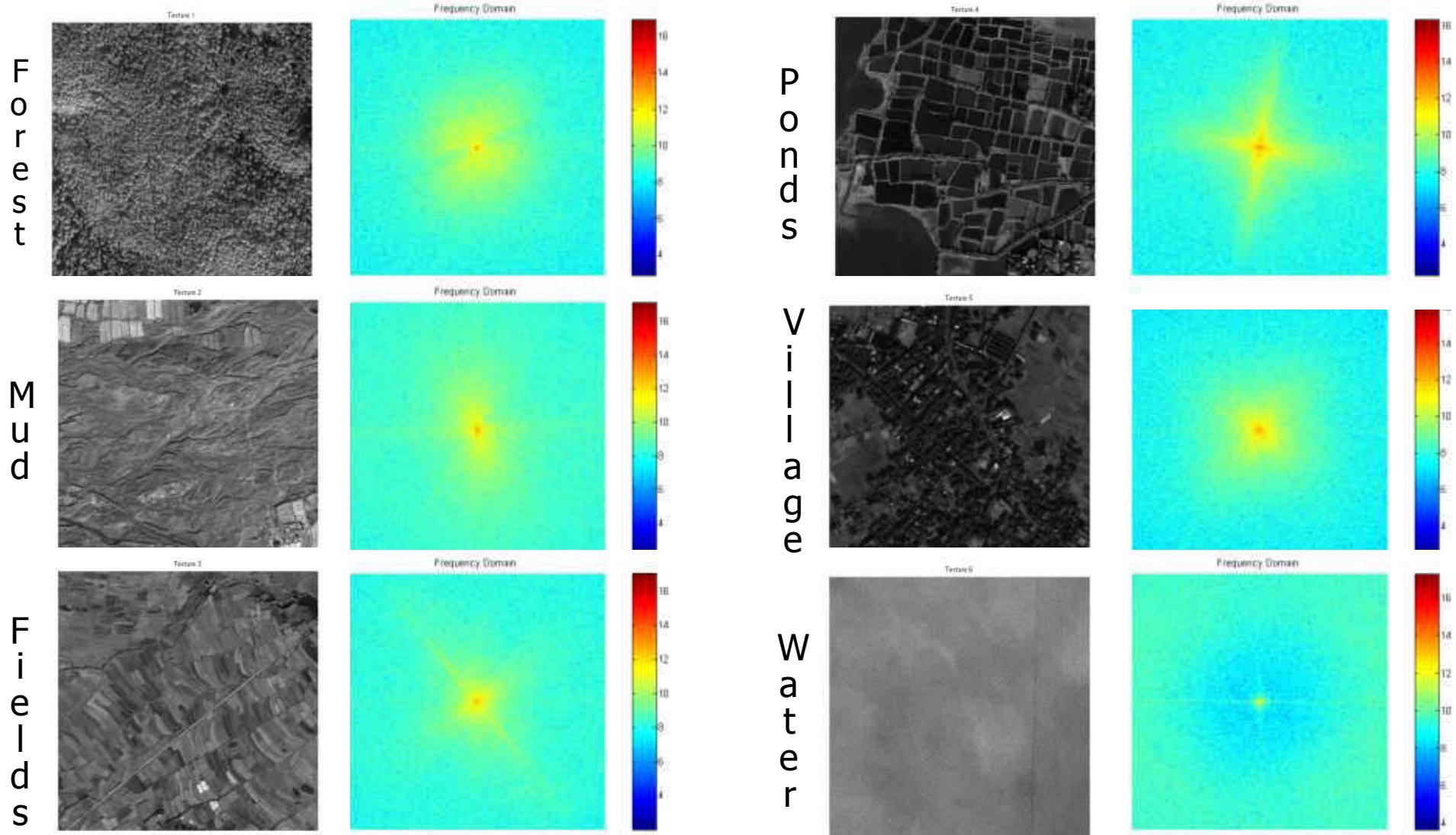


TEXTURE ANALYSIS



Angular and radial bins in the Fourier domain capture the directionality and rapidity of fluctuation of an image texture.

TEXTURE CLASSIFICATION



LIMITATIONS



- DFT on small patches is subject to severe boundary effects.
- Only applicable if texture is uniform over large areas.
- Improved results by using wavelets instead.

STATISTICAL METRICS



Most natural textures are best modeled using such methods.

First order gray-level statistics:

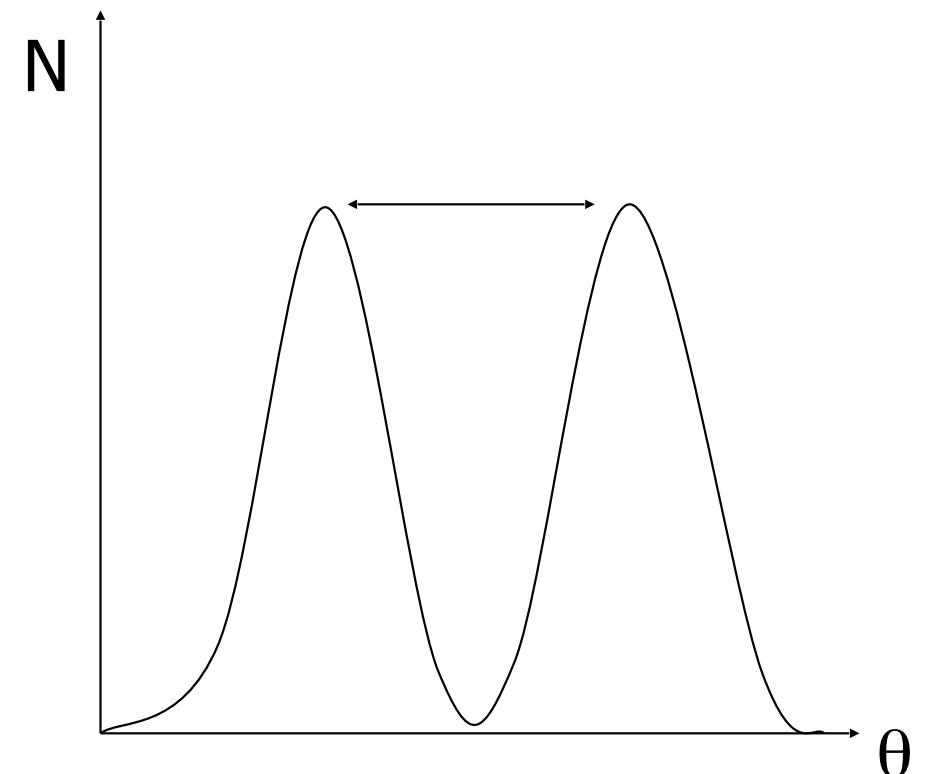
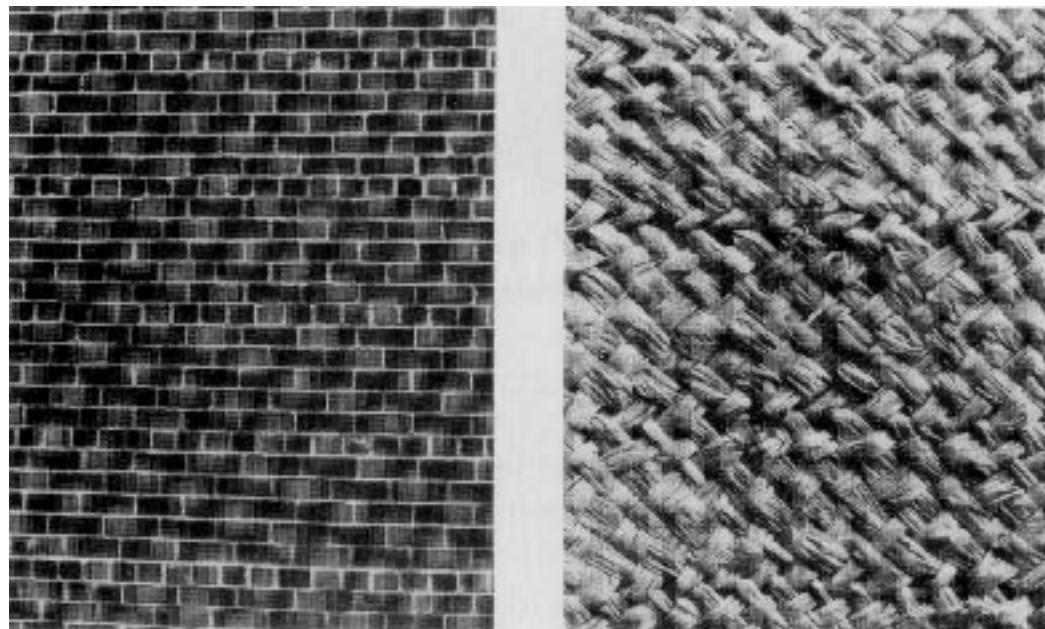
- Statistics of single pixels in terms of histograms.
- Insensitive to neighborhood relationships.

Second order gray-level statistics:

$$P(l, m, \Delta i, \Delta j) : P(i, j) = l \text{ and } P(i + \Delta i, j + \Delta j) = m$$

Given g gray-levels, for each $(\Delta i, \Delta j)$, P is represented as the $g \times g$ matrix H.

FIRST ORDER TEXTURE



Orientation histogram gives a clue to the orientation of the underlying plane.

FIRST ORDER MEASURES



Edge Density and Direction

- Edge detection as a first step in texture analysis.
- The number of edge pixels in a fixed-size region tells us how busy that region is.
- The directions of the edges also help characterize the texture

Edgeness per unit area

- $\{ p \mid \text{gradient_magnitude}(p) \geq \text{threshold} \} / N$ where N is the unit area or region.

Edge magnitude and direction histograms

- $F_{\text{magdir}} = (H_{\text{magnitude}}, H_{\text{direction}})$

SECOND ORDER MEASURES



Histogram of the co-occurrence of particular intensity values in the image.

- Specified in terms of geometric relationships between pixel pairs:
 - Distance
 - Orientation
- $P(i,j,d,\theta)$ Frequency with which a pixel with value j occurs at distance d and orientation θ from a pixel with value i .

SIMPLE EXAMPLE

If $I = \begin{bmatrix} 0 & 1 & 2 & 3 & 1 \\ 2 & 1 & 3 & 1 & 1 \\ 0 & 0 & 2 & 2 & 1 \\ 1 & 2 & 0 & 3 & 1 \\ 0 & 0 & 0 & 0 & 3 \end{bmatrix}$,

then $H = \begin{bmatrix} 4 & 1 & 1 & 2 \\ 0 & 1 & 2 & 1 \\ 1 & 2 & 1 & 1 \\ 0 & 3 & 0 & 0 \end{bmatrix}$,

and $P(l,m,1,0) = \frac{H(l,m)}{20}$.

CO-OCCURRENCE MATRIX



No need to distinguish between

$$P(m, l, \Delta i, \Delta j)$$

and

$$P(l, m, \Delta i, \Delta j)$$

→ Co-Occurrence matrix C:

$$C = H + H^T$$

2ND ORDER TEXTURE MEASURES



Contrast:

$$\sum_{i,j=0}^{N-1} P_{i,j}(i - j)^2$$

Dissimilarity:

$$\sum_{i,j=0}^{N-1} P_{i,j}|i - j|$$

Homogeneity:

$$\sum_{i,j=0}^{N-1} \frac{P_{i,j}}{1 + (i - j)^2}$$

Angular Second Moment (Energy, Uniformity):

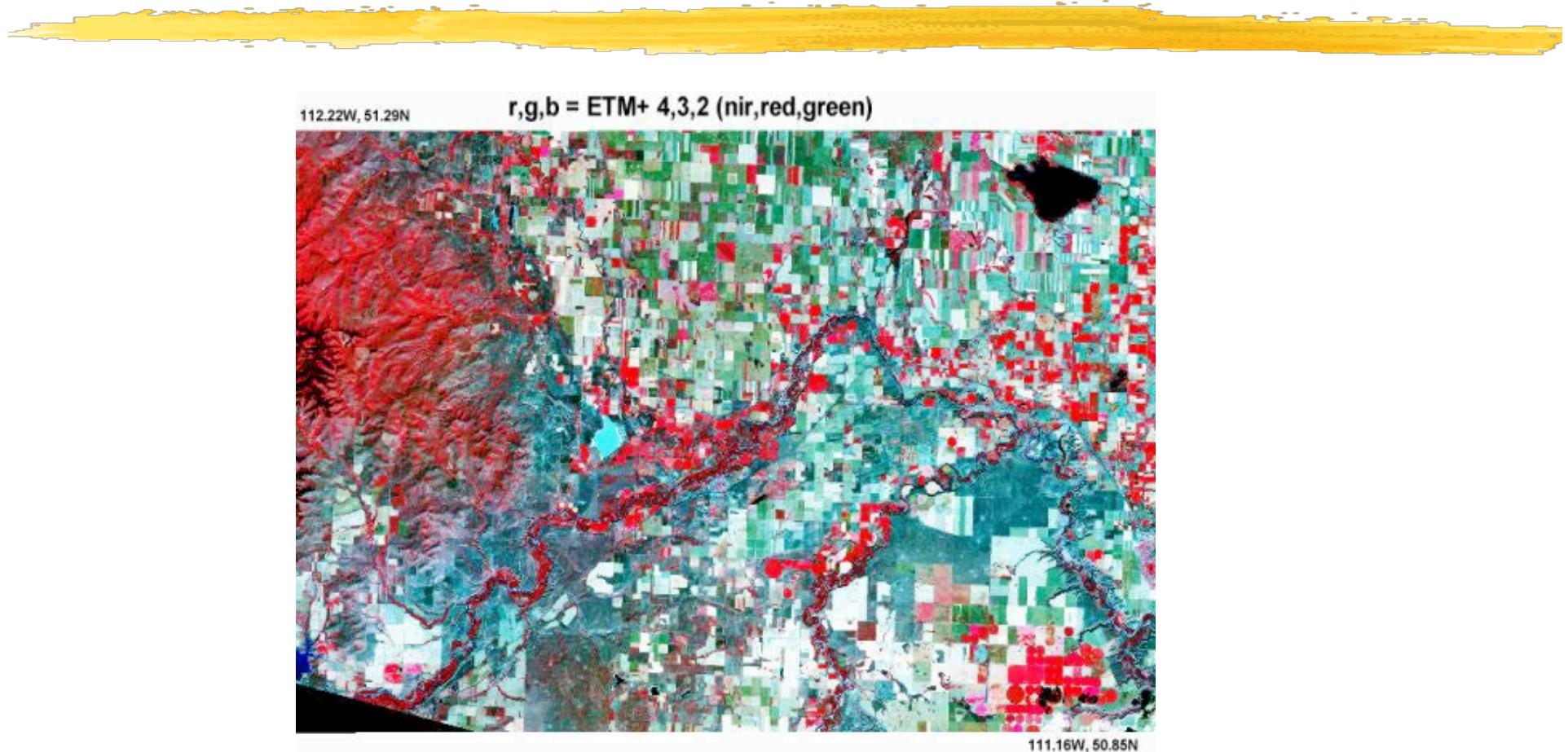
$$\sum_{i,j=0}^{N-1} P_{i,j}^2$$

and many more

Entropy:

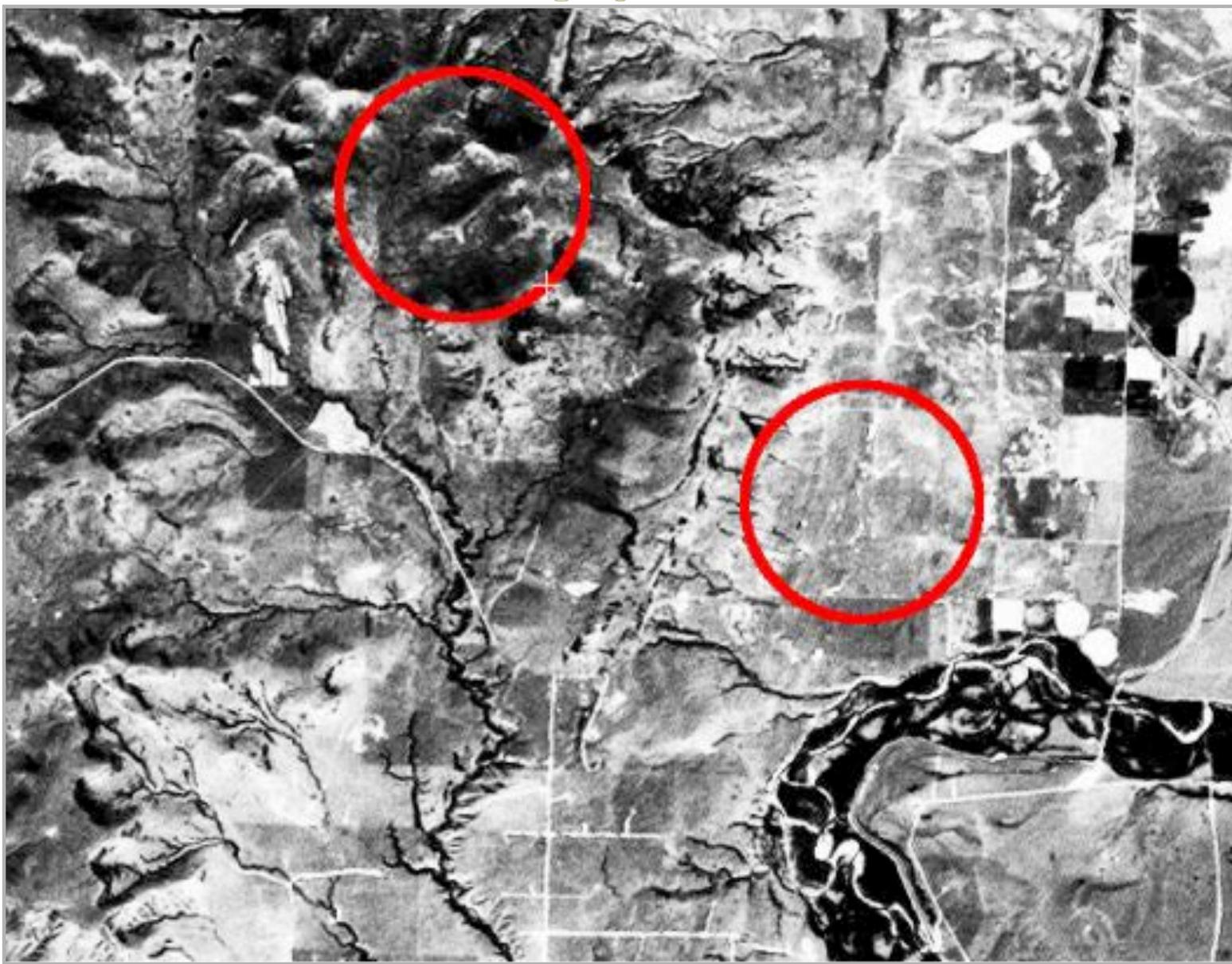
$$\sum_{i,j=0}^{N-1} P_{i,j}(-\ln P_{i,j})$$

LANDSAT IMAGE



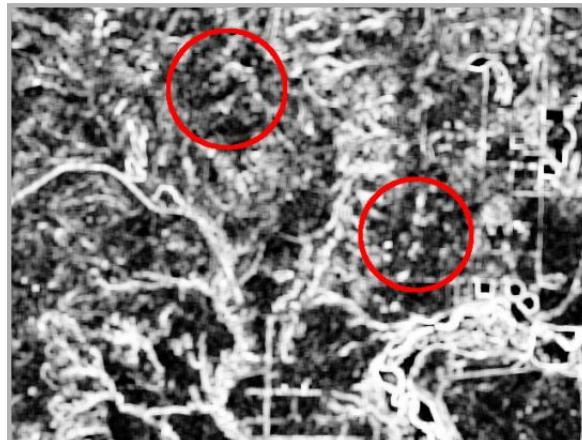
The image is excerpted from Path 41, Row 25 of Landsat 7 ETM+, dated 4 September 1999. This is an area in the Rocky Mountain Foothills near Waterton National Park, Alberta. The western edge of the image contains steep slopes and deep valleys. To the east is both grassland and annual crops, mostly grains. The eastern area is bisected by numerous small streams.

FULL RESOLUTION

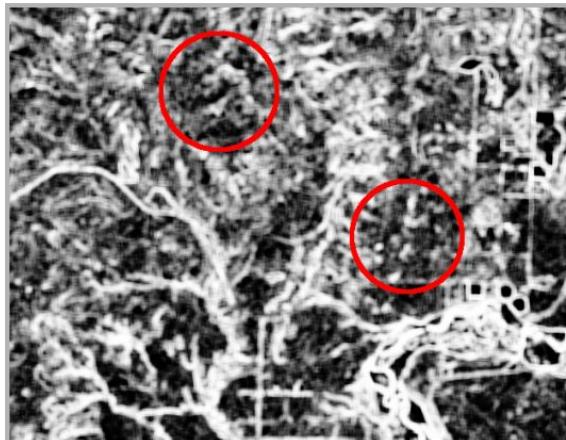


COMPARISON

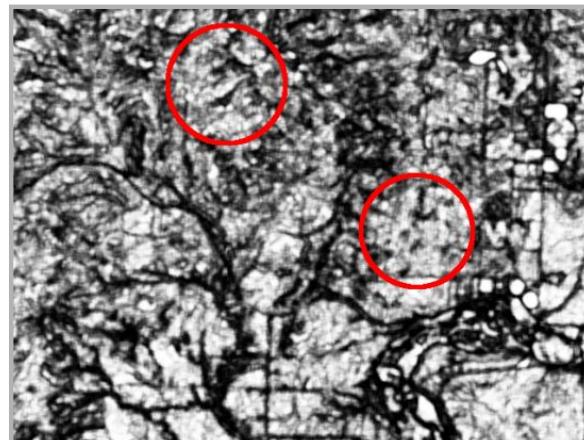
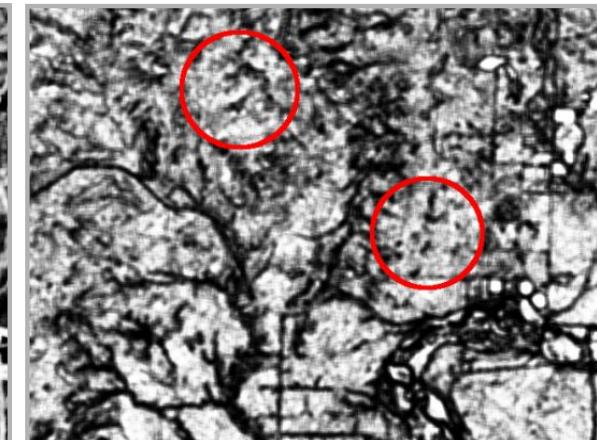
Contrast



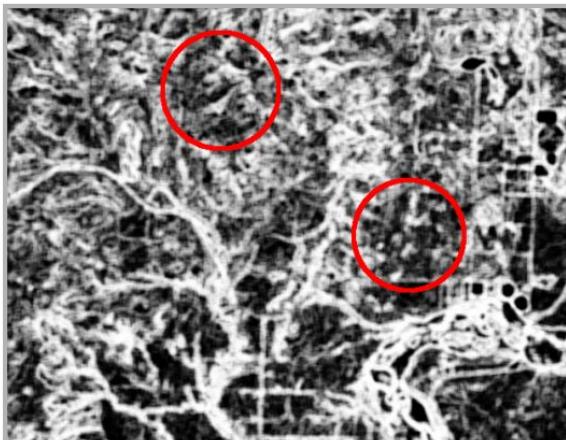
Dissimilarity



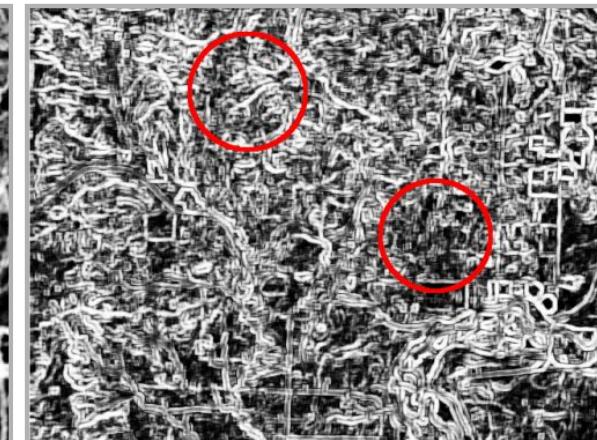
Homogeneity



ASM



Entropy



Correlation

CLASSIFICATION



Used on aerial images to identify eight terrain classes:

- Old residential
- New residential
- Urban
- Lake
- Swamp
- Scrub
- Wood

AERIAL TEXTURES



PARAMETER CHOICES



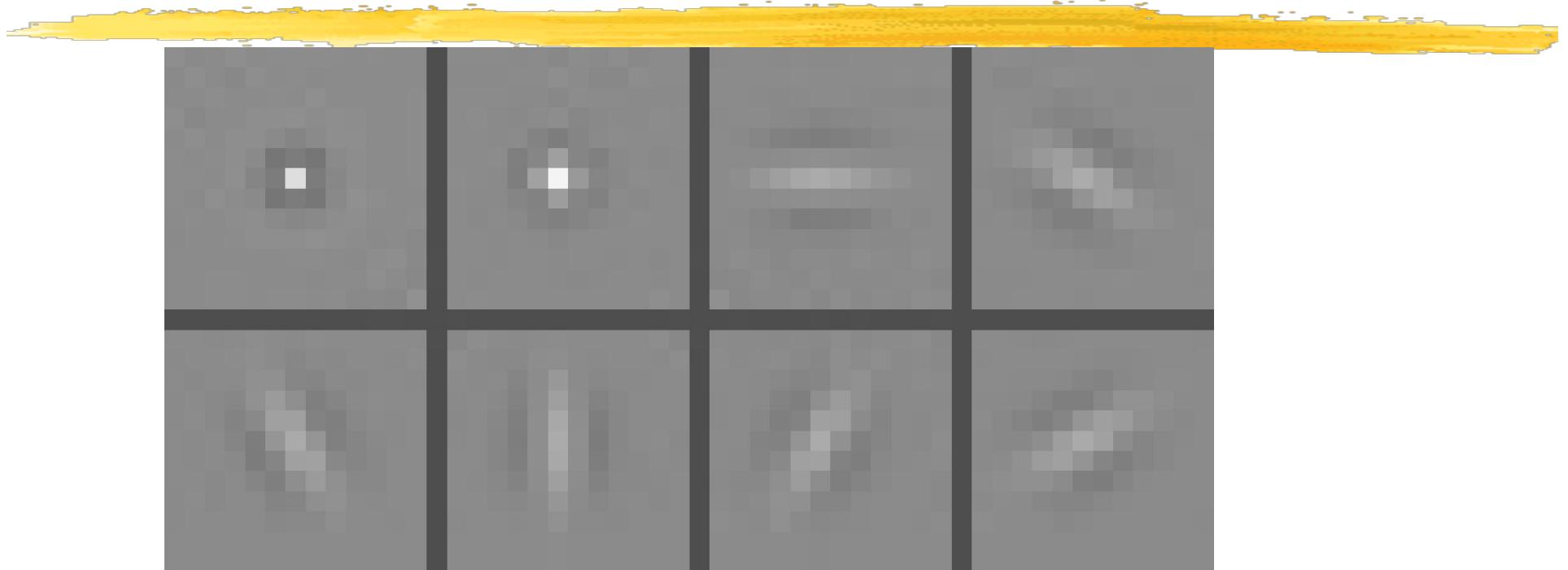
Using co-occurrence matrices requires choosing:

- window size,
- direction of offset,
- offset distance,
- which channels to use,
- which measures to use.

How do we choose these parameters?

- Critical question for **all** statistical texture methods.
- Can be addressed using Machine Learning.

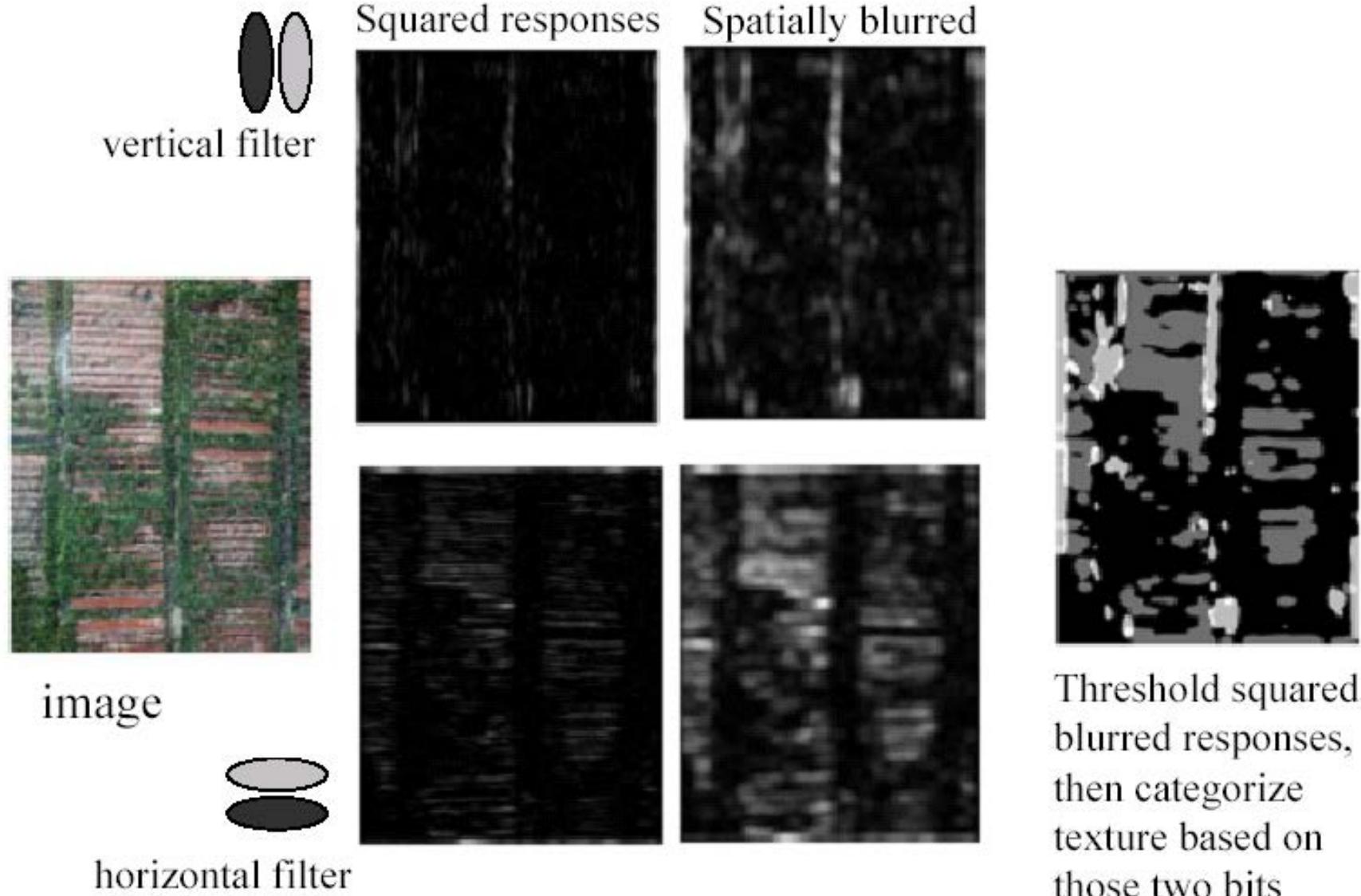
FILTER BASED MEASURES



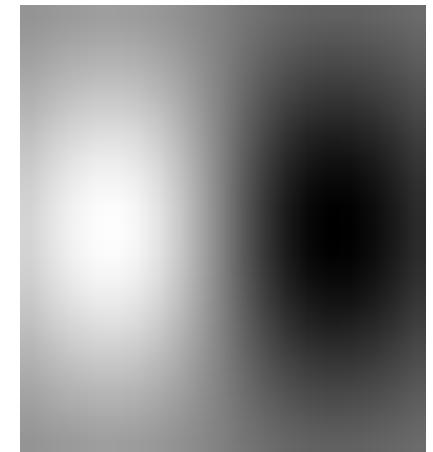
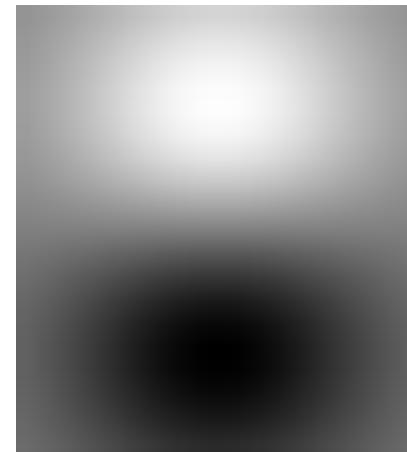
Represent image textures using the responses of a collection of filters.

- An appropriate filter bank will extract useful information such as spots and edges
- Traditionally one or two spot filters and several oriented bar filters

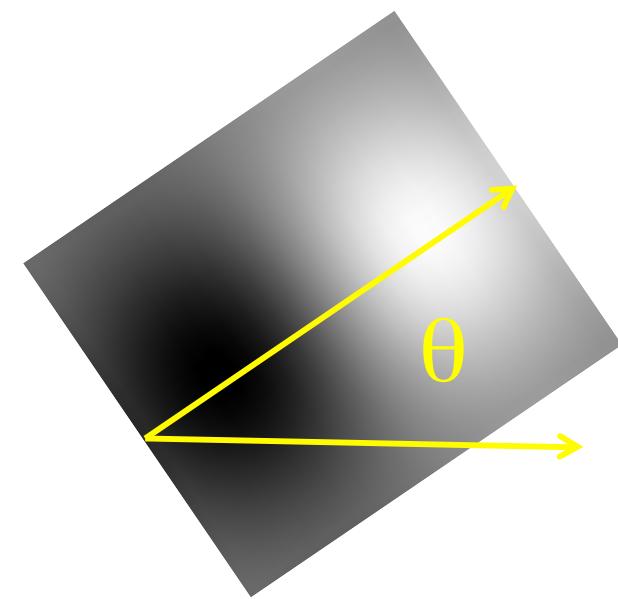
HORIZONTAL AND VERTICAL STRUCTURES



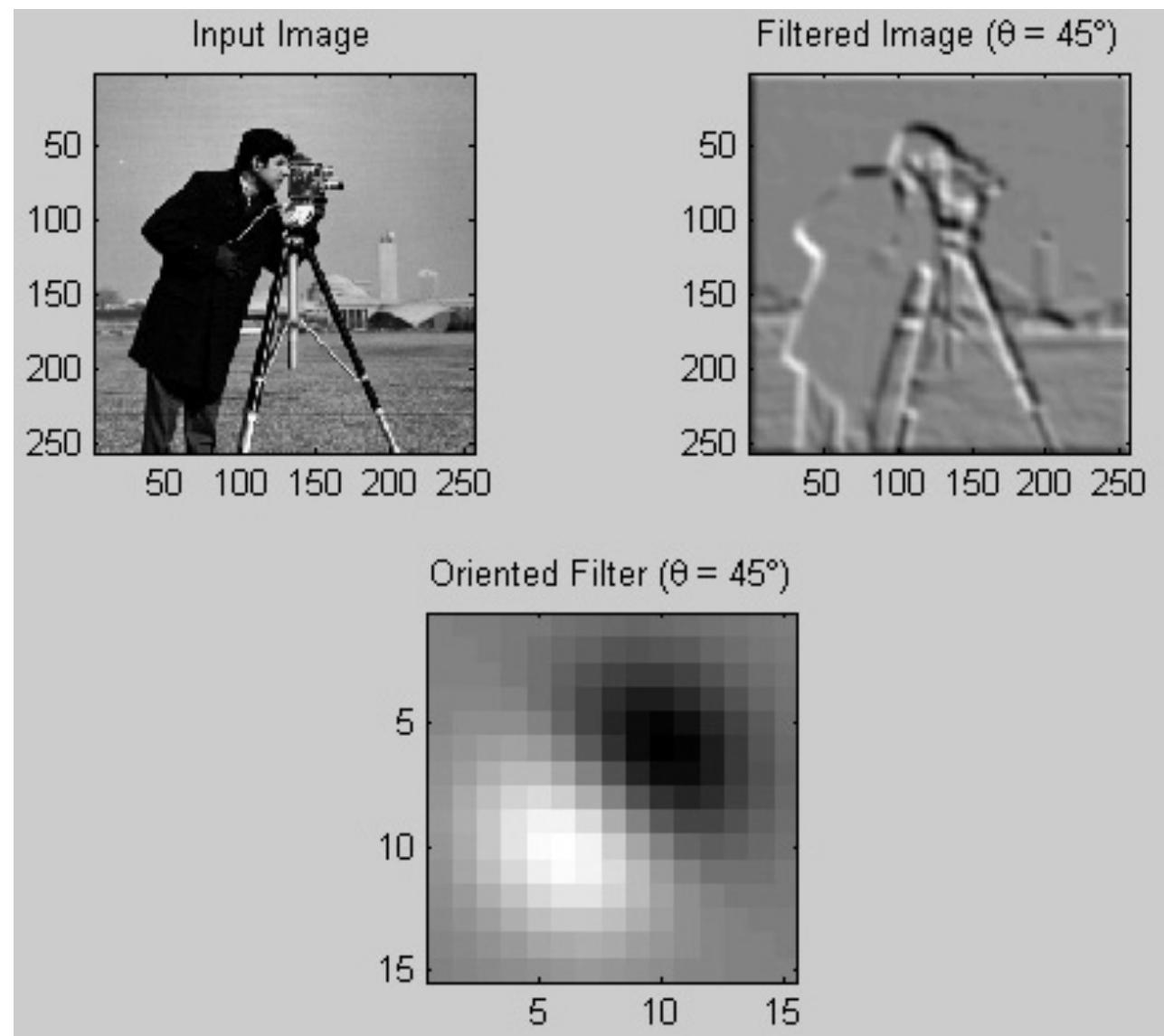
ORIENTED FILTERS



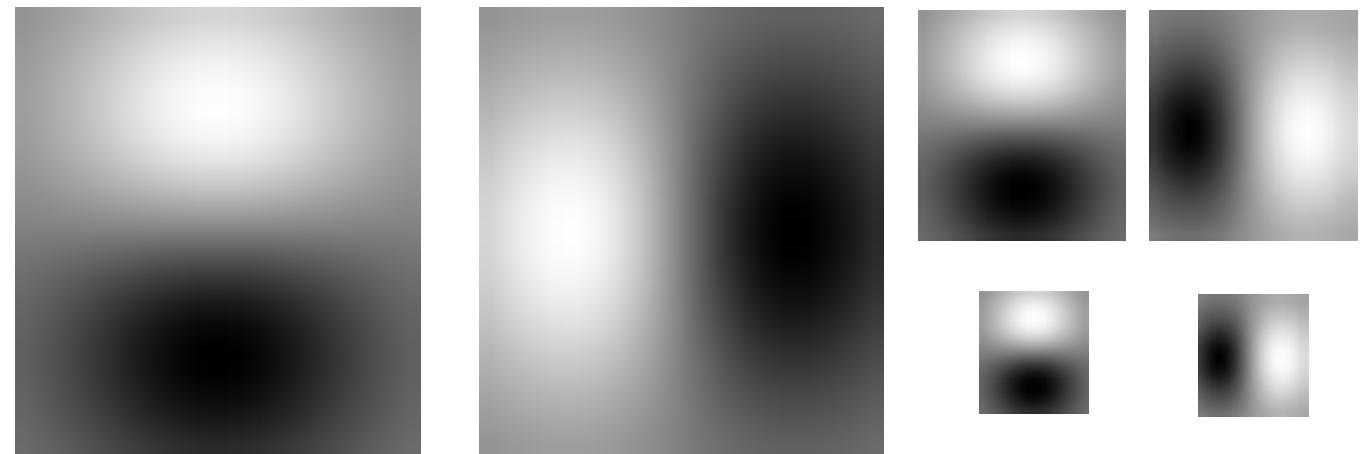
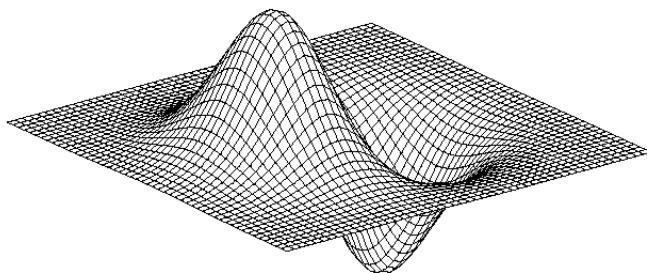
$$\frac{\partial I}{\partial \theta} = \cos(\theta) \frac{\partial I}{\partial x} + \sin(\theta) \frac{\partial I}{\partial y}$$



DIRECTIONAL GRADIENTS



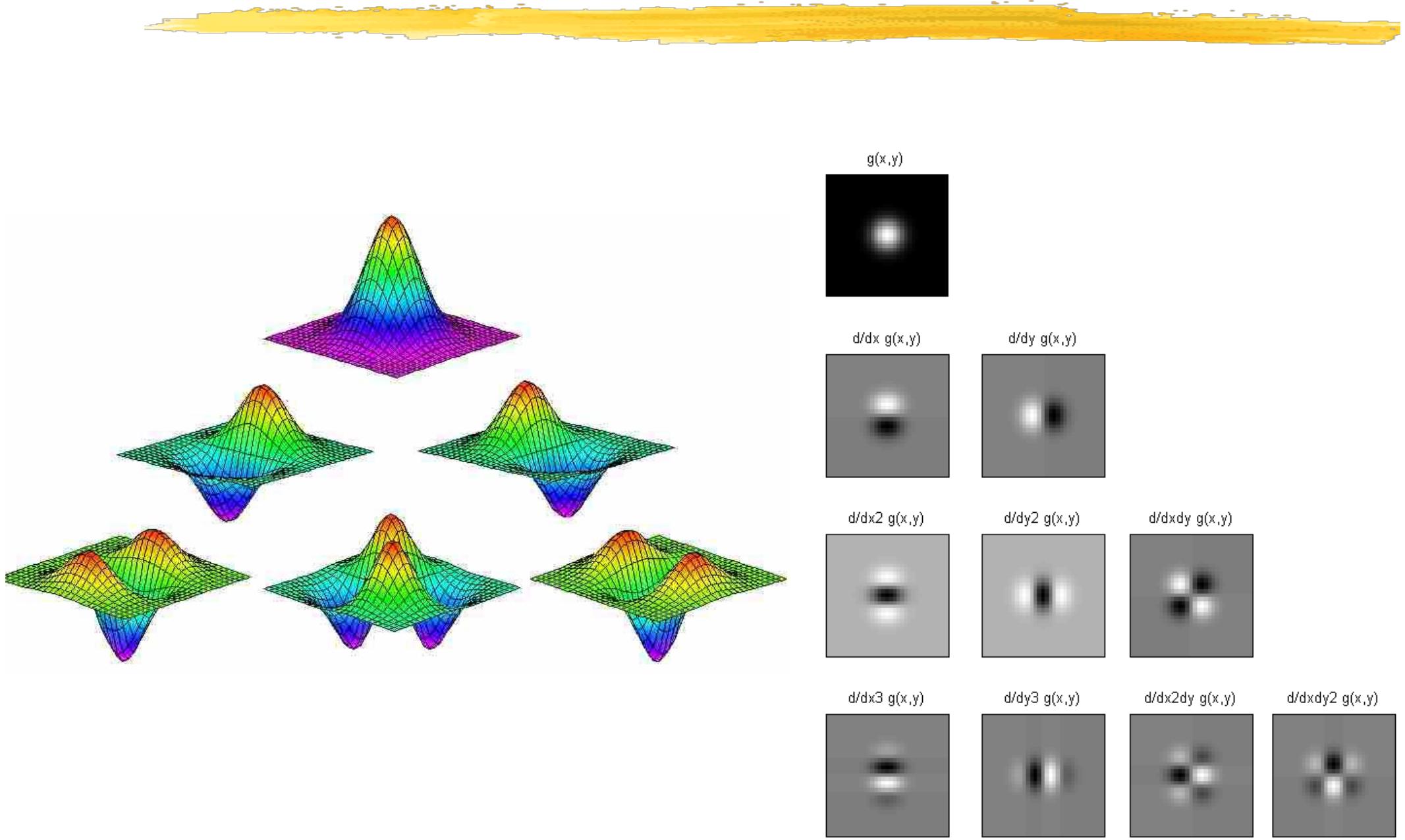
GAUSSIAN FILTER DERIVATIVES



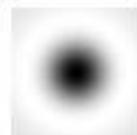
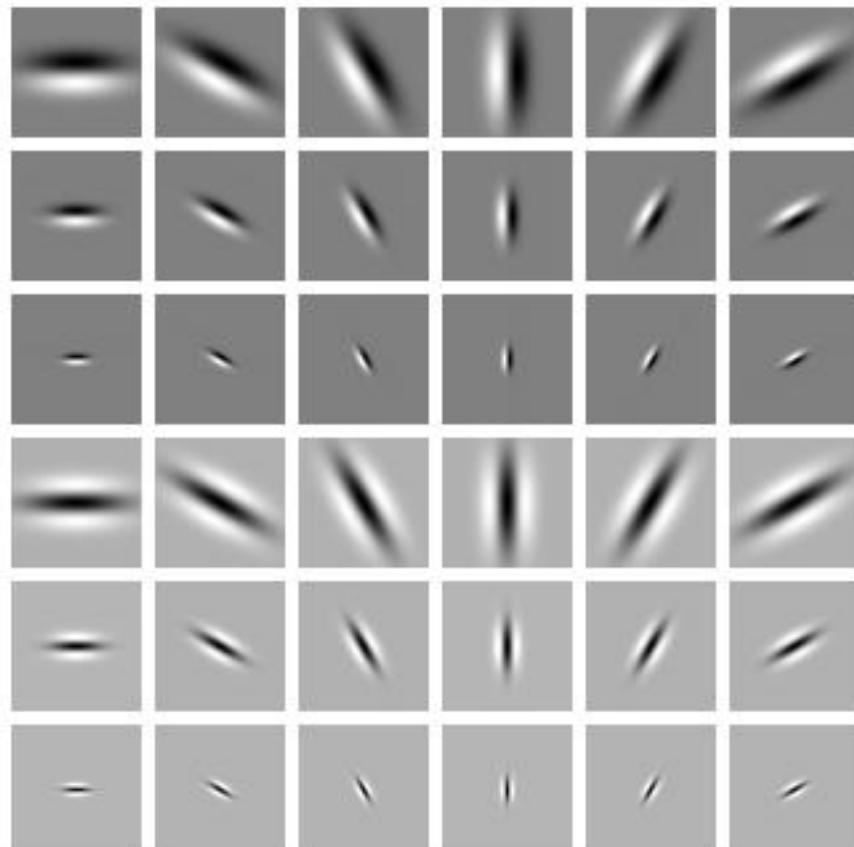
Gaussian Derivative

x and y derivatives at different scales

HIGHER ORDER DERIVATIVES

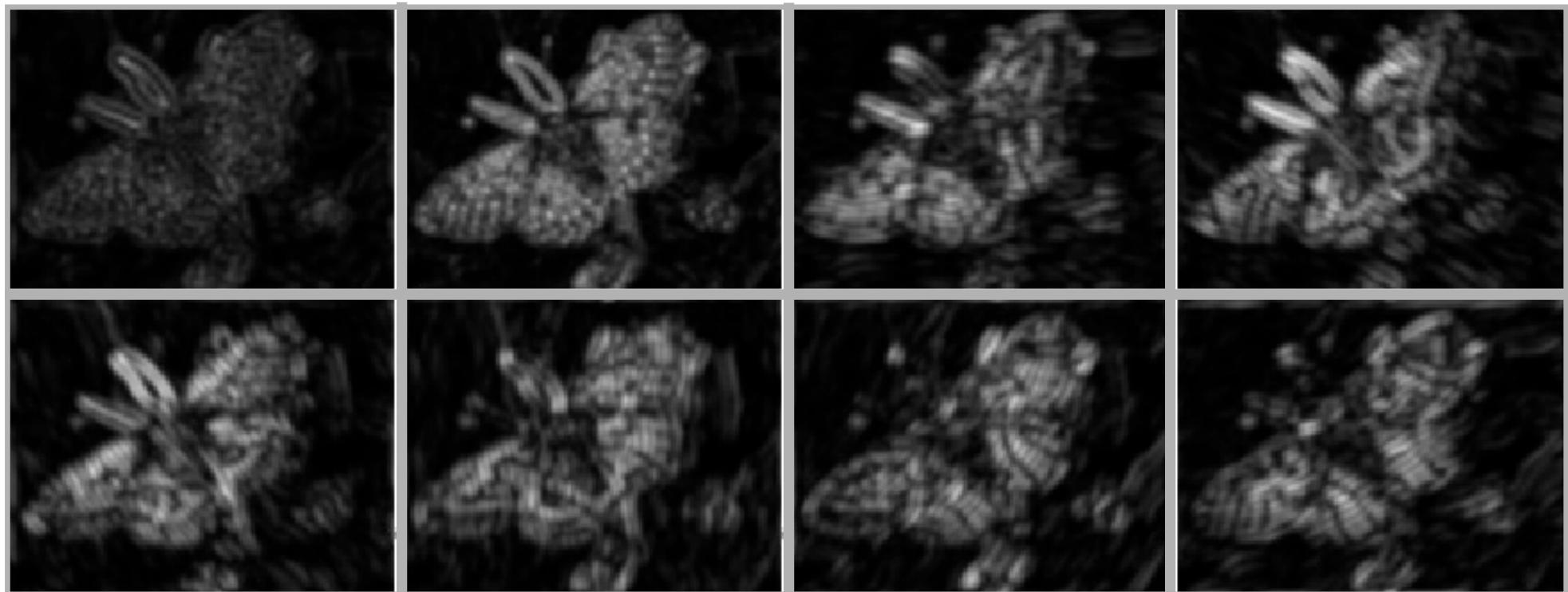
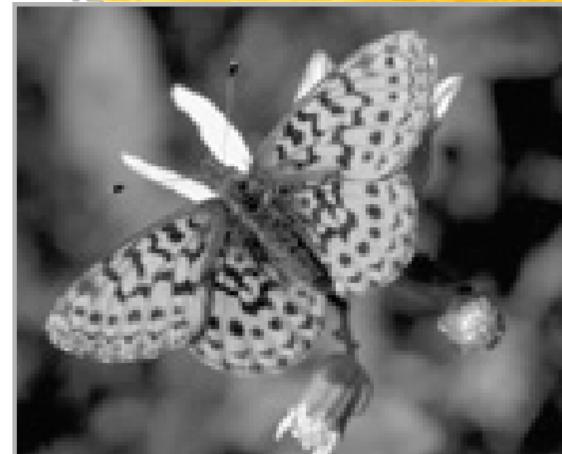


FILTER BANKS

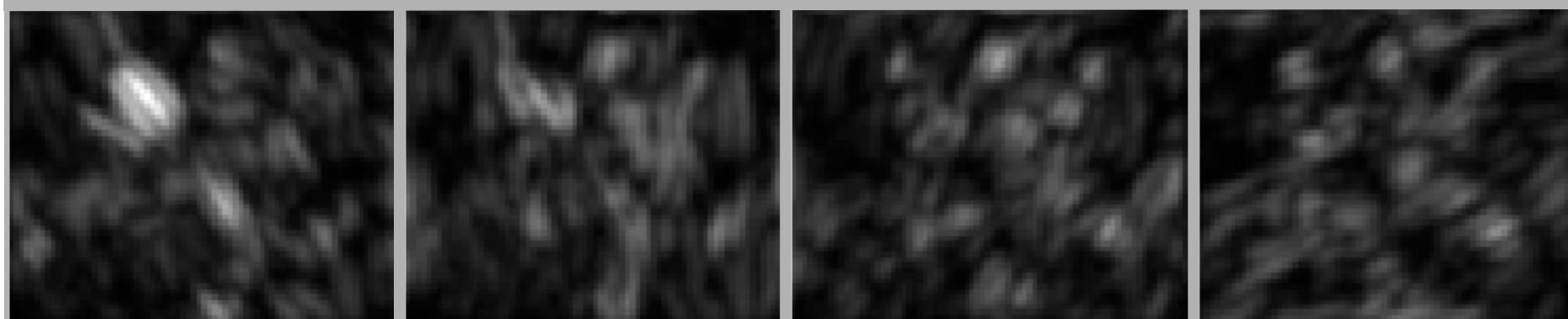
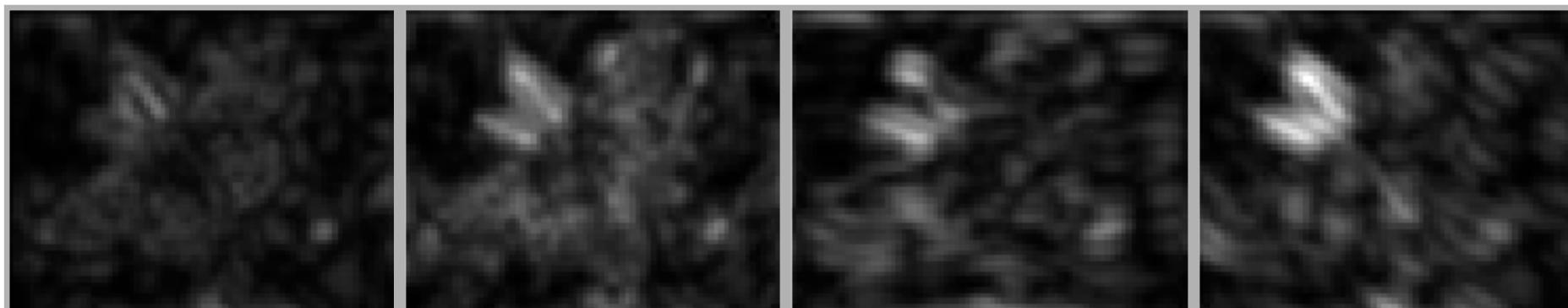


- Different scales.
- Different orientations.
- Derivatives order 0, 1, 2 ..

FILTER RESPONSES: HIGH RESOLUTION



FILTER RESPONSES: LOW RESOLUTION



GABOR FILTERS



Gabor filters are the products of a Gaussian filter with oriented sinusoids. They come in pairs, each consisting of a symmetric filter and an anti-symmetric filter:

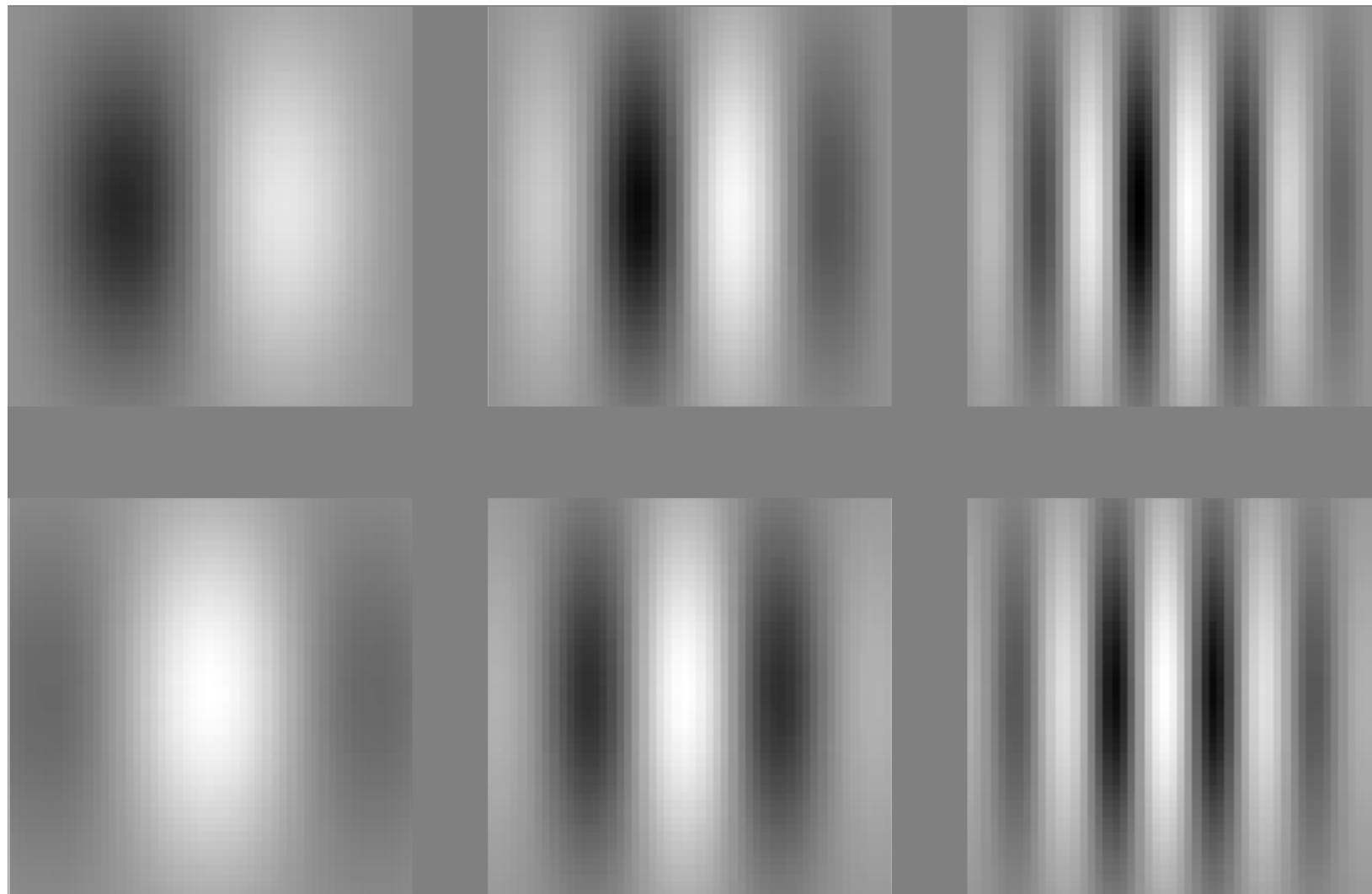
$$G_{\text{sym}}(x, y) = \cos(k_x x + k_y y) \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right)$$

$$G_{\text{asym}}(x, y) = \sin(k_x x + k_y y) \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right)$$

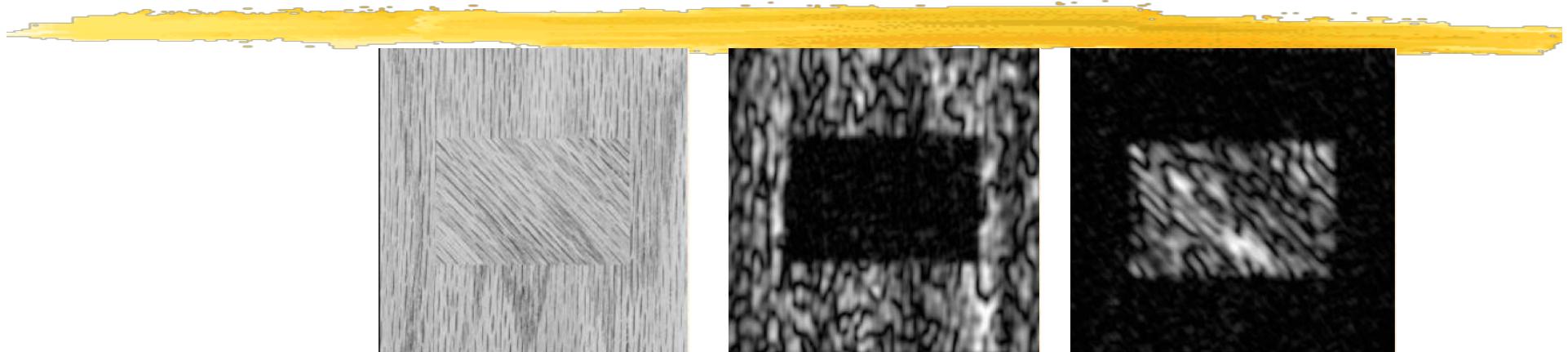
where k_x and k_y determine the spatial frequency and the orientation of the filter and σ determines the scale.

→ A filter bank is formed by varying the frequency, the scale, and the filter orientation

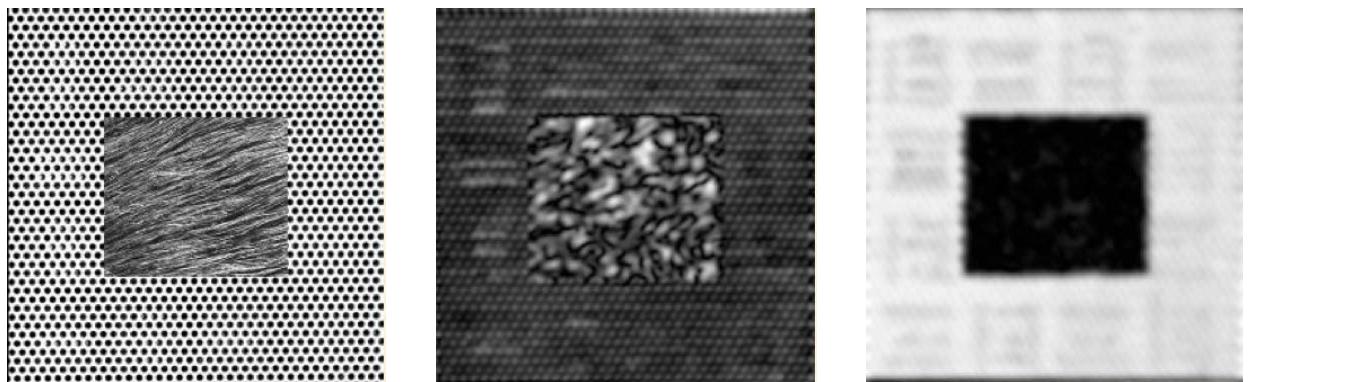
GABOR FILTERS



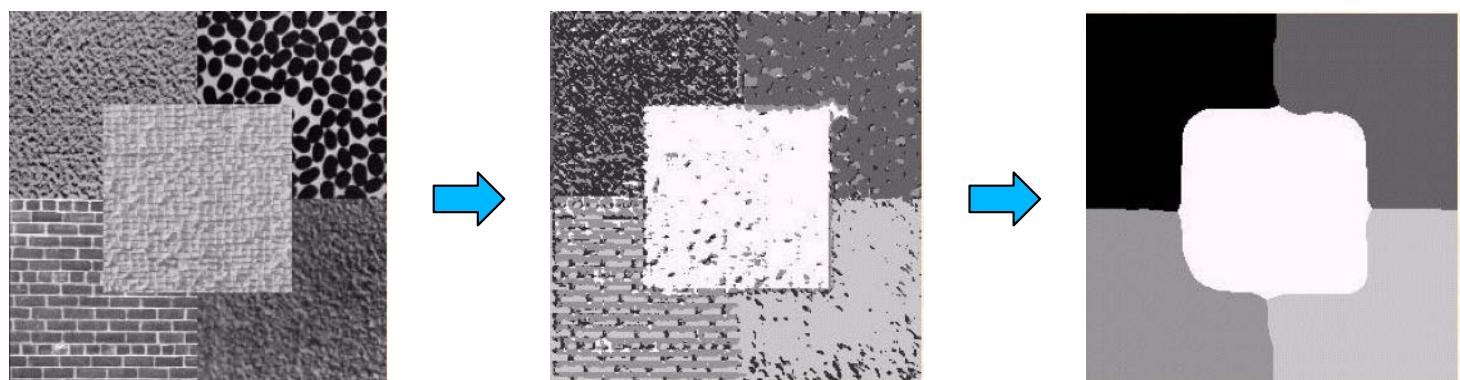
GABOR RESPONSES



Responses:



Segmentation:

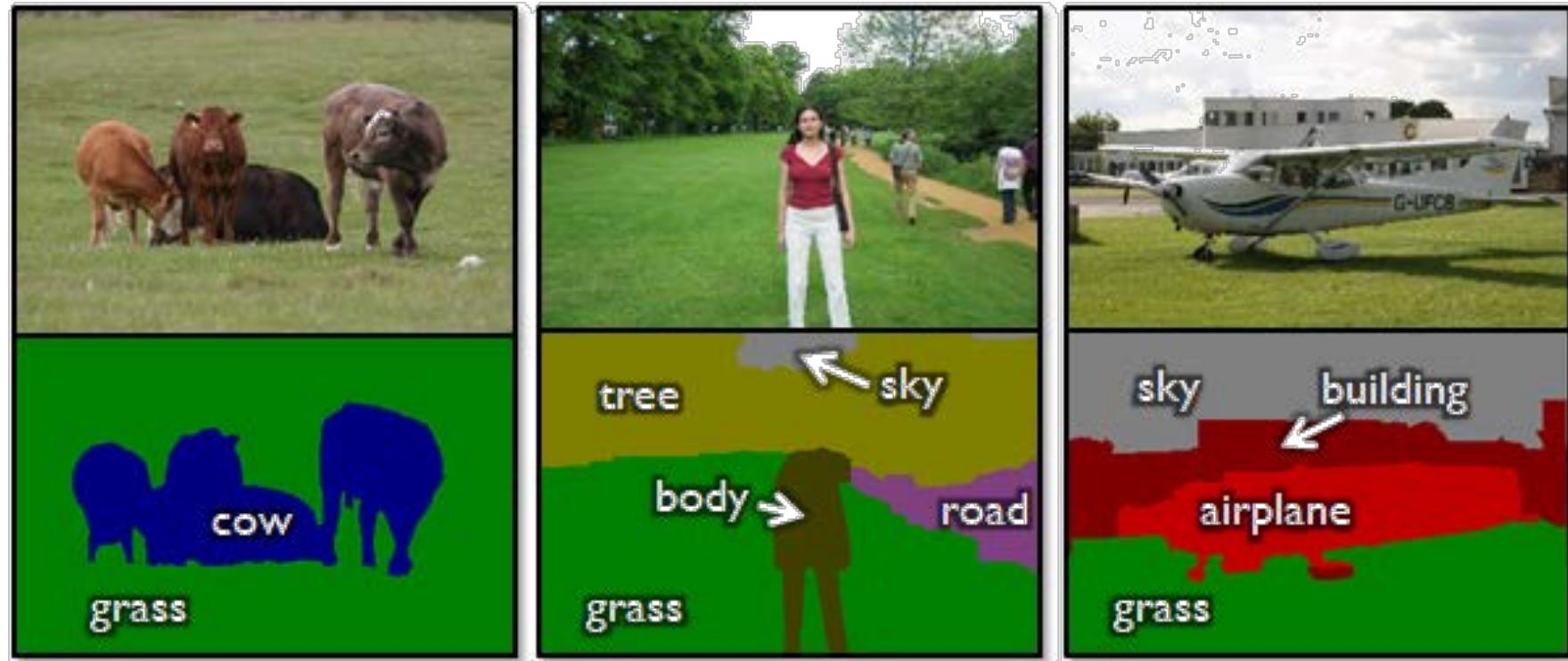


GABOR FILTER CHARACTERISTICS



- Respond strongly at points in an image where there are components that locally have a particular spatial frequency and orientation.
- In theory, by applying a very large number of Gabor filters at different scales, orientations and spatial frequencies, one can analyze an image into a detailed local description.
- In practice, it is not known how many filters, at what scale, frequencies, and orientations, to use. This tends to be application dependent and can be estimated using Machine Learning techniques.

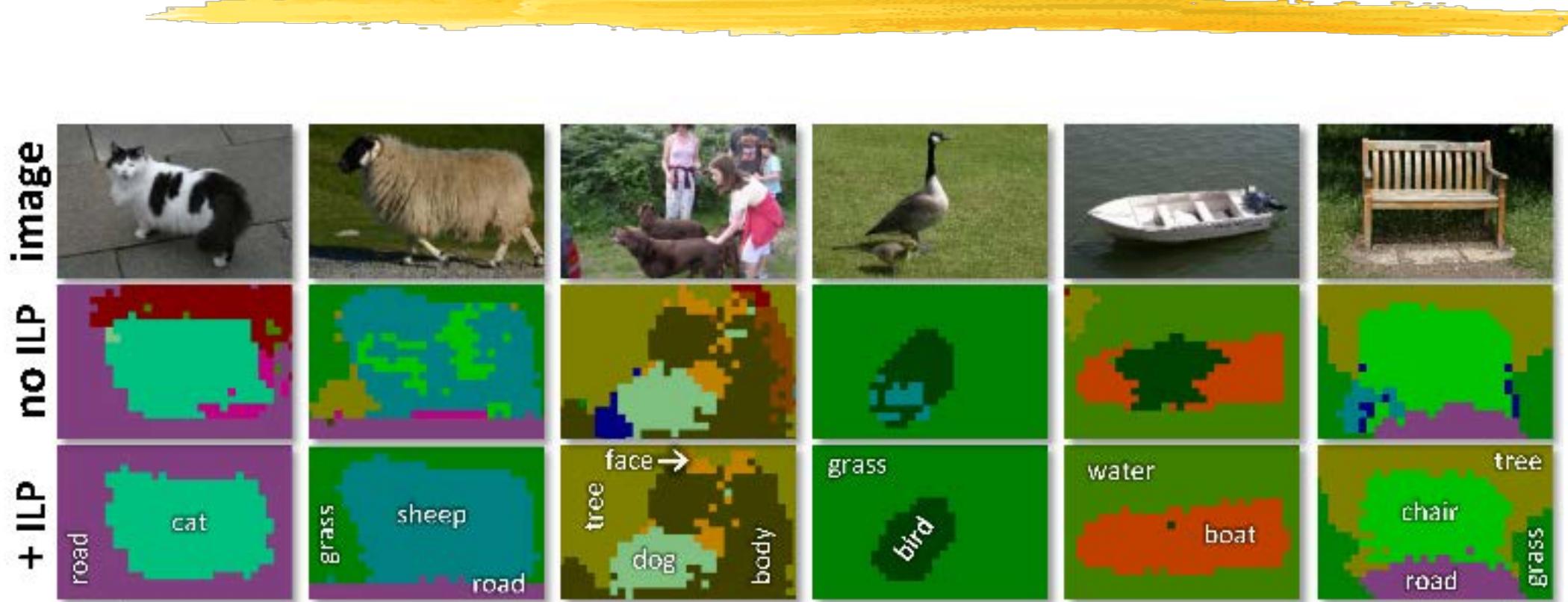
TEXTON BOOST



object classes	building	grass	tree	cow	sheep	sky	airplane	water	face	car
bicycle	flower	sign	bird	book	chair	road	cat	dog	body	boat

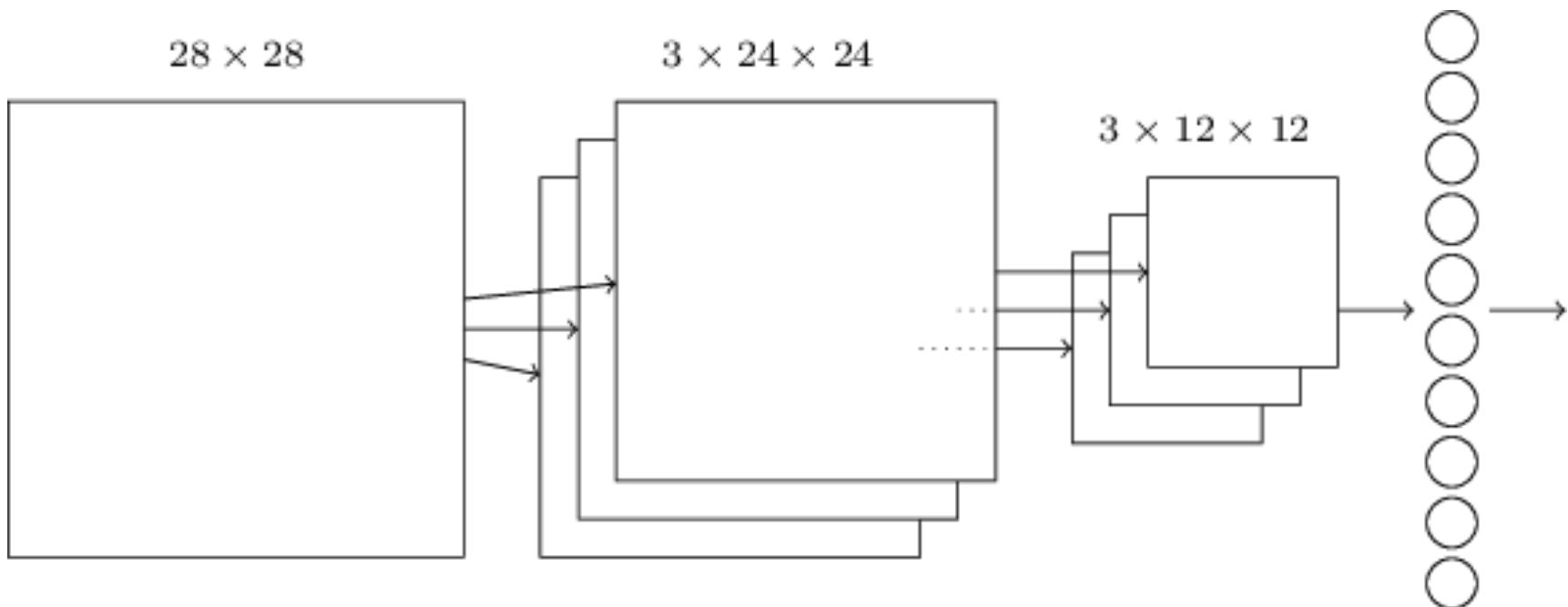
Performs classification on the output of oriented filters.

TEXTON FORESTS

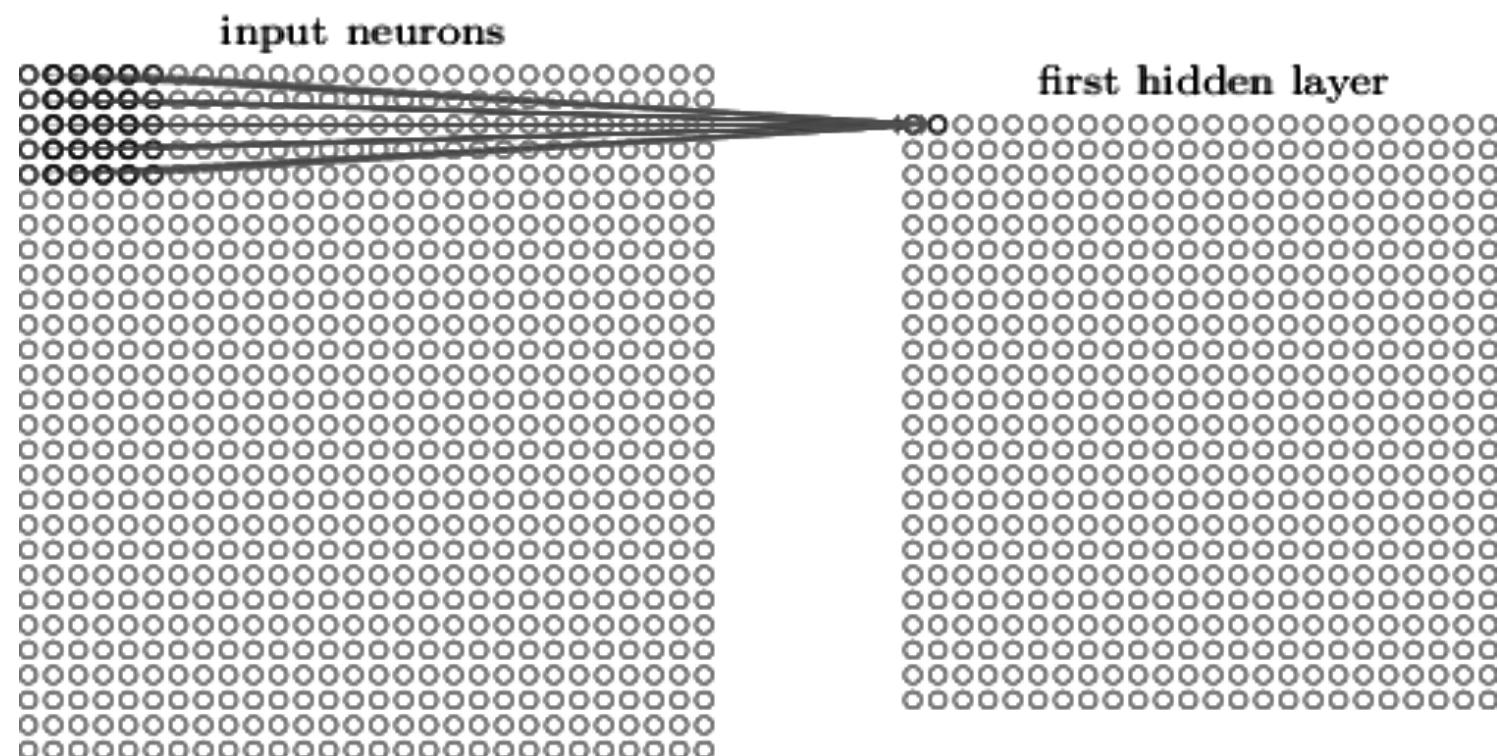


Relies on decision trees and comparisons of colors of pixels belonging to small image patches.

BACK TO CONVOLUTIONAL NETS

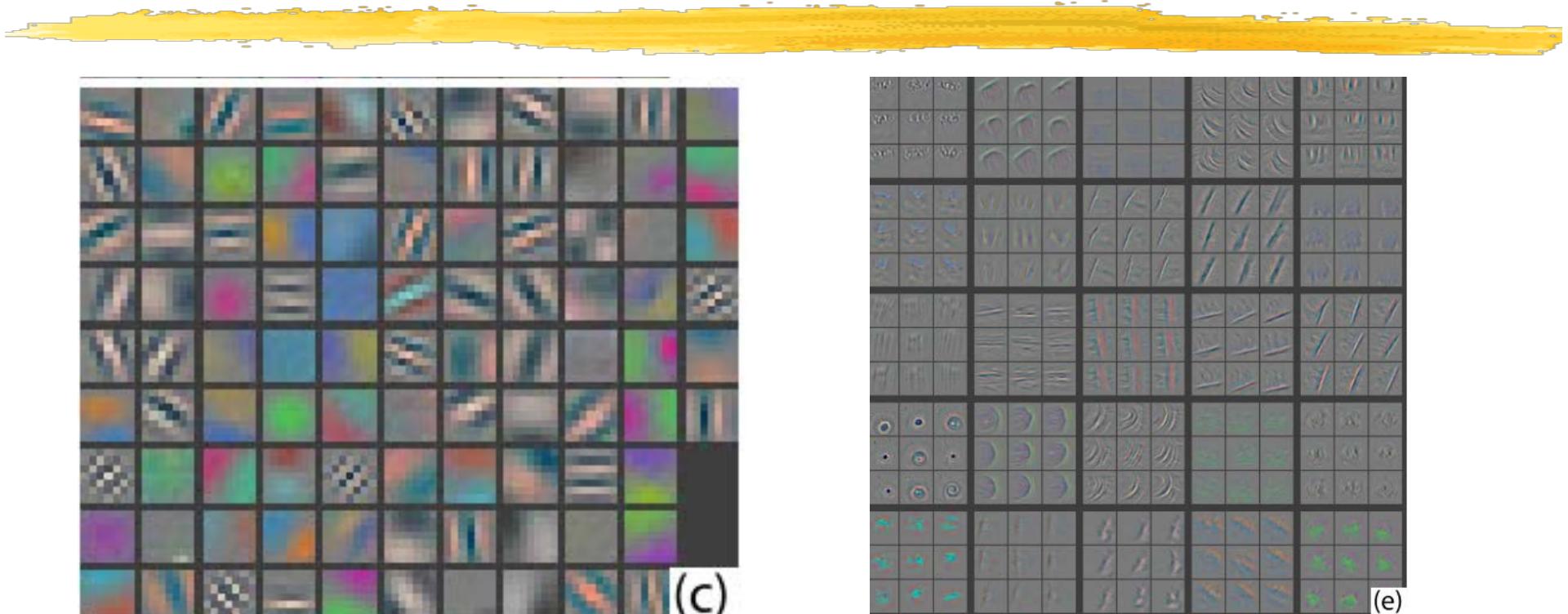


CONVOLUTIONAL LAYER



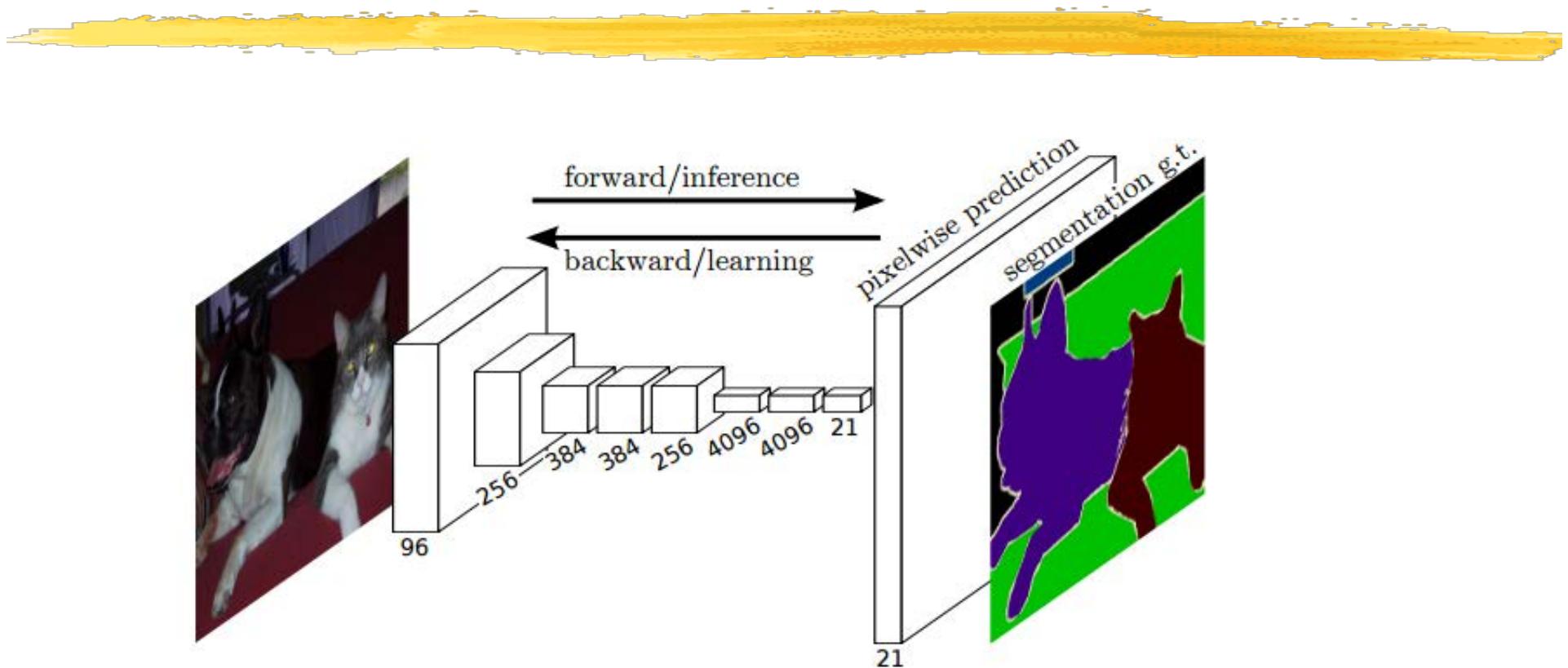
$$\sigma \left(b + \sum_{x=0}^{n_x} \sum_{y=0}^{n_y} w_{i,j} a_{i+x,j+y} \right)$$

FEATURE MAPS



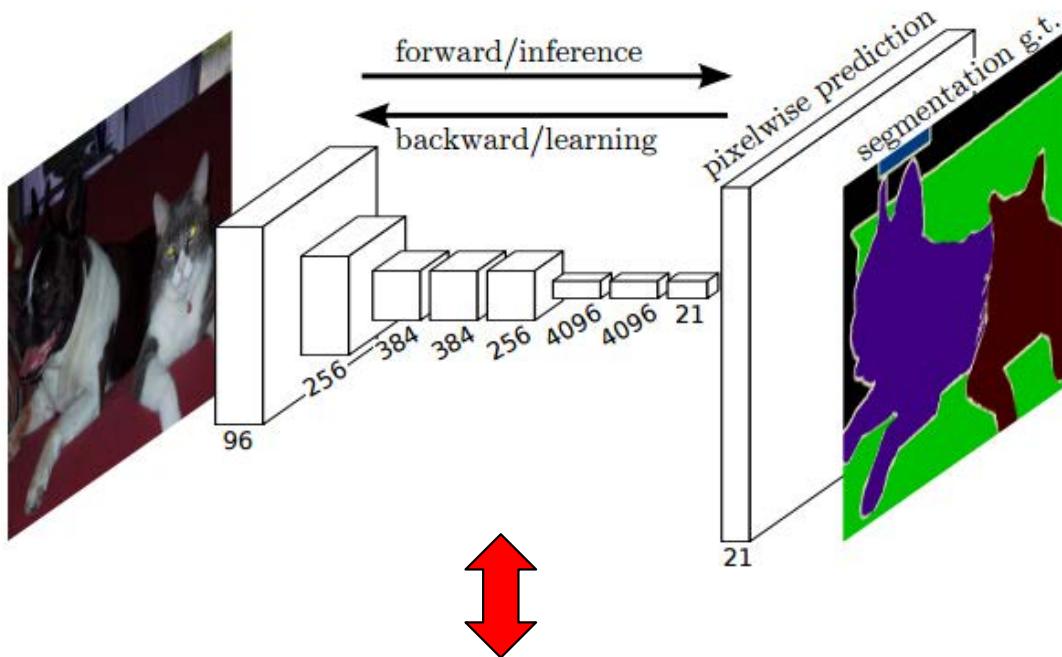
- Some of these convolutional filters look very Gabor like.
- The network learns the right filter bank but still depends on many arbitrary parameters.

CONVOLUTIONAL NETS

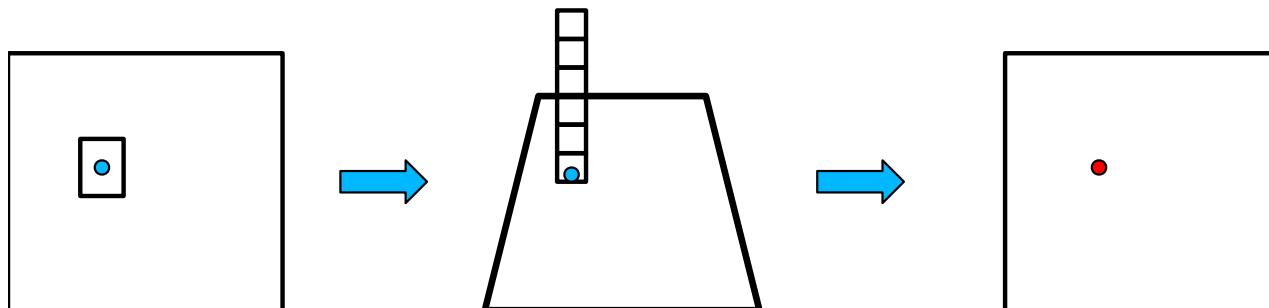


- Connect input layer to output one made of segmentation labels.
- Need layers that both downscale and upscale.
- Connect the lower layers directly to the upper ones.

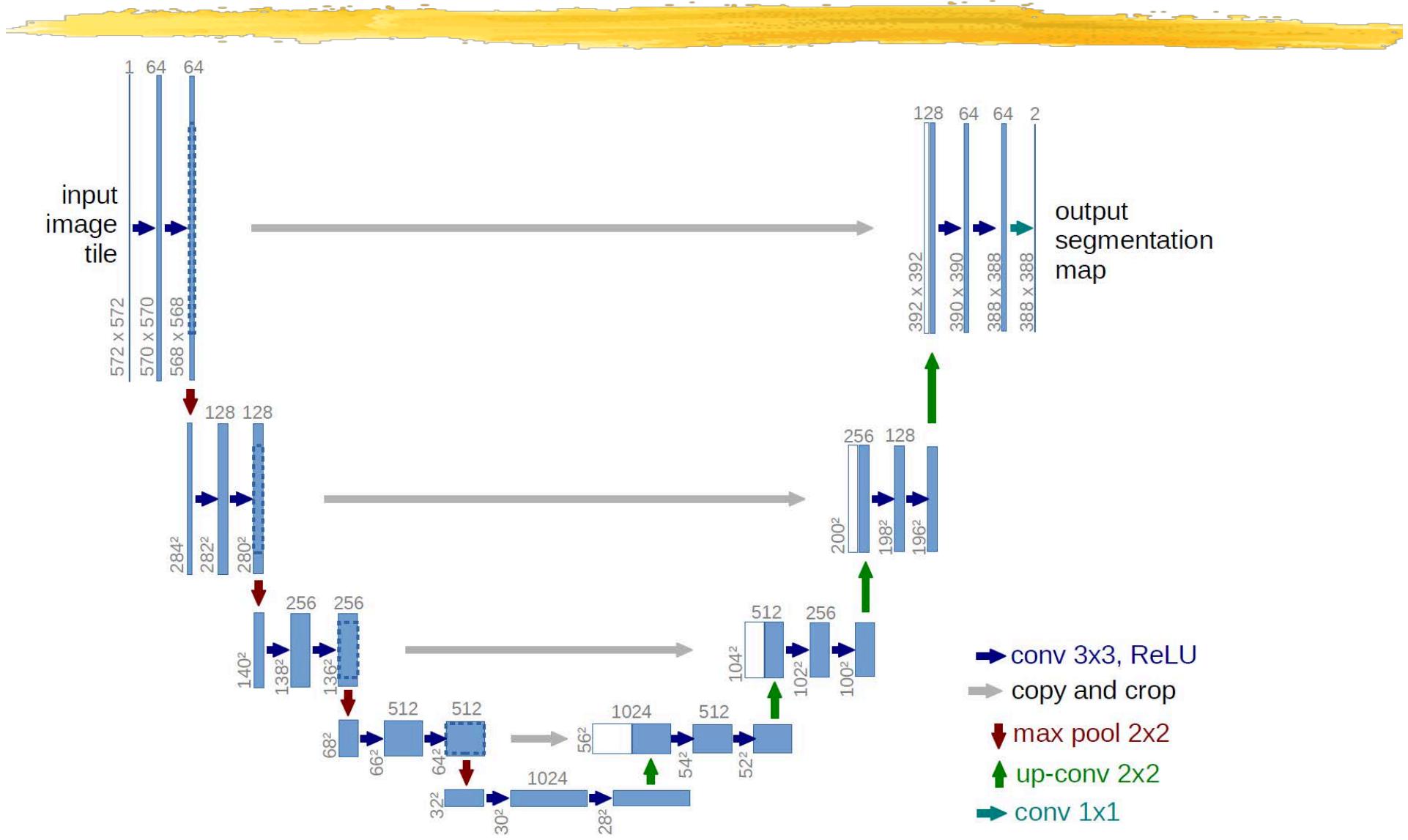
AN INTERPRETATION



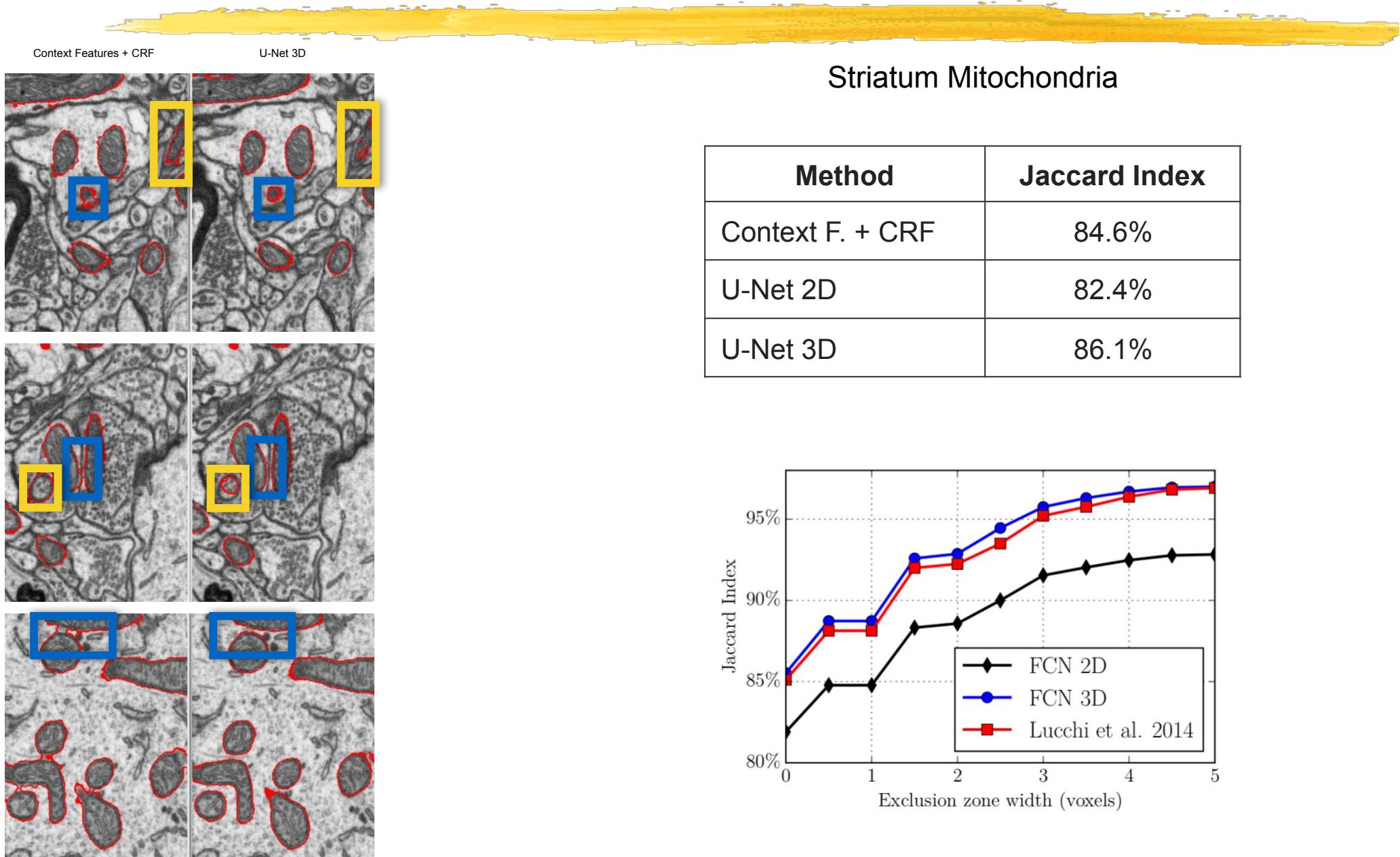
- Can be understood as generating for every output pixel a feature vector containing the output of all the intermediate layers.



U-NET ARCHITECTURE



FROM GRAPH CUT TO U-NET



IN SHORT

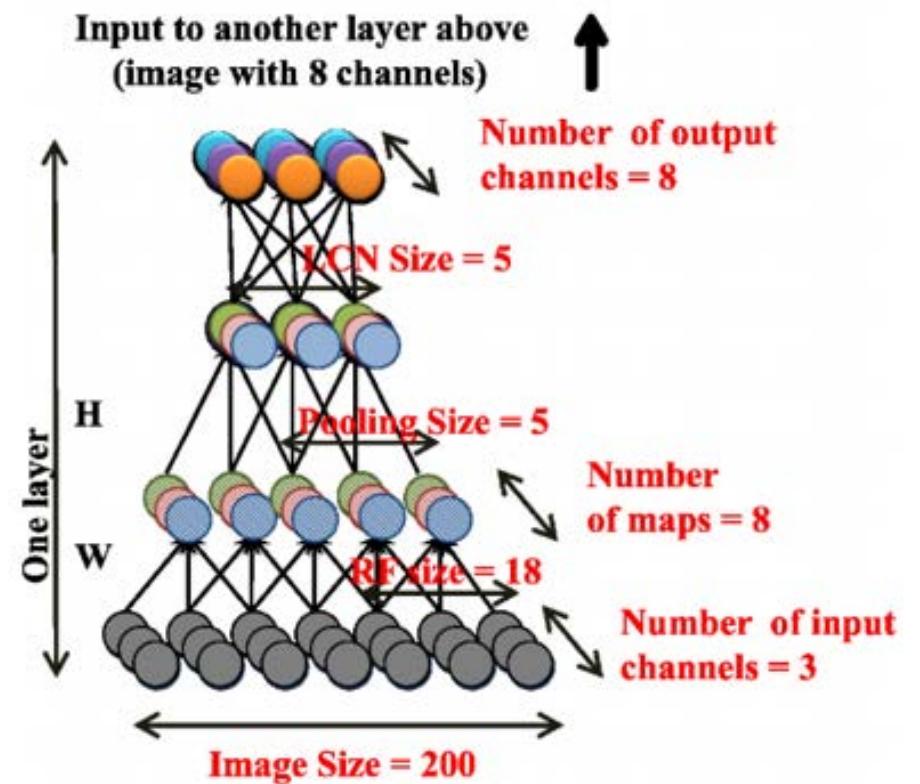


Texture is a key property of objects which is

- Non local
 - Non trivial to measure
 - Subject to deformations
- Hard to characterize formally and best used in conjunction with effective Machine Learning techniques.
- This may be exactly what Convolutional Neural Nets do.

DEEP LEARNING CRASH COURSE

- Single Layer Perceptron
- Multiple Layer Perceptron
- Convolutional Neural Net



ARTIFICAL INTELLIGENCE

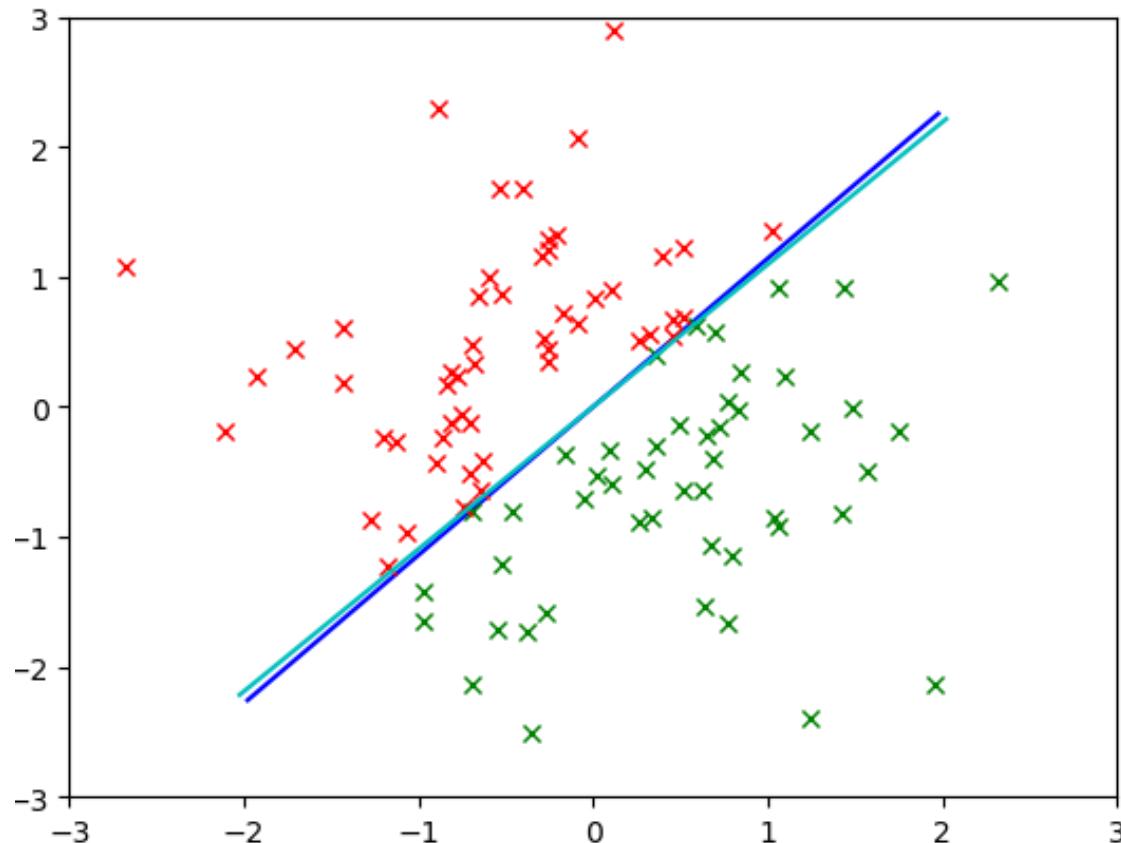


1997: Deep Blue beats chess World Champion



2016: AlphaGo beats go world champion

PERCEPTRON

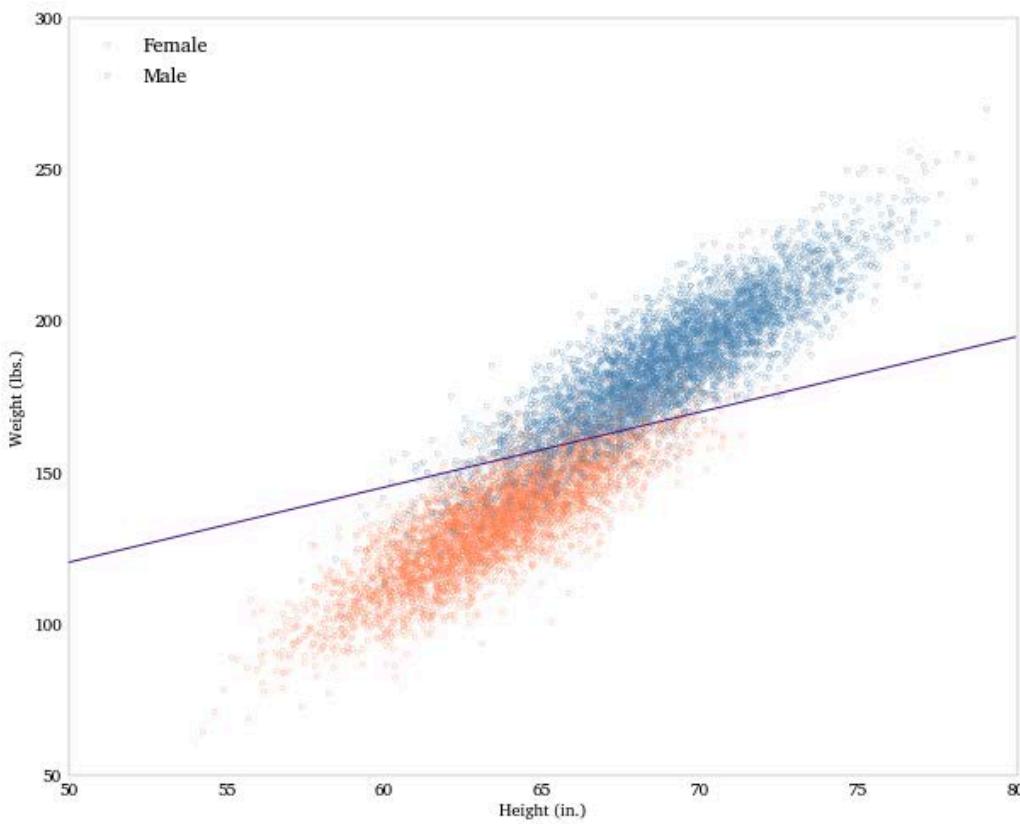


$$y(\mathbf{x}; \tilde{\mathbf{w}}) = \begin{cases} 1 & \text{if } \tilde{\mathbf{w}}^T \tilde{\mathbf{x}} \geq 0, \\ -1 & \text{otherwise.} \end{cases}$$
$$\tilde{\mathbf{x}} = [1, x_1, \dots, x_n]$$

Choose the \mathbf{w} that minimizes:

$$E(\tilde{\mathbf{w}}) = - \sum_{n=1}^N (\tilde{\mathbf{w}}^T \tilde{\mathbf{x}}_n) t_n$$

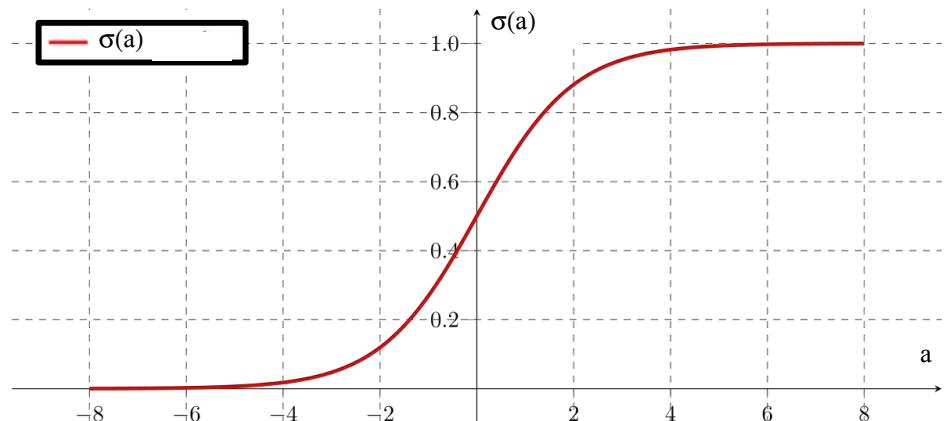
LOGISTIC REGRESSION



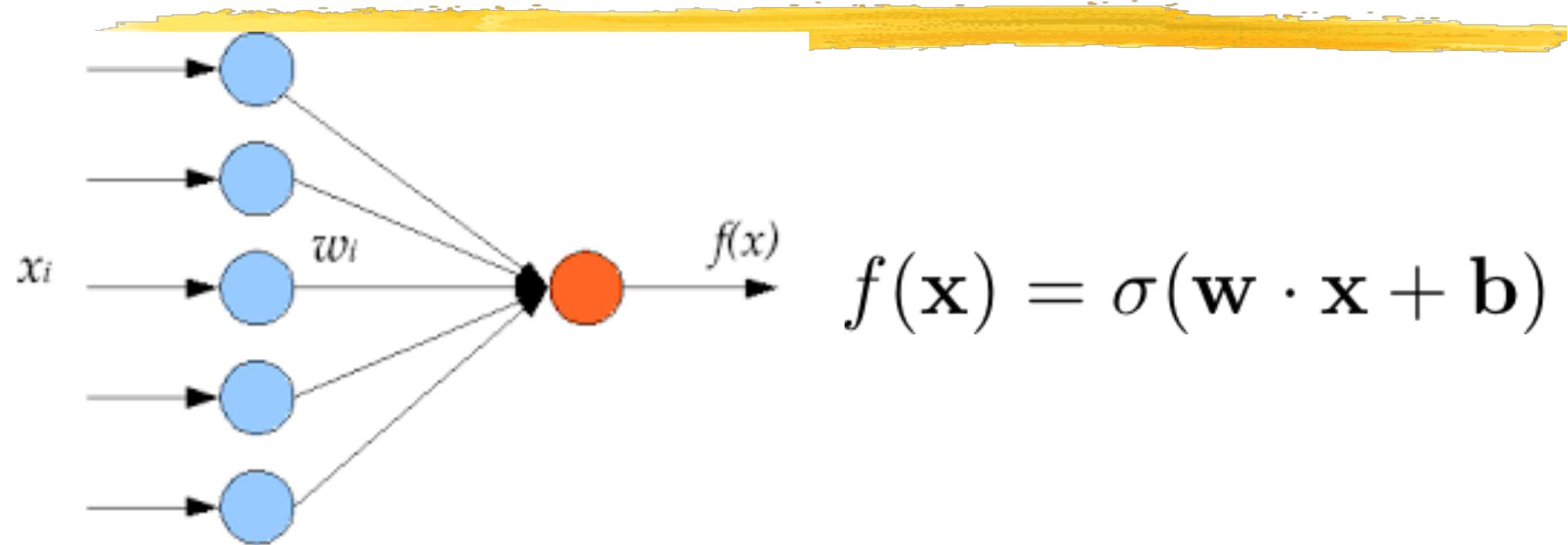
$$y(\mathbf{x}; \mathbf{w}) = \sigma(\mathbf{w}^T \mathbf{x} + w_0)$$

Choose the \mathbf{w} that minimizes:

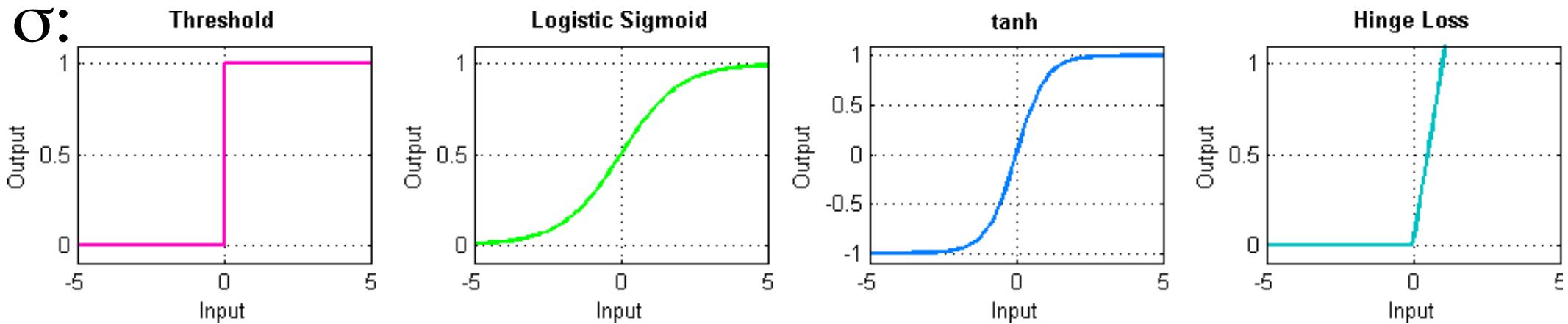
$$E(\mathbf{w}) = - \sum_n \{t_n \ln y_n + (1 - t_n) \ln(1 - y_n)\}$$



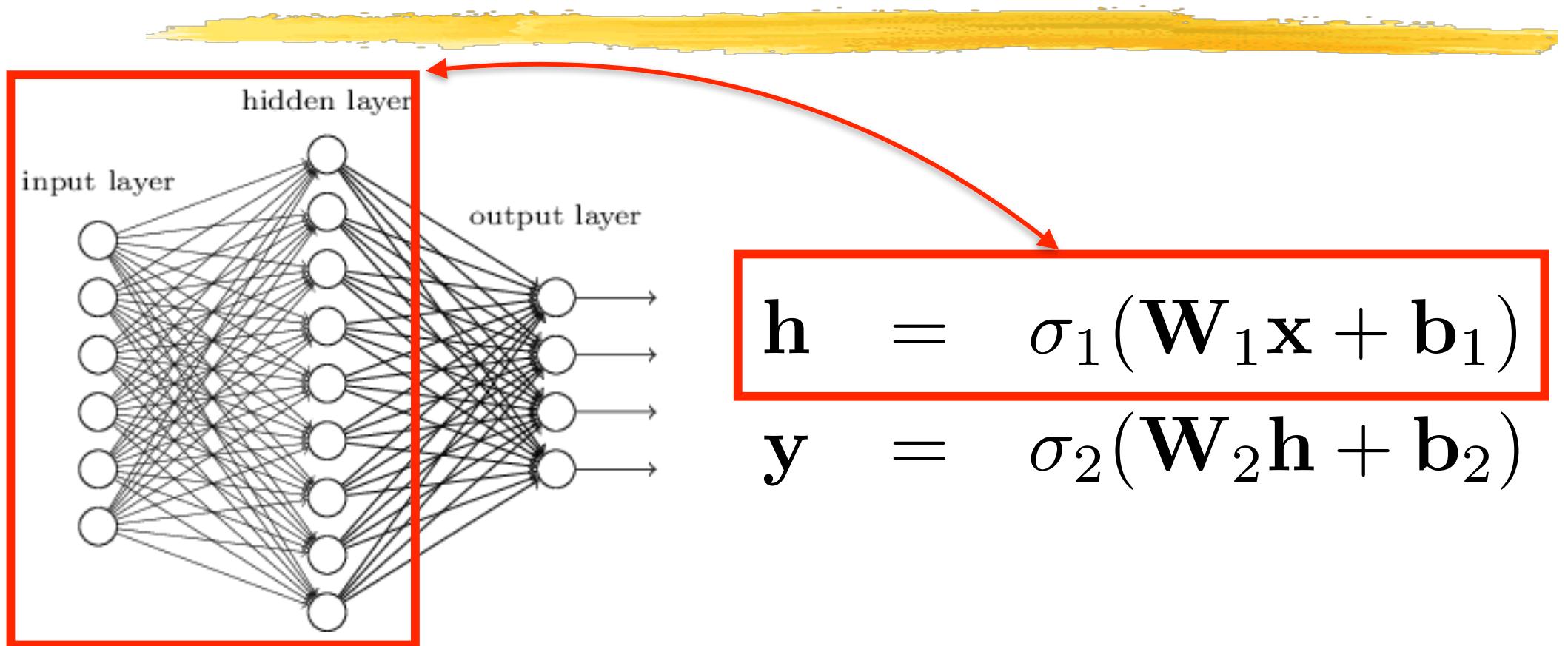
SINGLE LAYER PERCEPTRON



σ :

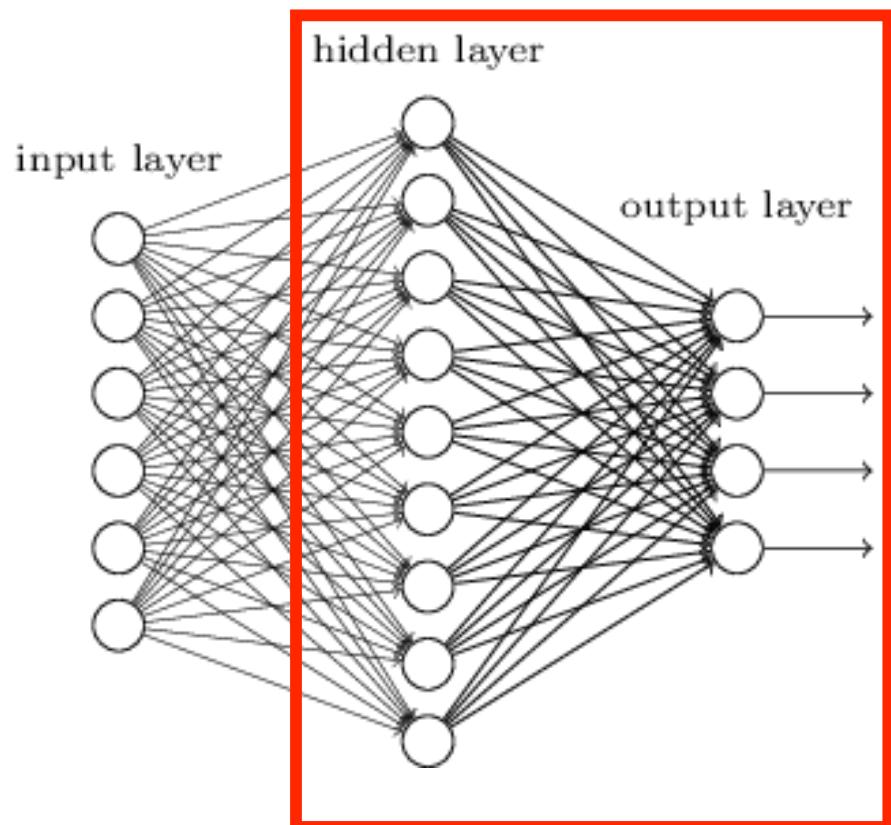


MULTILAYER PERCEPTRON



- The process can be repeated several times to create a vector \mathbf{h} .

MULTILAYER PERCEPTRON



$$\begin{aligned} \mathbf{h} &= \sigma_1(\mathbf{W}_1 \mathbf{x} + \mathbf{b}_1) \\ \mathbf{y} &= \sigma_2(\mathbf{W}_2 \mathbf{h} + \mathbf{b}_2) \end{aligned}$$

- The process can be repeated several times to create a vector \mathbf{h} .
- It can then be done again to produce an output \mathbf{y} .
- → This output is a **differentiable** function of the weights.

BINARY CASE

Given a training set $\{\mathbf{x}_n, t_n\}_{1 \leq n \leq N}$ where $t_n \in \{0, 1\}$, minimize

$$\begin{aligned} E(\mathbf{W}, \mathbf{b}) &= -\frac{1}{N} \sum_1^N [t_i \log(y_i) + (1 - t_i) \log(1 - y_i)] , \\ y_i &= f(\mathbf{x}_i) \\ &= \sigma(\mathbf{W}_2(\sigma(\mathbf{W}_1 \mathbf{x}_i + \mathbf{b}_1)) + \mathbf{b}_2), \end{aligned}$$

that is, minimize the number of misclassified samples.

Since E is a differentiable function of \mathbf{W} and \mathbf{b} , this can be done using a gradient-based technique, also known as back propagation.

MULTI-CLASS CASE

In the multi-class case, the probability that input vector \mathbf{x} belongs to class c is taken to be

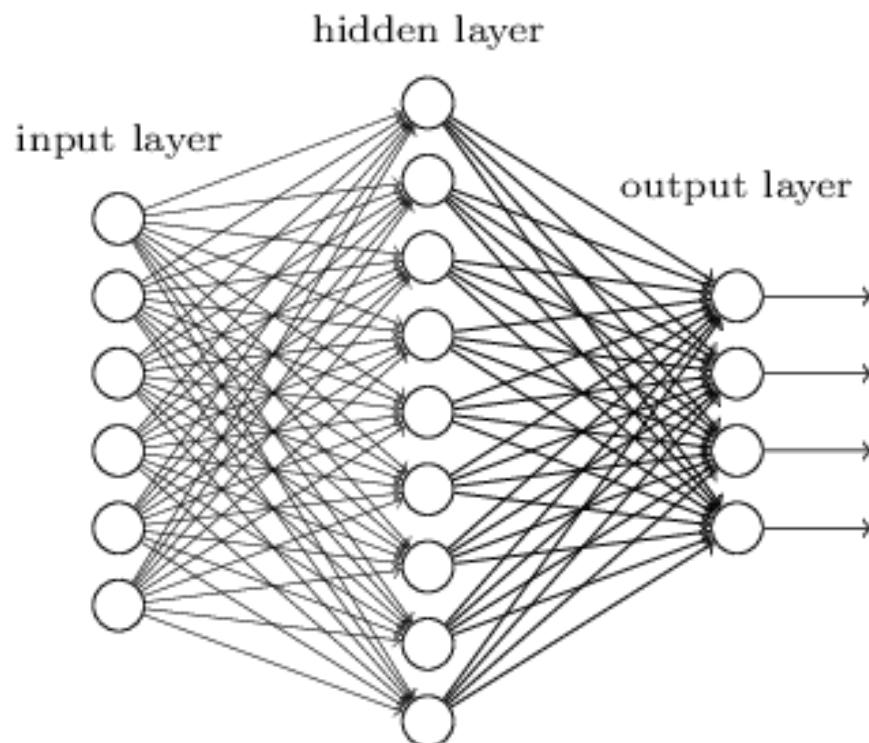
$$P(\mathbf{x} \in c | \mathbf{W}, \mathbf{b}) = \frac{y^c(\mathbf{x}; \mathbf{W}, \mathbf{b})}{\sum_k y^k(\mathbf{x}; \mathbf{W}, \mathbf{b})} .$$

Given a training set $\{\mathbf{x}_n, t_n^1, \dots, t_n^C\}_{1 \leq n \leq N}$ where $t_n^c \in \{0, 1\}$, minimize

$$E(\mathbf{W}, \mathbf{b}) = - \sum_n \sum_c t_n^c \log(P(\mathbf{x}_n \in c | \mathbf{W}, \mathbf{b})) .$$

Since E remains a differentiable function of \mathbf{W} and \mathbf{b} , this can also be done using a gradient-based technique, also known as back propagation.

HINGE LOSS OR RELU



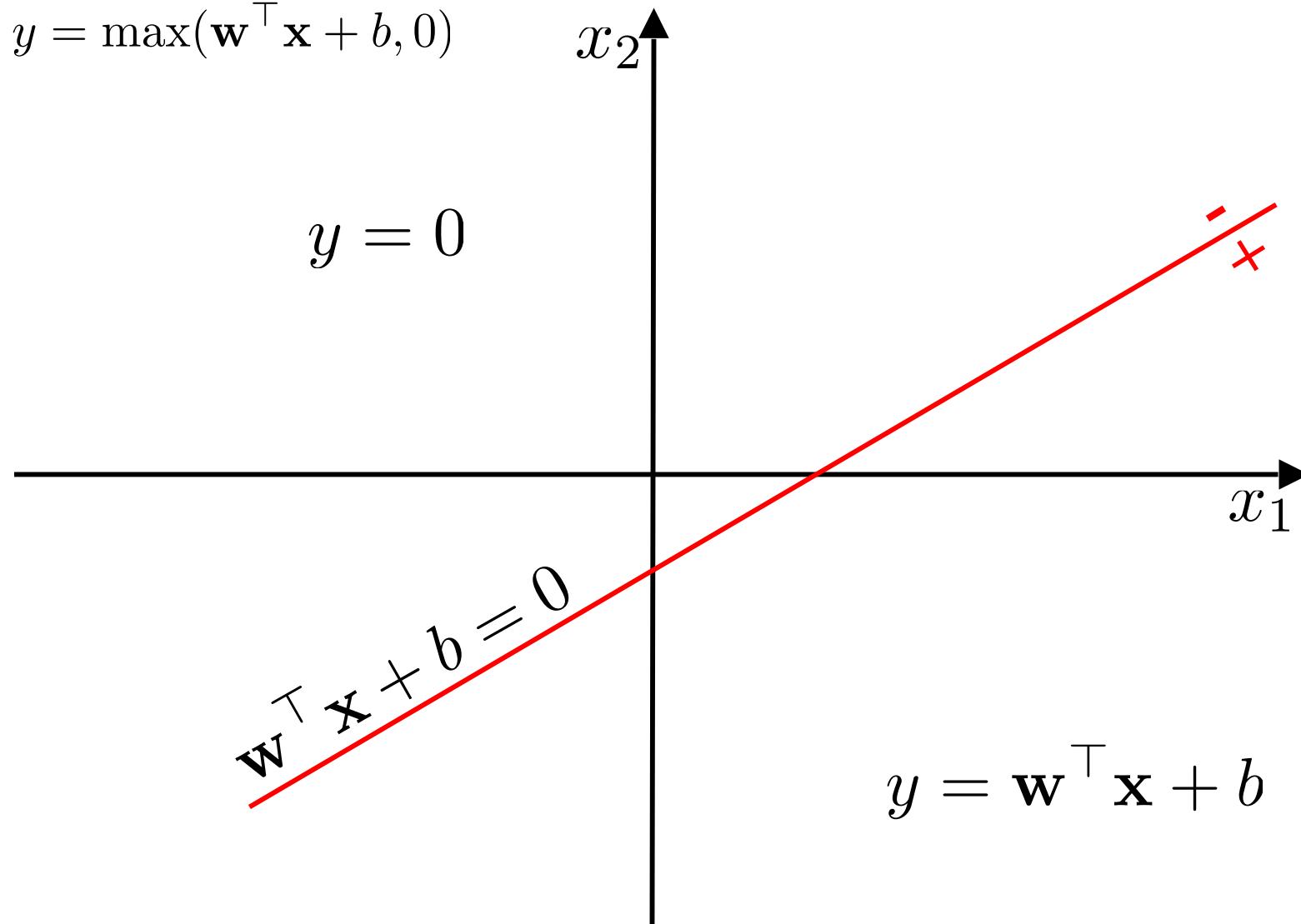
$$\begin{aligned} \mathbf{h} &= \sigma(\mathbf{W}_1 \mathbf{x} + \mathbf{b}_1) \\ \mathbf{y} &= \sigma(\mathbf{W}_2 \mathbf{h} + \mathbf{b}_2) \end{aligned}$$

$$\sigma(\mathbf{x}) = \max(0, \mathbf{x}).$$

- Each node defines a hyperplane.
- The resulting function is piecewise linear affine and continuous.

ONE SINGLE HYPERPLANE

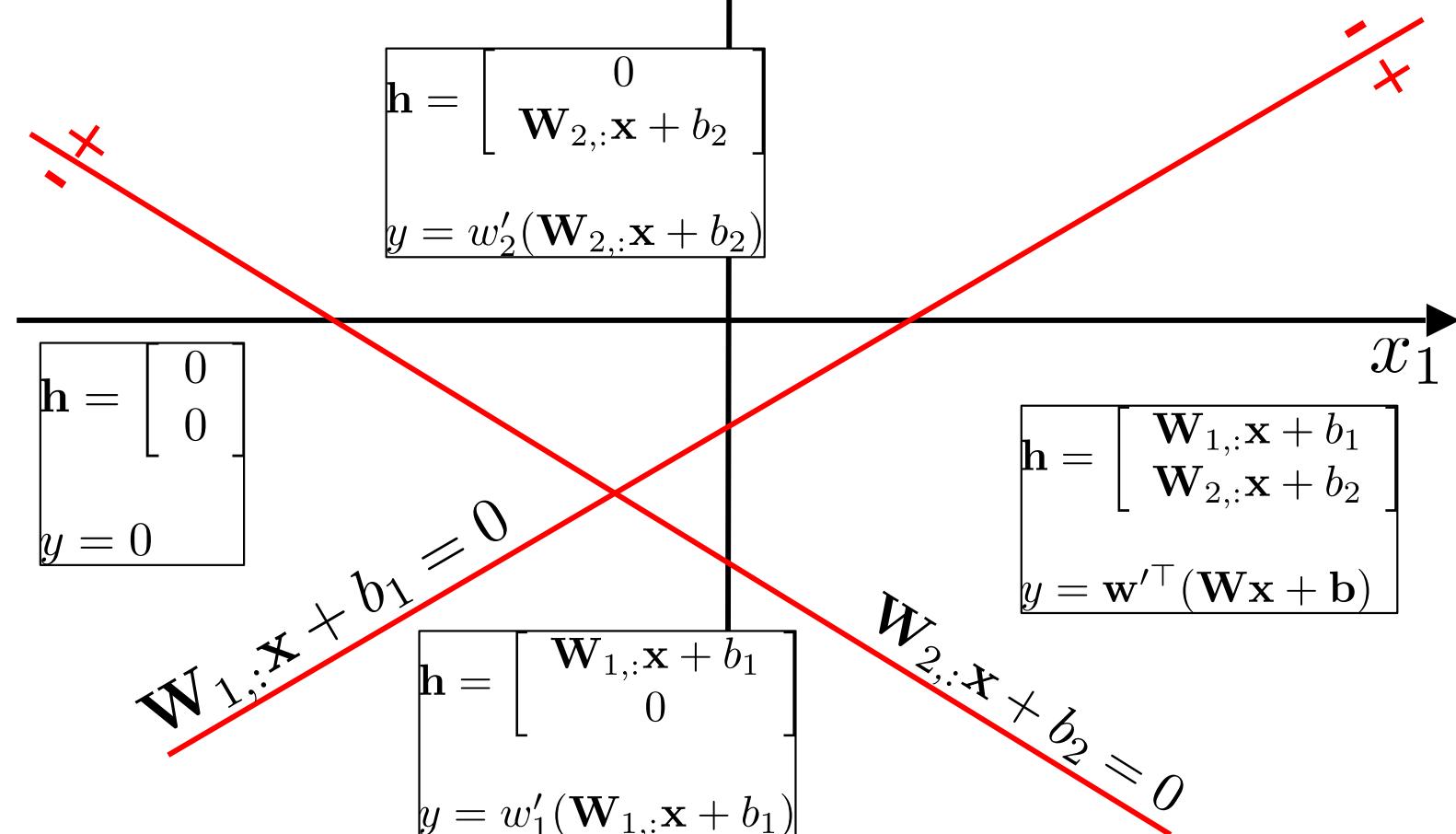
$$y = \max(\mathbf{w}^\top \mathbf{x} + b, 0)$$



TWO HYPERPLANES

$$\begin{cases} \mathbf{h} = \max(\mathbf{W}\mathbf{x} + \mathbf{b}, 0) \\ y = \mathbf{w}'^\top \mathbf{h} \end{cases}$$

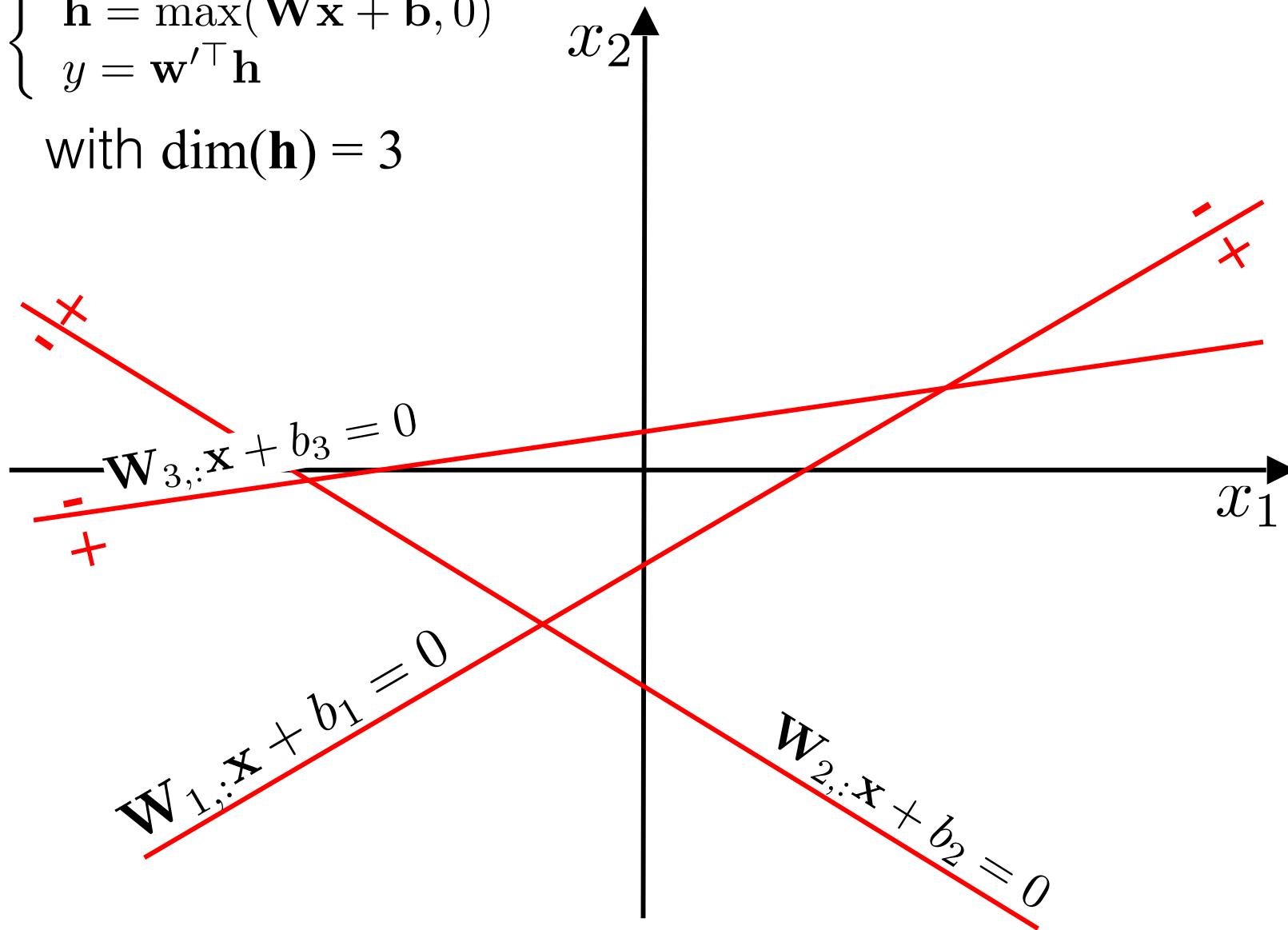
with $\dim(\mathbf{h}) = 2$



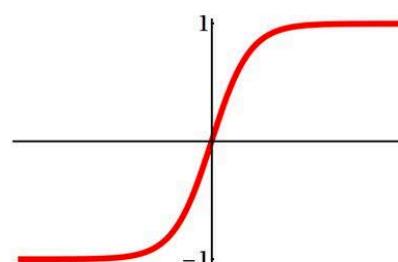
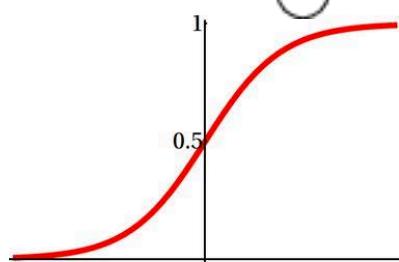
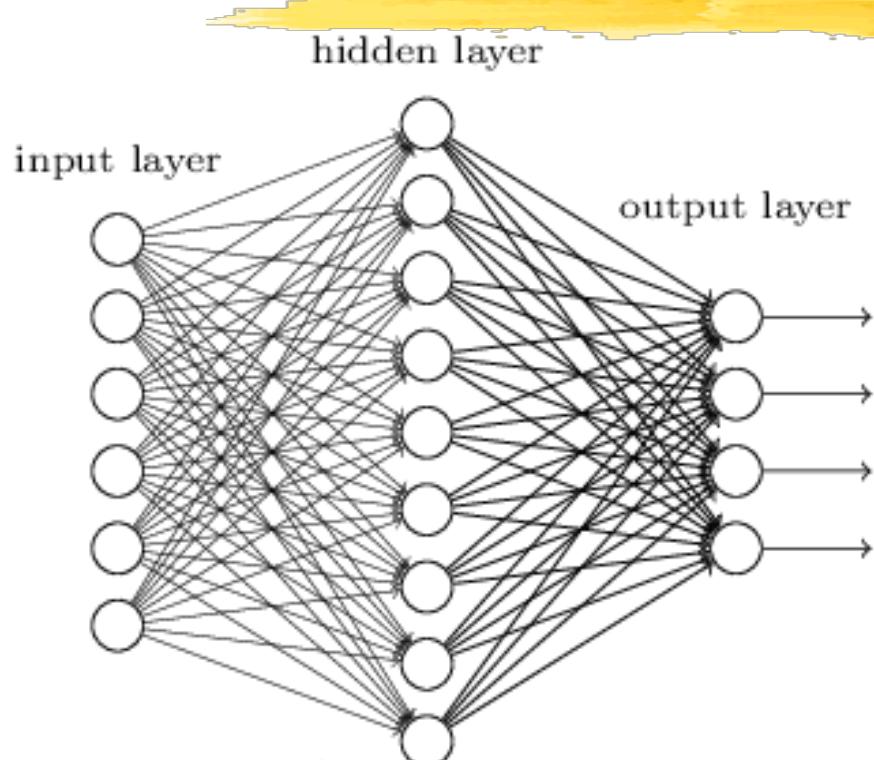
THREE HYPERPLANES

$$\begin{cases} \mathbf{h} = \max(\mathbf{W}\mathbf{x} + \mathbf{b}, 0) \\ y = \mathbf{w}'^\top \mathbf{h} \end{cases}$$

with $\dim(\mathbf{h}) = 3$



SIGMOID AND TANH

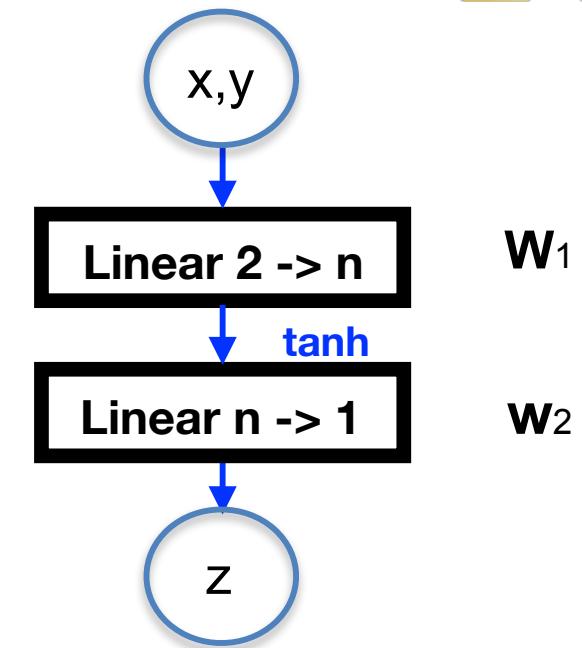
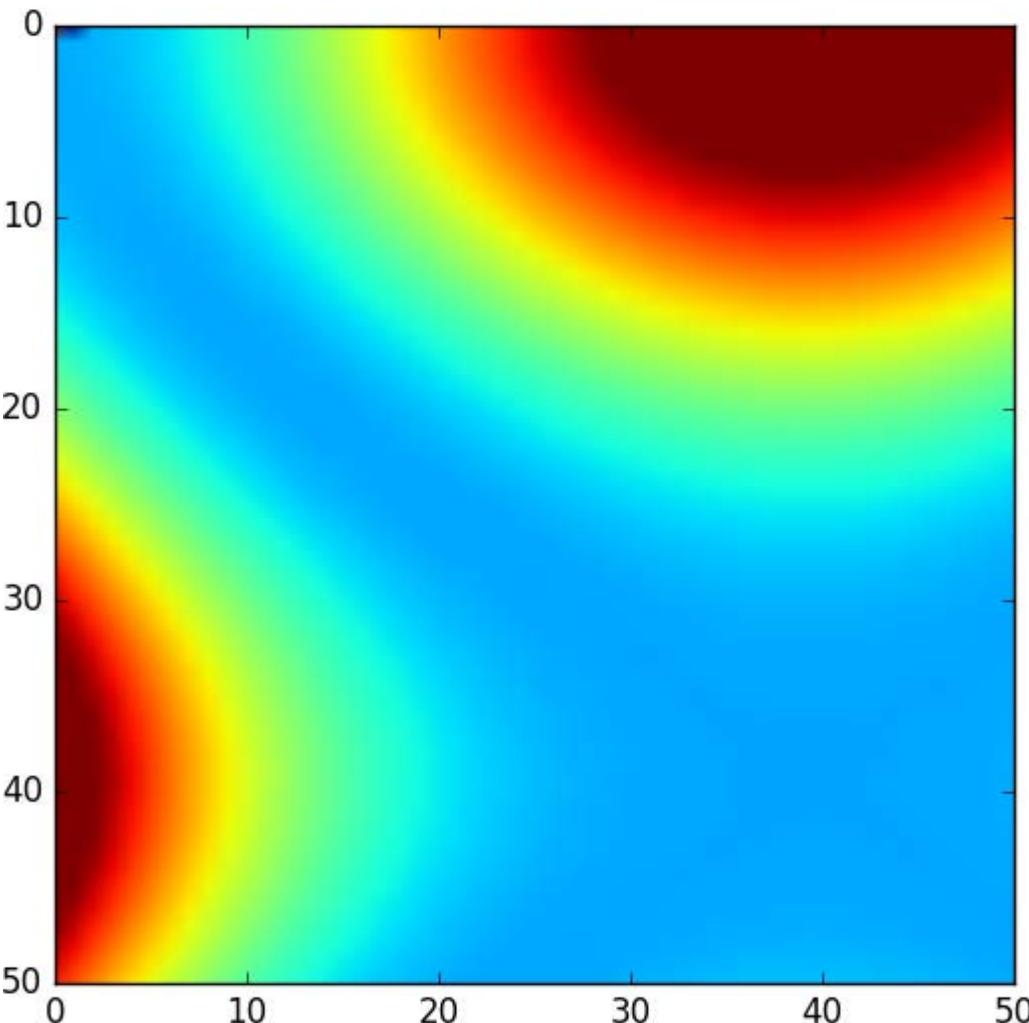


$$\begin{aligned} \mathbf{h} &= \sigma(\mathbf{W}_1 \mathbf{x} + \mathbf{b}_1) \\ \mathbf{y} &= \sigma(\mathbf{W}_2 \mathbf{h} + \mathbf{b}_2) \end{aligned}$$

$$\begin{aligned} \text{sigm: } \sigma(x) &= \frac{1}{1 + \exp(-x)} \\ \text{tanh: } \sigma(x) &= \frac{\exp(x) - \exp(-x)}{\exp(x) + \exp(-x)} \end{aligned}$$

- Each node defines a hyperplane.
- The resulting function is continuously differentiable.

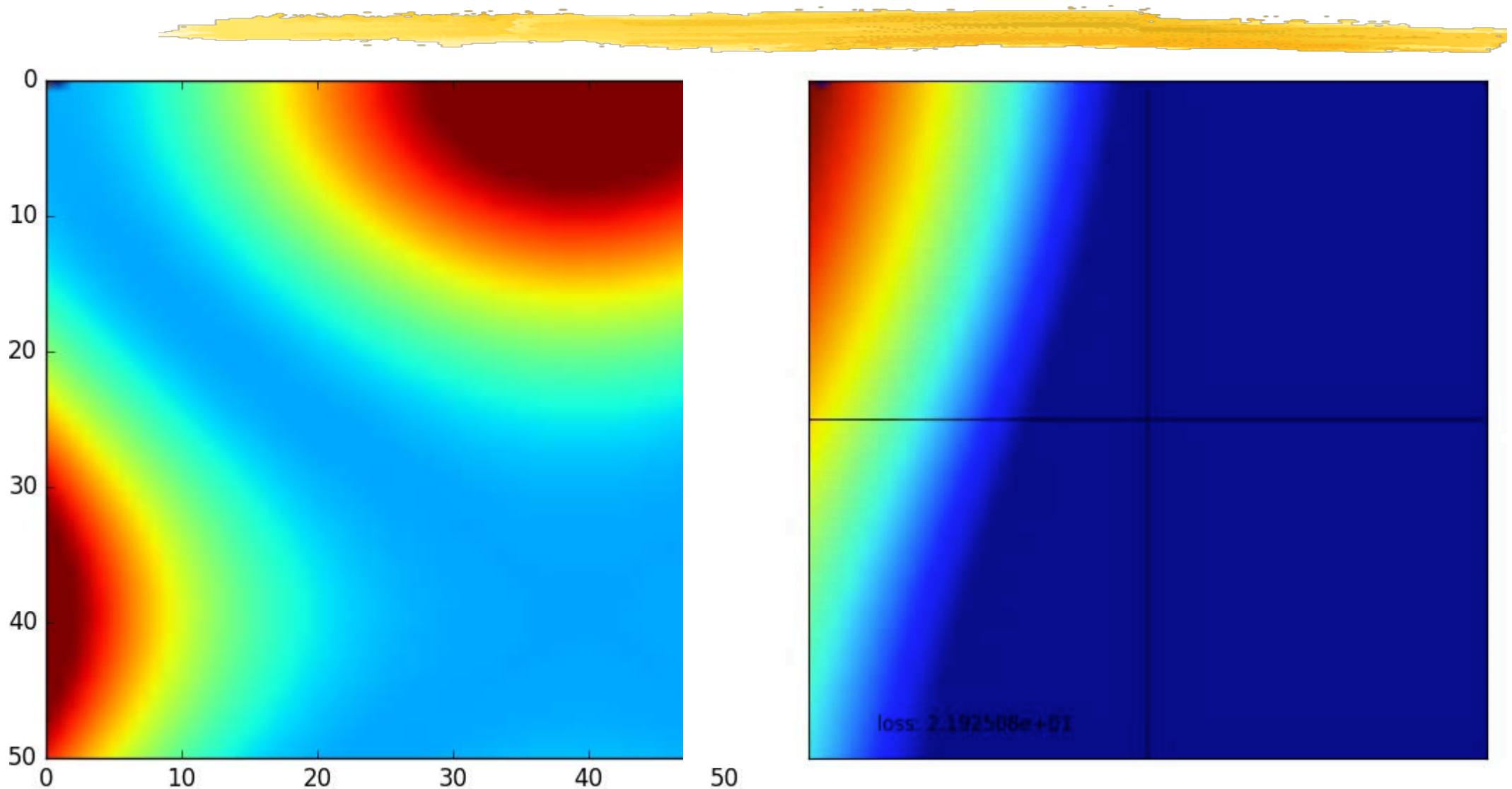
INTERPOLATING A SURFACE



$$\begin{aligned} z &= f(x, y) \\ &= \mathbf{w}_2 \sigma(\mathbf{W}_1 \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} b_x \\ b_y \end{bmatrix}) + b_z \end{aligned}$$

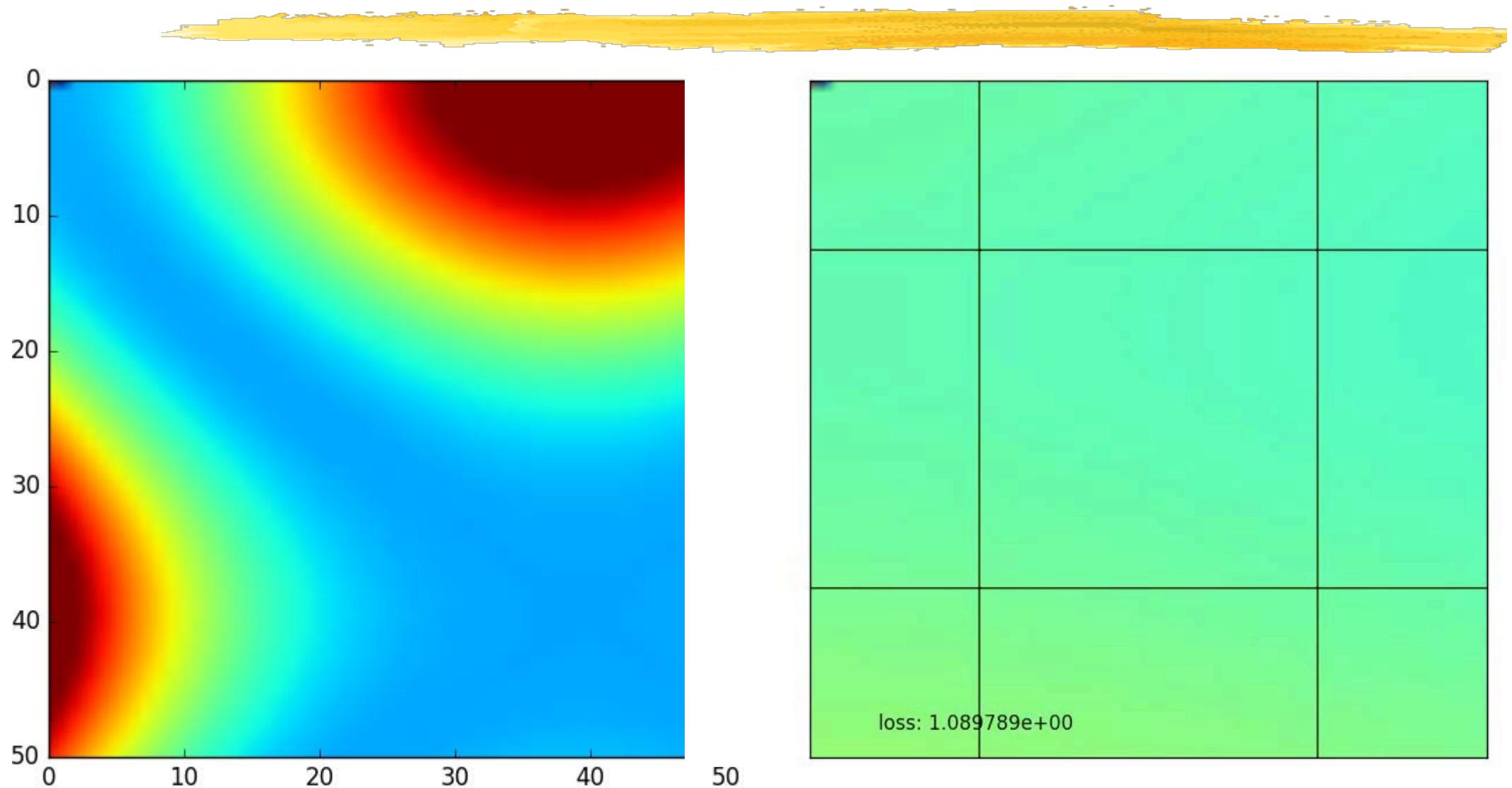
Minimize $\sum_i (z_i - f(x_i, y_i))^2$,
with respect to $\mathbf{W}_1, \mathbf{w}_2, b_x, b_y, b_z$.

INTERPOLATING A SURFACE



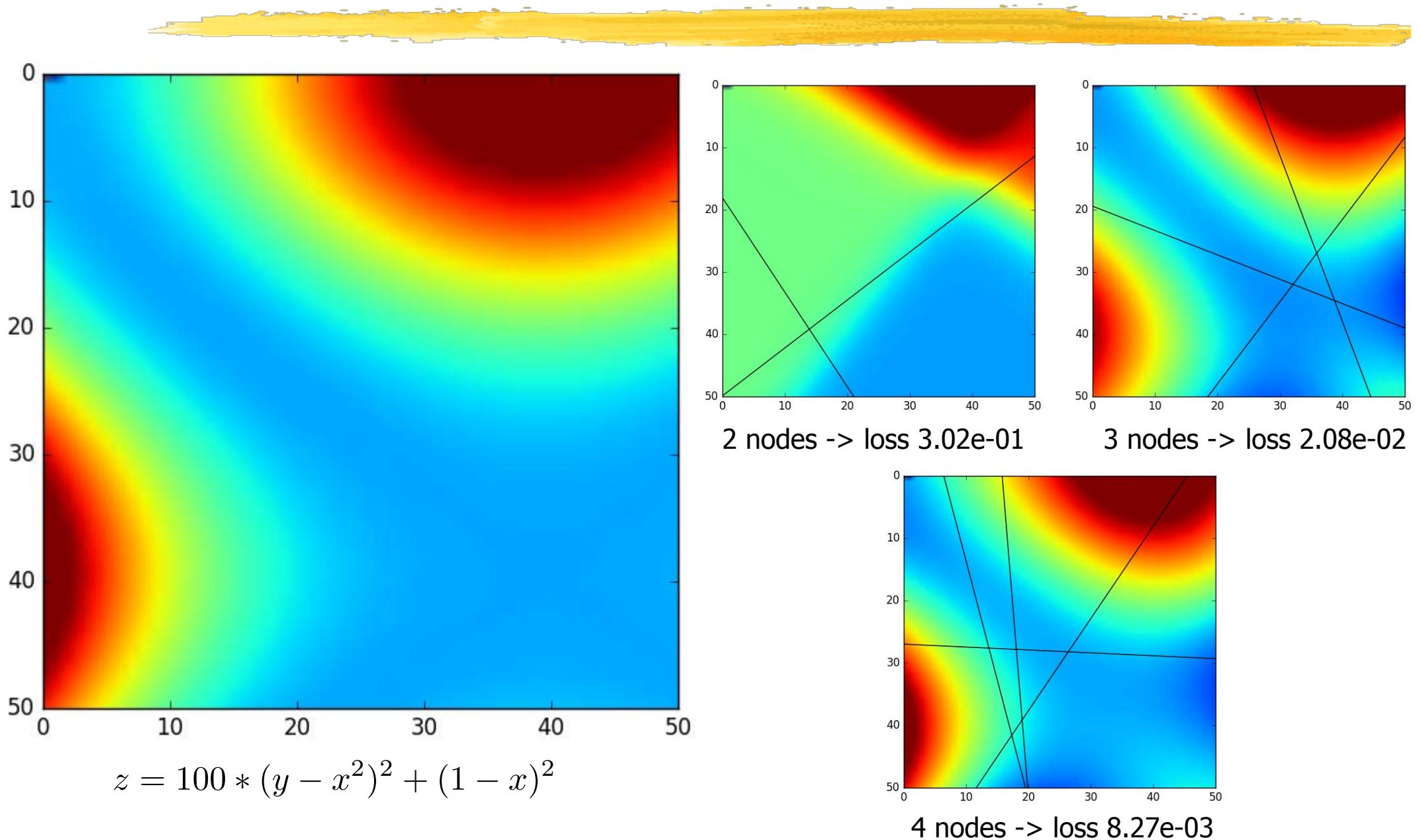
3-node hidden layer

INTERPOLATING A SURFACE

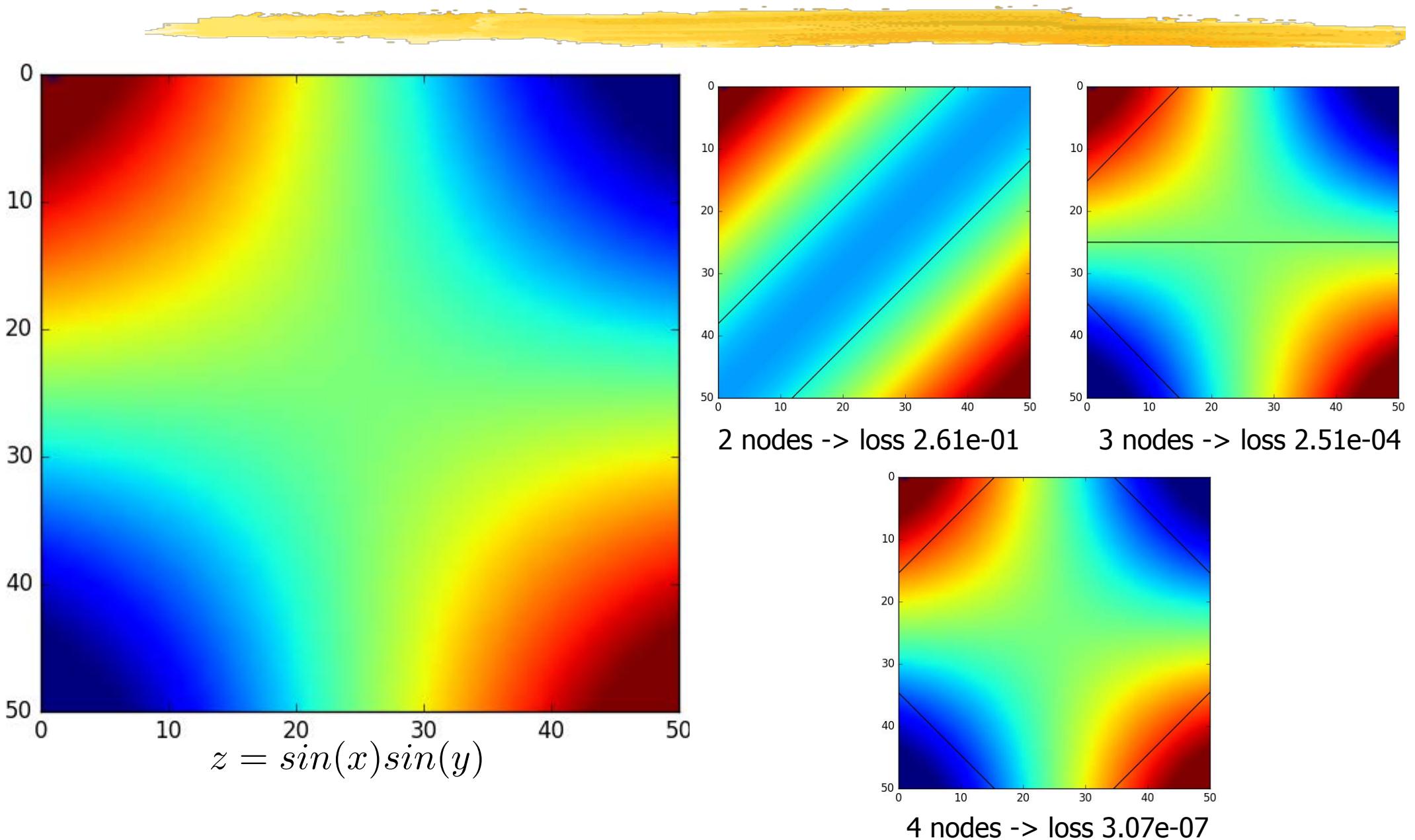


4-node hidden layer

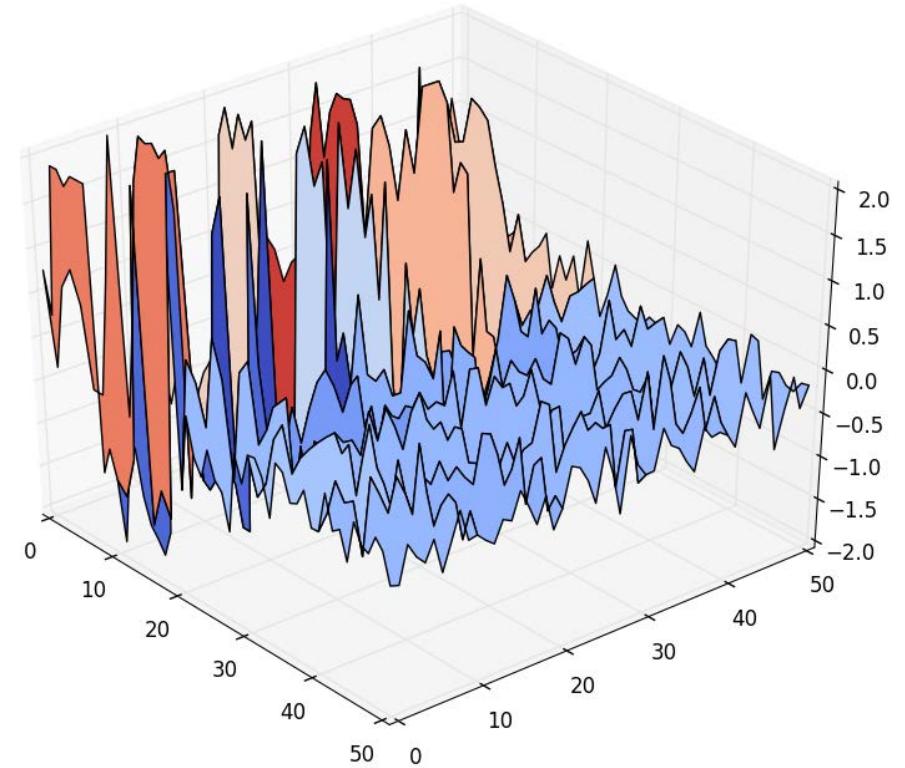
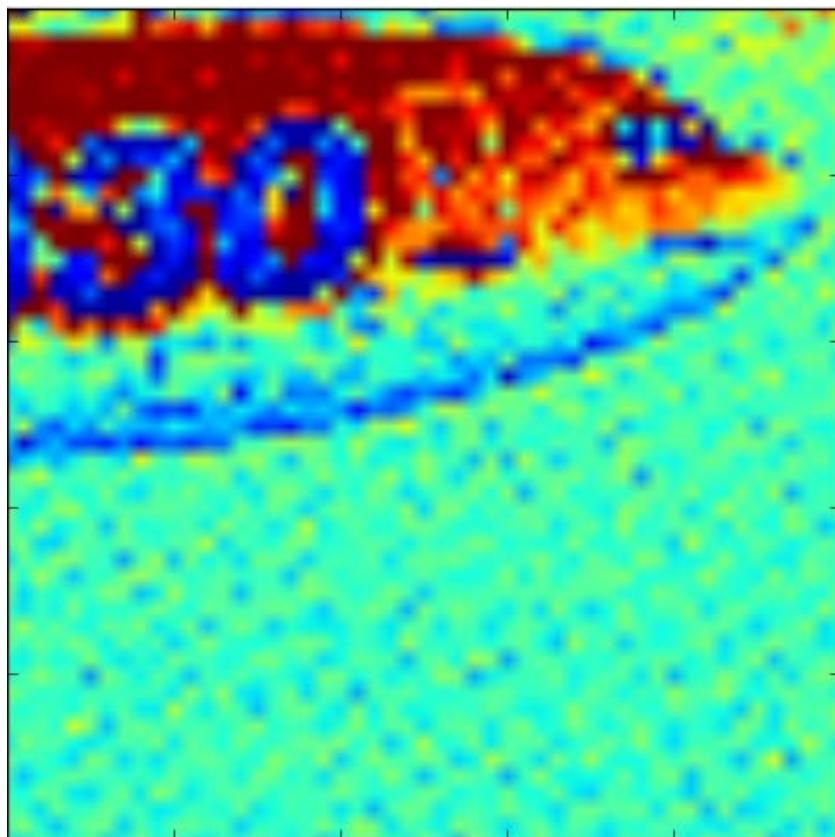
ADDING MORE NODES



ADDING MORE NODES

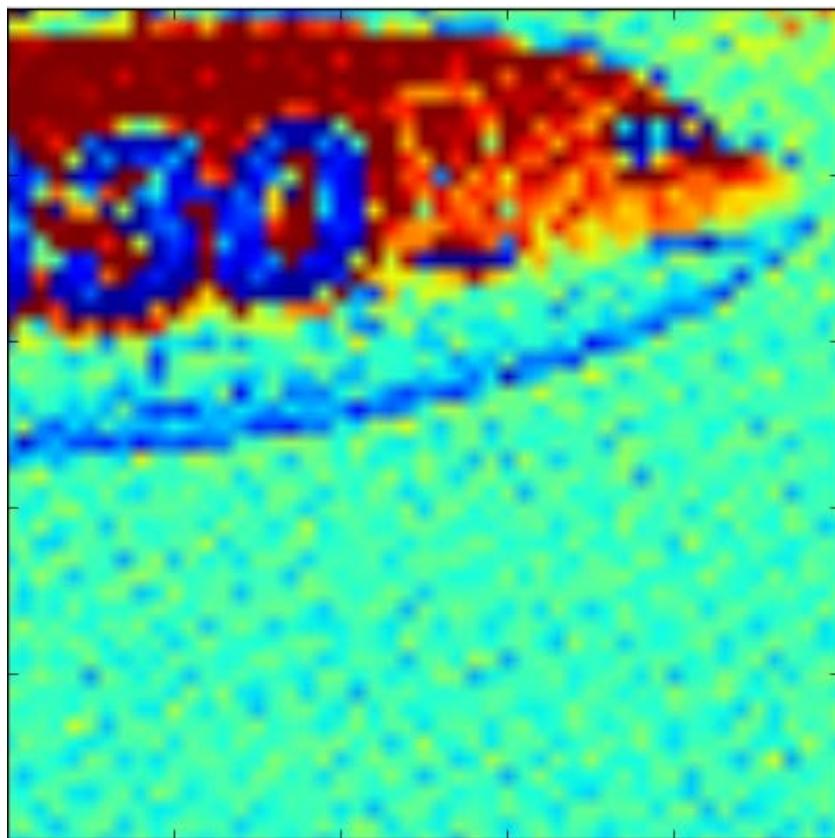


MORE COMPLEX SURFACE

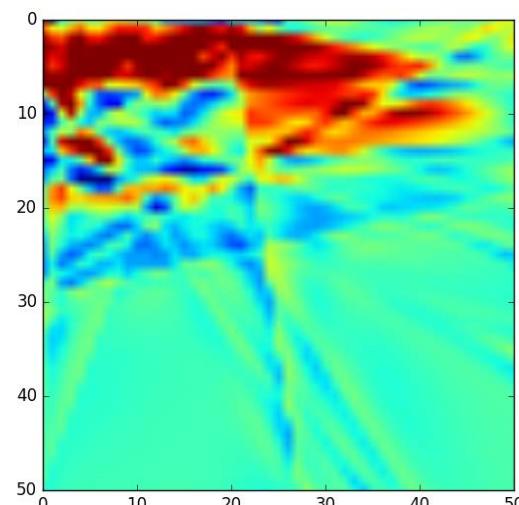


$$I = f(x, y)$$

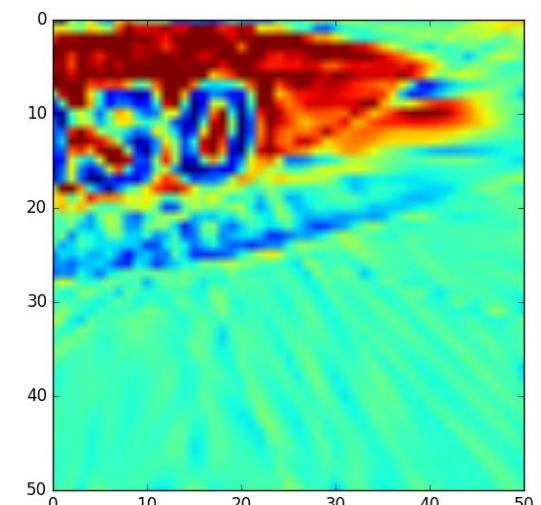
ADDING MORE NODES



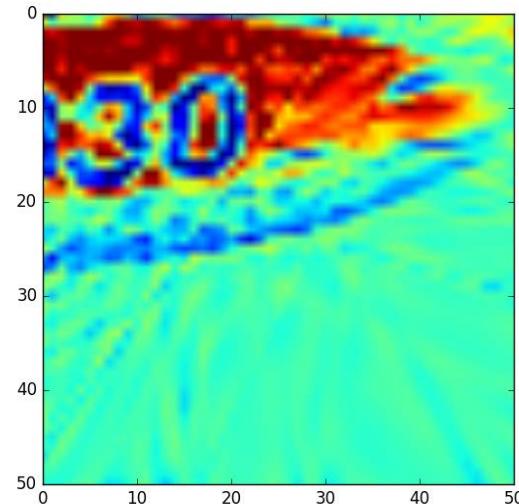
$$I = f(x, y)$$



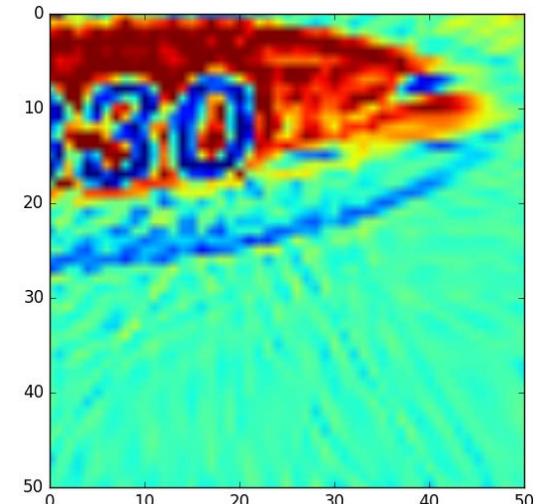
50 nodes -> loss 3.65e-01



100 nodes -> loss 2.50e-01



125 nodes -> loss 2.40e-01



300 nodes -> loss 1.92e-01

UNIVERSAL APPROXIMATION THEOREM

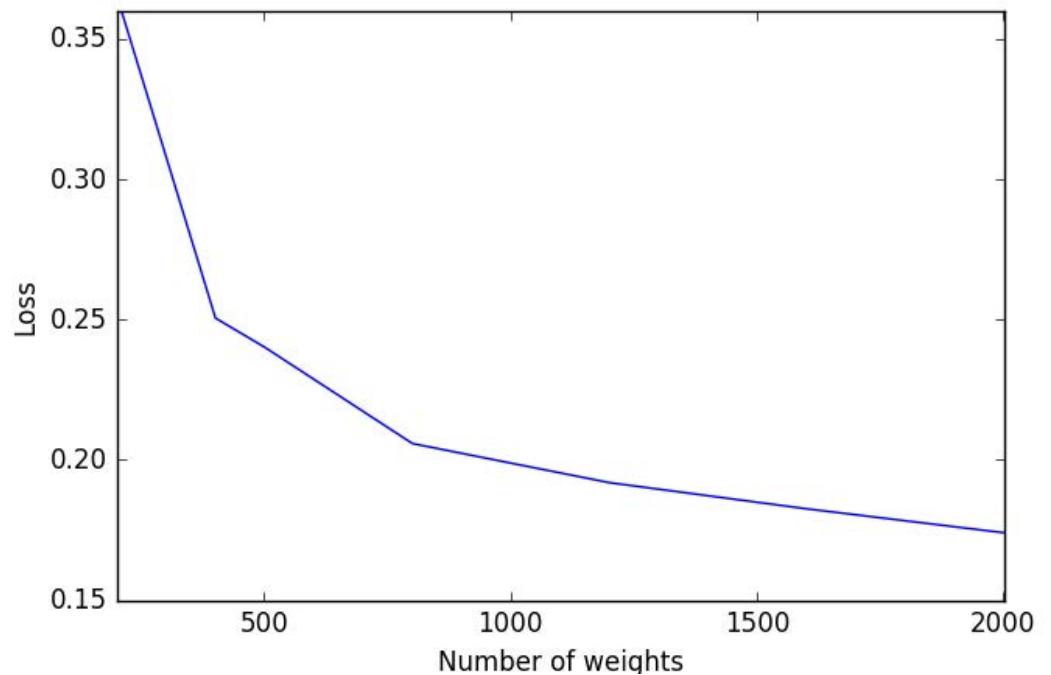
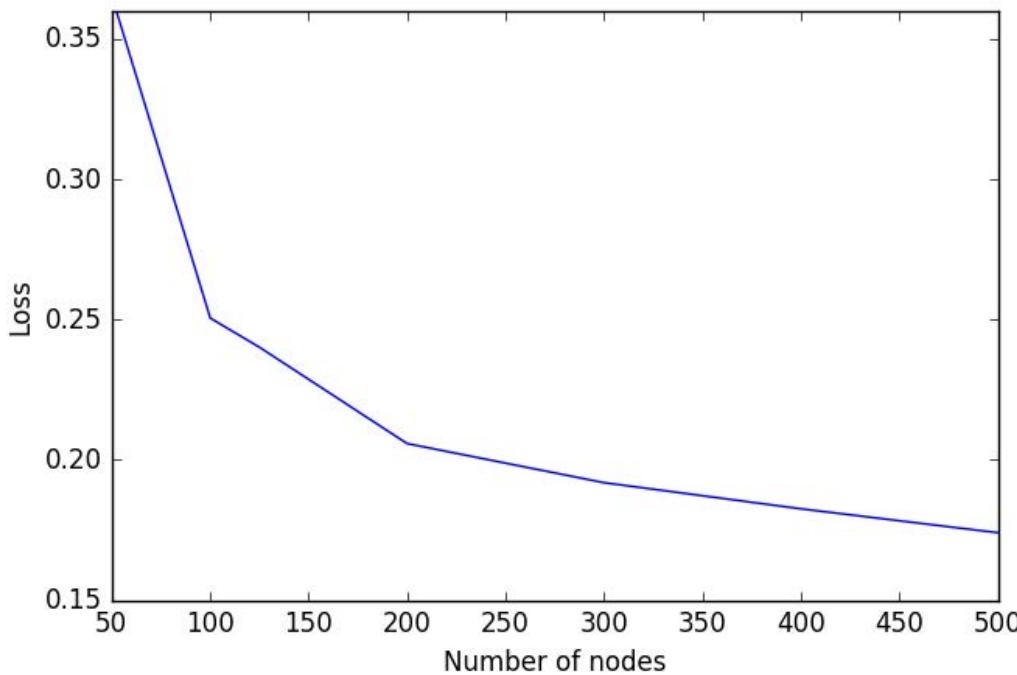


A feedforward network with a linear output layer and at least one hidden layer with any 'squashing' activation function (e.g. logistic sigmoid) can approximate any Borel measurable function (from one finite-dimensional space to another) with any desired nonzero error.

Any continuous function on a closed and bounded set of R^n is Borel-measurable.

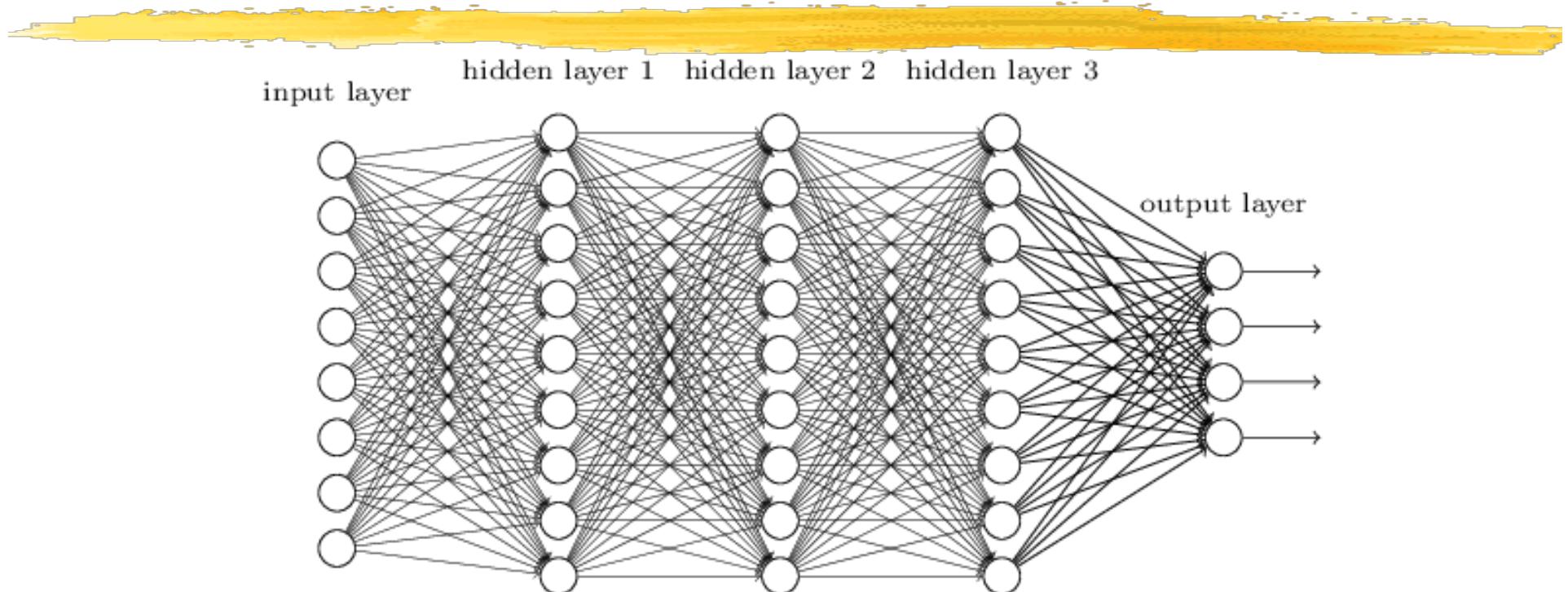
→ In theory, any reasonable function can be approximated by a two-layer network as long as it is continuous.

IN PRACTICE



- It may take an exponentially large number of parameters for a good approximation.
 - The optimization problem becomes increasingly difficult.
- > The one hidden layer perceptron may not converge to the best solution!

DEEP LEARNING

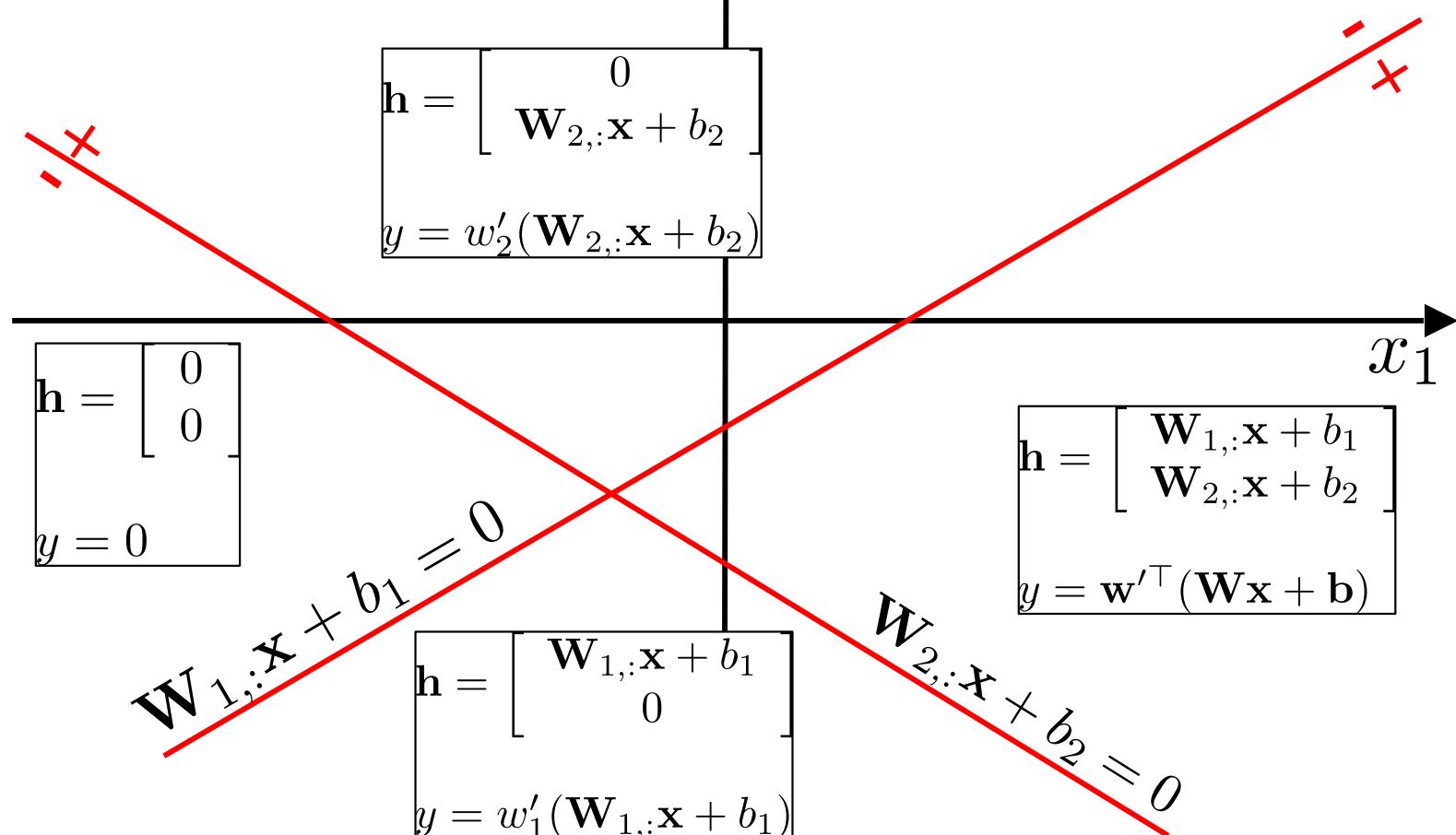


- The descriptive power of the net increases with the number of layers.
- In the case of a 1D signal, it is roughly proportional to $\prod_n W_n$ where W_n represents the width of a layer.

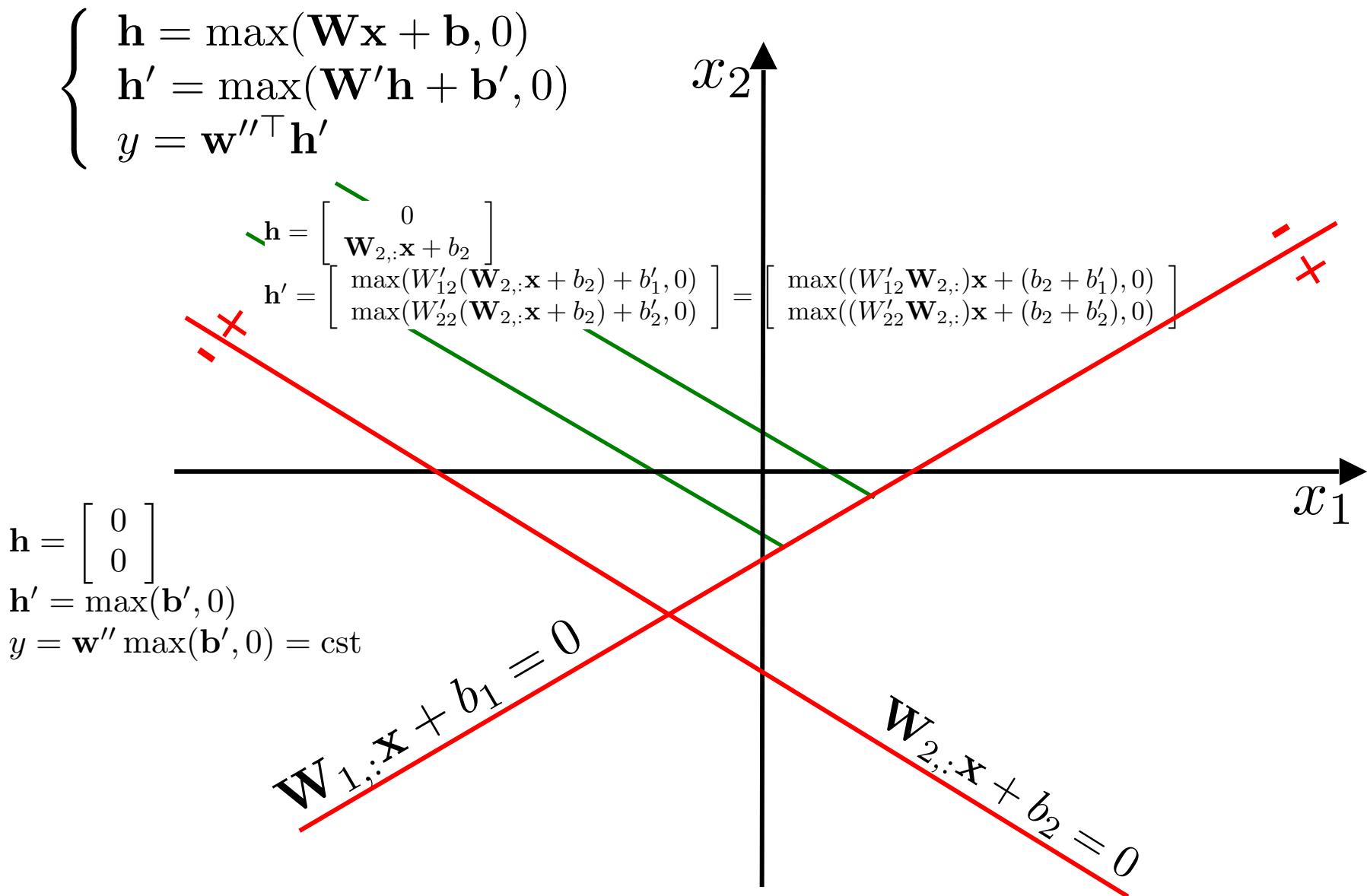
ONE LAYER: TWO HYPERPLANES

$$\begin{cases} \mathbf{h} = \max(\mathbf{W}\mathbf{x} + \mathbf{b}, 0) \\ y = \mathbf{w}'^\top \mathbf{h} \end{cases}$$

with $\dim(\mathbf{h}) = 2$

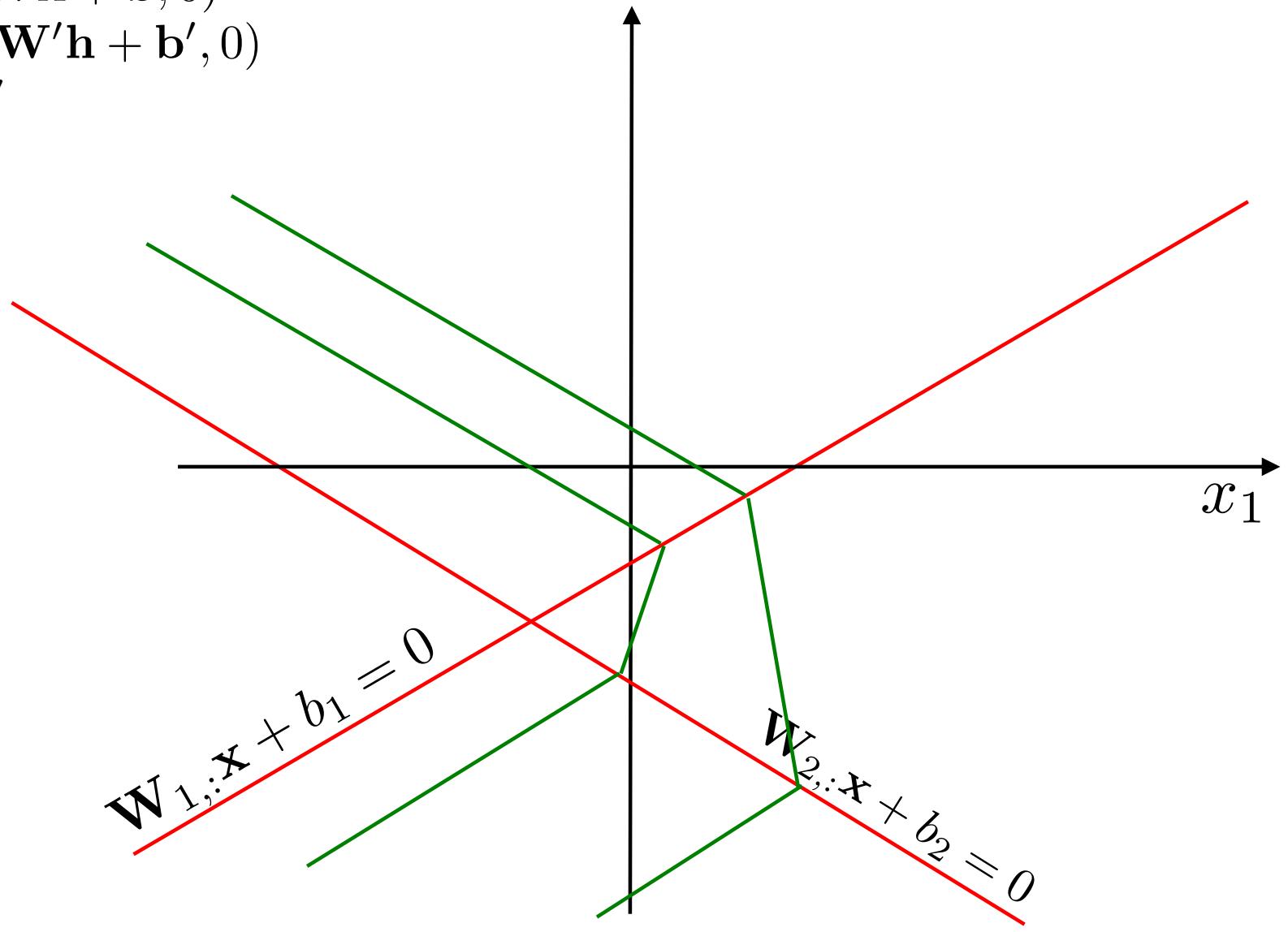


TWO LAYERS: TWO HYPERPLANES PER LAYER

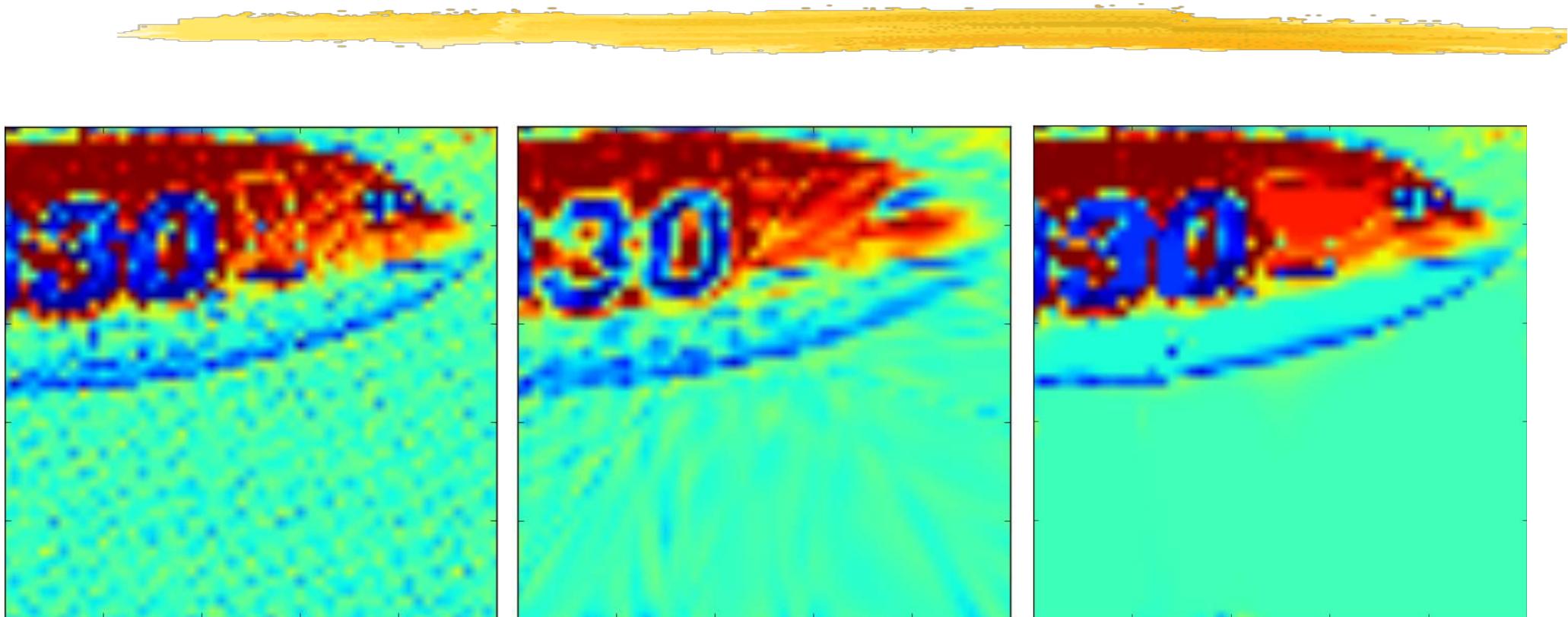


TWO LAYERS: TWO HYPERPLANES PER LAYER

$$\begin{cases} h = \max(Wx + b, 0) \\ h' = \max(W'h + b', 0) \\ y = w''^\top h' \end{cases}$$



ADDING A SECOND LAYER



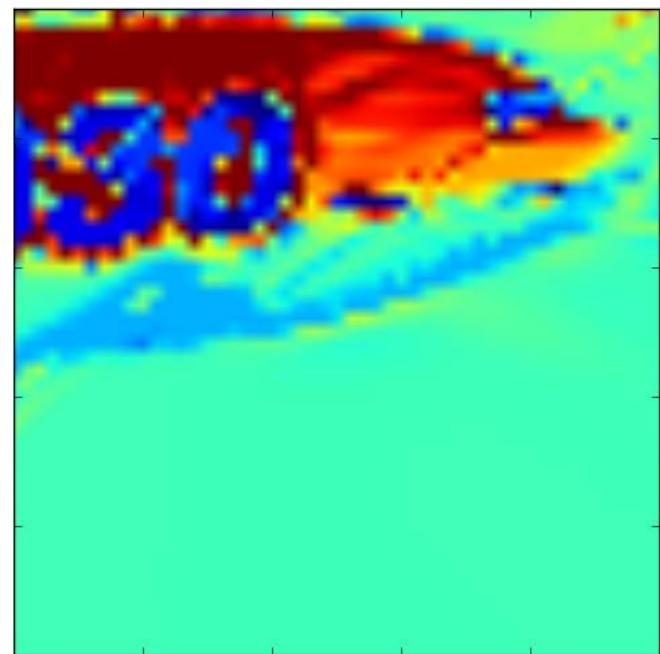
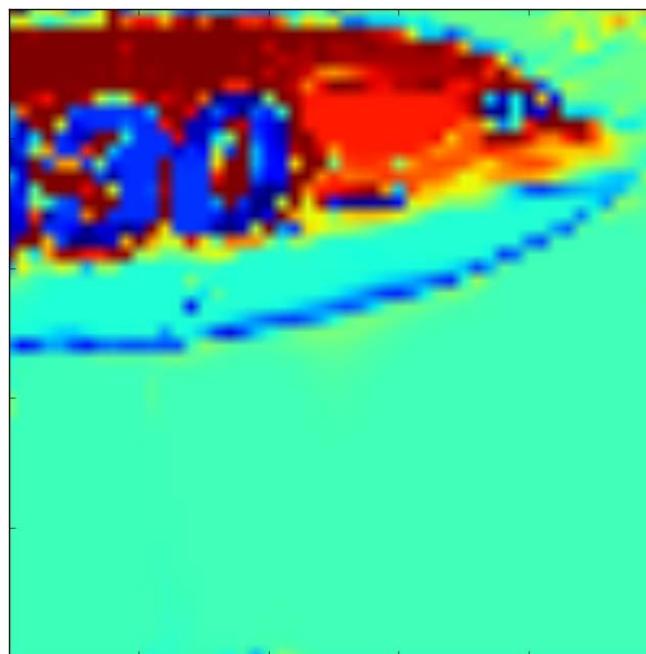
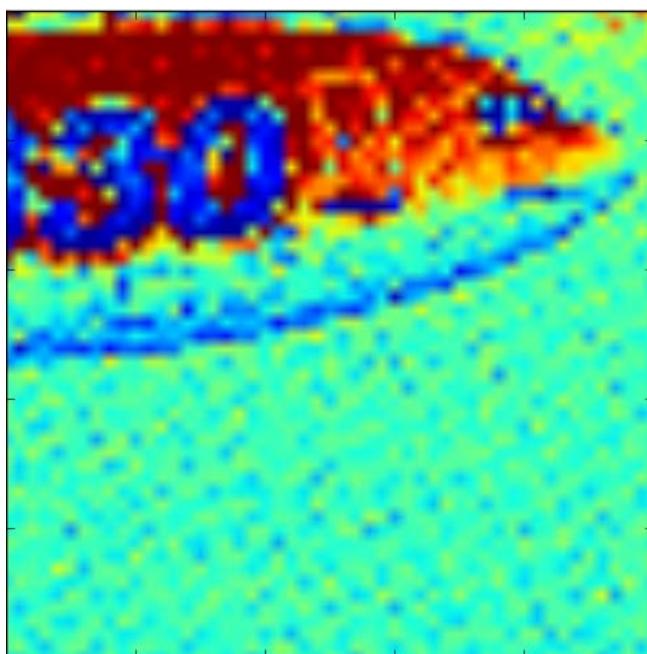
$$I = f(x, y)$$

1 Layer: 125 nodes -> loss 2.40e-01

2 Layers: 20 nodes -> loss 8.31e-02

501 weights in both cases

ADDING A THIRD LAYER



$$I = f(x, y)$$

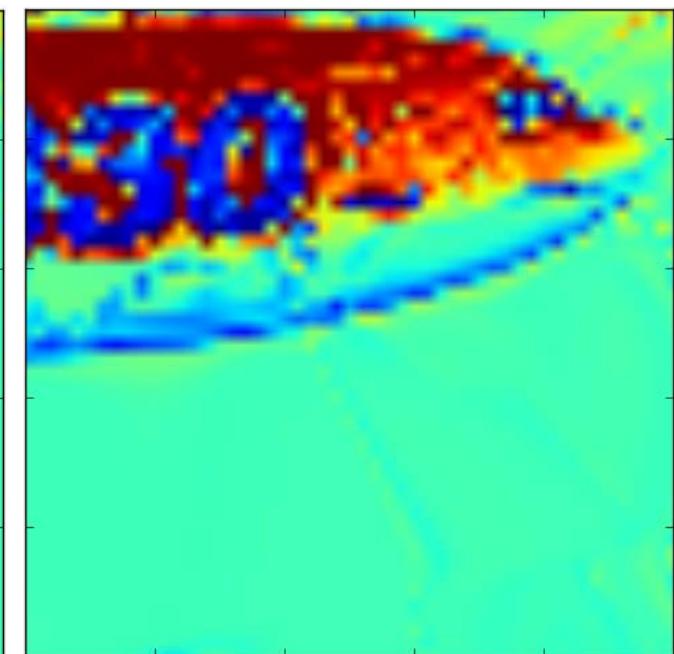
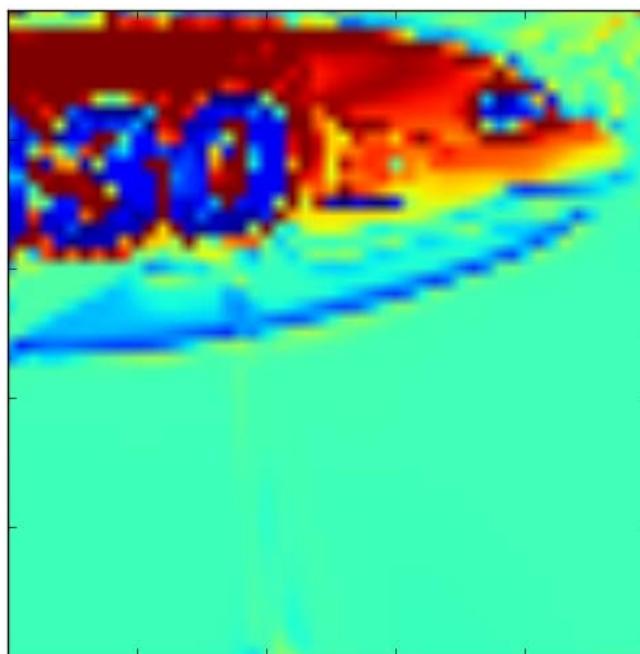
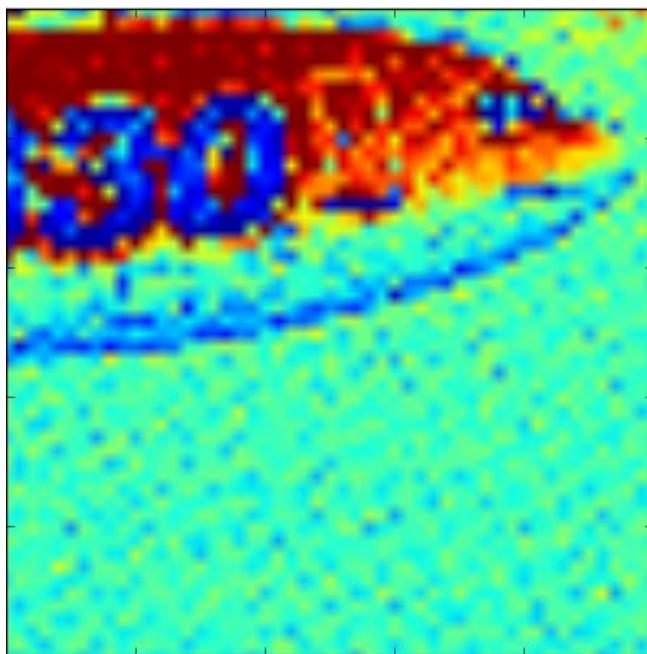
2 Layers: 20 nodes -> loss 8.31e-02

501 weights

3 Layers: 14 nodes -> loss 7.55e-02

477 weights

ADDING A THIRD LAYER



$$I = f(x, y)$$

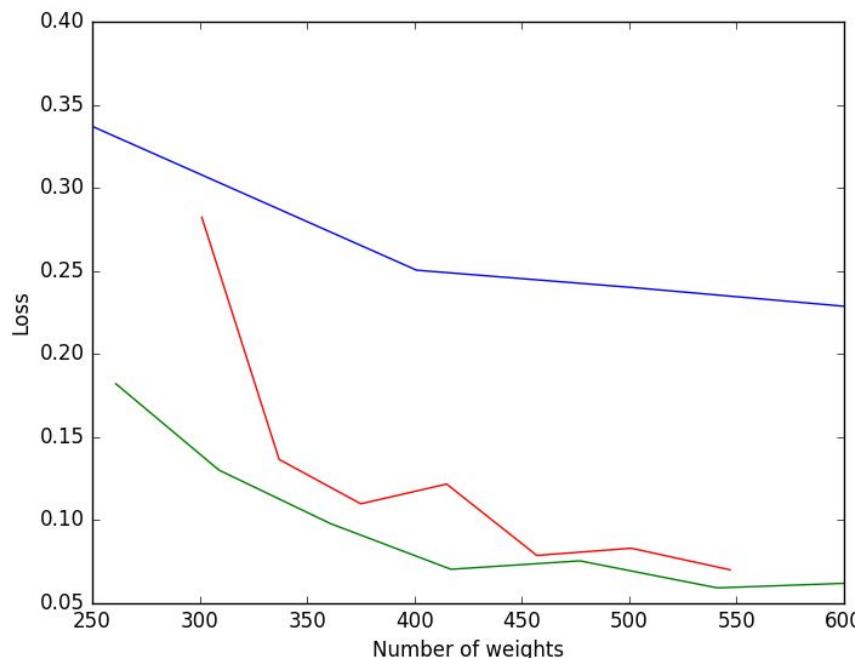
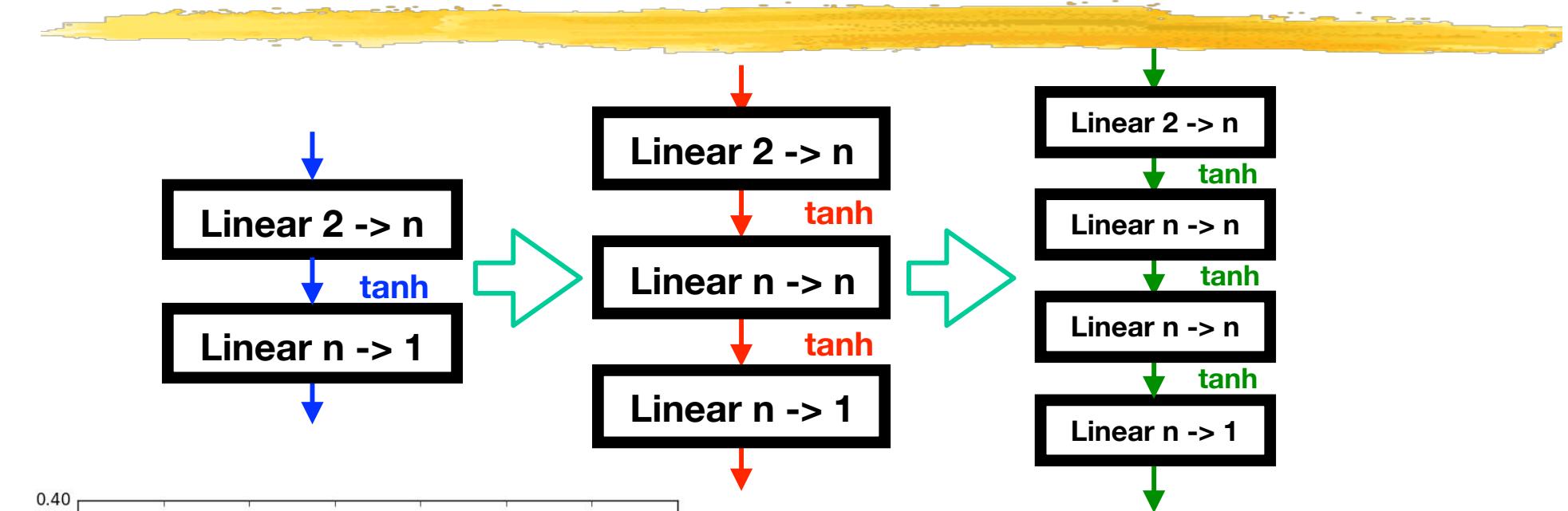
3 Layers: 15 nodes -> loss 5.93e-02

541 weights

3 Layers: 19 nodes -> loss 4.38e-02

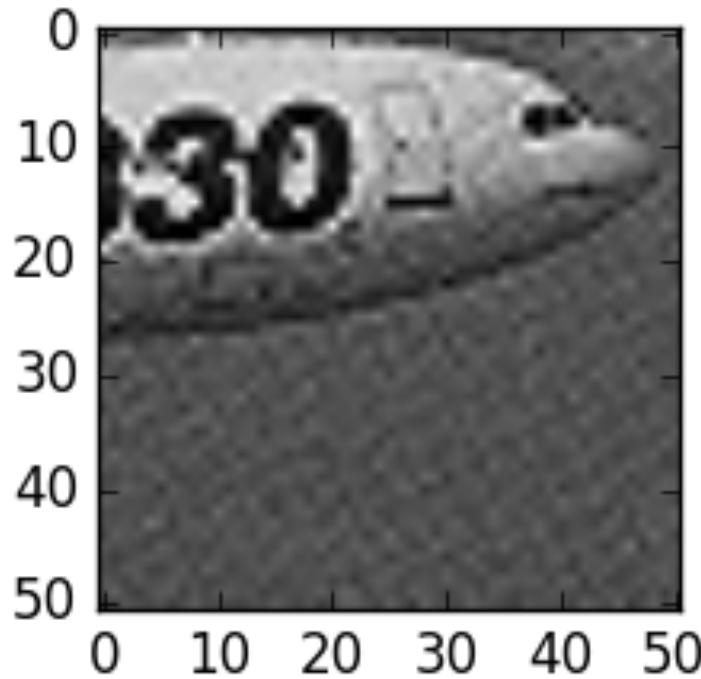
837 weights

MULTILAYER PERCEPTRONS

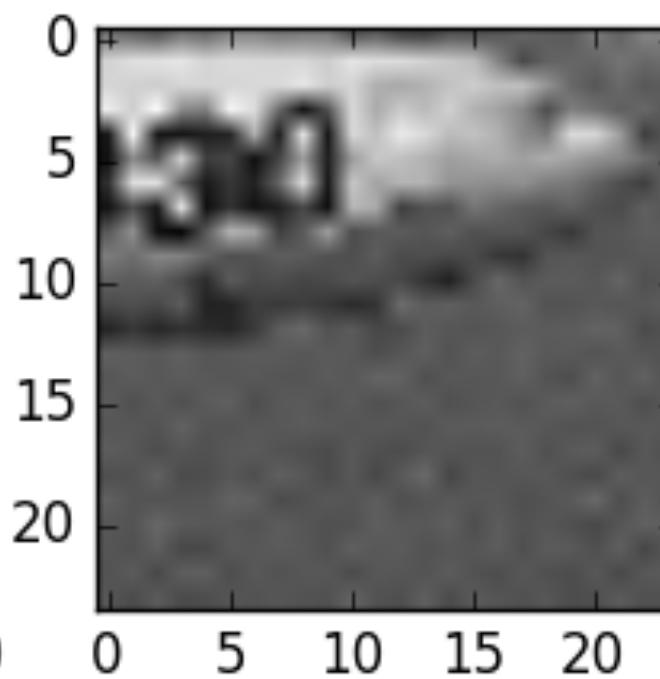


- Adding layers often yields better convergence properties.
- In current practice, deeper is usually better.

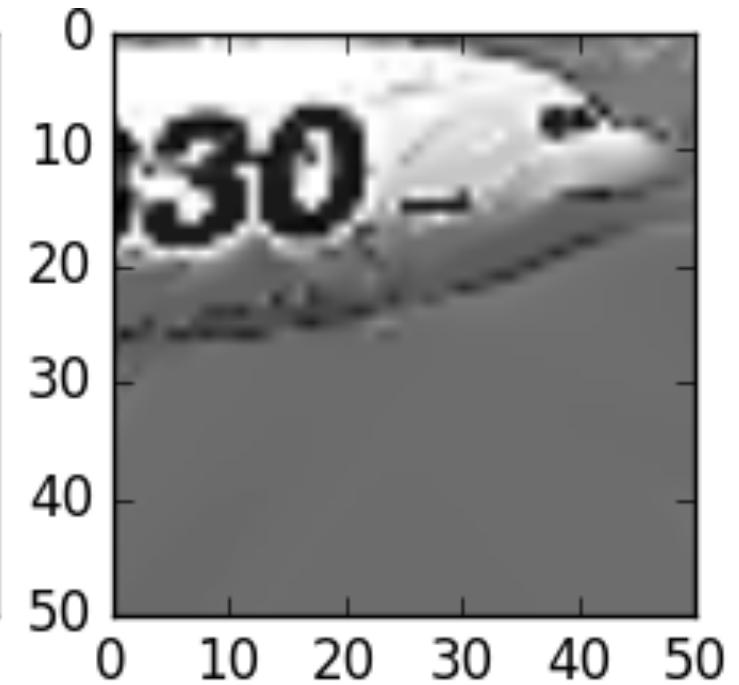
SIMPLER WAY TO INTERPOLATE



Original 51x51 image:
2601 gray level values.



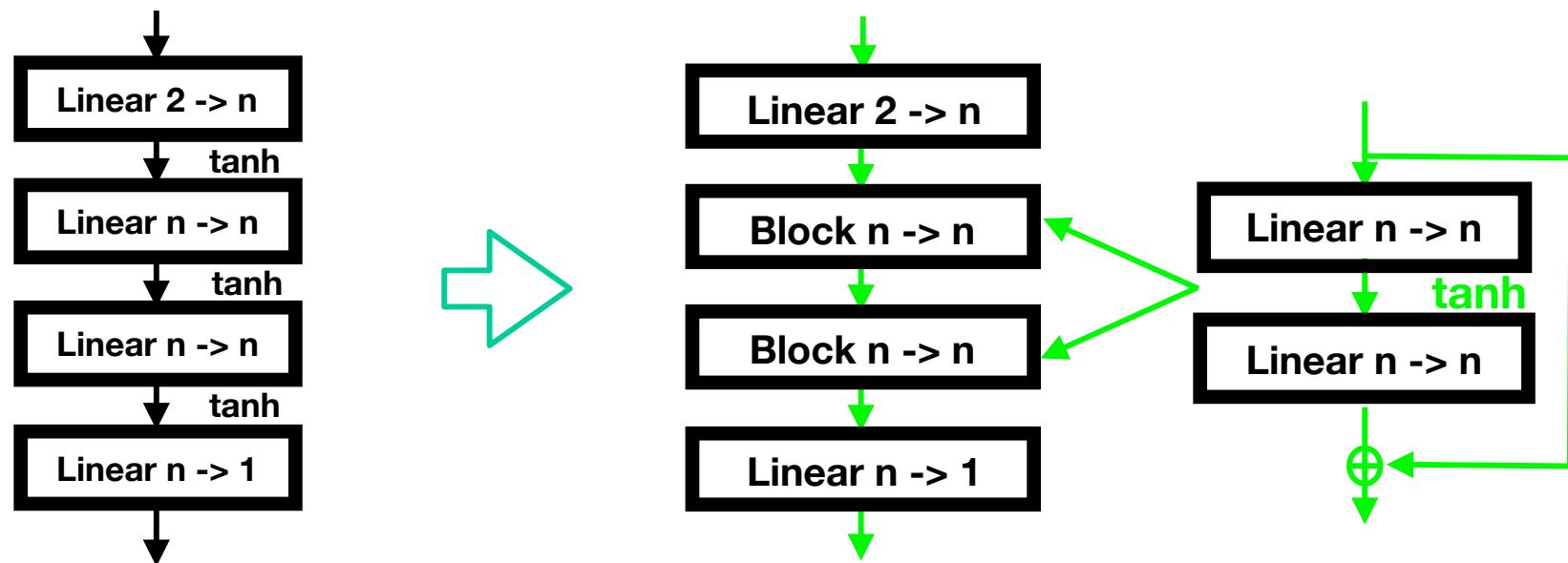
Scaled 24x24 image:
576 gray level values.



MLP 10/20/10 Interpolation:
471 weights.

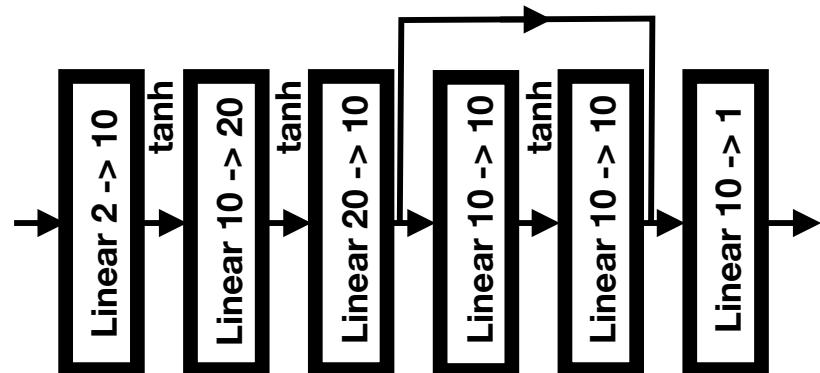
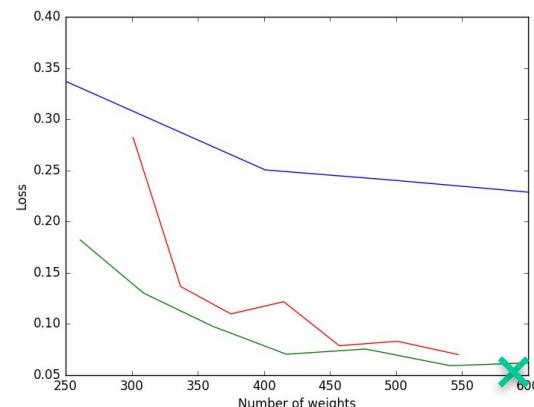
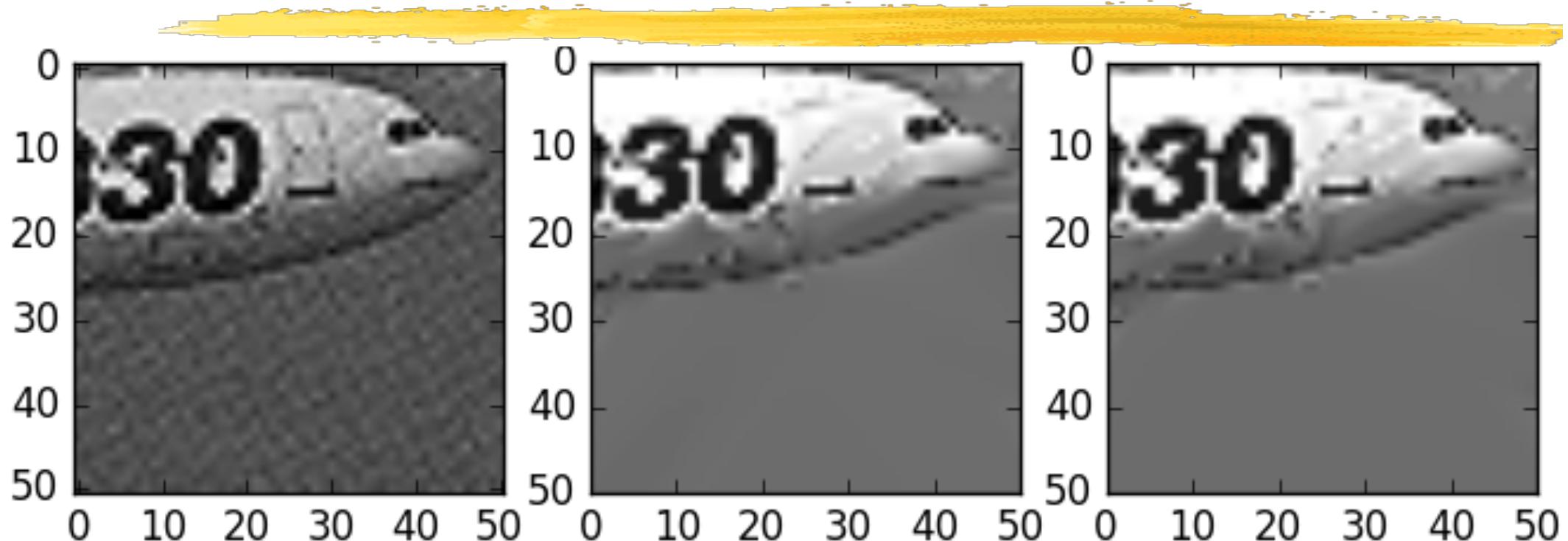
Simpler but not necessarily better!

MLP TO RESNET

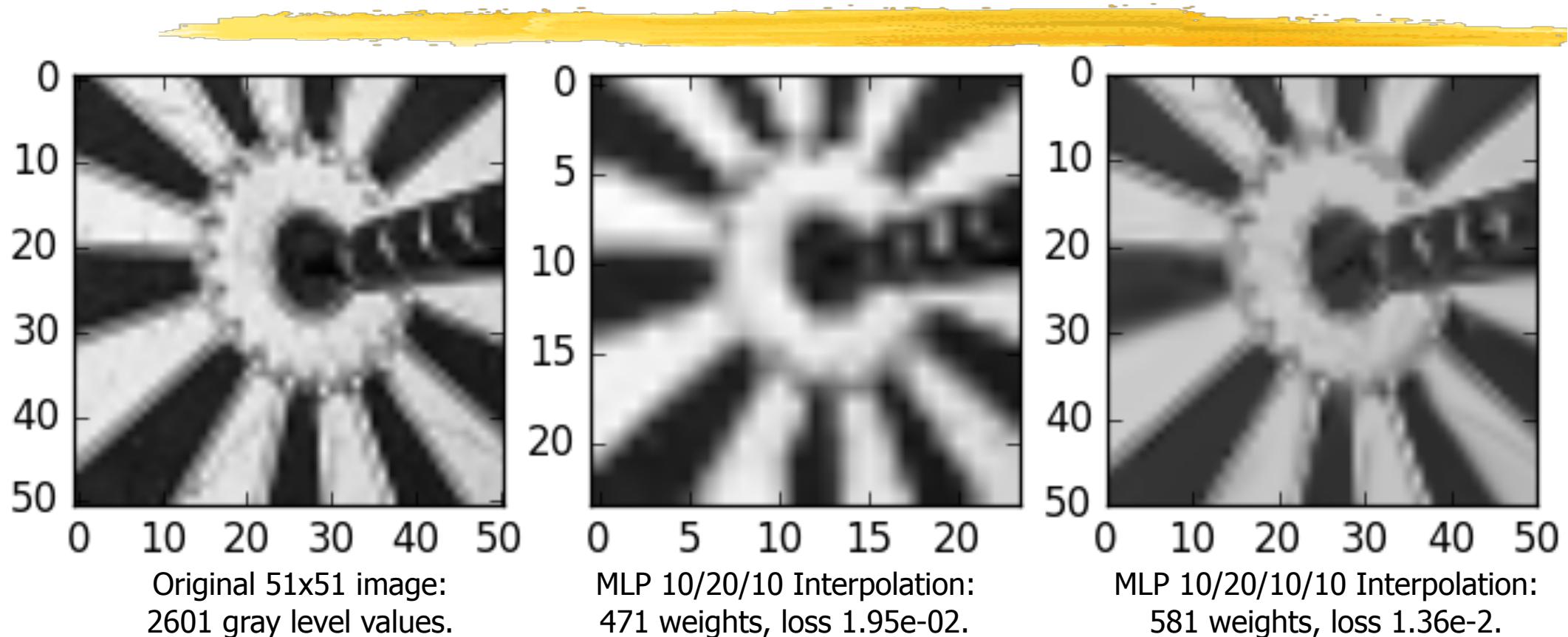


Further improvements in the convergence properties have been obtained by adding a bypass, which allows the final layers to only compute residuals.

IMPROVING THE NETWORK

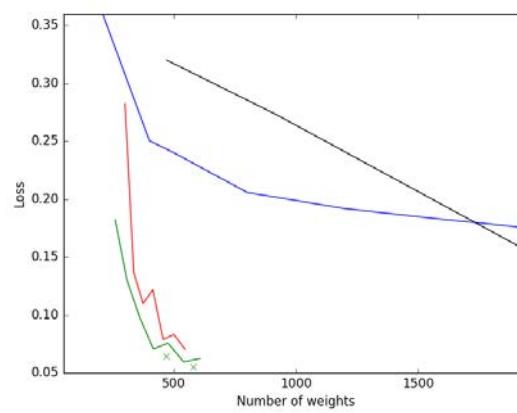
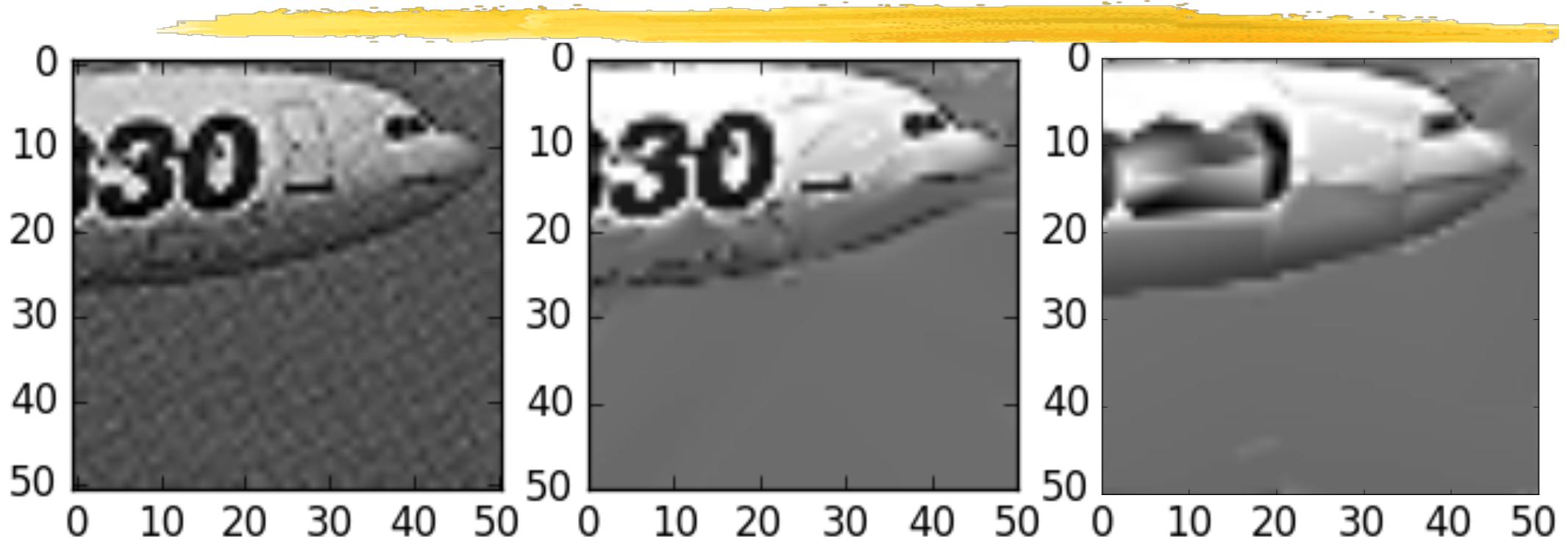


IMPROVING THE NETWORK



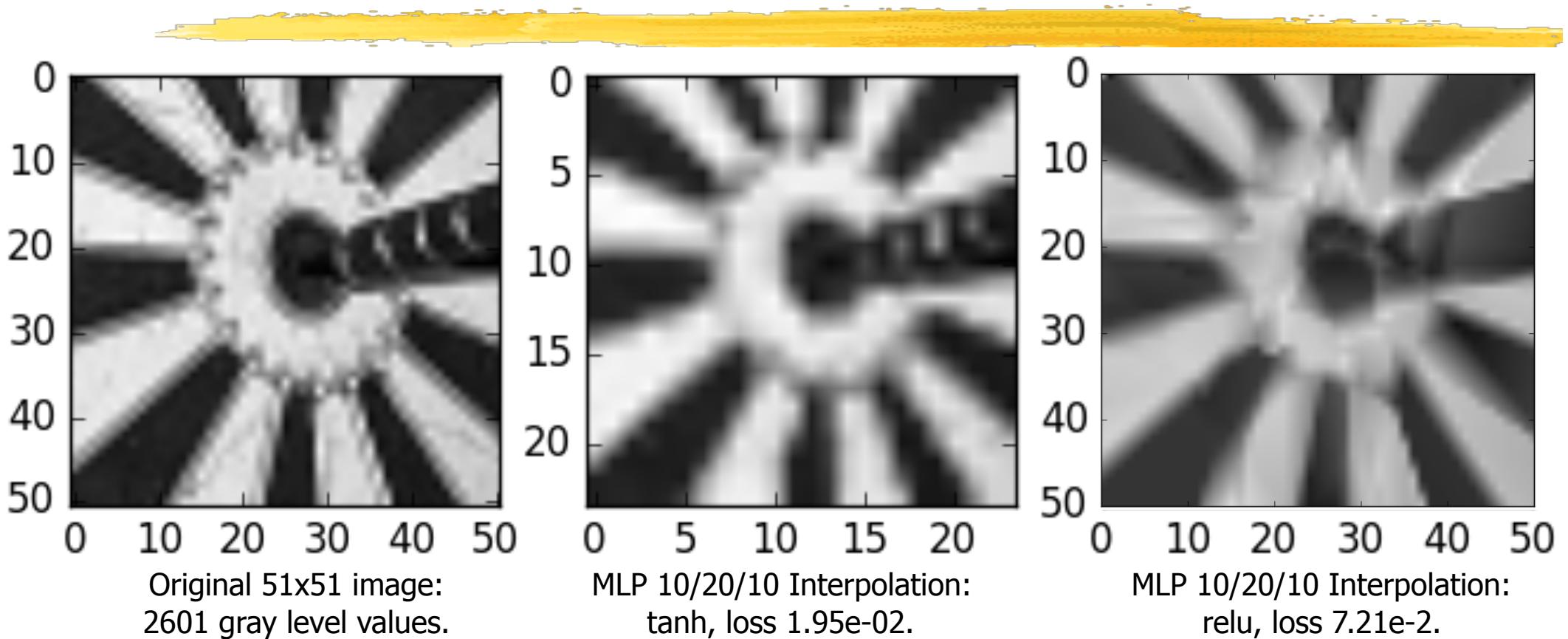
- Relatively small improvement in **this** case.
- The problem is probably too small.
→ Networks can behave very differently for small and large problems!

TANH vs ReLU



- Tanh, 1 layers
- Tanh, 2 layers
- Tanh, 3 layers
- ReLU, 3 layers
- ✖ Tanh, 4 layers

TANH vs ReLU



- Tanh works better than ReLU in **this** case.
- ReLU is widely credited with eliminating the vanishing gradient problem in large networks.

→ There is no substitute for experimentation!

MULTI LAYER PERCEPTRONS



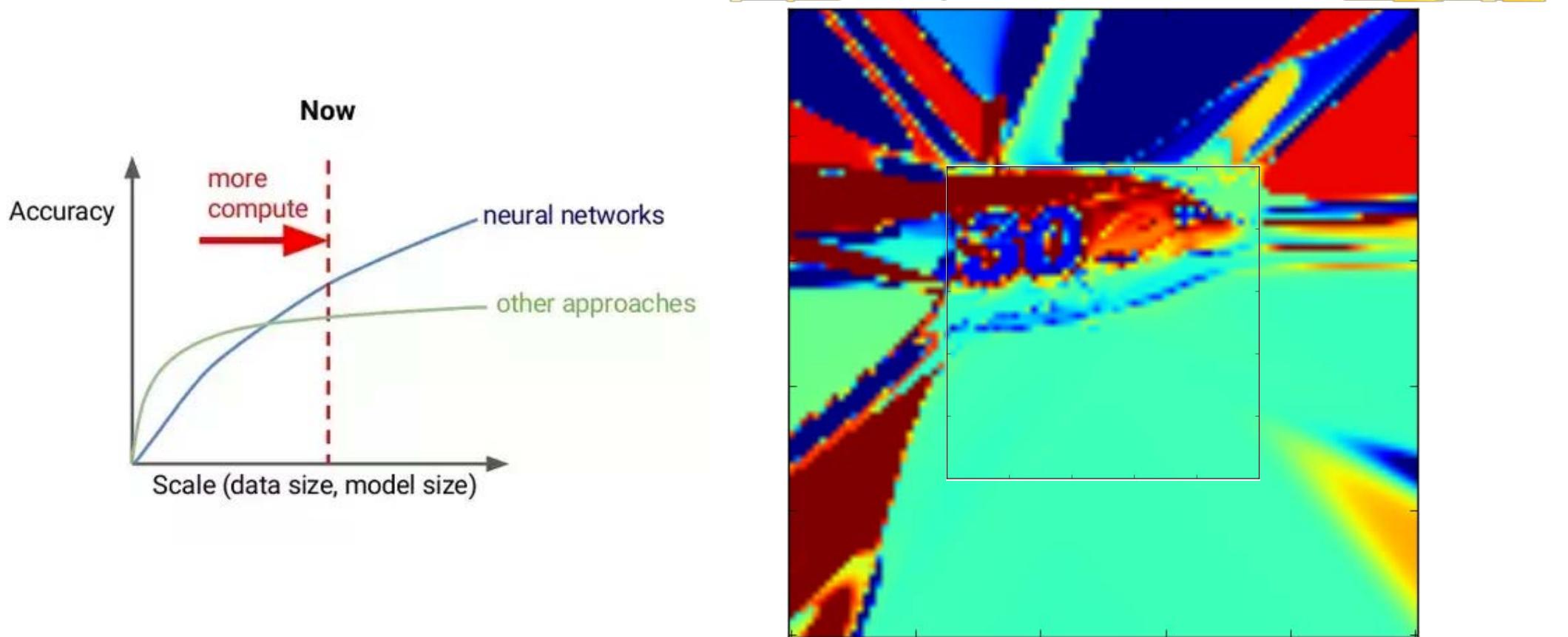
The function learned by a DNN using either the ReLU or Tanh operators is:

- piecewise affine or smooth;
- continuous because it is a composition of continuous functions.

Each region created by a layer is split into smaller regions:

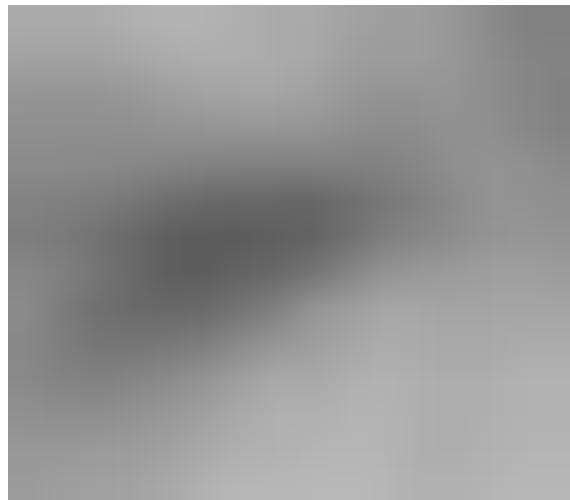
- Their boundaries are correlated in a complex way.
- Their descriptive power is larger than shallow networks for the same number of parameters.

STRENGTHS AND LIMITATIONS



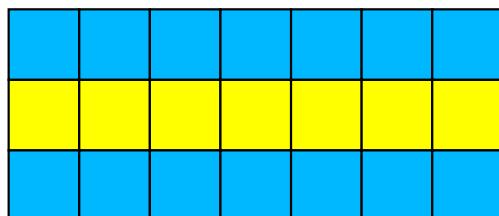
- Powerful regressors but require many parameters, and therefore large training databases.
- Excellent at interpolation but less good at extrapolation. The training data must cover all cases of interest.

DIGITAL IMAGES



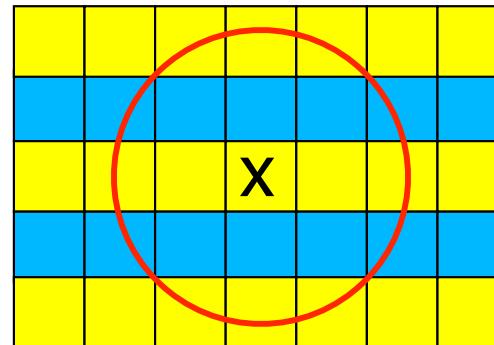
136 134 161 159 163 168 171 173 173 171 166 159 157 155
152 145 136 130 151 149 151 154 158 161 163 163 159 151
145 149 149 145 140 133 145 143 145 145 145 146 148 148
148 143 141 145 145 145 141 136 136 135 135 136 135 133
131 131 129 129 133 136 140 142 142 138 130 128 126 120
115 111 108 106 106 110 120 130 137 142 144 141 129 123
117 109 098 094 094 094 100 110 125 136 141 147 147 145
136 124 116 105 096 096 100 107 116 131 141 147 150 152
152 152 137 124 113 108 105 108 117 129 139 150 157 159
159 157 157 159 135 121 120 120 121 127 136 147 158 163
165 165 163 163 163 166 136 131 135 138 140 145 154 163
166 168 170 168 166 168 170 173 145 143 147 148 152 159
168 173 173 175 173 171 170 173 177 178 151 151 153 156
161 170 176 177 177 179 176 174 174 176 177 179 155 157
161 162 168 176 180 180 180 182 180 180 175 178 180 180

- A digital image is simply a 2D array of numbers.



- A $M \times N$ image can be represented as an MN vector, in which case neighborhood relationships are lost.
- By contrast, treating it as a 2D array preserves neighborhood relationships.

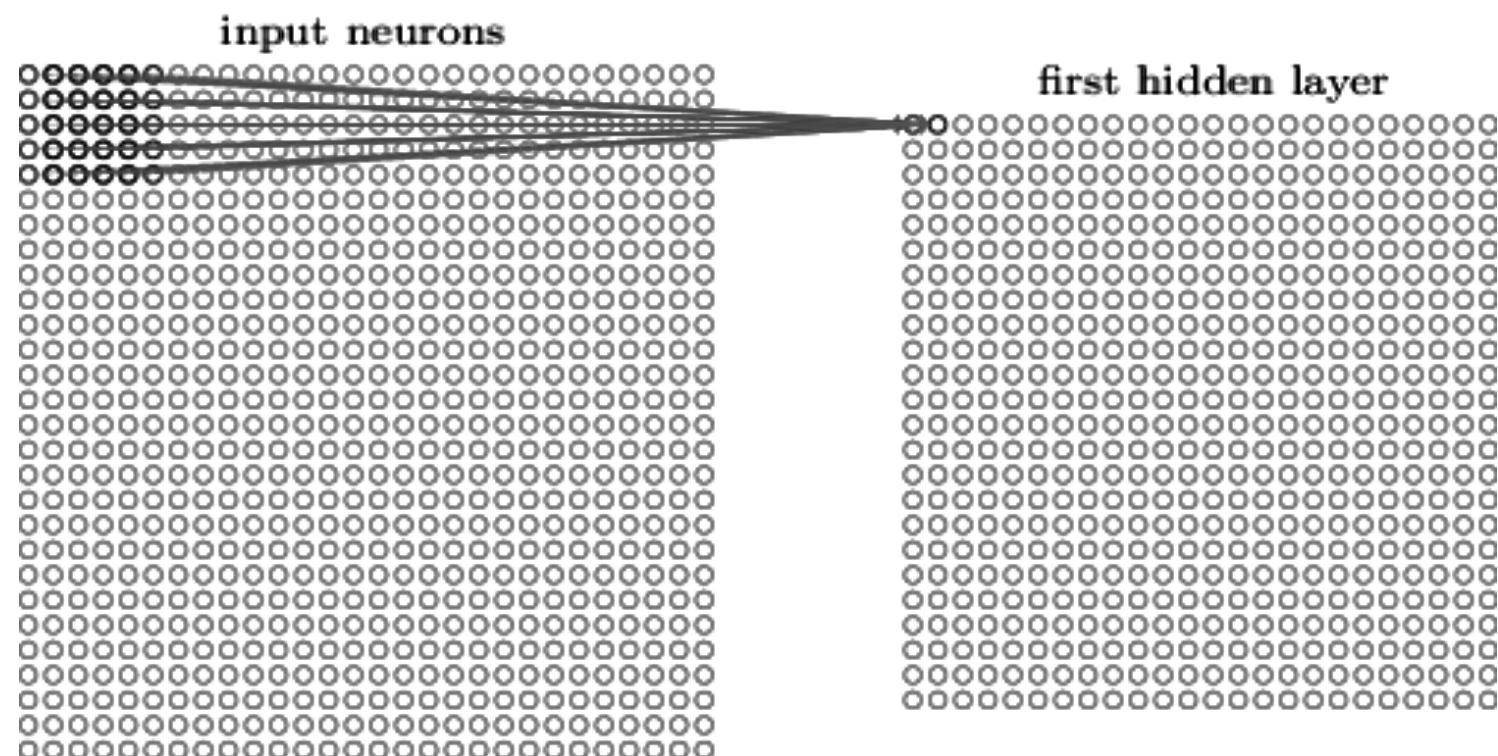
IMAGE SPECIFICITIES



- In a typical image, the values of **neighboring pixels** tend to be more highly correlated than those of distant ones.
- An image filter should be translation invariant.

→ These two properties can be exploited to drastically reduce the number of weights required by CNNs using so-called convolutional layers.

CONVOLUTIONAL LAYER

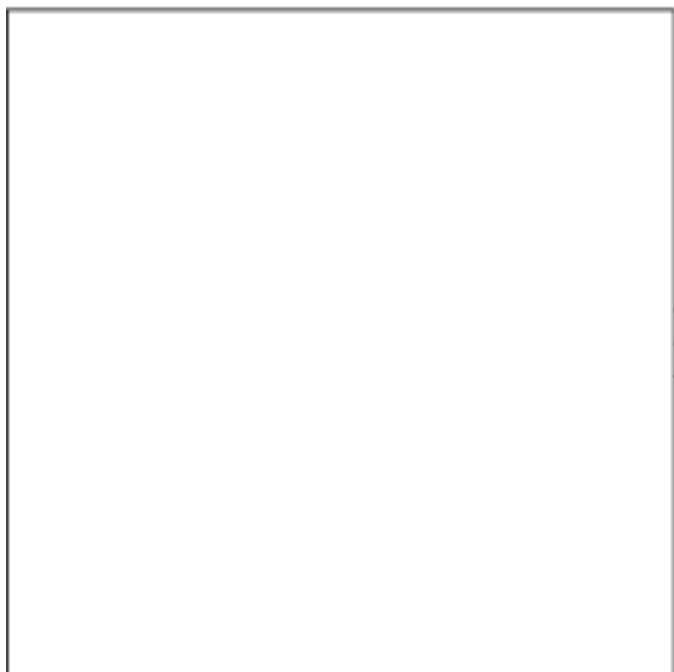


$$\sigma \left(b + \sum_{x=0}^{n_x} \sum_{y=0}^{n_y} w_{i,j} a_{i+x,j+y} \right)$$

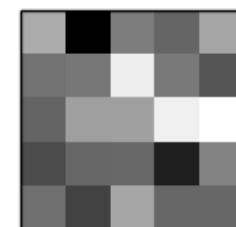
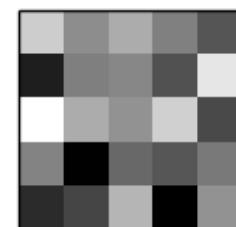
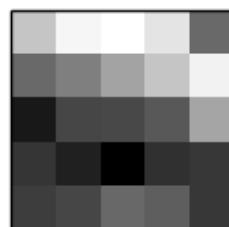
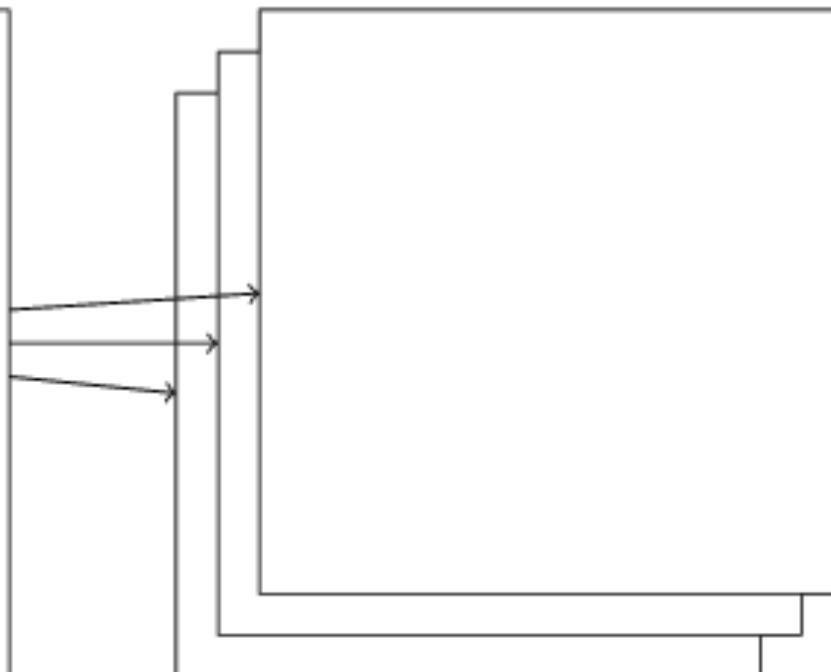
FEATURE MAPS



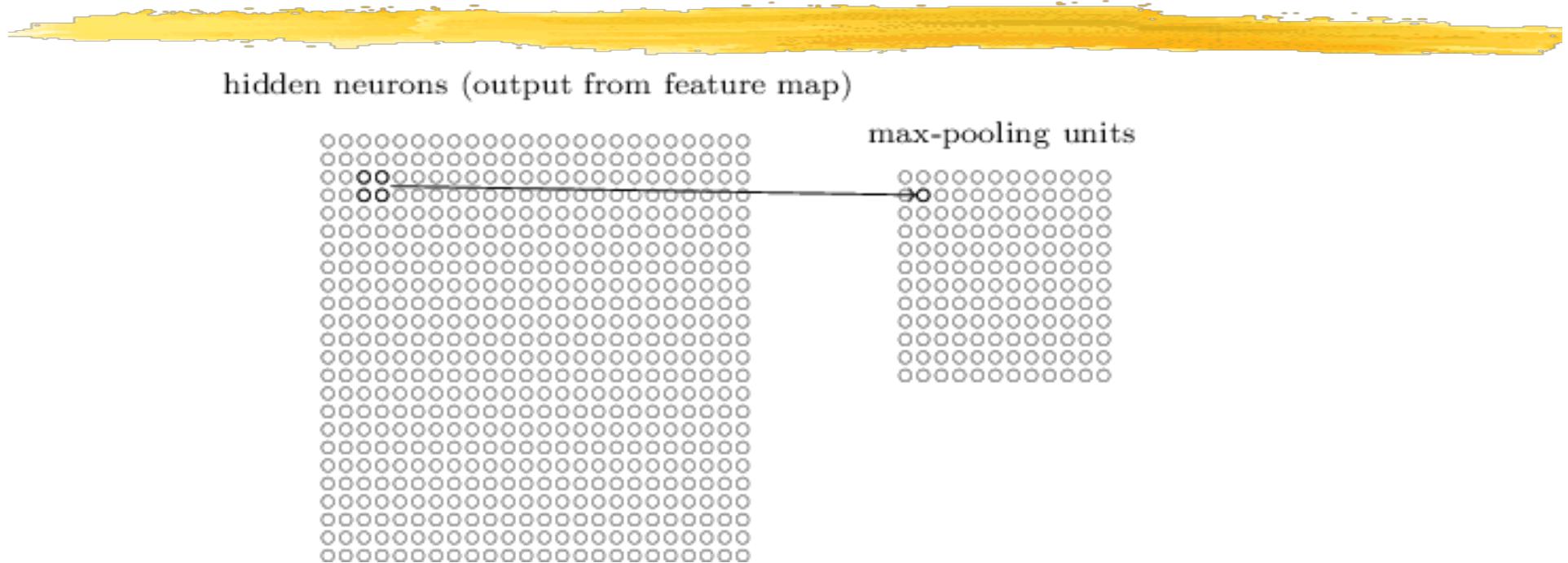
28 × 28 input neurons



first hidden layer: $3 \times 24 \times 24$ neurons

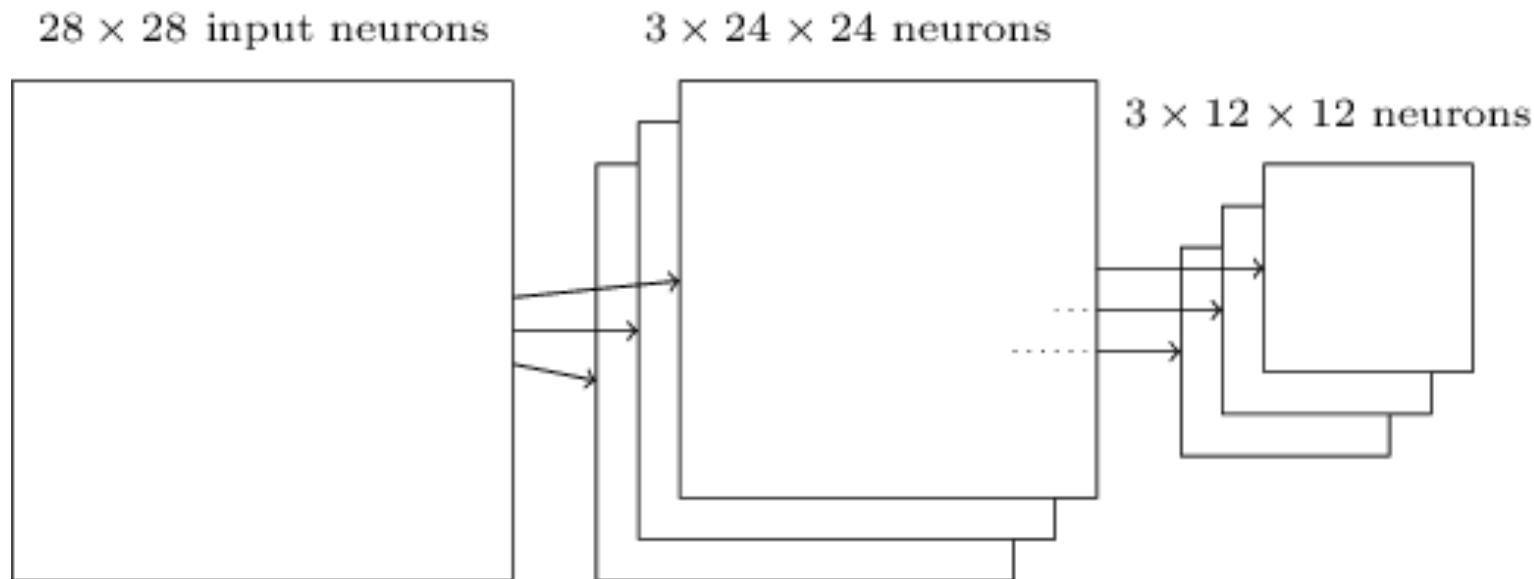


POOLING LAYER



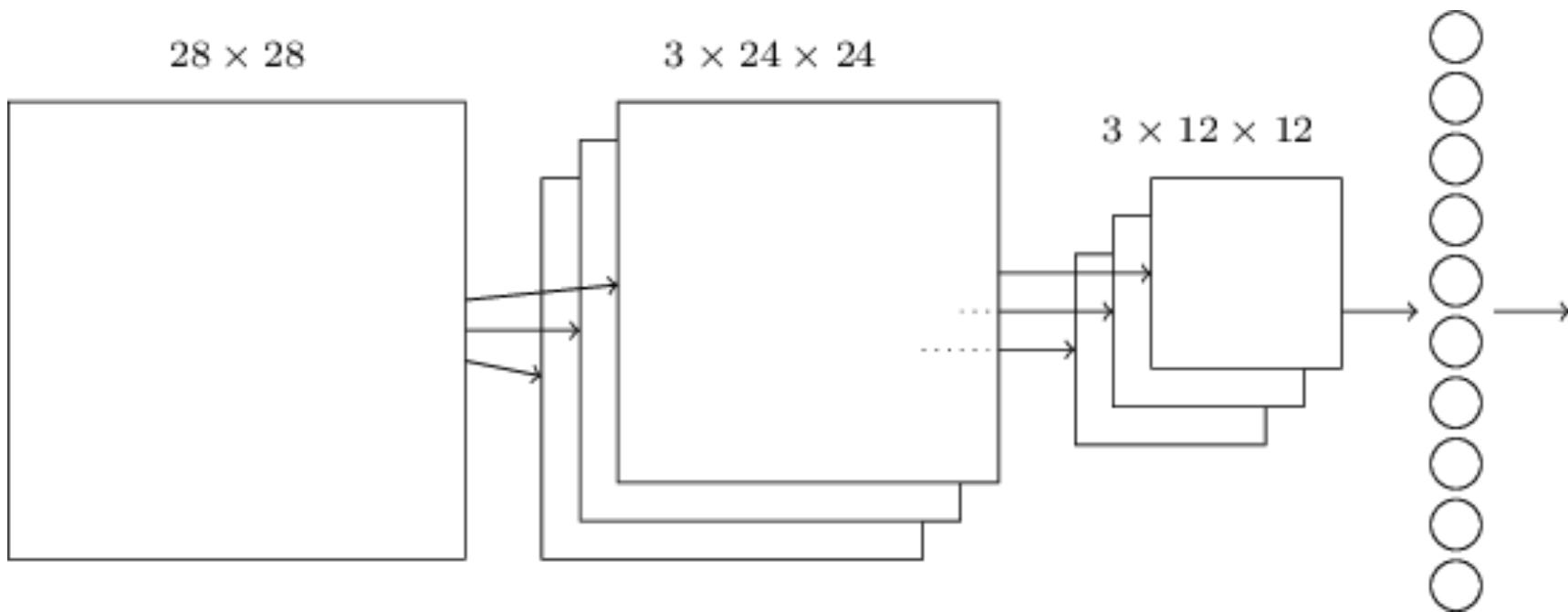
- Reduce the number of inputs by replacing all activations in a neighborhood by a single one.
- Can be thought as asking if a particular feature is present in that neighborhood while ignoring the exact location.

ADDING THE POOLING LAYERS



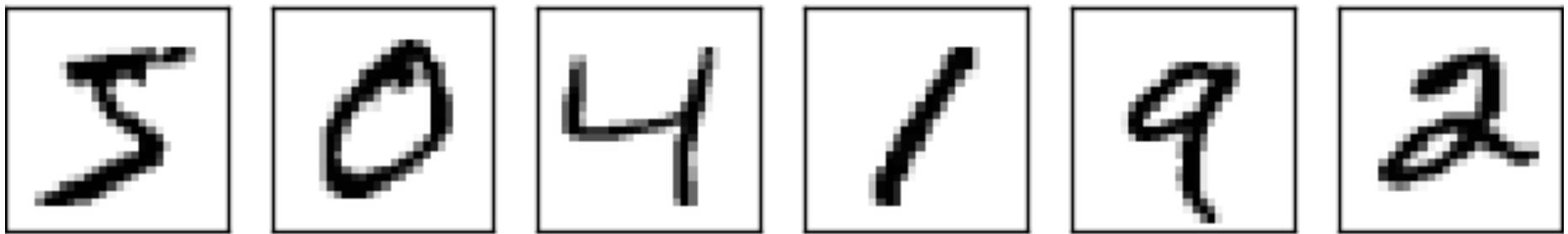
The output size is reduced by the pooling layers.

ADDING A FULLY CONNECTED LAYER



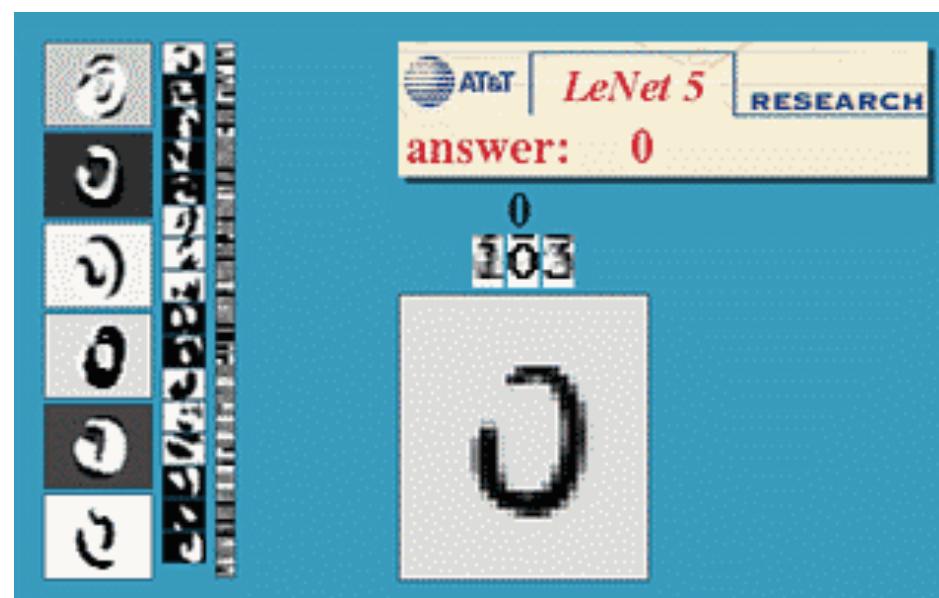
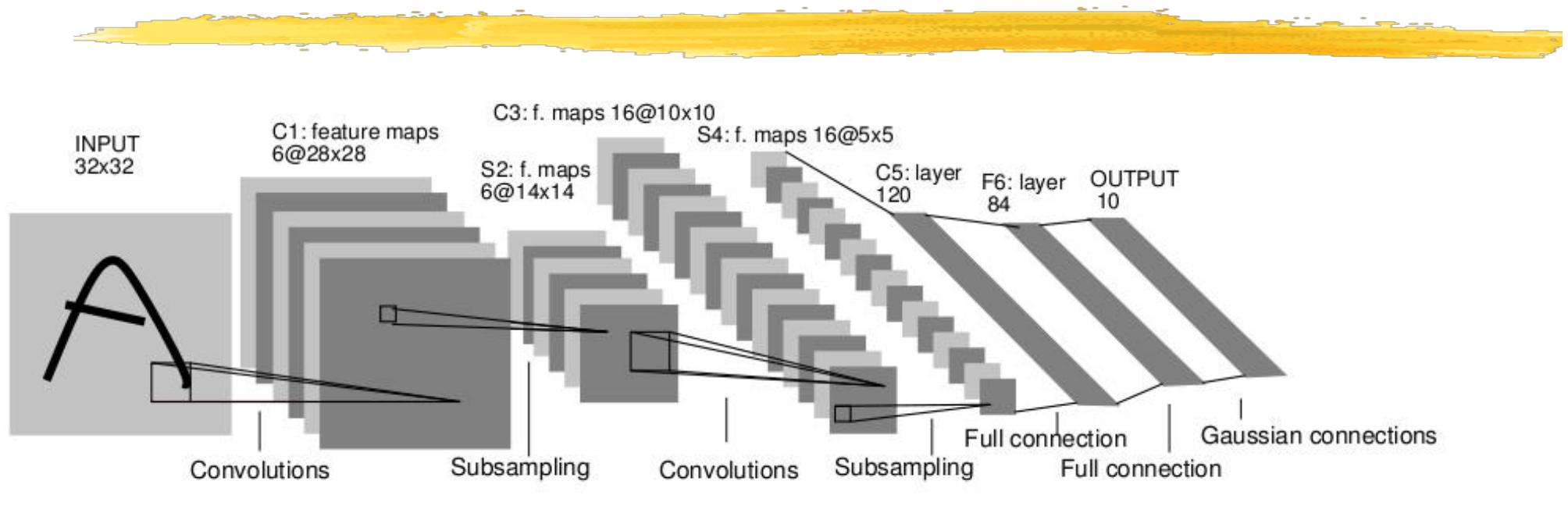
- Each neuron in the final fully connected layer is connected to all neurons in the preceding one.
- Deep architecture with many parameters to learn but still far fewer than an equivalent multilayer perceptron.

MNIST

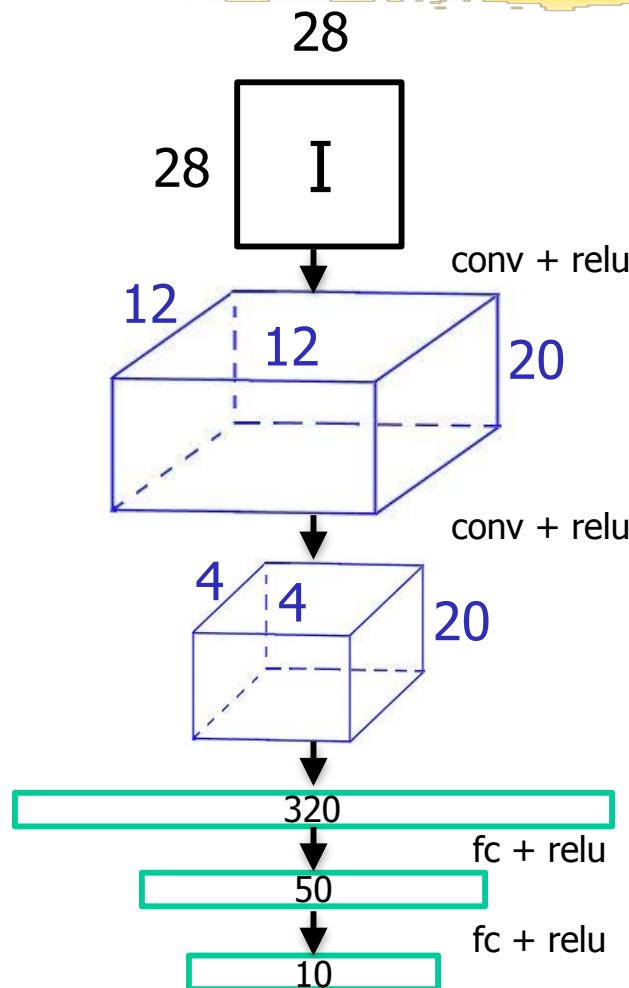


- The network takes as input 28x28 images represented as 784D vectors.
- The output is a 10D vector giving the probability of the image representing any of the 10 digits.
- There are 50'000 training pairs of images and the corresponding label, 10'000 validation pairs, and 5'000 testing pairs.

LeNet (1989-1999)

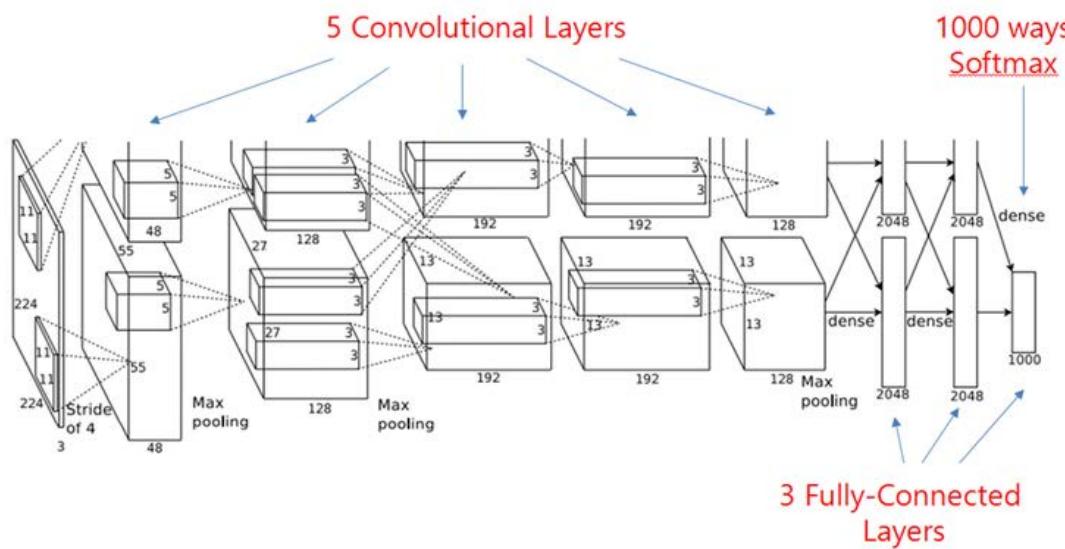


IS MAX POOL REQUIRED?



Accuracy	Train	Test
Conv 5x5, stride 1	99.58	98.77
Max pool 2x3		
Conv 5x5, stride 2	99.42	98.31
Conv 5x5, stride 1	99.38	98.57
Conv 3x3, stride 2		

AlexNet (2012)



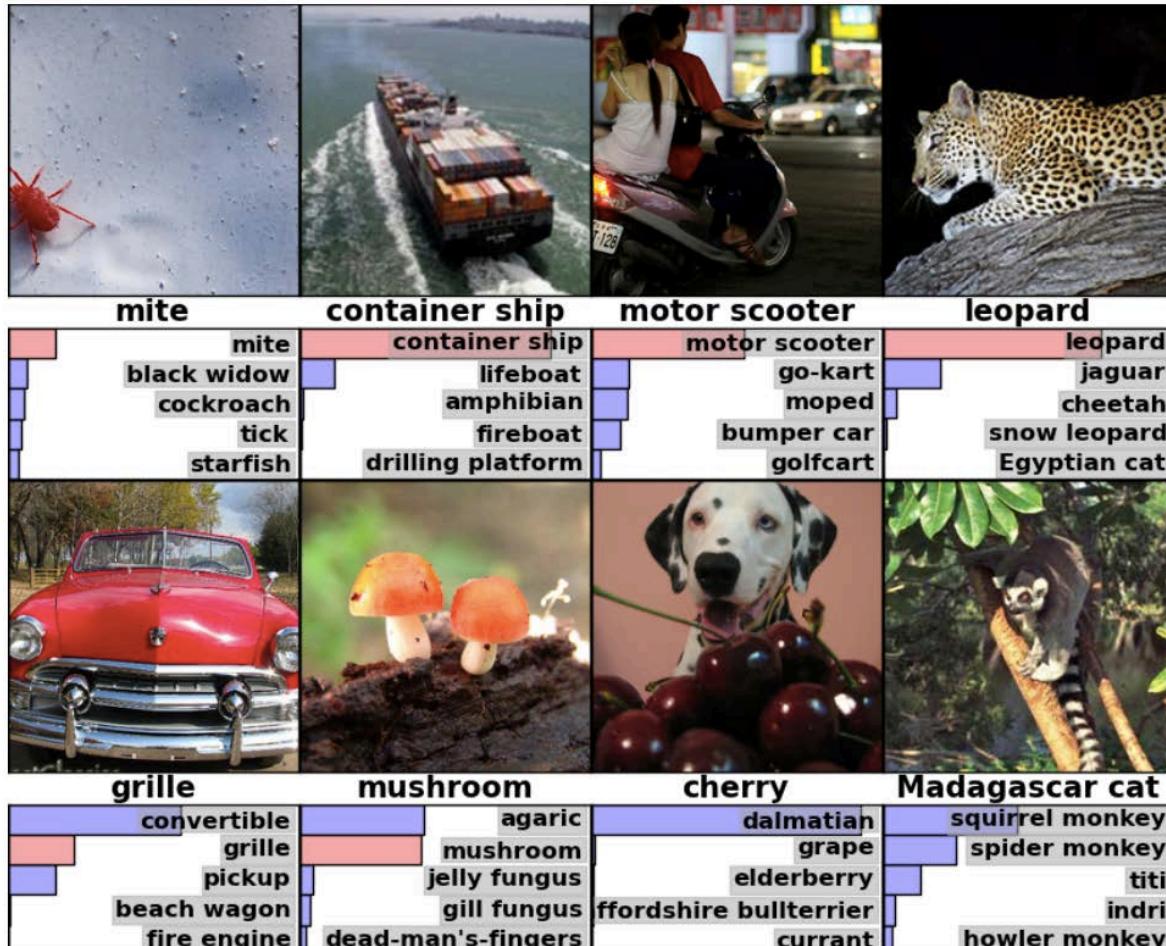
Task: Image classification

Training images: Large Scale Visual Recognition Challenge 2010

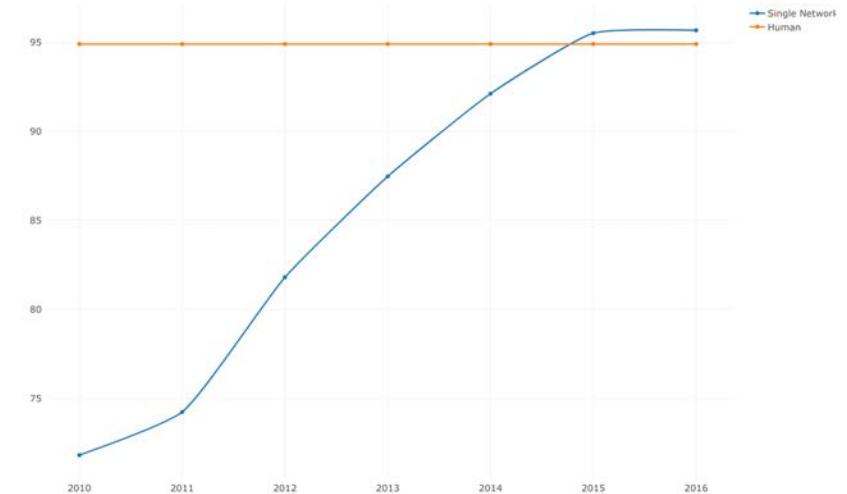
Training time: 2 weeks on 2 GPUs

Major Breakthrough: Training large networks has now been shown to be practical!!

AlexNet RESULTS

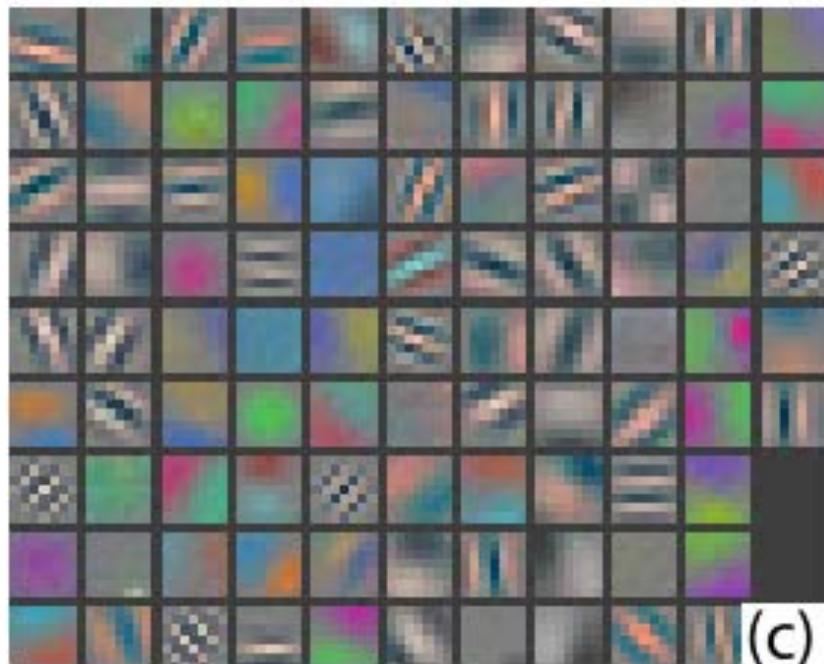


ImageNet Large Scale Visual Recognition Challenge Accuracy

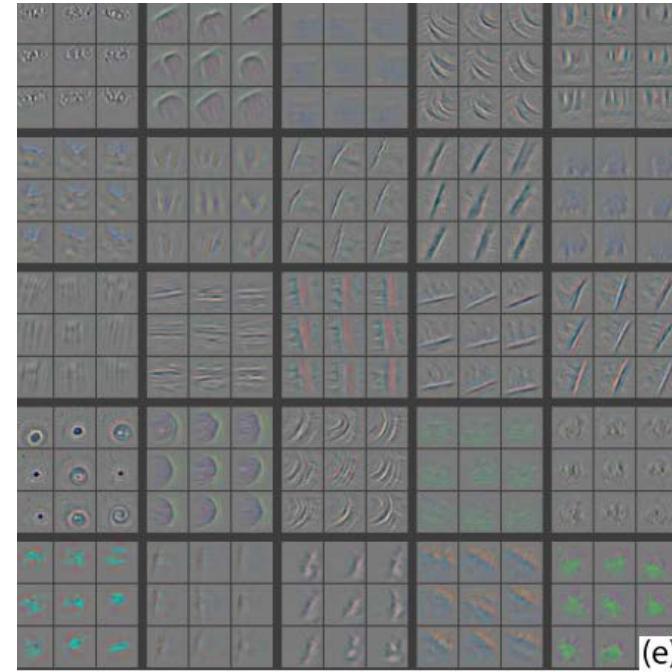


- At the 2012 ImageNet Large Scale Visual Recognition Challenge, AlexNet achieved a top-5 error of 15.3%, more than 10.8% lower than the runner up.
- Since 2015, networks outperform humans on this task.

FEATURE MAPS



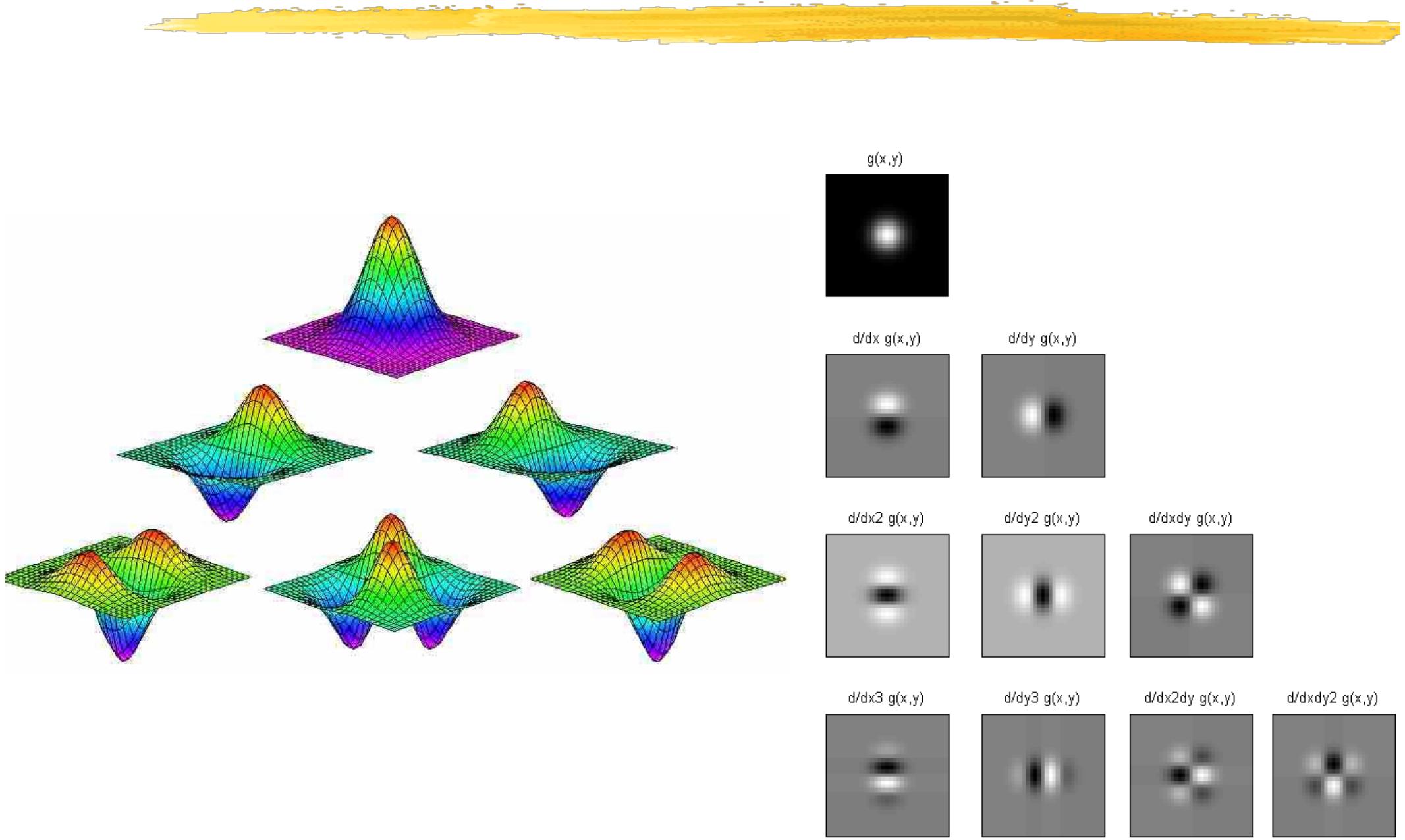
First convolutional layer



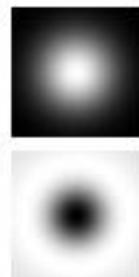
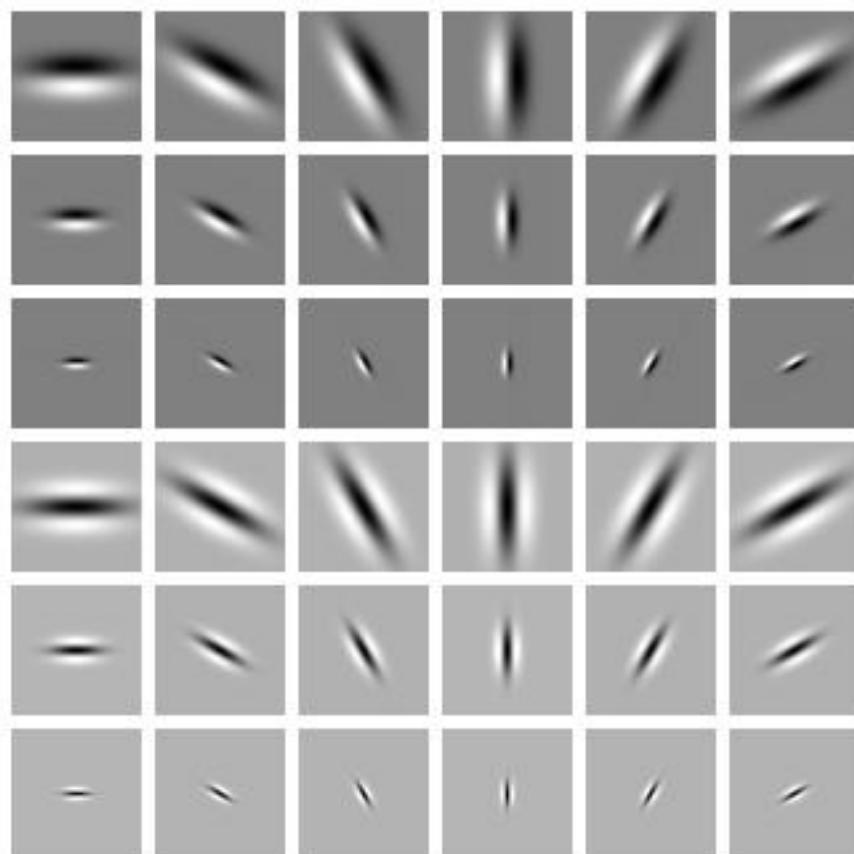
Second convolutional layer

- Some of the convolutional masks seem very similar to oriented Gaussian or Gabor filters!
- Much ongoing work to better understand this.

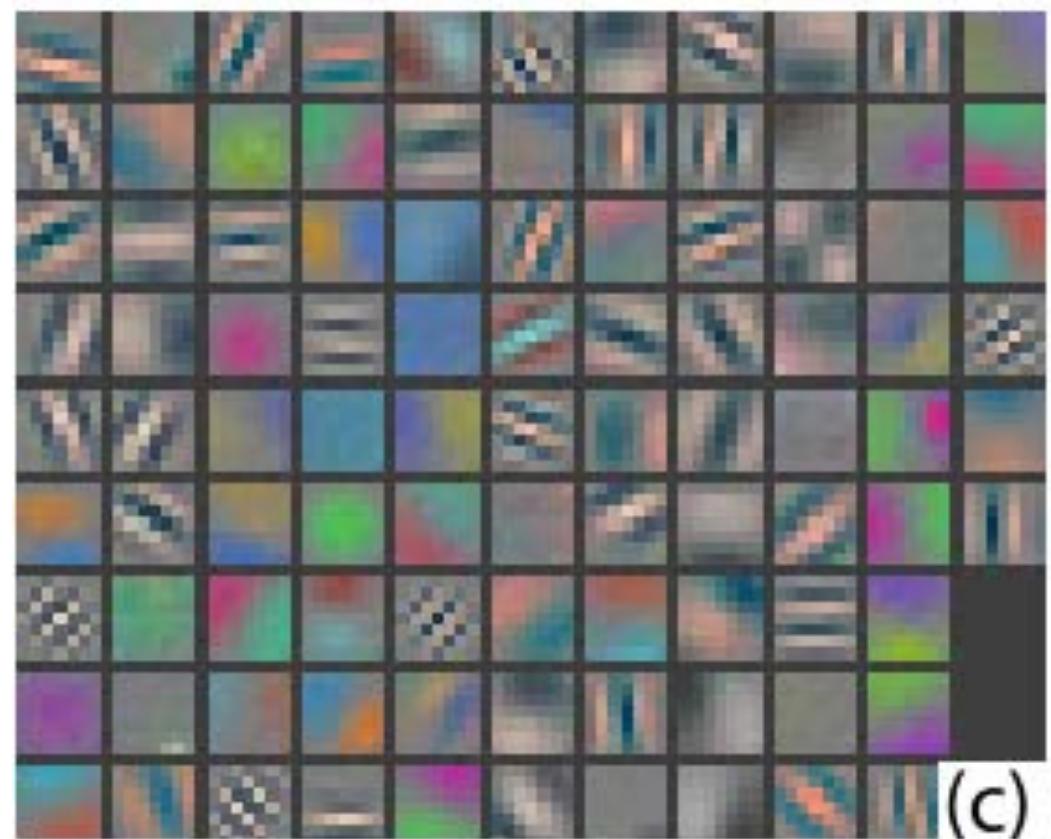
HIGHER ORDER DERIVATIVES



FILTER BANKS



Hand-Designed



Learned

(c)

SIZE AND DEPTH MATTER



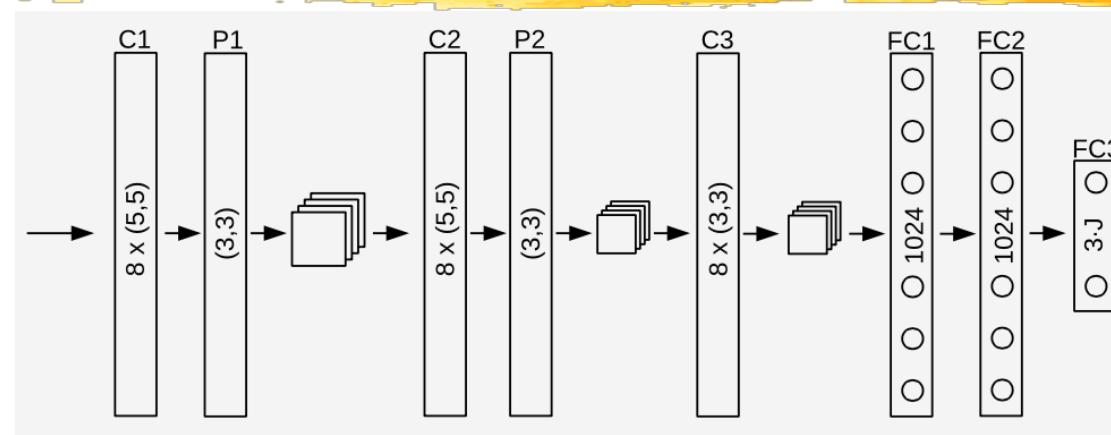
VGG19, 3 weeks of training.

GoogleLeNet.

“It was demonstrated that the representation depth is beneficial for the classification accuracy, and that state-of-the-art performance on the ImageNet challenge dataset can be achieved using a conventional ConvNet architecture.”

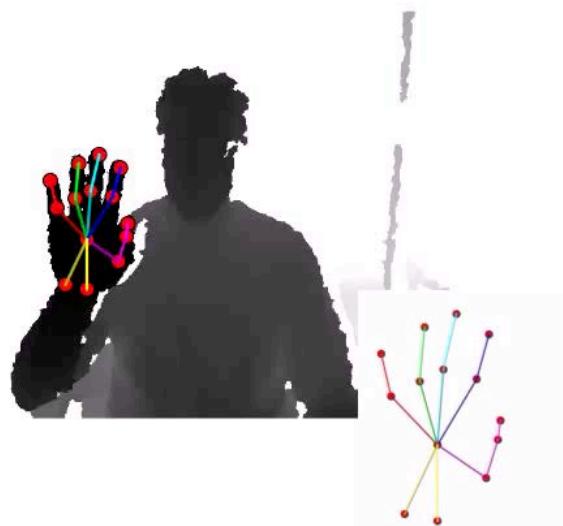
Simonyan & Zisserman, ICLR’15

HAND POSE ESTIMATION (2015)



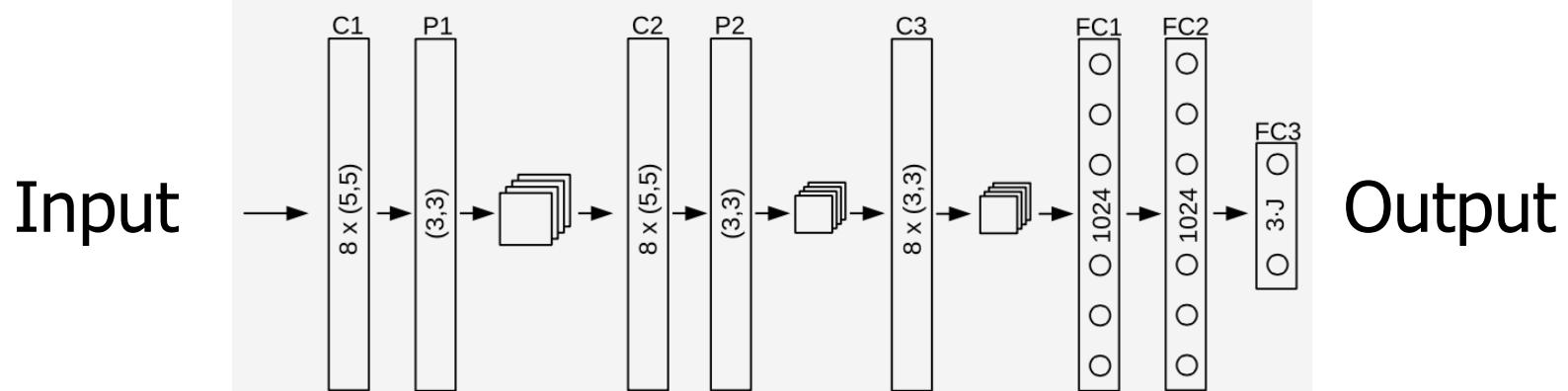
Input: Depth image.

Output: 3D pose vector.



Oberweger et al., ICCV'15

REGRESSION



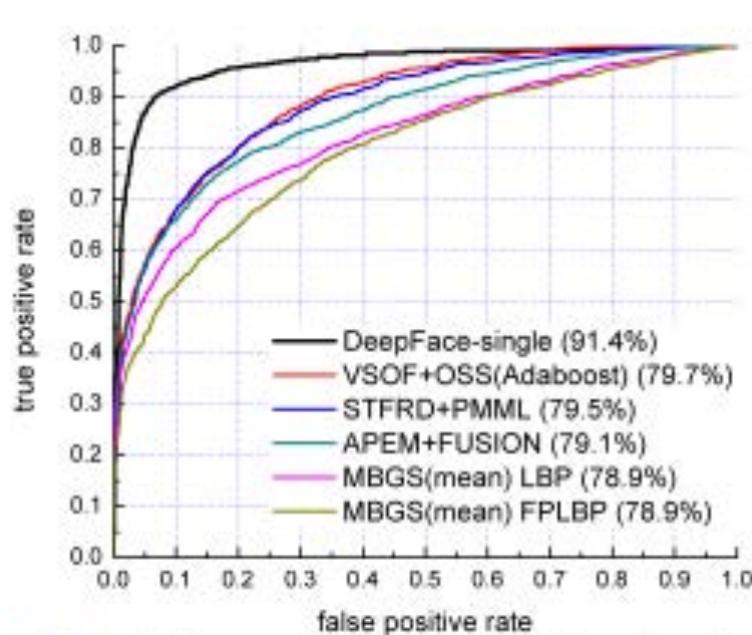
Network parameters are found by minimizing an objective function of the form

$$\min_{\mathbf{W}_l, \mathbf{B}_l} \sum_i \|\mathbf{F}(\mathbf{x}_i, \mathbf{W}_1, \dots, \mathbf{W}_L, \mathbf{b}_1, \dots, \mathbf{b}_L) - \mathbf{y}_i\|^2$$

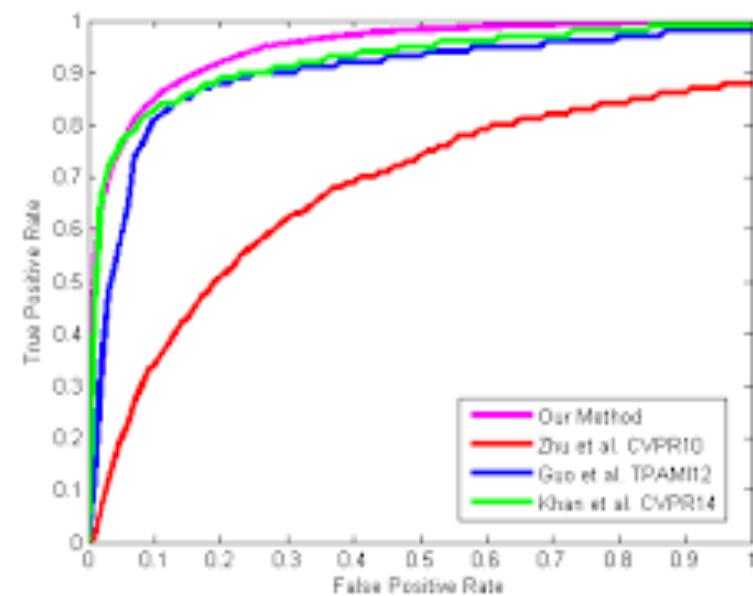
using

- stochastic gradient descent on mini-batches,
- dropout,
- hard example mining,
-

ROC HUNTING



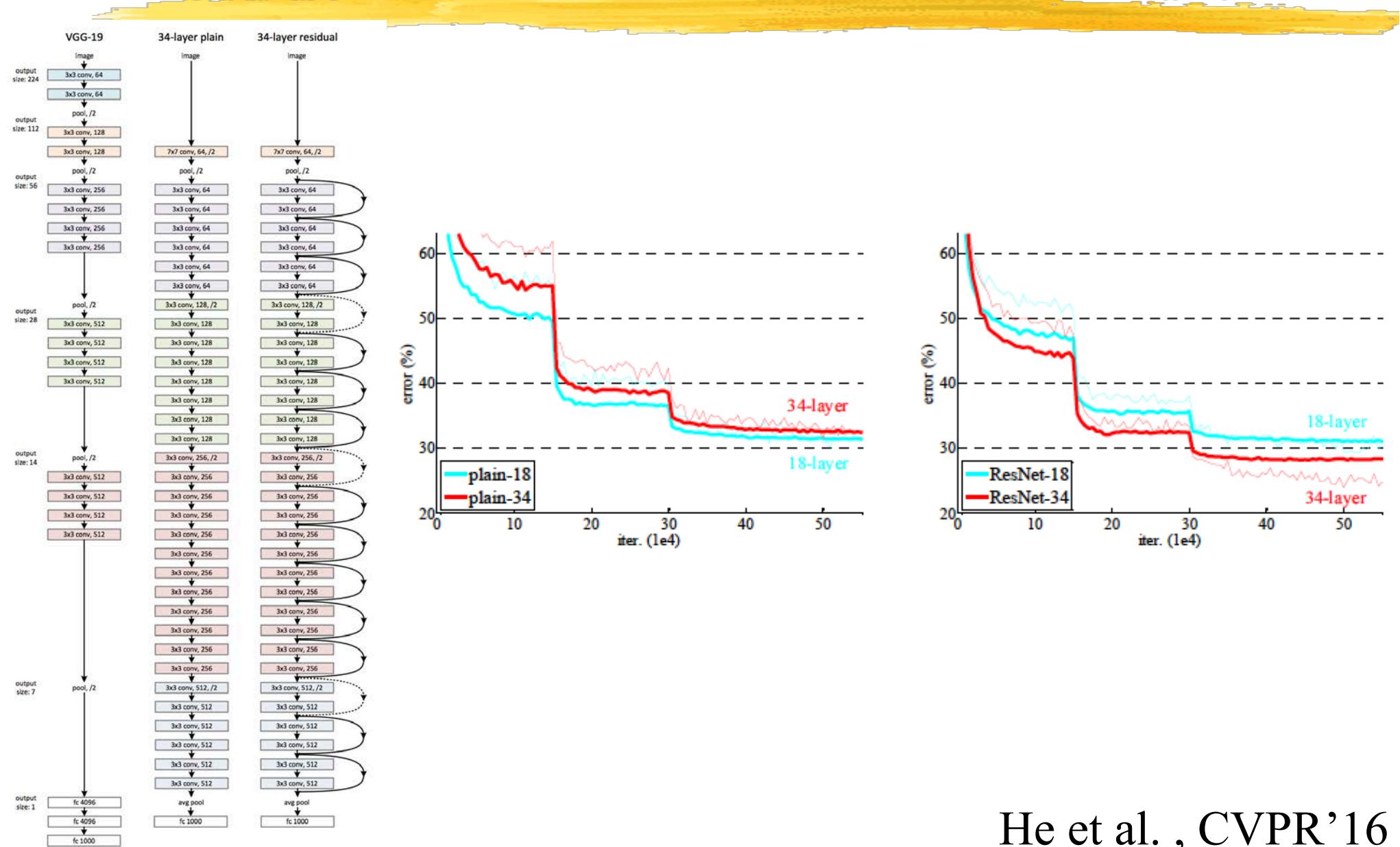
DeepFace
Taigman et al. 2014



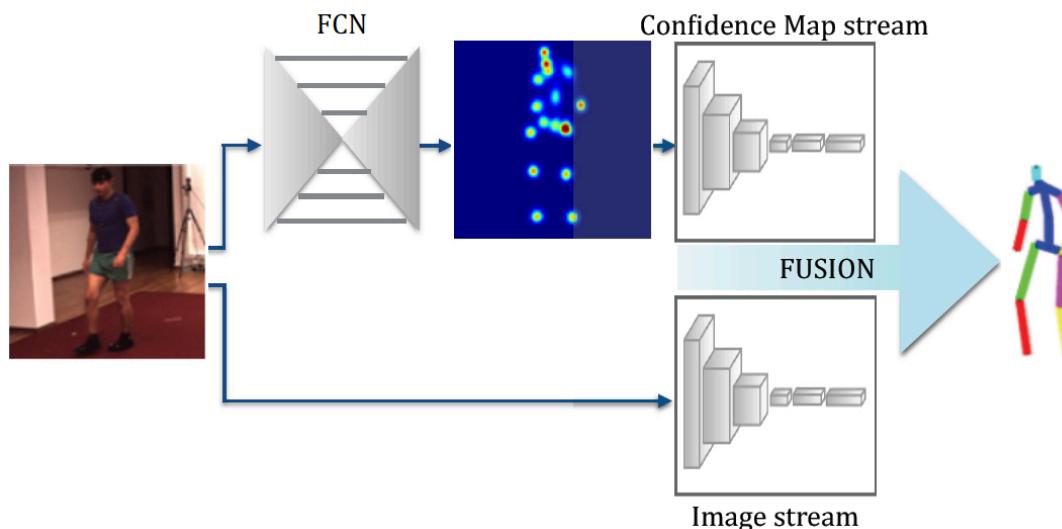
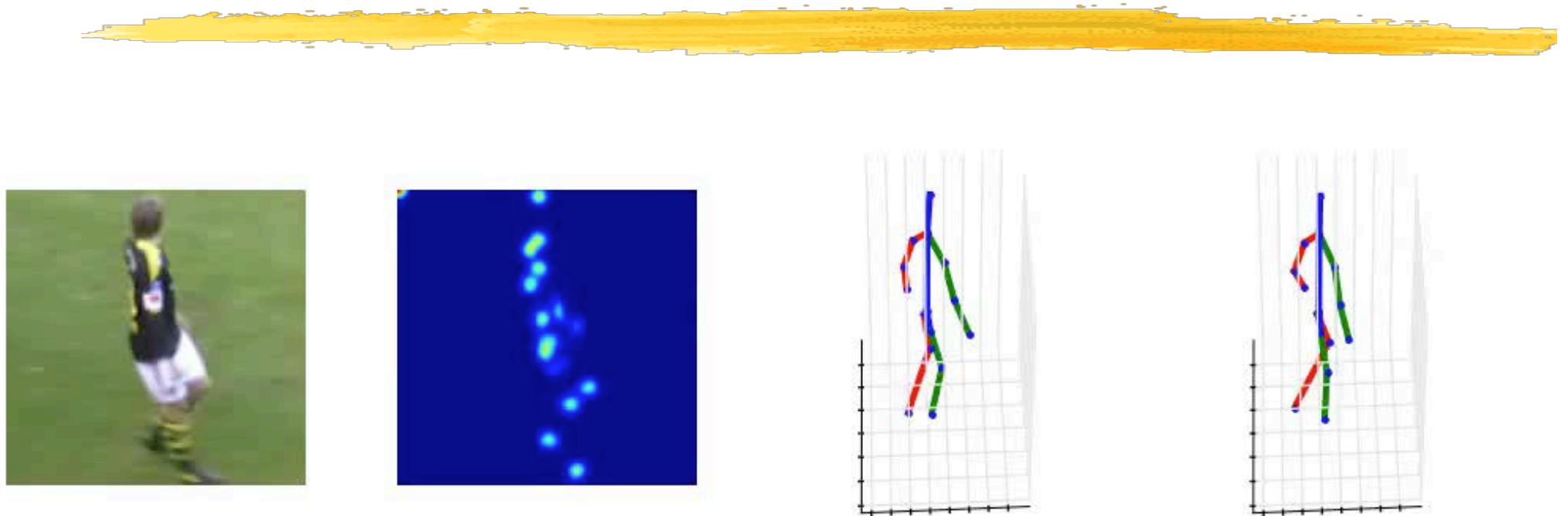
Deep Edge Detection
Shen et al. 2015



DEEPER AND DEEPER

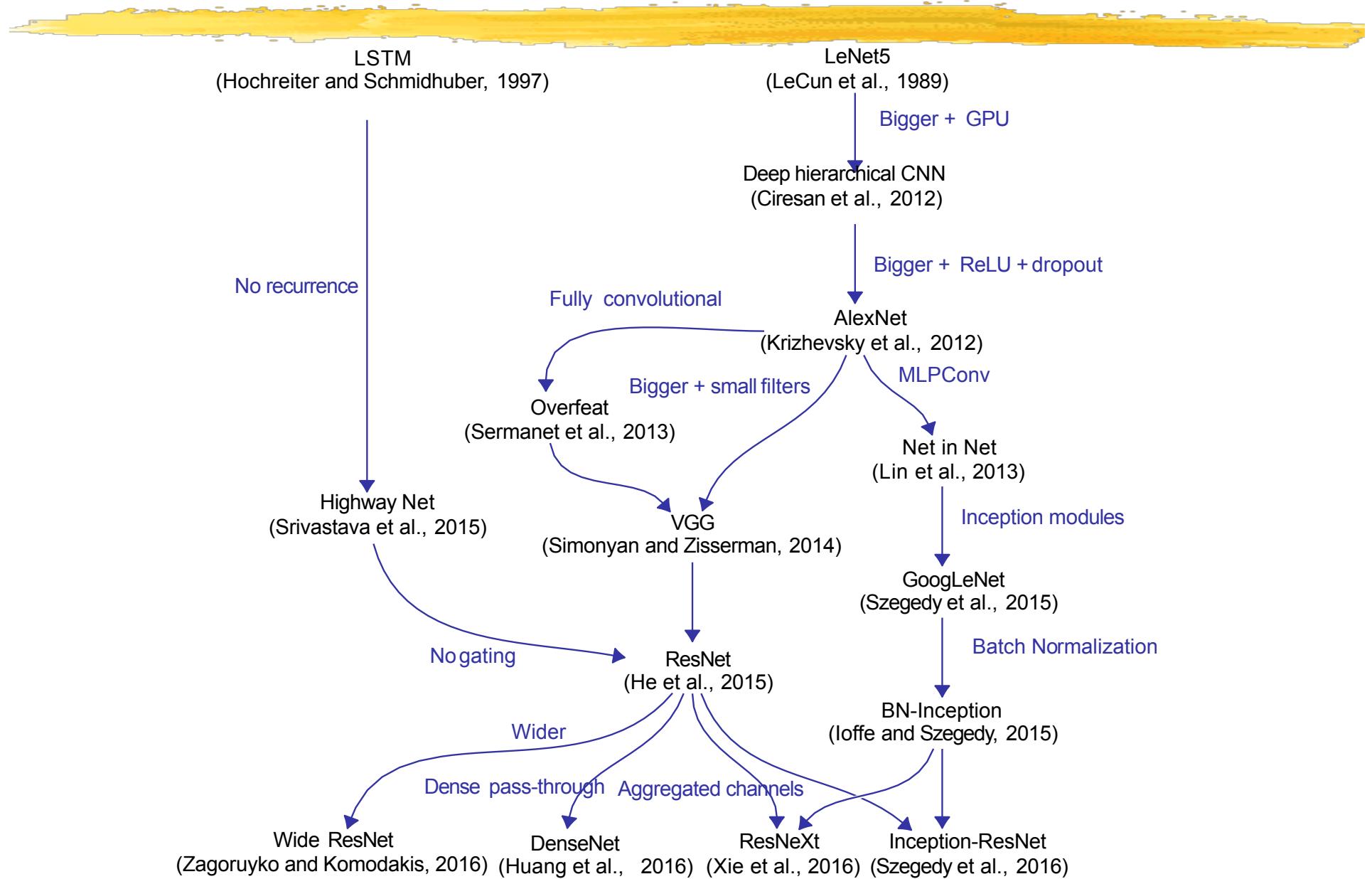


MONOCULAR POSE ESTIMATION



Tekin et al. , ICCV'17

IMAGE CLASSIFICATION TAXONOMY



ALPHA GO



- Uses Deep Nets to find the most promising locations to focus on.
- Performs Tree based search when possible.
- Relies on reinforcement learning and other ML techniques to train.

→ Beat the world champion in 2017.

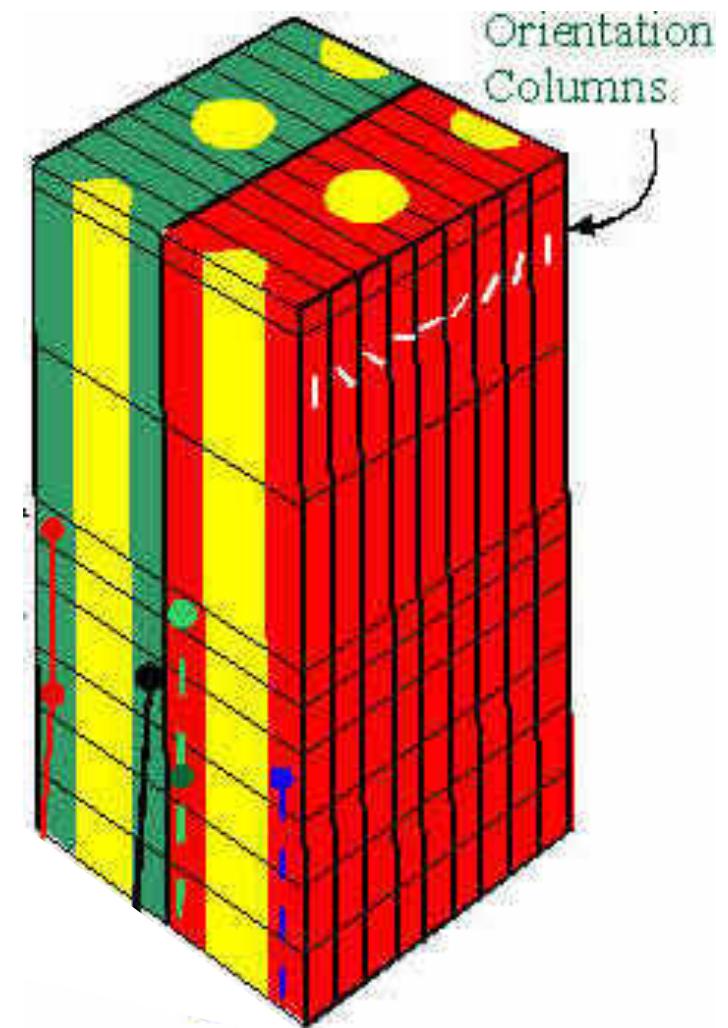
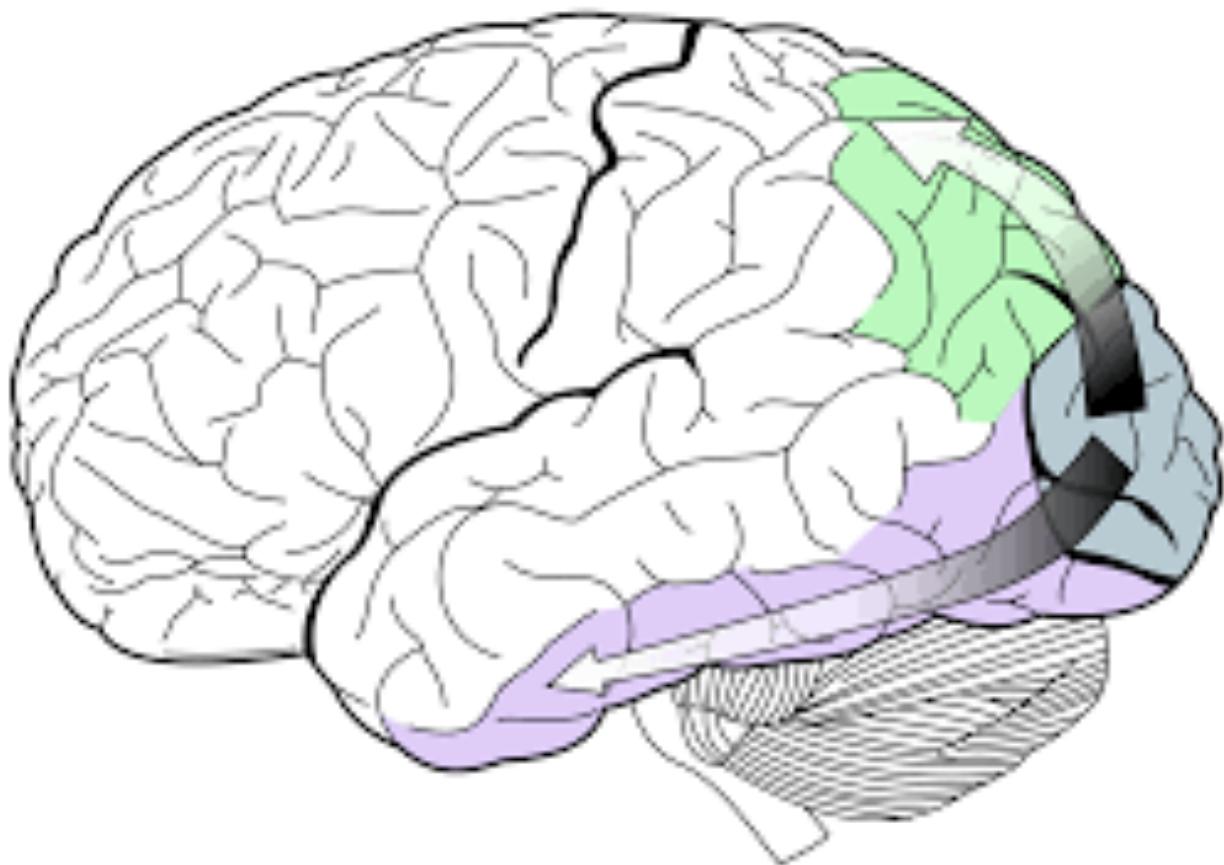
ANOTHER POINT OF VIEW



To summarize roughly the evolution of convnets for image classification:

- Standard ones are extensions of LeNet5.
- Everybody loves ReLU.
- Newer ones have 100s of channels and 10s of layers.
- They can (should?) be fully convolutional.
- Pass-through connections allow deeper “residual” nets.
- Bottleneck local structures reduce the number of parameters.
- Aggregated pathways reduce the number of parameters.

VISUAL CORTEX

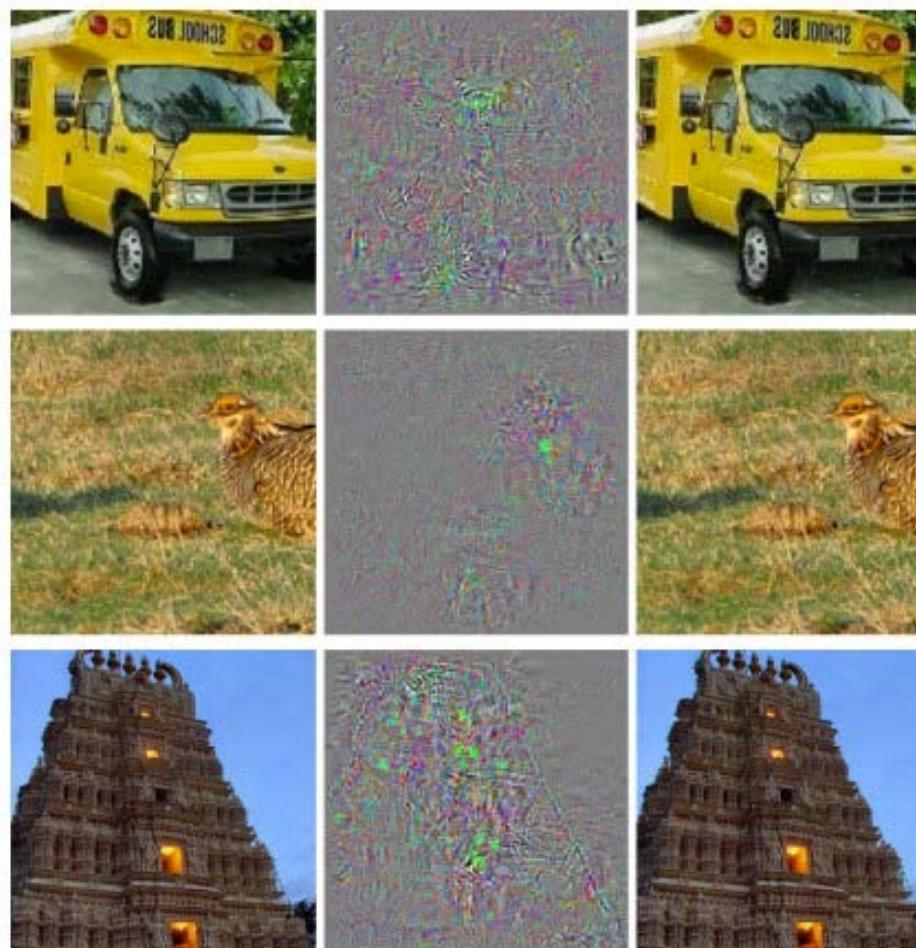


AlphaGo



- Uses Deep Nets to find the most promising locations to focus on.
- Performs Tree based search when possible.
- Relies on reinforcement learning and other ML techniques to train.

ADVERSARIAL IMAGES



Szegedy et al. 2013

XKCD'S VIEW ON THE MATTER



<https://xkcd.com/>

IN SHORT



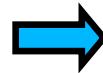
- Deep Belief Networks in general and Convolutional Neural Nets in particular outperform conventional Computer Vision algorithms on many benchmarks.
 - It is not fully understood why and unexpected failure cases have been demonstrated.
 - They require a lot of manual tuning to perform well and performance is hard to predict.
- Many questions are still open and there is much work left to do.

DELINEATION

- Dynamic Programming
- Deformable Models
- Hough Transform
- Graph Based Approaches



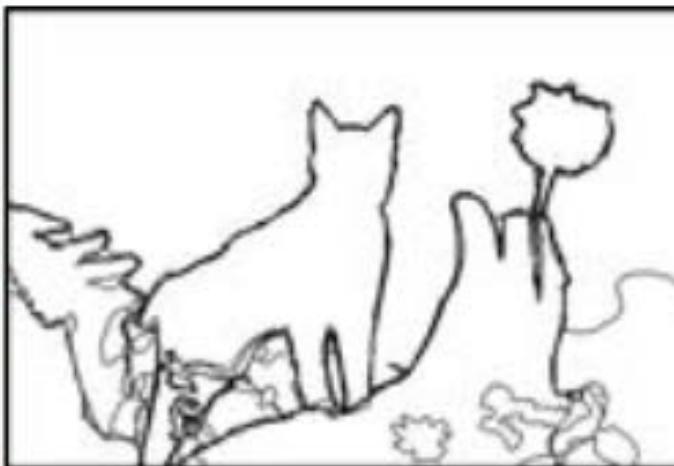
FROM GRADIENTS TO OUTLINES



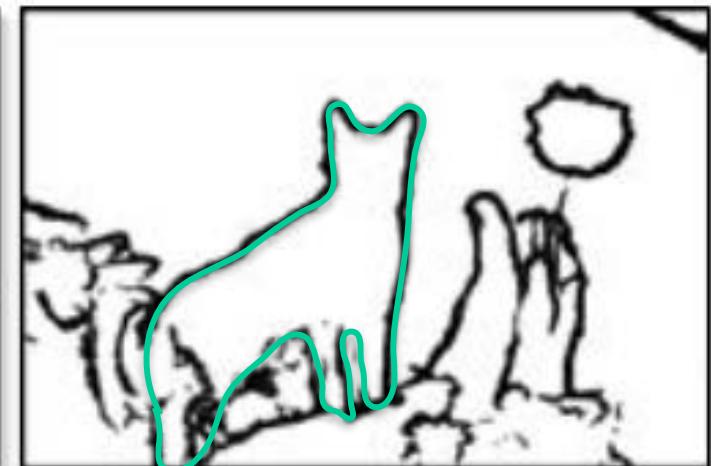
FROM DEEP-NETS TO OUTLINES



(a) original image



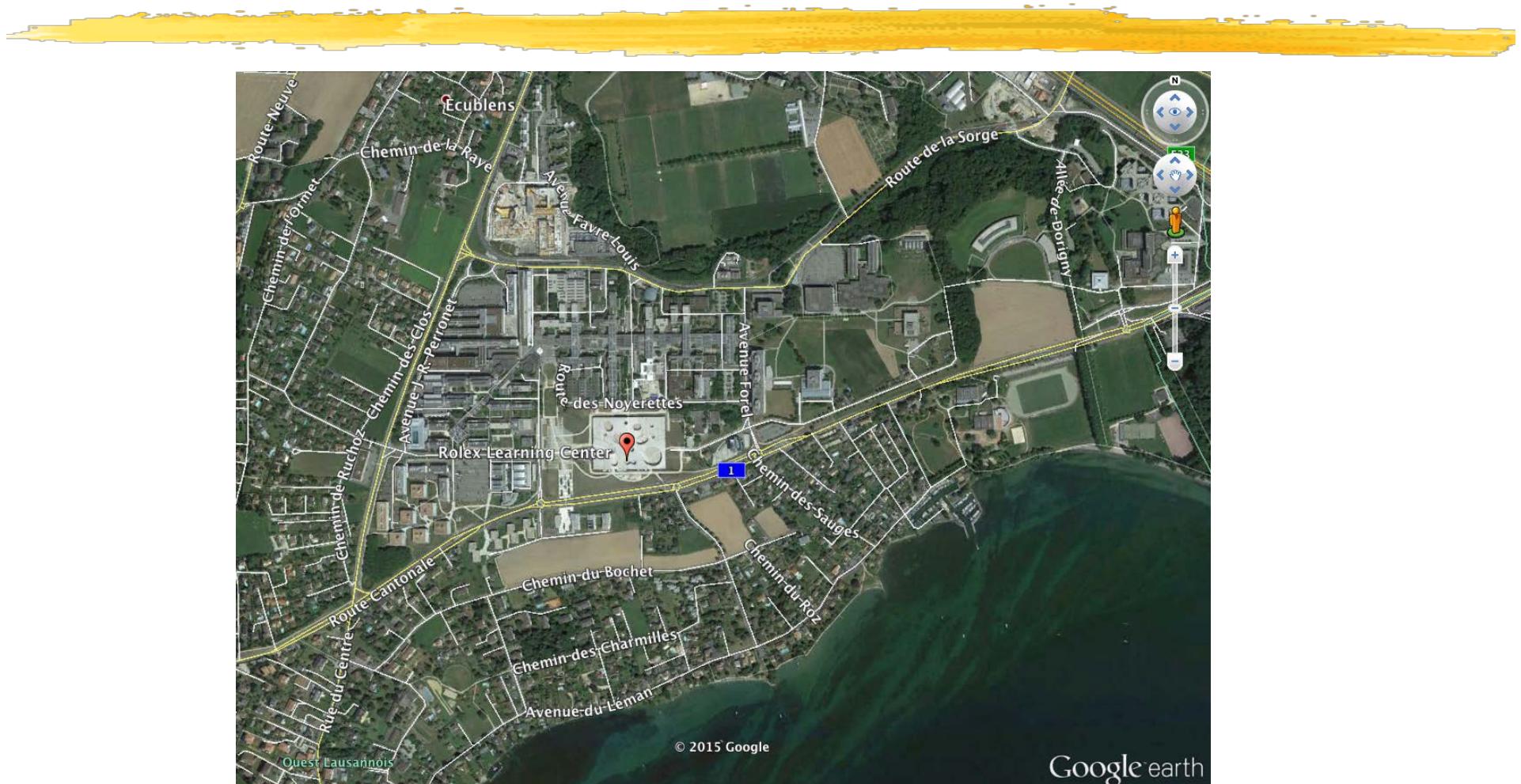
(b) ground truth



(c) HED: output

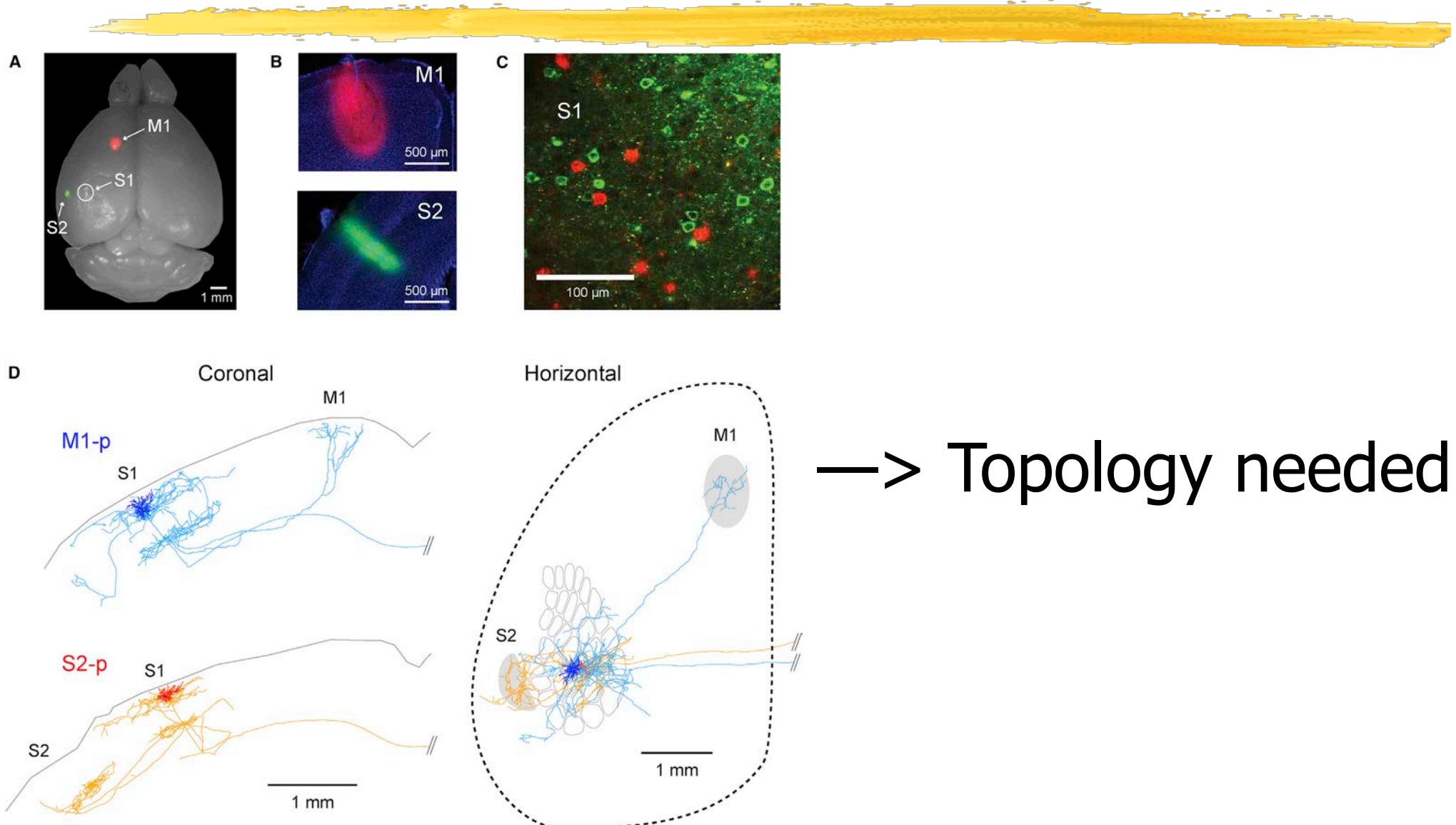
→ Still work to do!

MAPPING AND OVERLAYS



Connectivity matters!

CONNECTOMICS



Courtesy of C. Petersen

ANALOGY



Low level processing

- Uses Deep Nets to find the most promising locations to focus on.

High level processing

- Performs Tree based search when possible.
- Relies on reinforcement learning and other ML techniques to train.

TECHNIQUES



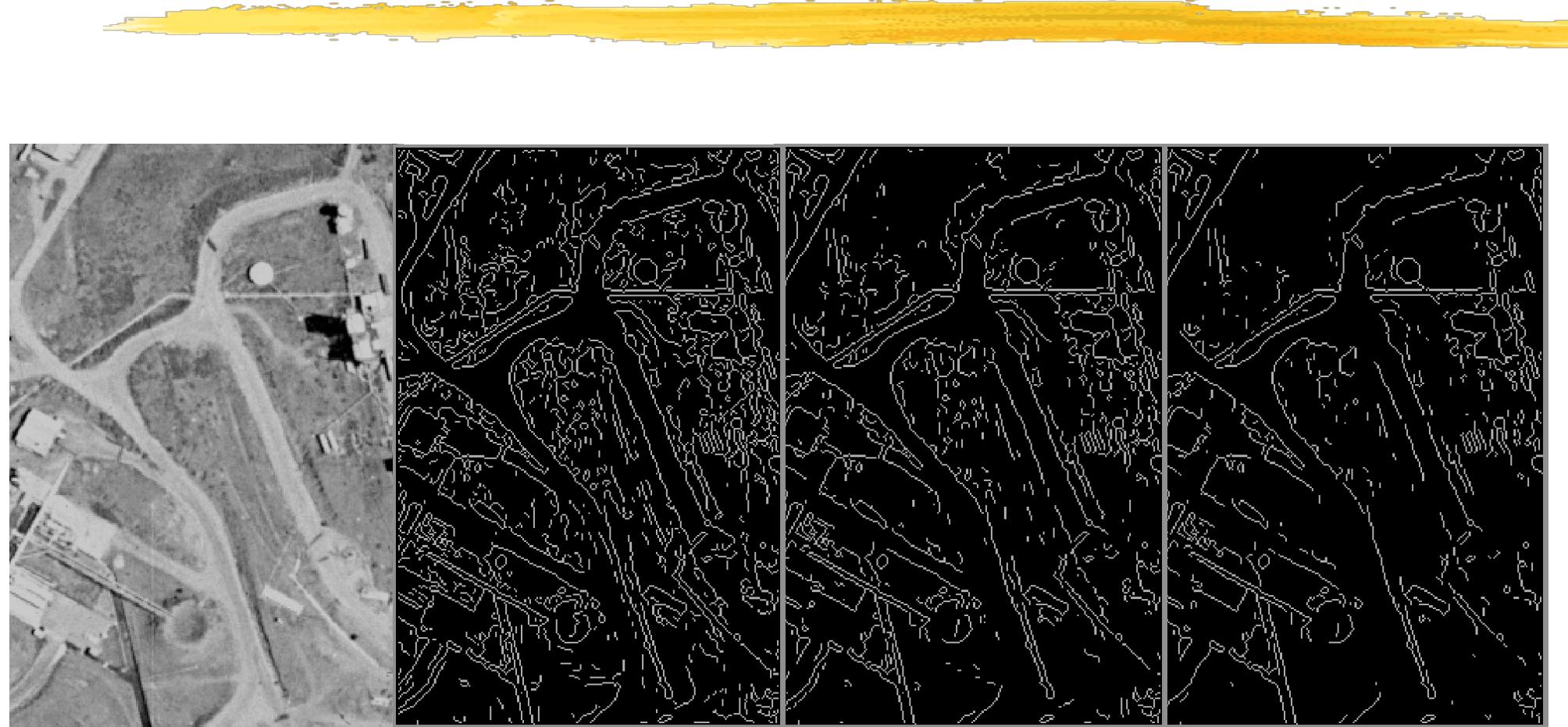
Semi-Automated Techniques:

- Dynamic Programming
- Deformable Models

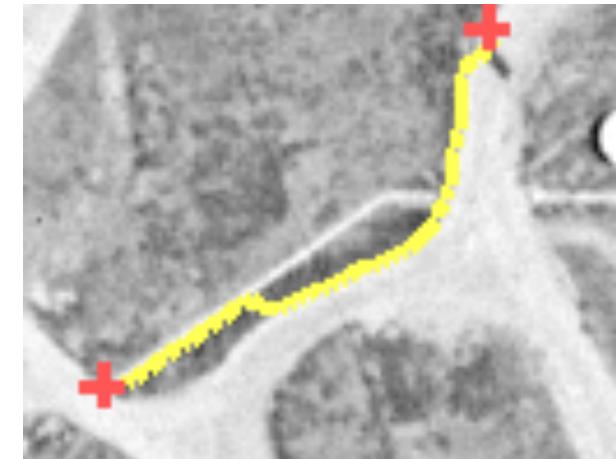
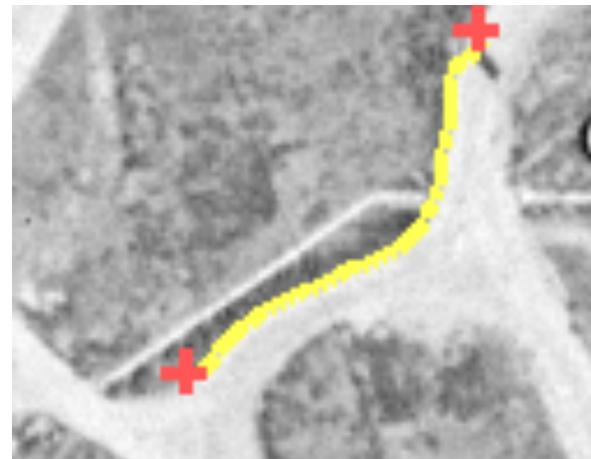
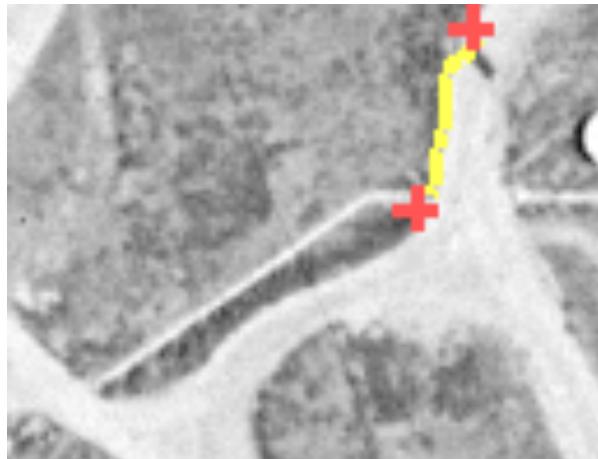
Fully Automated Techniques:

- Hough Transform
- Graph Based Approaches

ROAD IMAGE



INTERACTIVE DELINEATION

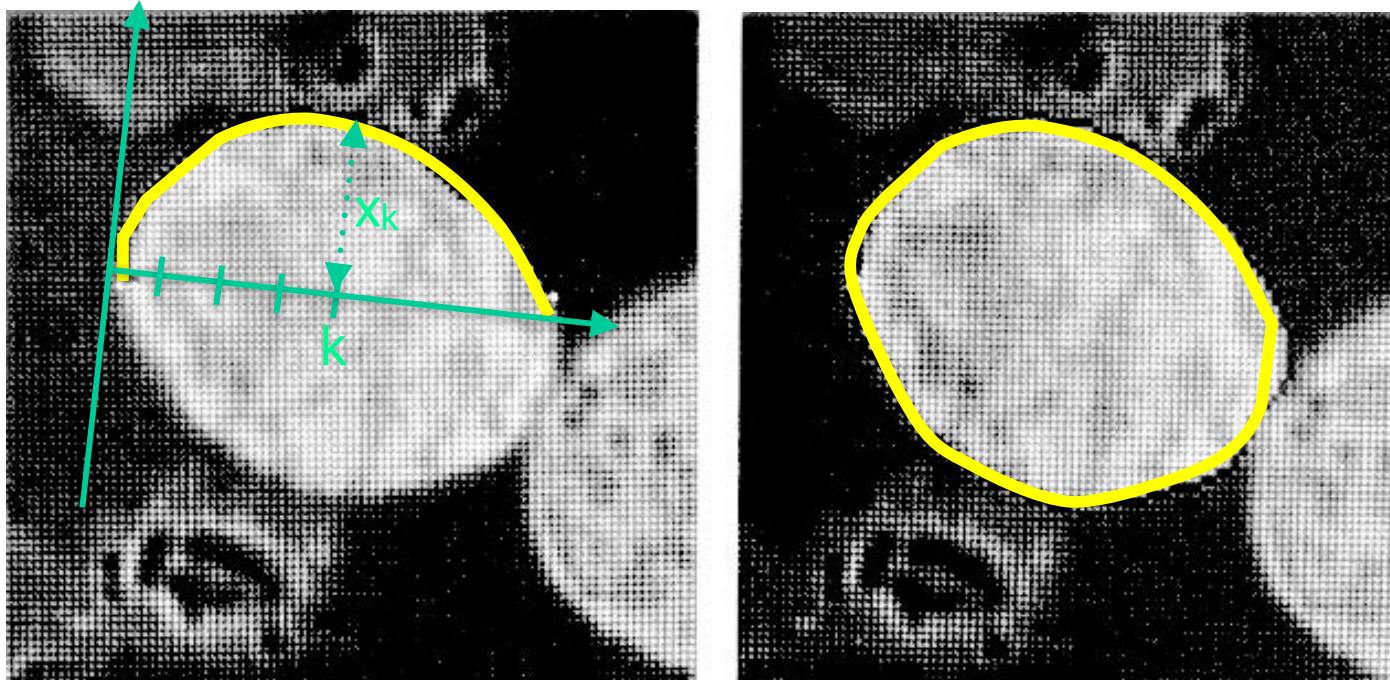


LIVE WIRE



Mortensen and Barrett,
SIGGRAPH, 1995

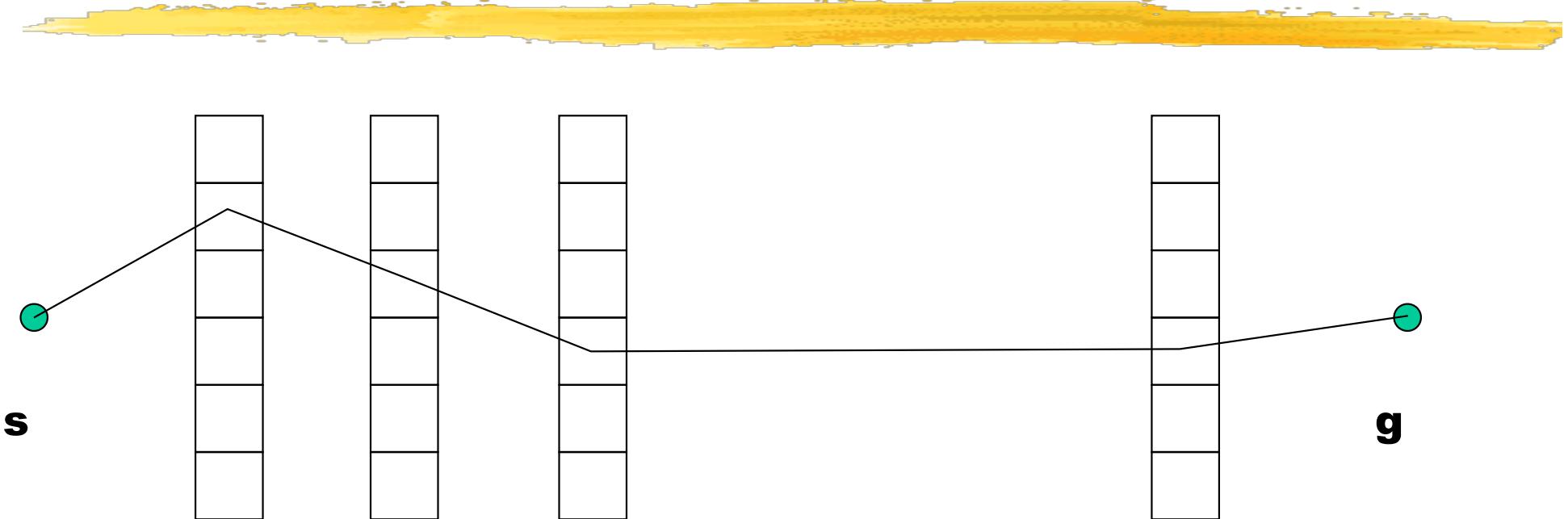
1D DYNAMIC PROGRAMMING



$$h(x_1, x_2, \dots, x_n) = - \sum_{k=1}^n g(x_k) + \sum_{k=1}^{n-1} h(x_k, x_k + 1) ,$$
$$h(x_k, x_k + 1) = \text{diff}(\phi(x_k), \phi(x_{k+1})) ,$$

where ϕ denotes the gradient orientation.

1D DYNAMIC PROGRAMMING



- N Locations
- Q Quantized values

→ Global optimum $O(NQ^2)$

1D DYNAMIC PROGRAMMING

To find

$$\min_{x_i} h(x_1, x_2, \dots, x_n)$$

where

$$h(x_1, x_2, \dots, x_n) = h(s, x_1) + \sum_{i=1}^{n-1} h(x_i, x_{i+1}) + h(x_n, g)$$

define

$$f_1(x_2) = \min_{x_1} (h(s, x_1) + h(x_1, x_2))$$

$$f_2(x_3) = \min_{x_2} (h(x_2, x_3) + f_1(x_2))$$

$$\vdots \quad \vdots \quad \vdots$$

$$f_{n-1}(x_n) = \min_{x_{n-1}} (h(x_{n-1}, x_n) + f_{n-2}(x_{n-1}))$$

$$\Rightarrow \min h(x_1, x_2, \dots, x_n) = \min_{x_n} (h(x_n, g) + f_{n-1}(x_n))$$

2D DYNAMIC PROGRAMMING

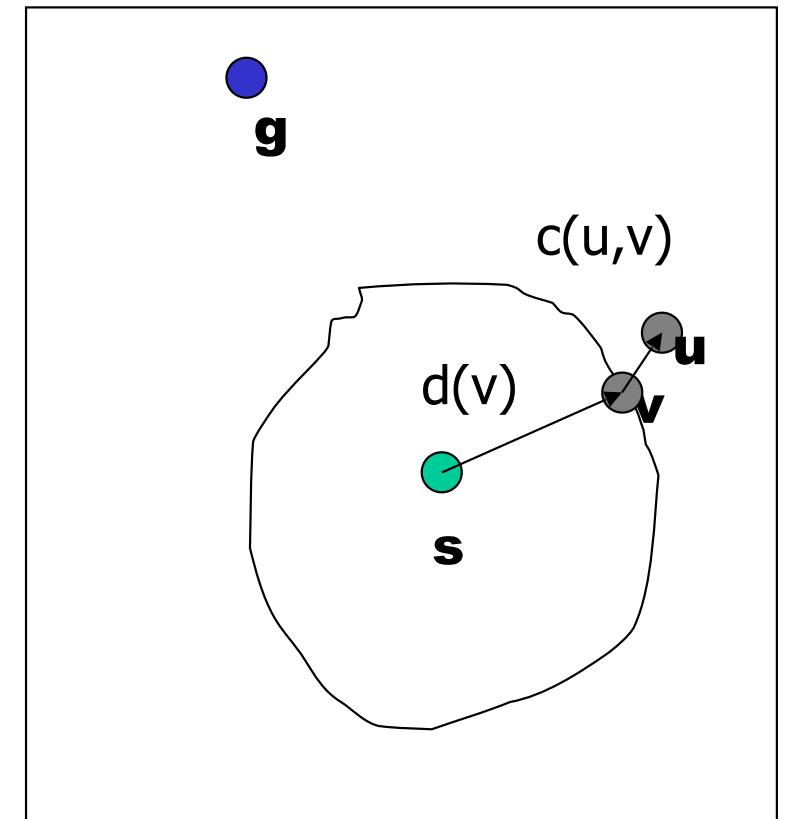
Notations:

s Start point

L List of active nodes

$c(u,v)$ Local costs for link $u \rightarrow v$

$d(v)$ Total cost from s to v



DIJKSTRA'S ALGORITHM

Initialization :

$$d(s) \leftarrow 0 \text{ and } d(u) \leftarrow \infty \text{ for } u \neq s$$

$$T = \emptyset$$

$$v = s$$

Loop until goal is reached :

$$T \leftarrow T \cup \{v\}$$

for all $v \rightarrow u$ edges such that $u \notin T$

$$\text{if } d(v) + c(v,u) < d(u)$$

$$d(u) \leftarrow d(v) + c(v,u)$$

end

end

$$v = \operatorname{argmin}_{w \notin T} d(w)$$

Maintain a sorted list of paths

LIVE WIRE

- Sorting is the expensive operation. Normally $n \log(n)$, but can be reduced to $\log(n)$ if all costs are integer costs
- Local costs computed using gradient:

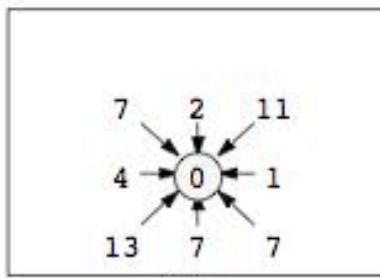
$$c(u,v) = 255 - \frac{1}{2} (g(u) + g(v))$$

- Diagonal penalized by multiplying cost of non diagonal edges by:
$$\frac{1}{\sqrt{2}} = \frac{5}{7}$$
- Add a constant cost for each edge.

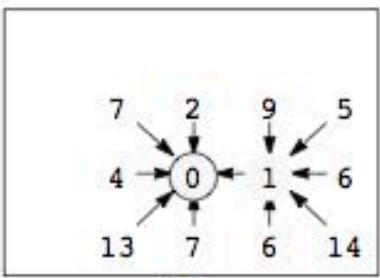
COST EXPANSION

11	13	12	9	5	8	3	1	2	4	10
14	11	7	4	2	5	8	4	6	3	8
11	6	3	5	7	9	12	11	10	7	4
7	4	6	11	13	18	17	14	8	5	2
6	2	7	10	15	15	21	19	8	3	5
8	3	4	7	9	13	14	15	9	5	6
11	5	2	8	3	4	5	7	2	5	9
12	4	(2)	1	5	6	3	2	4	8	12
10	9	7	5	9	8	5	3	7	8	15

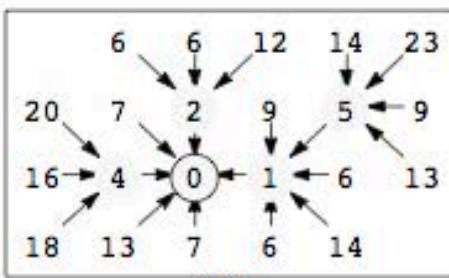
(a)



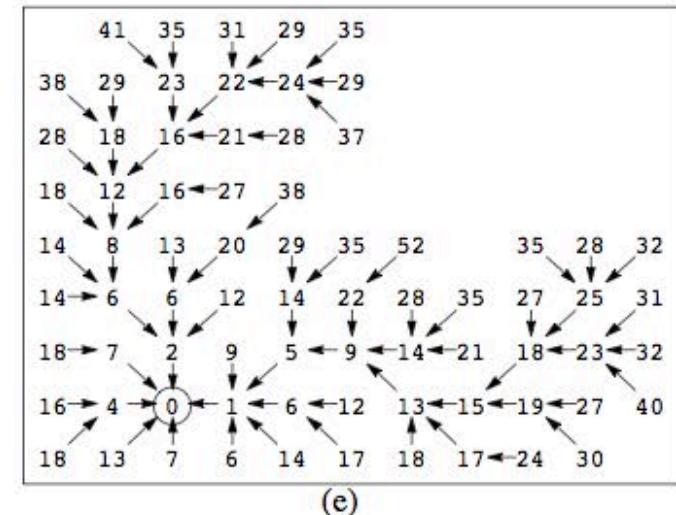
(b)



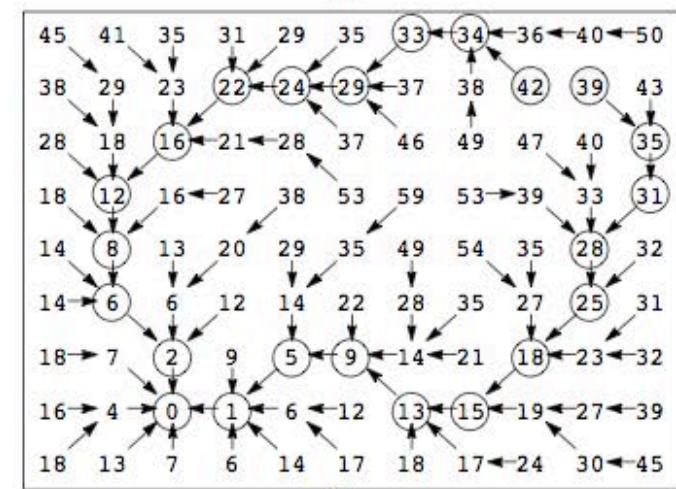
(c)



(d)



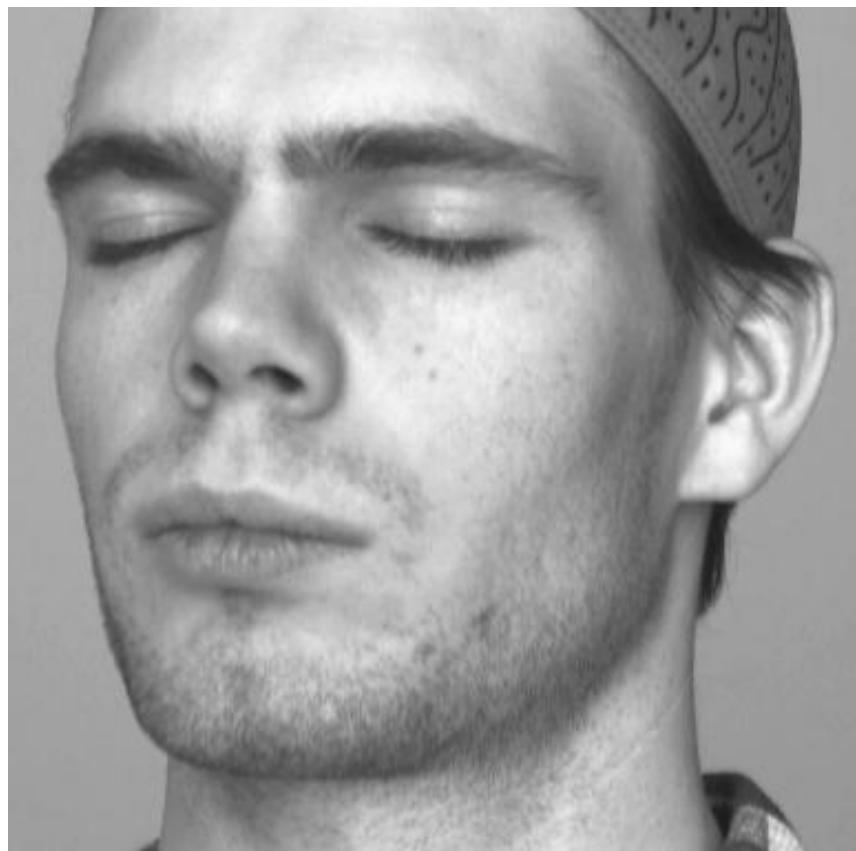
(e)



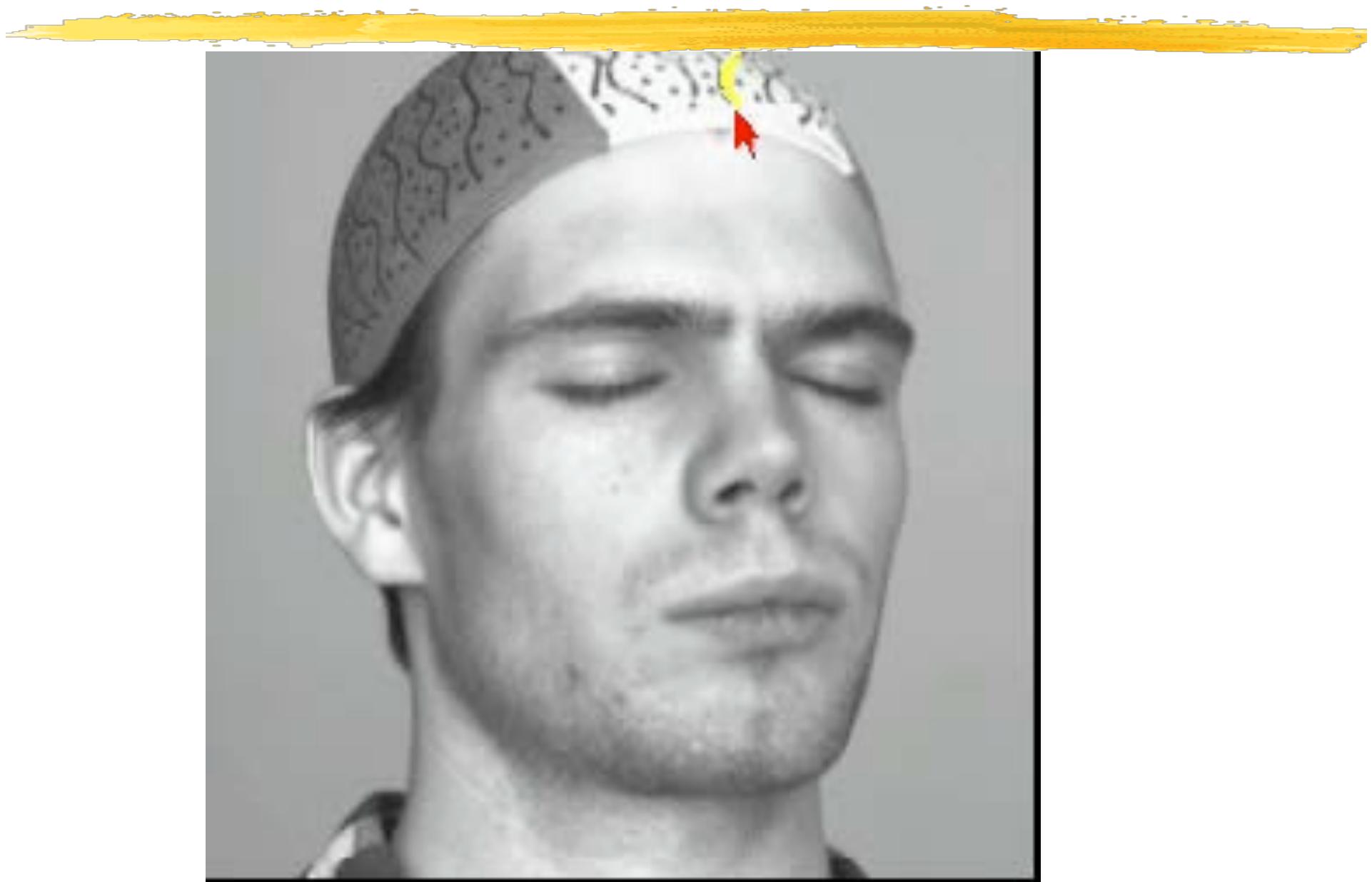
(f)

(a) Local cost map. (b) Seed point expanded. (c) 2 points expanded. (d) 5 points expanded. (e) 47 points expanded. (f) Completed cost path-pointer map with optimal paths shown from nodes with total costs 42 and 39.

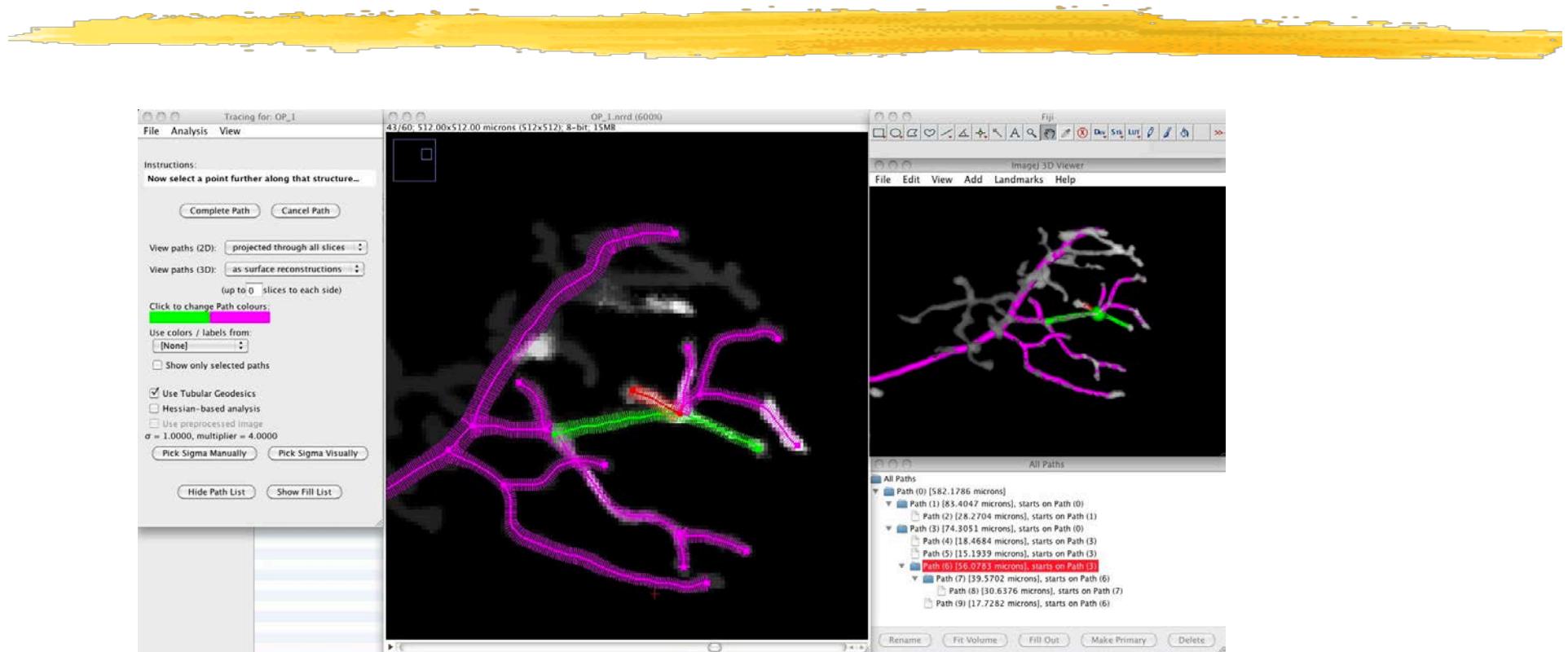
FACE IMAGE



LIVE WIRE

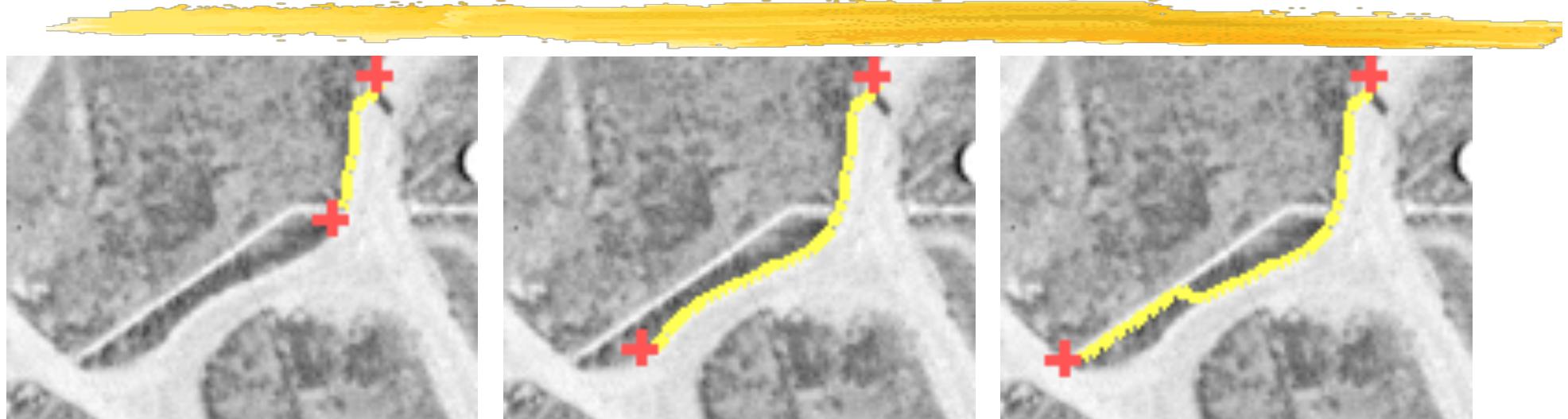


TRACING NEURONS



- Link user provided points and extracts centerline points.
- Fiji Plug-In downloadable from <http://cvlab.epfl.ch/software/delin>

OPEN ISSUES



- The “optimal” path is not always the “best” one.
 - Difficult to impose global constraints.
 - The cost grows exponentially with the dimension of the problem.
- > Must often look for local, as opposed to global, optimum using gradient descent techniques.

TECHNIQUES



Semi-Automated Techniques:

- Dynamic programming
- Deformable Models

Fully Automated Techniques:

- Hough transform
- Graph Based Approaches

SNAKES



ITERATING



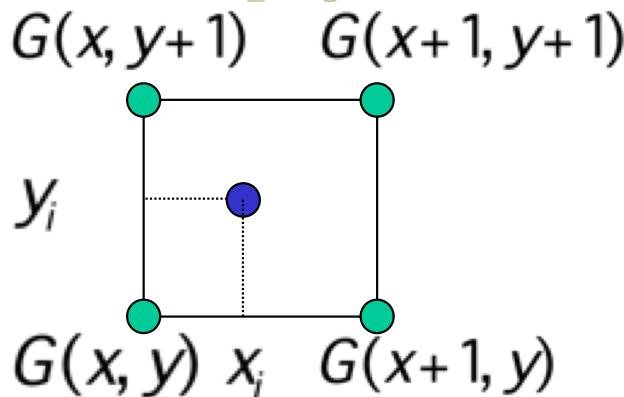
At every step:

$$0 = \frac{\delta E_G}{\delta X} + KX_t + \alpha(X_t - X_{t-1}) \Rightarrow (K + \alpha I)X_t = \alpha X_{t-1} - \frac{\delta E_G}{\delta X}$$

$$0 = \frac{\delta E_G}{\delta Y} + KY_t + \alpha(Y_t - Y_{t-1}) \Rightarrow (K + \alpha I)Y_t = \alpha Y_{t-1} - \frac{\delta E_G}{\delta Y}$$

→ Solve two linear equations at each iteration.

DERIVATIVES OF THE GRADIENT

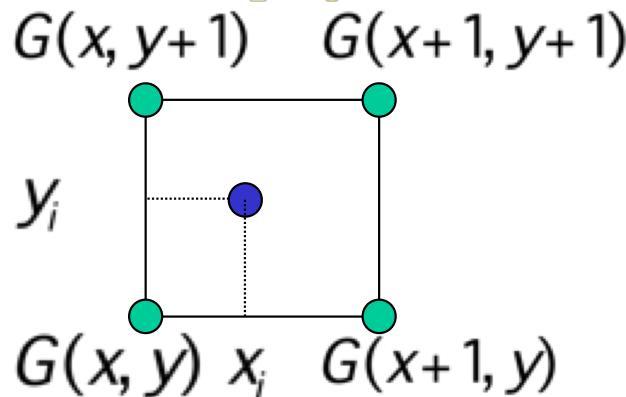


$$E_G = -\frac{I}{N} \sum_{i=1}^N G(x_i, y_i)$$

$$\frac{\partial E_G}{\partial X} = \begin{bmatrix} \frac{\partial E_G}{\partial x_1} & \dots & \frac{\partial E_G}{\partial x_N} \end{bmatrix}, \quad \frac{\partial E_G}{\partial Y} = \begin{bmatrix} \frac{\partial E_G}{\partial y_1} & \dots & \frac{\partial E_G}{\partial y_N} \end{bmatrix}$$

$$\frac{\partial E_G}{\partial x_i} = -\frac{I}{N} \frac{\partial G}{\partial x_i}(x_i, y_i), \quad \frac{\partial E_G}{\partial y_i} = -\frac{I}{N} \frac{\partial G}{\partial y_i}(x_i, y_i)$$

BILINEAR INTERPOLATION



$$p = x_i - x$$

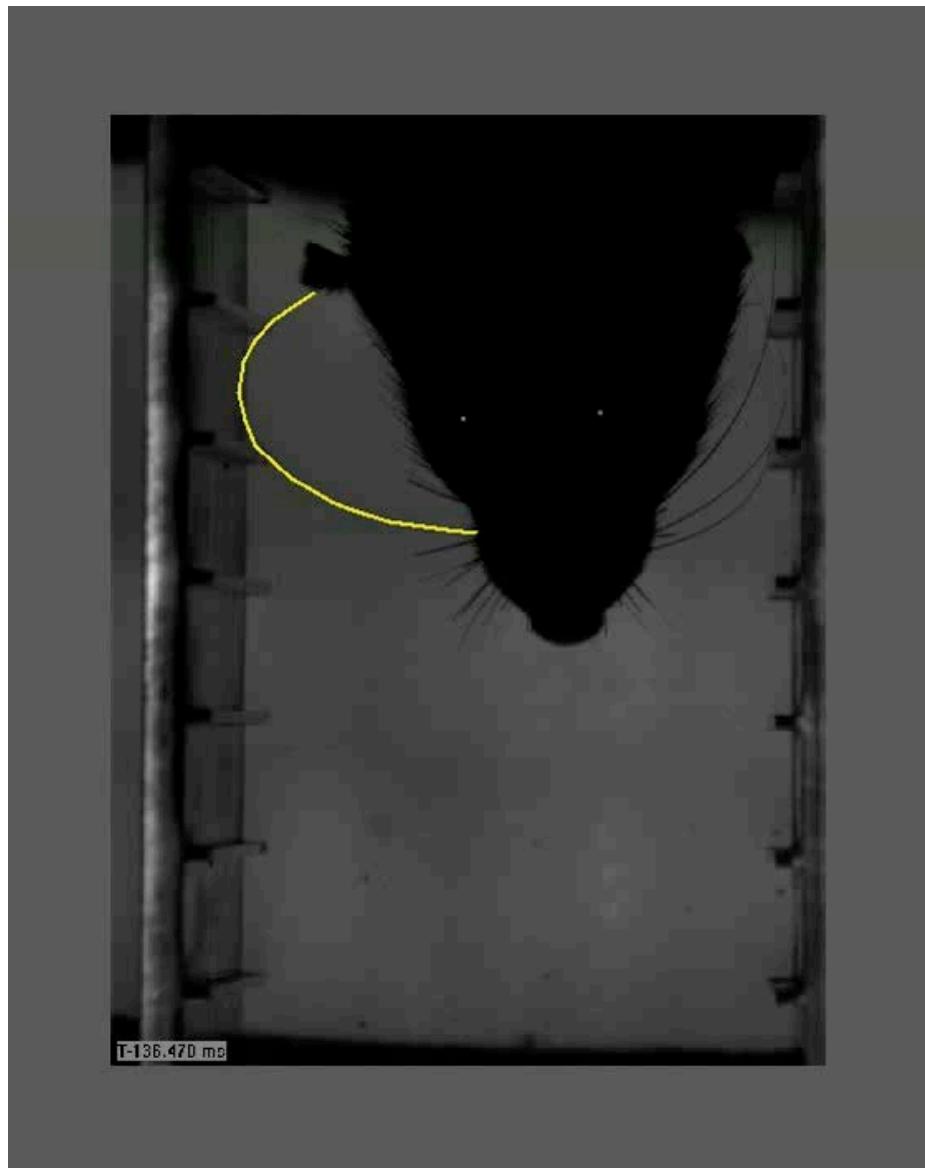
$$q = y_i - y$$

$$G(x_i, y_i) = (1 - p)(1 - q)G(x, y) + (1 - p)qG(x, y + 1) + p(1 - q)G(x + 1, y) + pqG(x + 1, y + 1)$$

$$\frac{\partial G}{\partial x_i} = (1 - q)(G(x + 1, y) - G(x, y)) + q(G(x + 1, y + 1) - G(x, y + 1))$$

$$\frac{\partial G}{\partial y_i} = (1 - p)(G(x, y + 1) - G(x, y)) + p(G(x + 1, y + 1) - G(x + 1, y))$$

OPEN AND CLOSED SNAKES



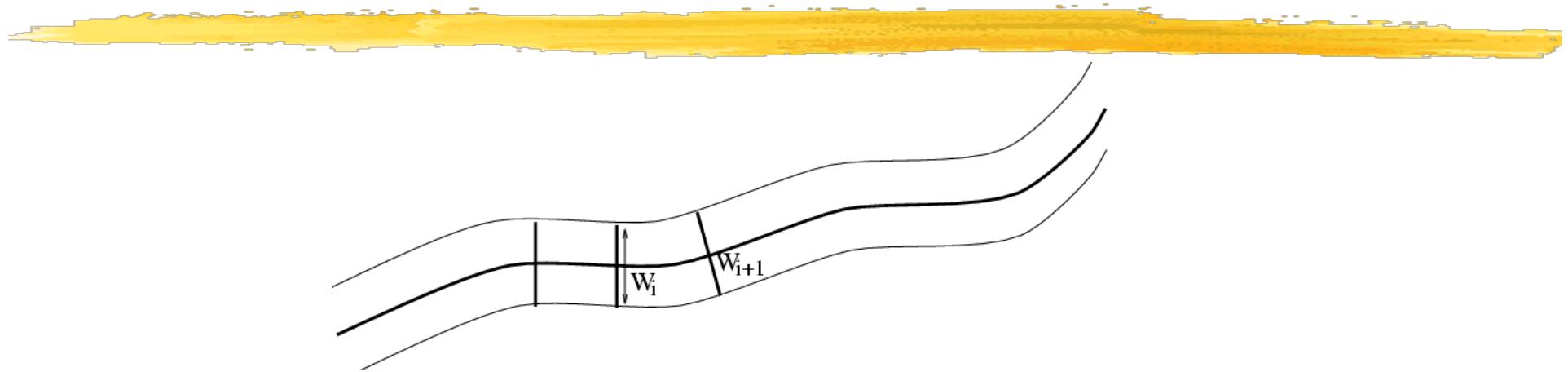
NETWORK SNAKES



--> Updated field boundaries.

Butenuth, Phd 2008

RIBBON SNAKES



$$E = E_G + 1/2X^t K X + 1/2Y^t K Y$$

$$W = [w_1, \dots, w_N]^t$$

DYNAMICS EQUATIONS



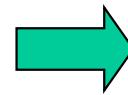
$$(K + \alpha I)X_t = \alpha X_{t-1} - \frac{\delta E_G}{\delta X}$$

$$(K + \alpha I)Y_t = \alpha Y_{t-1} - \frac{\delta E_G}{\delta Y}$$

$$(K + \alpha I)W_t = \alpha W_{t-1} - \frac{\delta E_G}{\delta W}$$

→ Solve three linear equations at each iteration.

DELINEATING ROADS



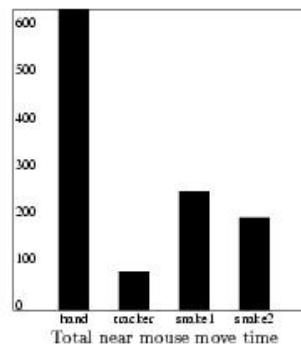
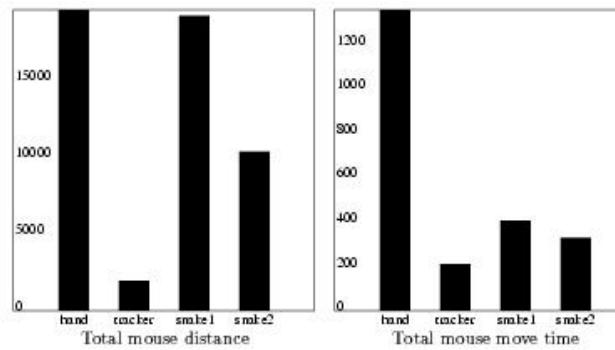
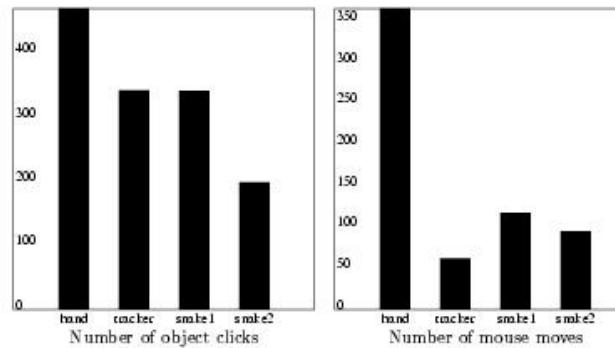
DELINEATING ROADS



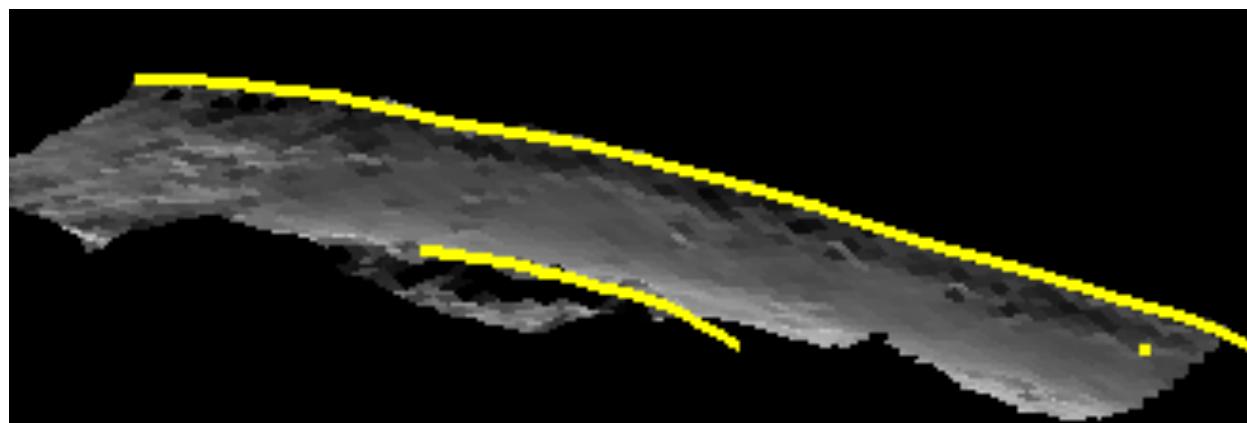
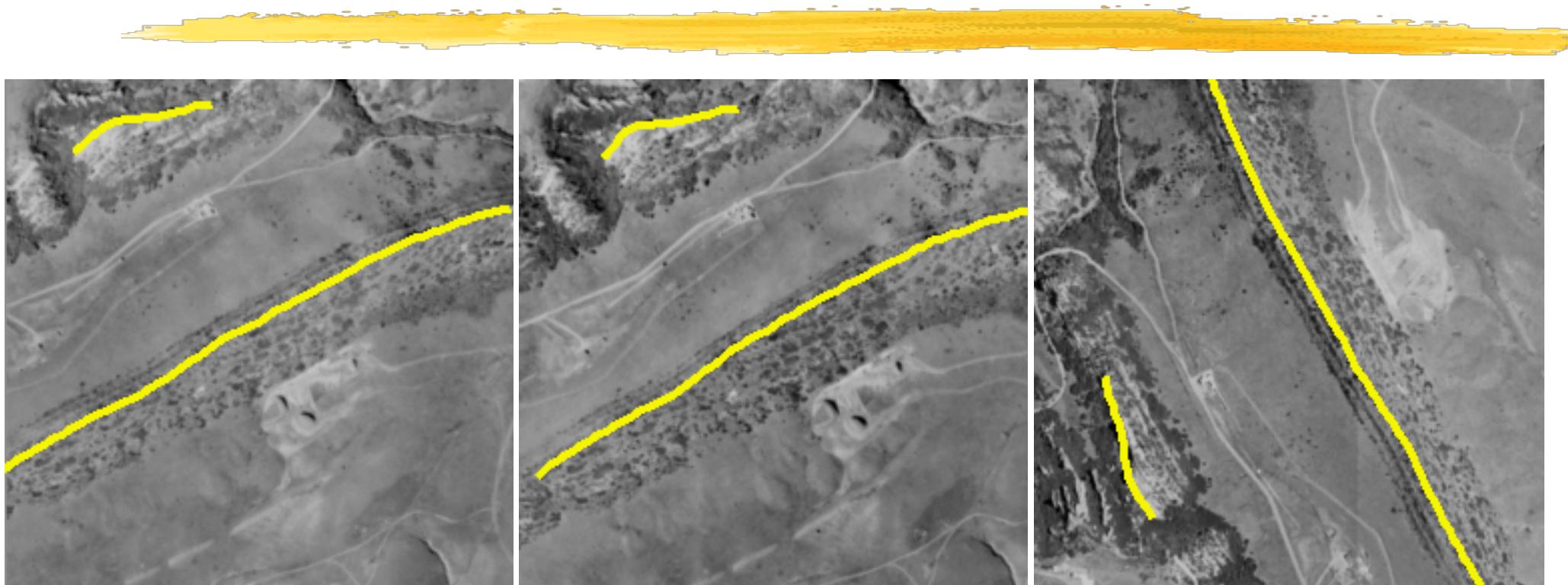
DELINEATING ROADS



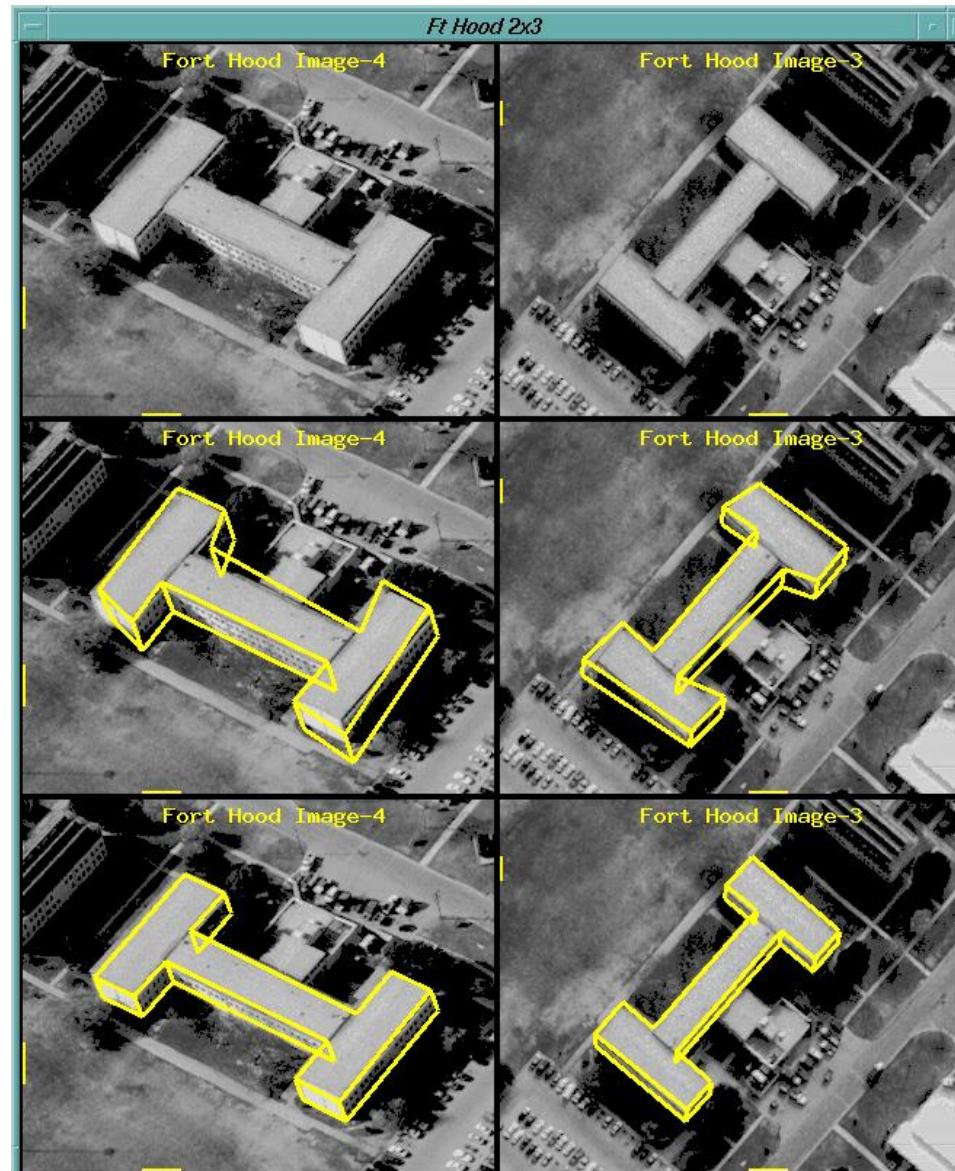
EVALUATION



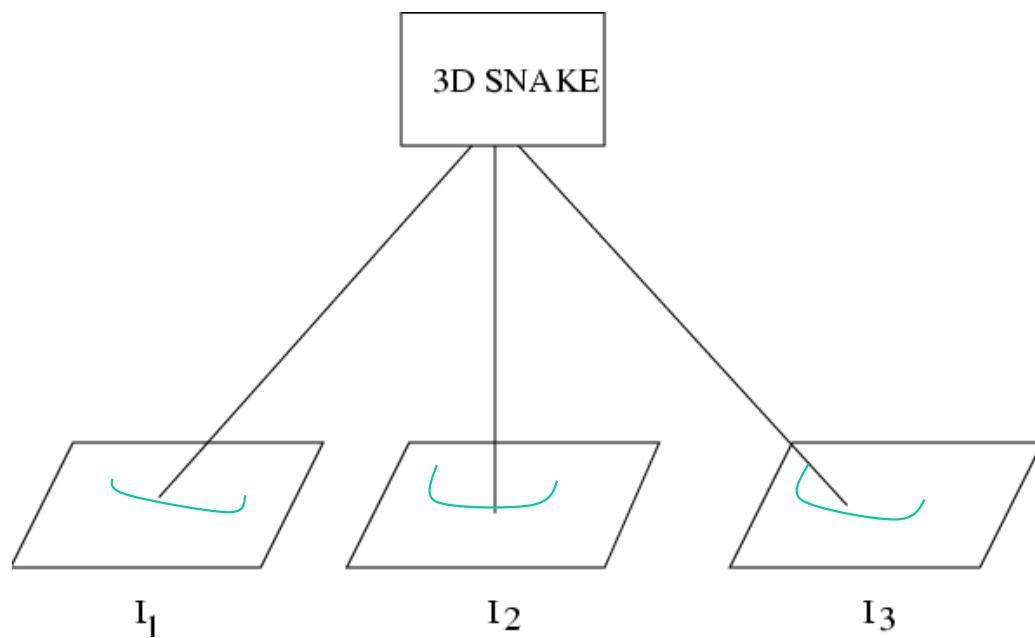
RIDGE LINES



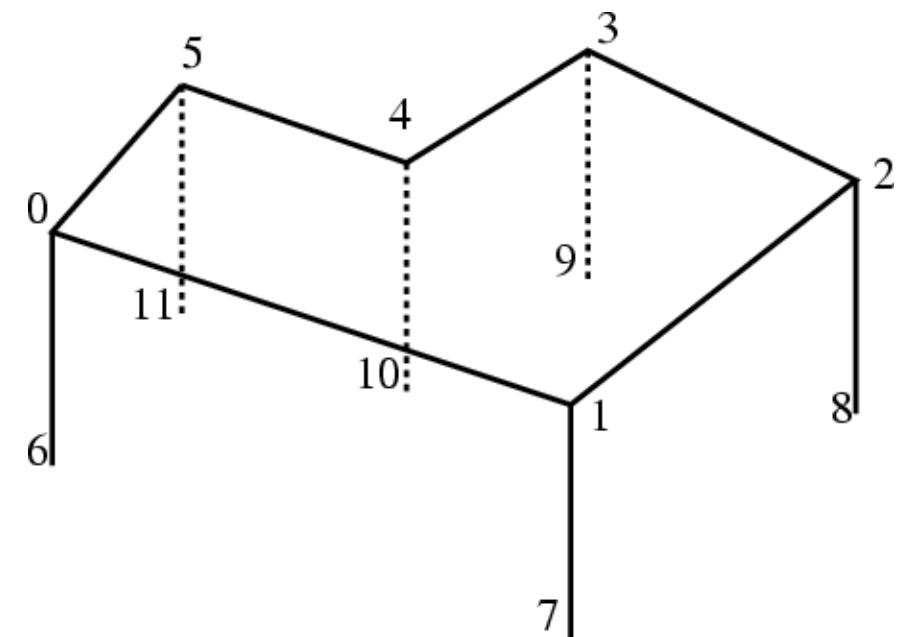
BUILDING



3—D SNAKES



Smooth 3—D snake



Rectilinear 3—D snake

DYNAMICS EQUATIONS

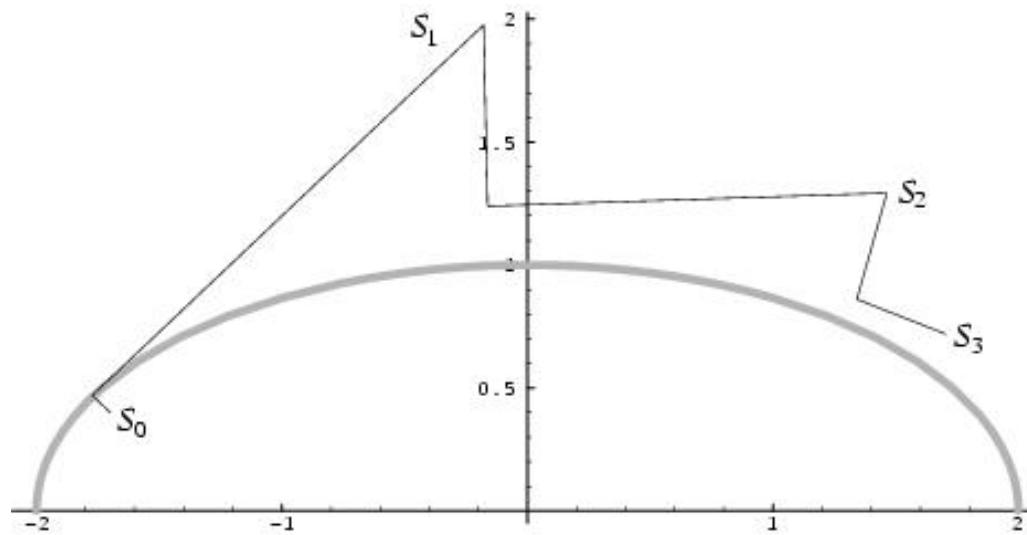
$$(K + \alpha I)X_t = \alpha X_{t-1} - \frac{\delta E_G}{\delta X}$$

$$(K + \alpha I)Y_t = \alpha Y_{t-1} - \frac{\delta E_G}{\delta Y}$$

$$(K + \alpha I)Z_t = \alpha Z_{t-1} - \frac{\delta E_G}{\delta Z}$$

→ Solve three linear equations at each iteration.

CONSTRAINED OPTIMIZATION

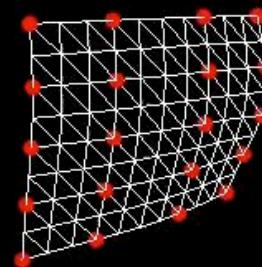
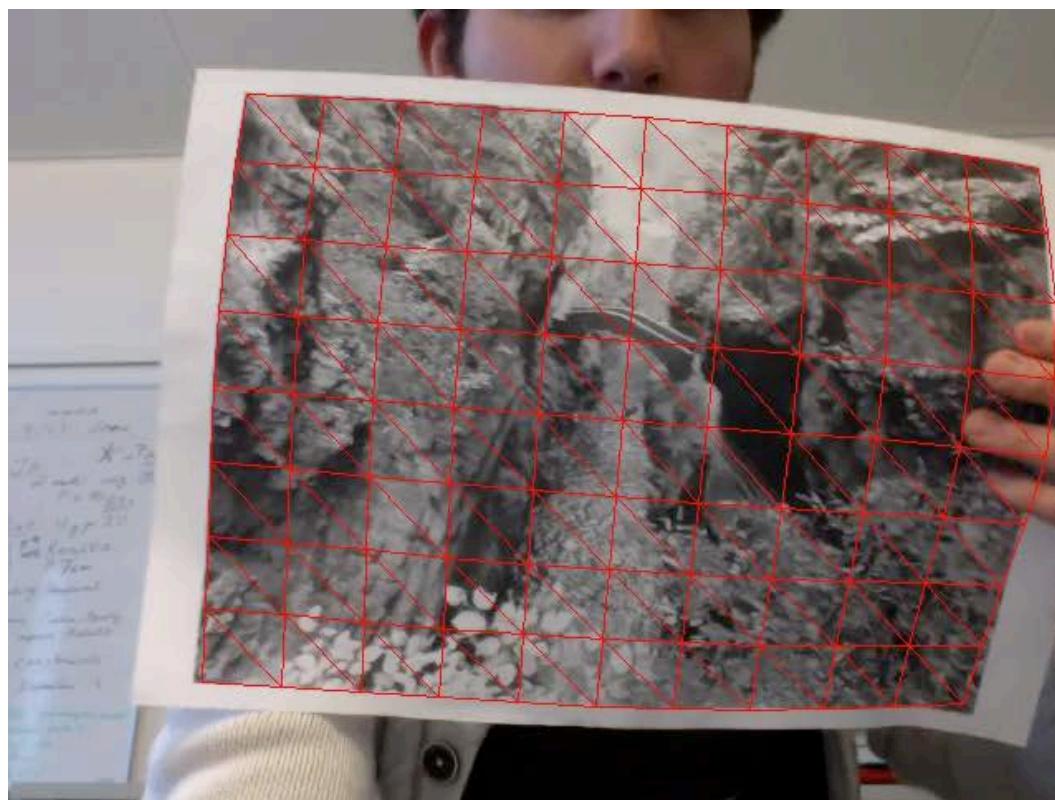


Minimize $F(S)$ subject to $C(S) = 0$

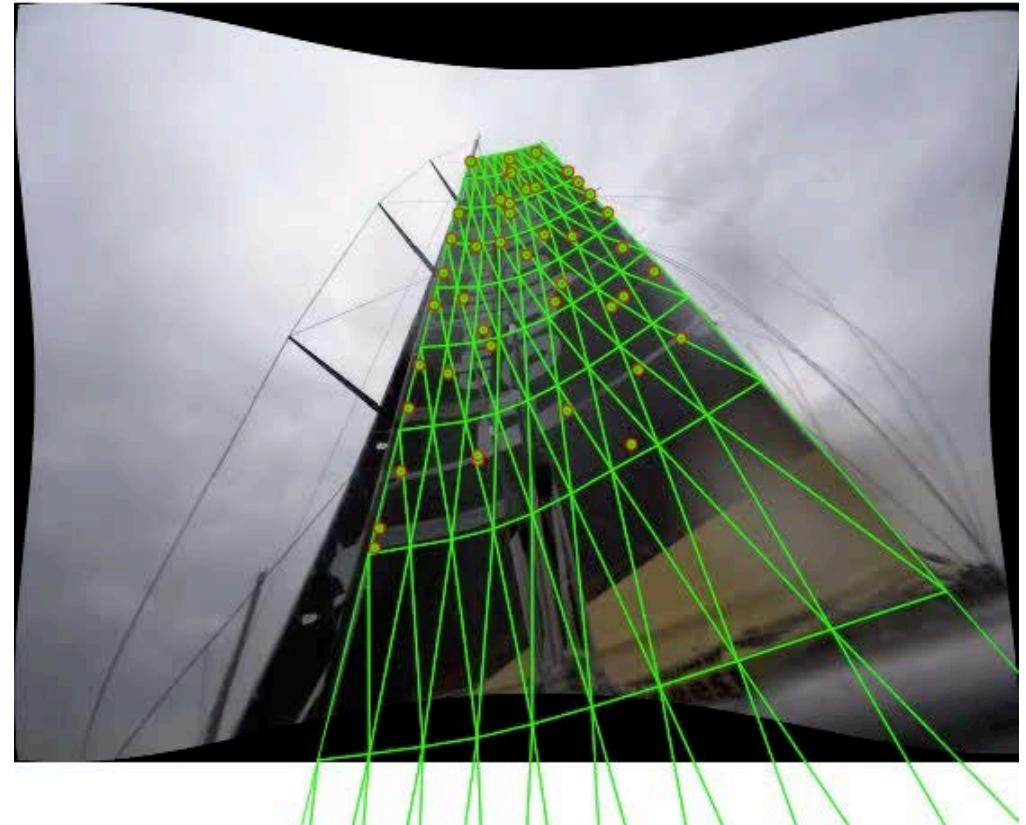
SITE MODELING



REAL-TIME 3D SHAPE



SAILS

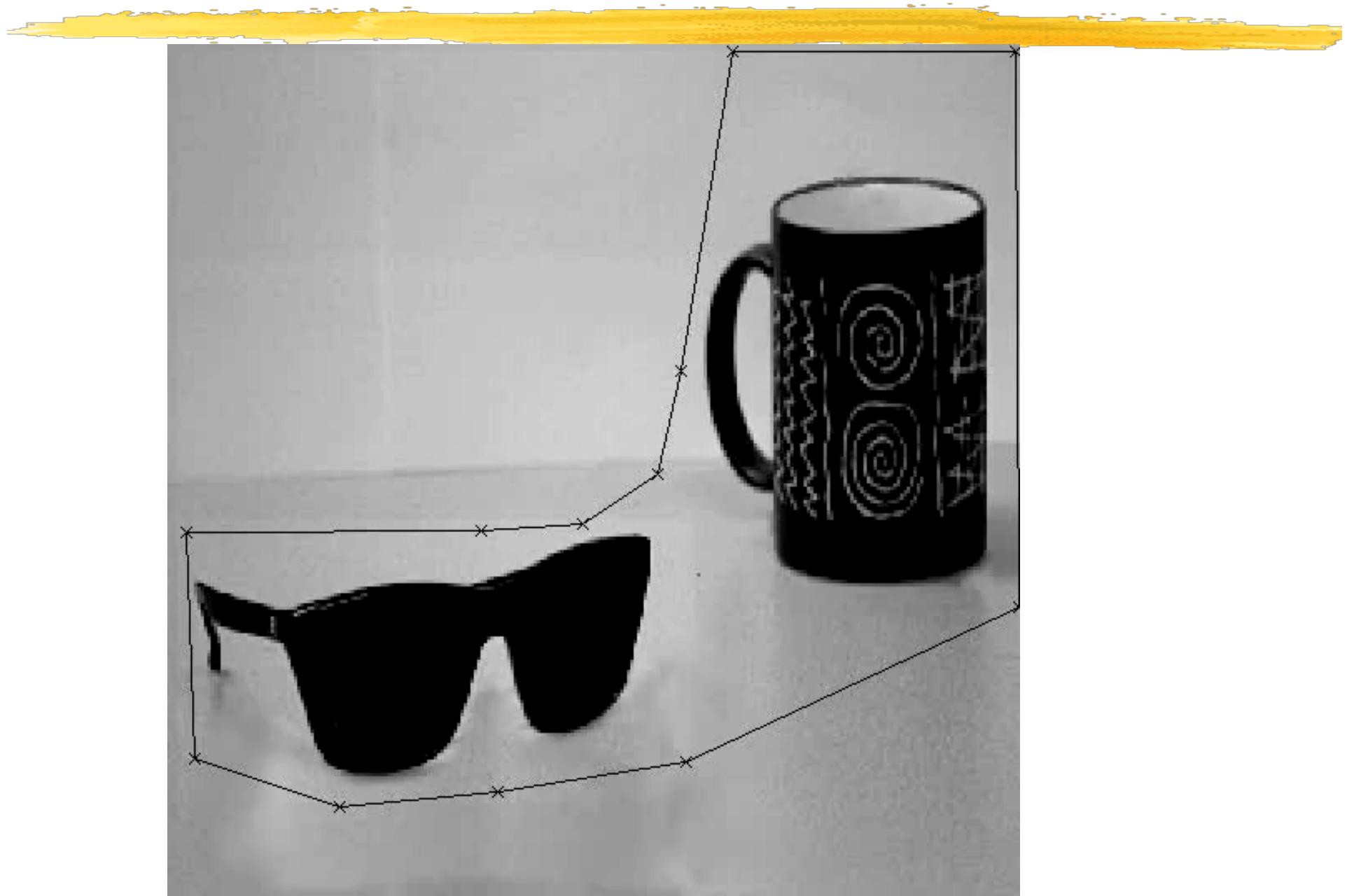


→ Real-time performance evaluation.

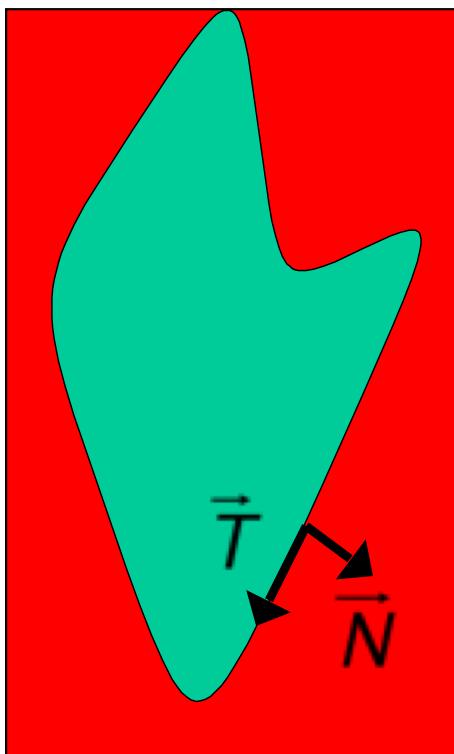
WINGS



LEVEL SETS



CURVE EVOLUTION



Generic formulation: $\frac{\partial C}{\partial t} = \alpha(s, t)\vec{T} + \beta(s, t)\vec{N}$

Reparameterization: $\frac{\partial C}{\partial t} = \beta(s, t)\vec{N}$

Special cases:

Prairie fire $\beta = \pm 1$

Diffusion $\beta = \beta_0 - \beta_1 \kappa$

Problems:

1. Involves curve sampling and resampling.
2. Curvature estimates are numerically unstable.

LEVEL SETS

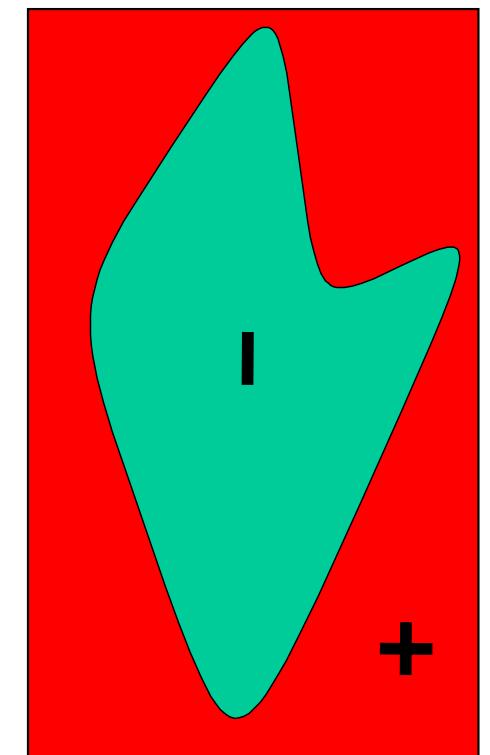
Consider the curve as the zero level set of the surface.

$$z = \Phi(x, y, t)$$

Corresponding evolution equation.

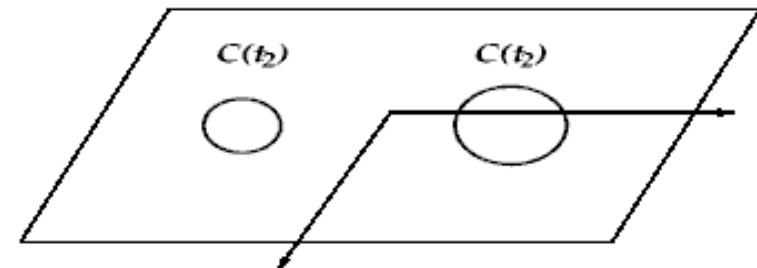
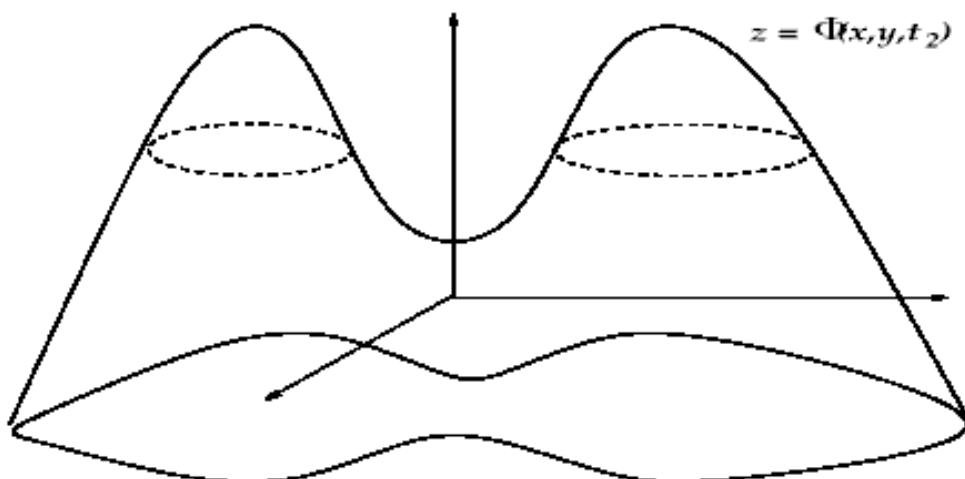
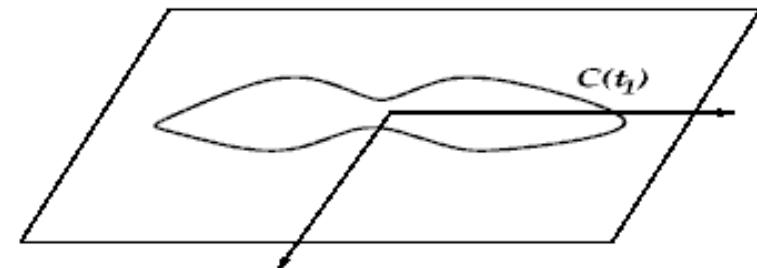
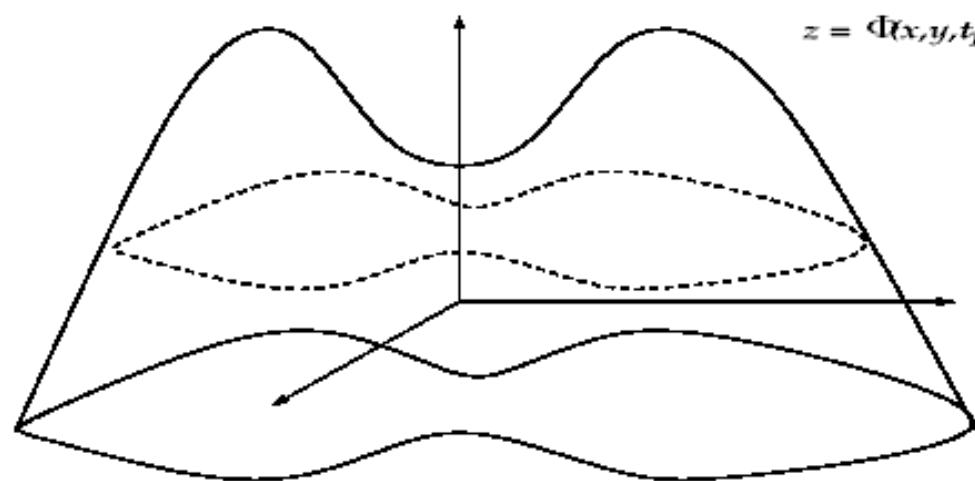
$$0 = \Phi_t + \beta(\kappa) |\nabla \Phi|$$

where $\kappa = \frac{\Phi_{xx}\Phi_y^2 - 2\Phi_{xy}\Phi_x\Phi_y + \Phi_{yy}\Phi_x^2}{\Phi_x^2 + \Phi_y^2}$



→ Much better numerical stability.

TOPOLOGY CHANGE



LEVEL SET SMOOTHING



Smoothing occurs when $\beta(\kappa) = -\kappa$

Desirable properties:

- Converges towards circles.
- Total curvature decreases.
- Number of curvature extrema and zeros of curvature decreases.

Relationship with Gaussian smoothing:

- Analogous to Gaussian smoothing of boundary over the short run, but does not cause self-intersections or overemphasize elongated parts.
- Can be implemented by Gaussian smoothing the characteristic function of a region.

SHAPE RECOVERY



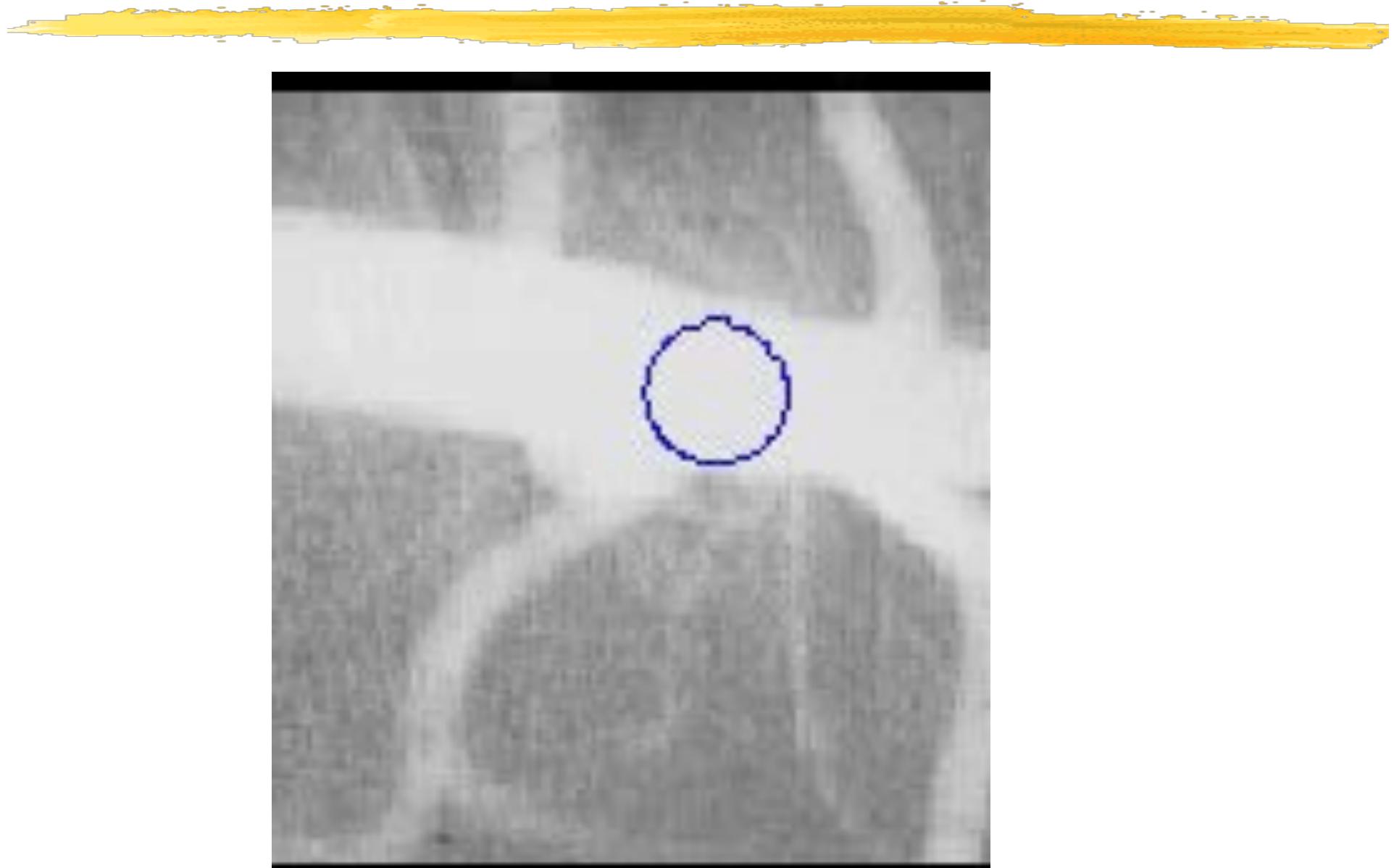
Evolution equation: $0 = \Phi_t + \beta(\kappa) |\nabla \Phi|$

where: $\beta(\kappa) = k_I(1 - \epsilon\kappa)$

$$k_I = \frac{1}{1 + \nabla I}$$

→ Expansion stops at the boundaries.

LEVEL SETS



LEVEL SETS



TECHNIQUES



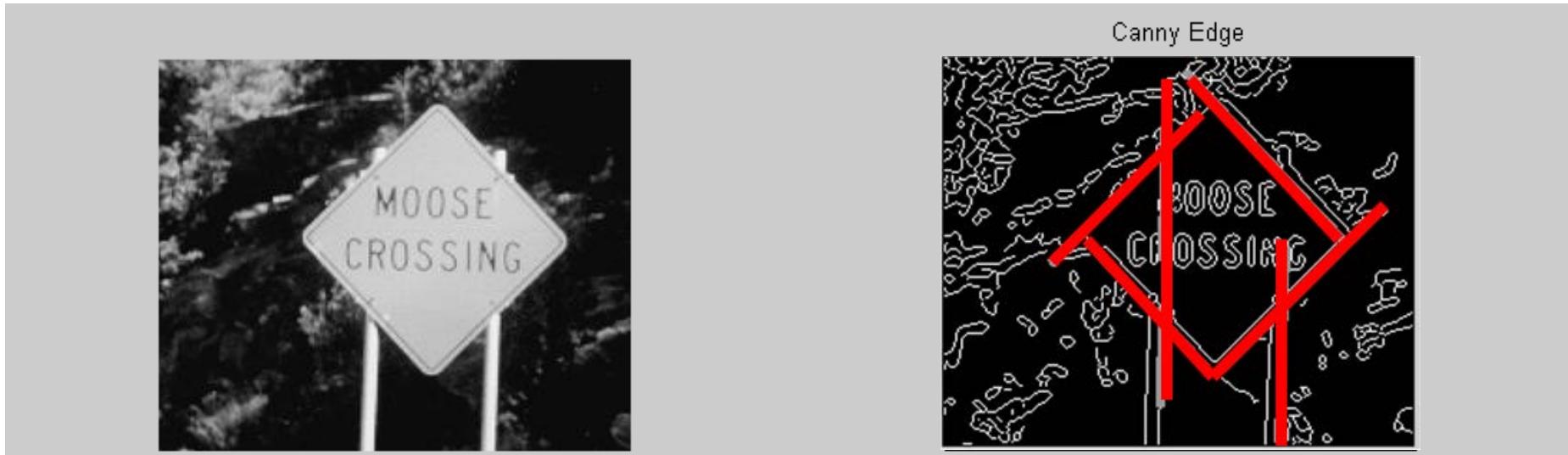
Semi-Automated Techniques:

- Dynamic programming
- Deformable Models

Fully Automated Techniques:

- Hough transform
- Graph Based Approaches

FINDING LINES



Input:

- Canny edge points.
- Gradient magnitude and orientation.

Output:

- All straight lines in image.

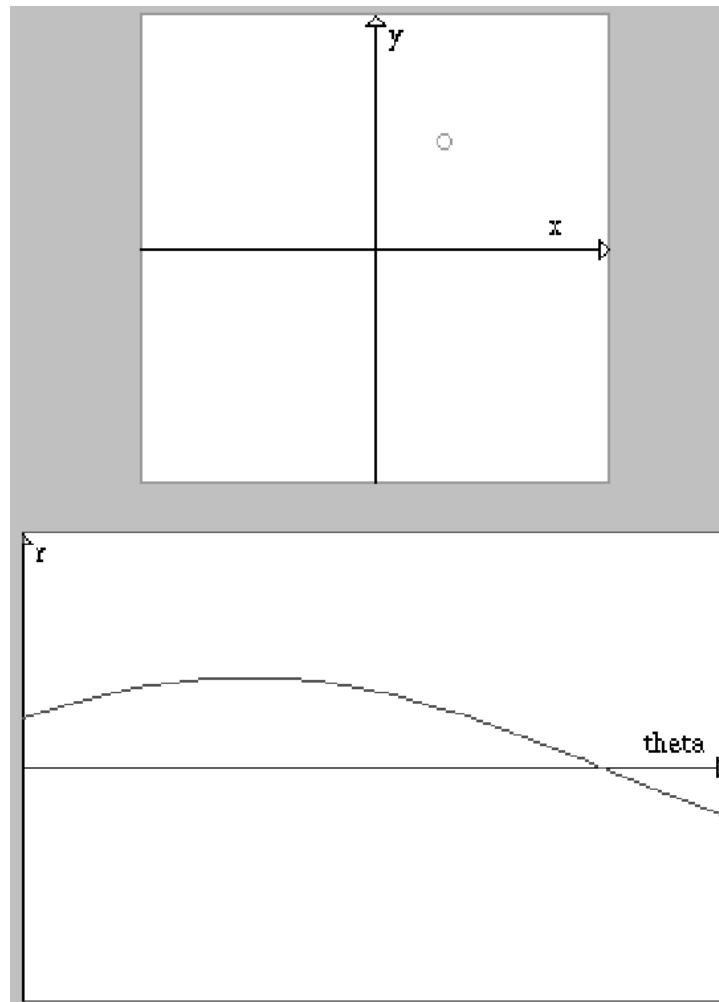
HOUGH TRANSFORM



Given a parametric model of a curve:

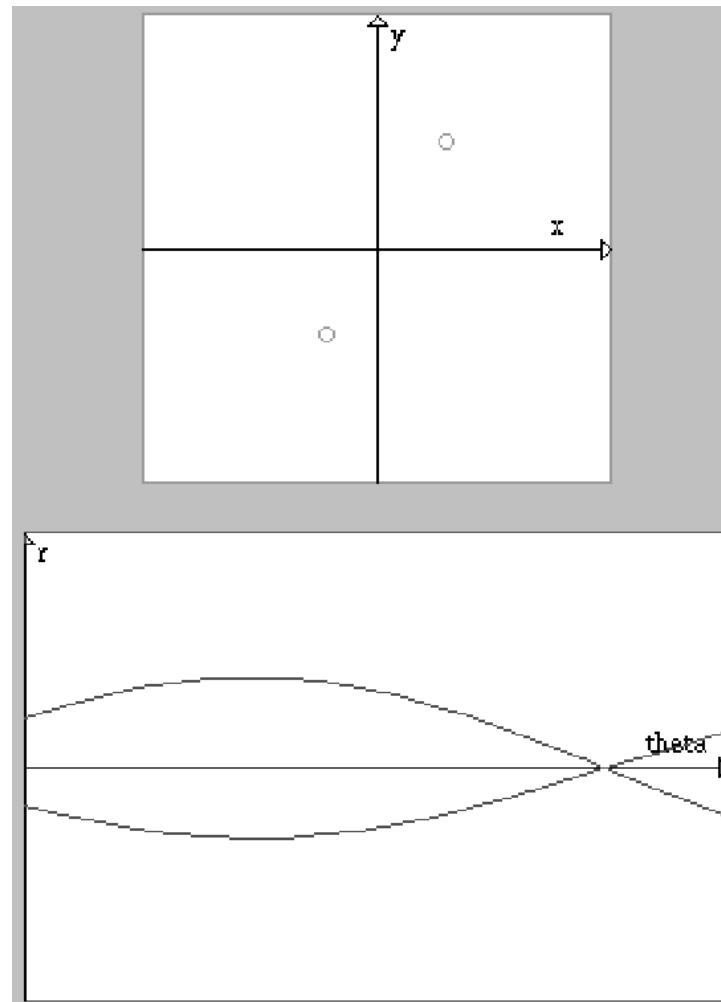
- Map each contour point onto the set of parameter values for which the curves passes through it.
- Find the intersection for all parameter sets thus mapped.

VOTING SCHEME

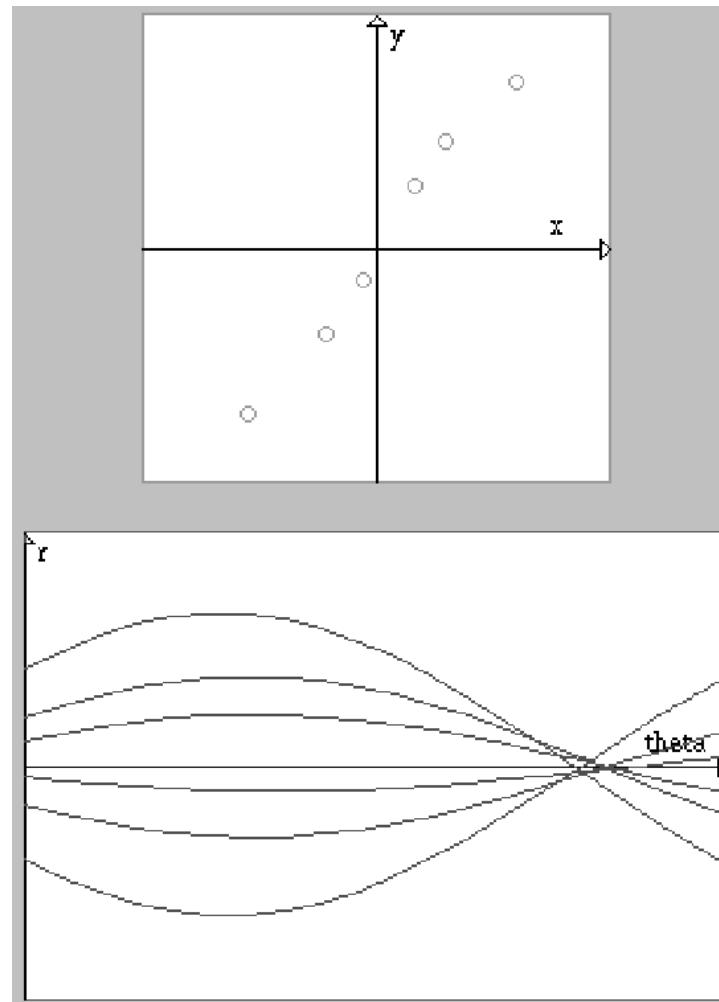


$$x \cos(\theta) + y \sin(\theta) = r, \quad 0 \leq \theta < \pi$$

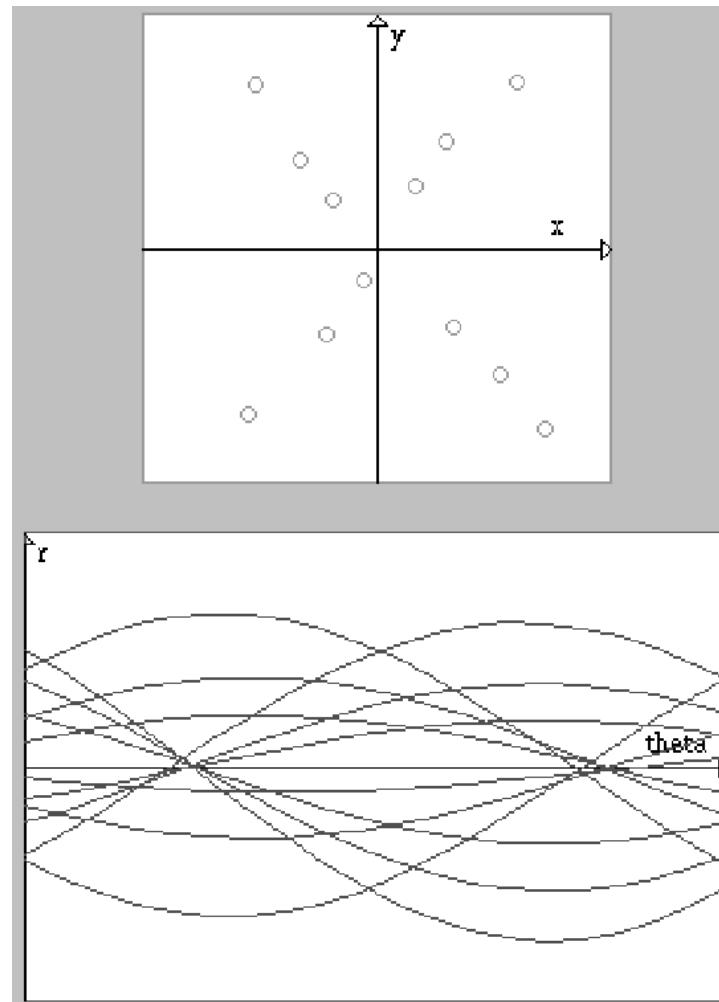
VOTING SCHEME



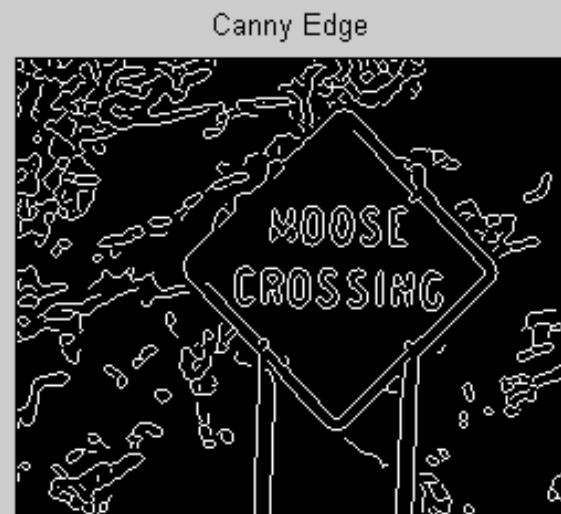
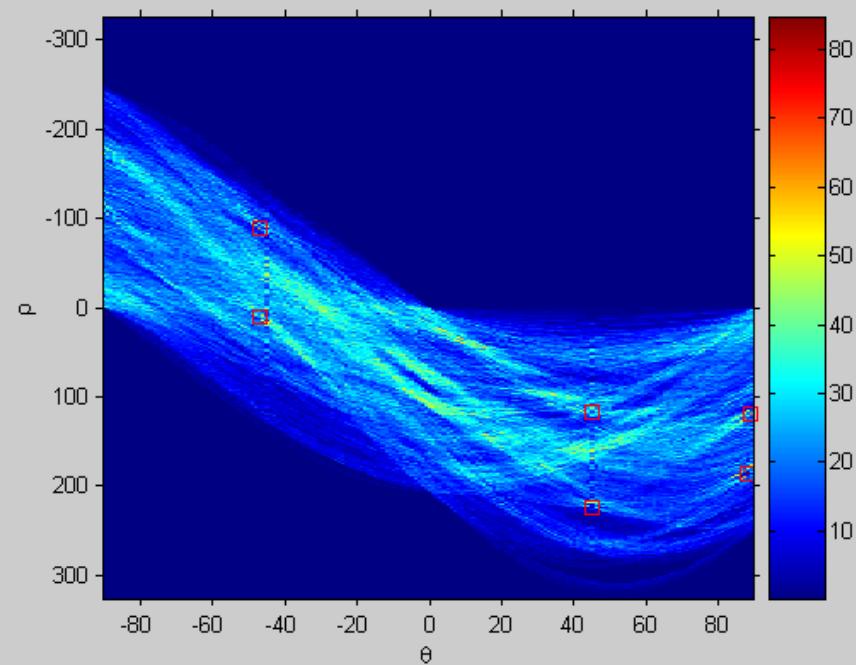
VOTING SCHEME



VOTING SCHEME



REAL WORLD LINES

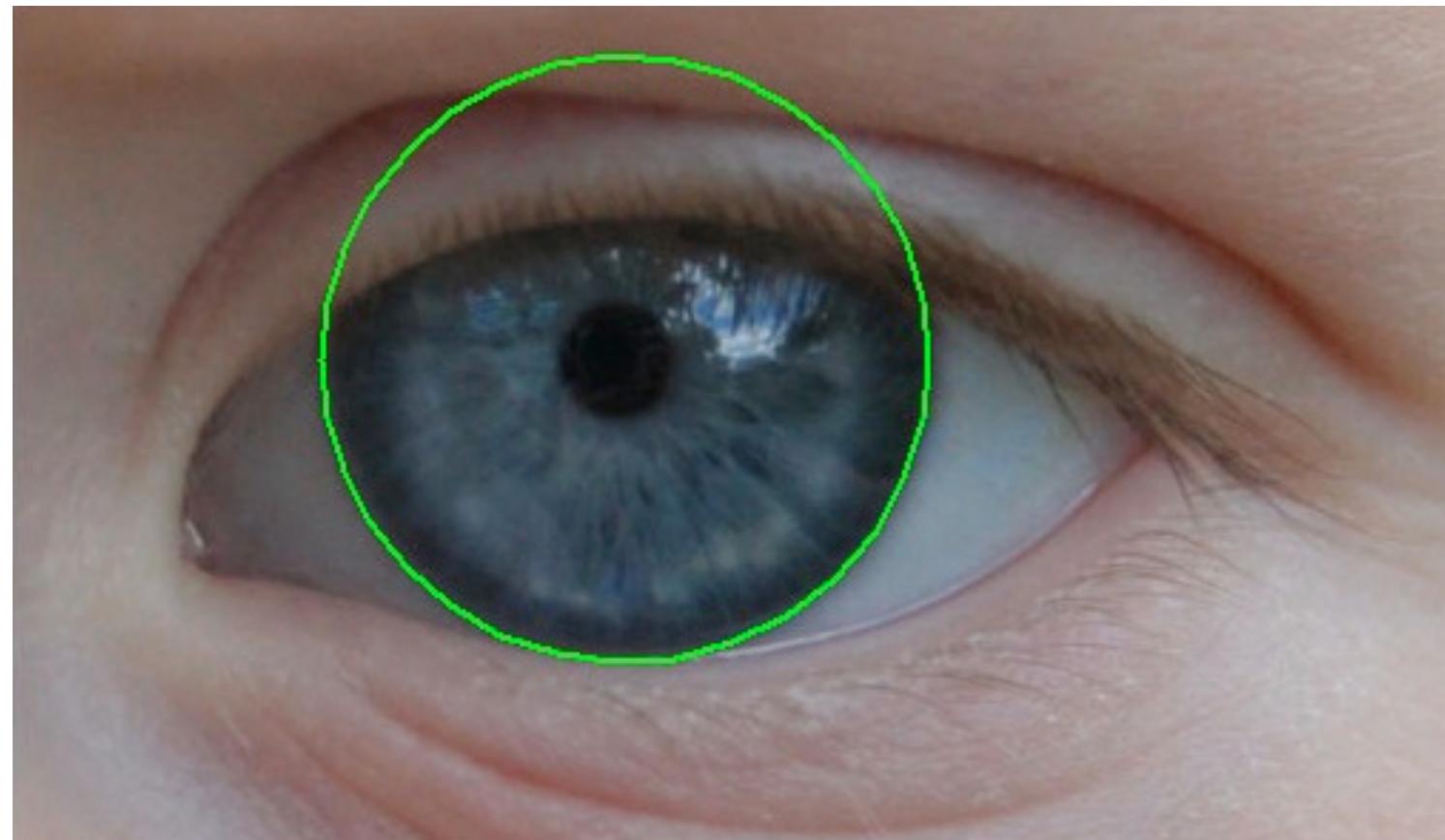


GENERIC ALGORITHM



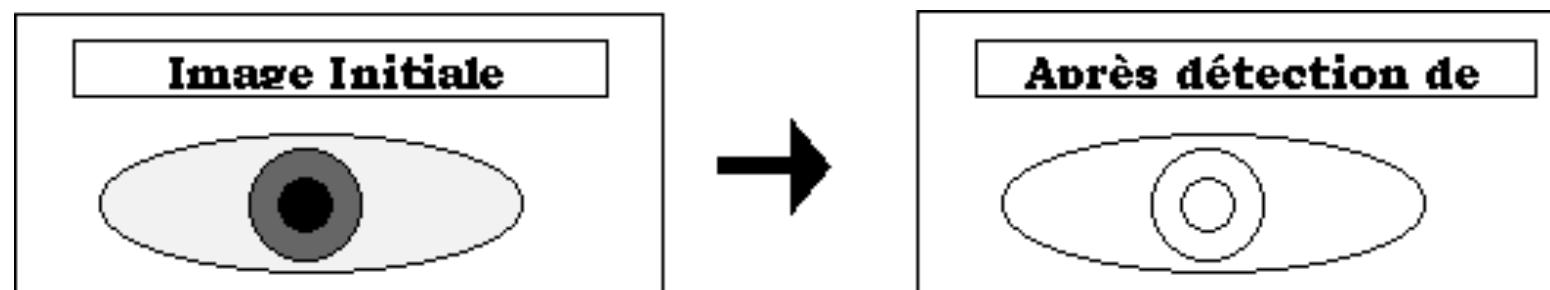
- Quantize parameter space with 1 dimension per parameter.
- Form an accumulator array.
- For each point in the gradient image such that the gradient strength exceeds a threshold, increment appropriate element of the accumulator.
- Find local maxima in the accumulator.

IRIS DETECTION



OCCLUSIONS

In theory:



In practice:



CIRCLE DETECTION

Circle of equation:

$$x = x_0 + r \cos(\theta)$$

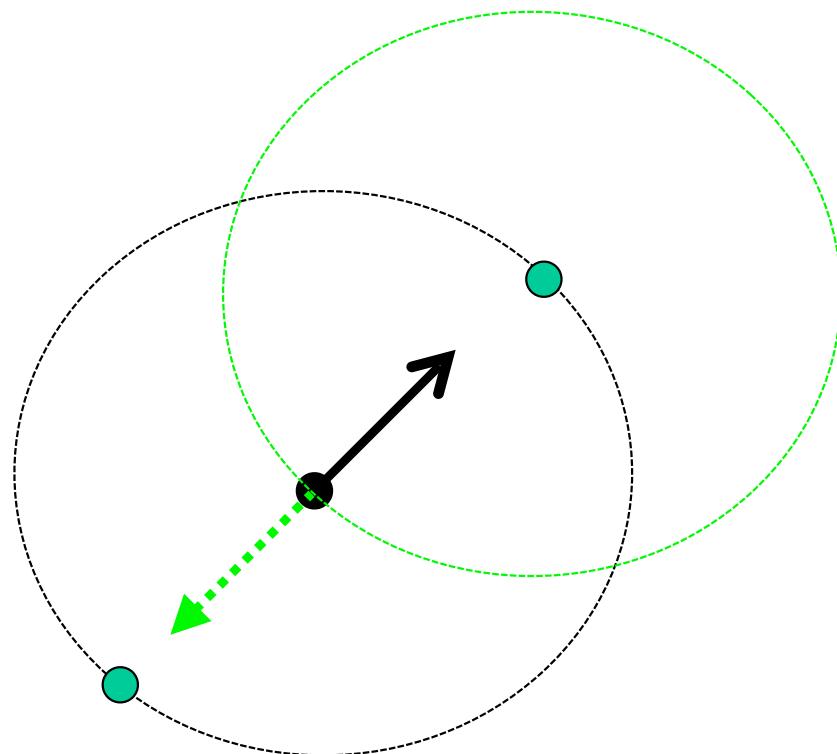
$$y = y_0 + r \sin(\theta)$$

Therefore:

$$x_0 = x - r \cos(\theta)$$

$$y_0 = y - r \sin(\theta)$$

GRADIENT ORIENTATION

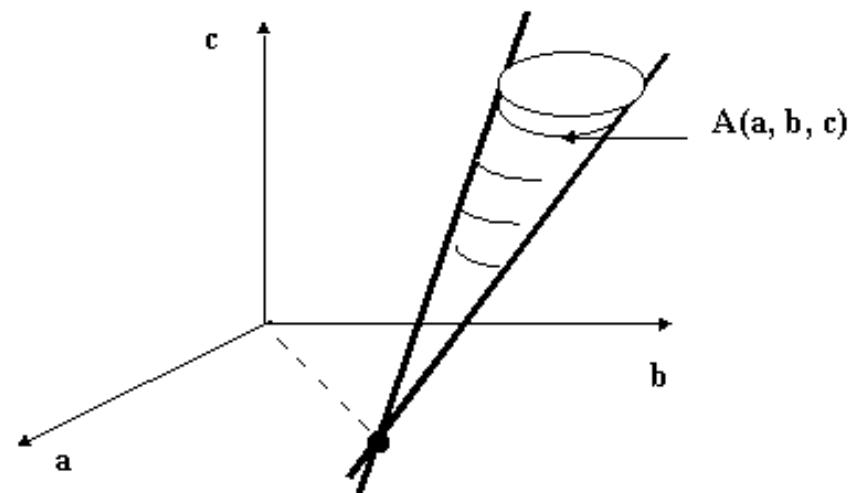


Can vote either along the entire circle or only at two points per value of the radius.

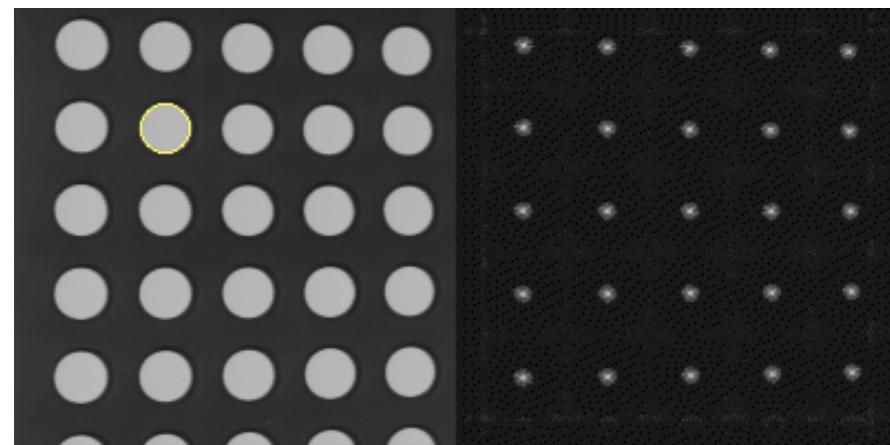
SIMPLE IMAGE



Voting scheme:



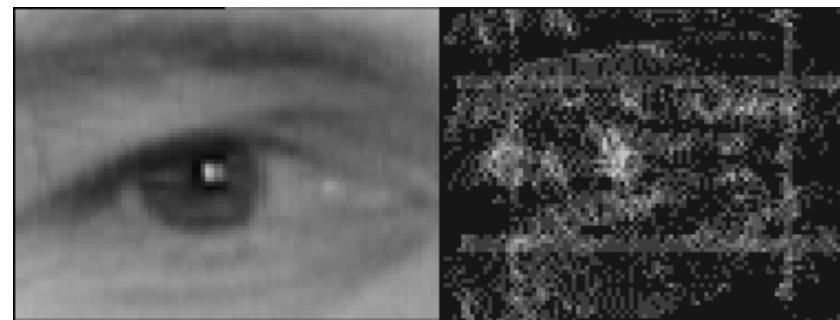
Result:



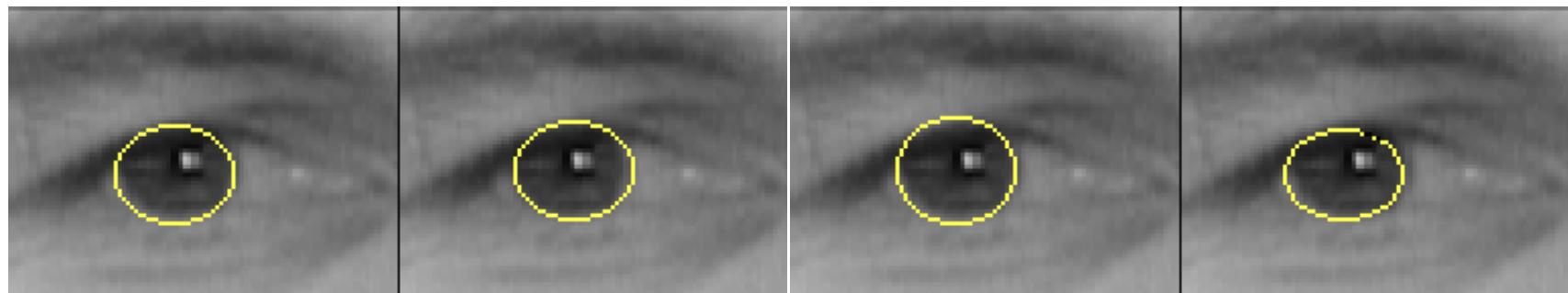
EYE IMAGES



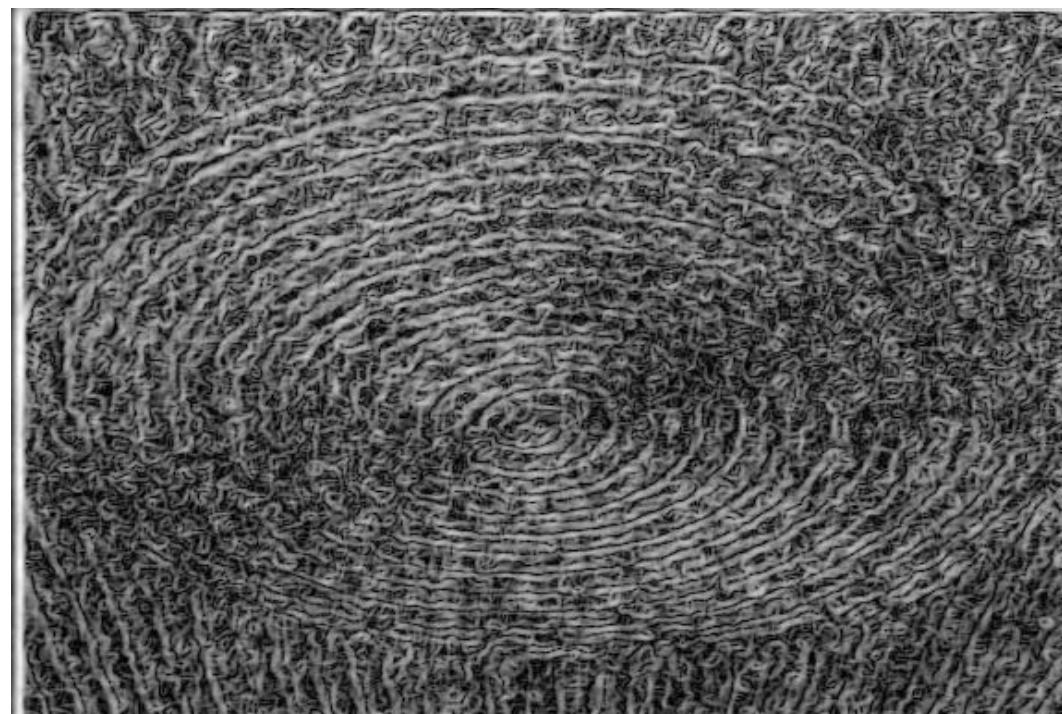
Image and accumulator:



Best four candidates:



ELLIPSES



ELLIPSE DETECTION

Ellipse of equation:

$$x = x_0 + a \cos(\theta)$$

$$y = y_0 + b \sin(\theta)$$

Therefore:

$$x_0 = x - a \cos(\theta)$$

$$y_0 = y - b \sin(\theta)$$

GRADIENT ORIENTATION



$$\frac{dx}{d\theta} = -a \sin(\theta)$$

$$\frac{dy}{d\theta} = b \cos(\theta)$$

$$\phi = \text{atan}\left(-\frac{dx}{d\theta}, \frac{dy}{d\theta}\right)$$

$$= \text{atan}(a \sin(\theta), b \cos(\theta))$$

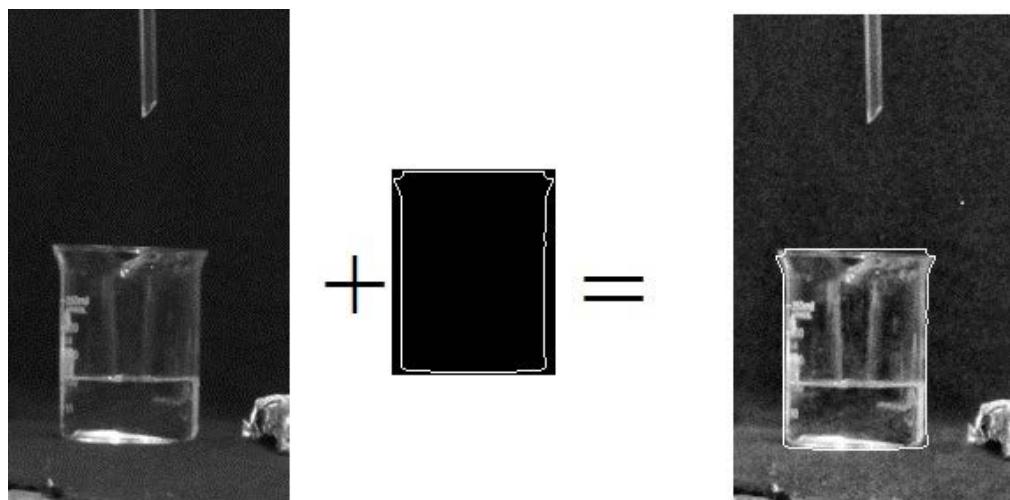
$$= \text{atan}\left(\frac{a}{b} \tan(\theta)\right)$$

$$\tan(\phi) = \frac{a}{b} \tan(\theta)$$

$$\Rightarrow \theta = \text{atan}\left(\frac{b}{a} \tan(\phi)\right)$$

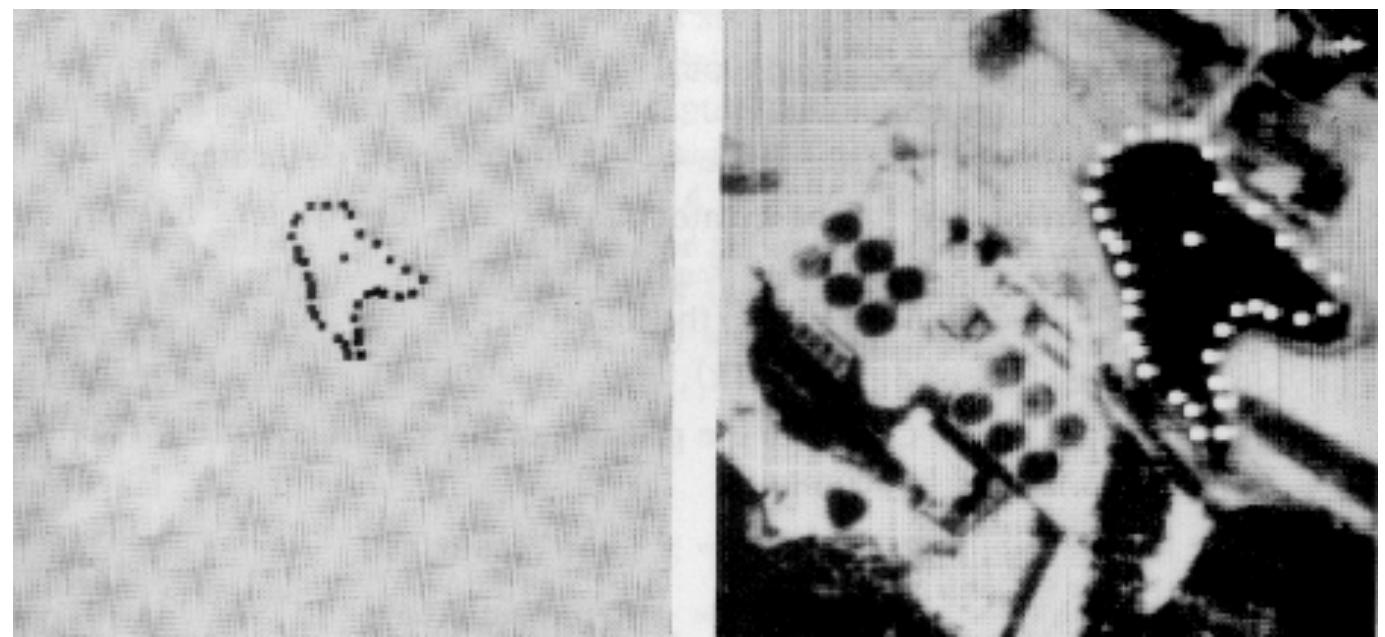
The accumulator need only be incremented for this θ .

GENERALIZED HOUGH

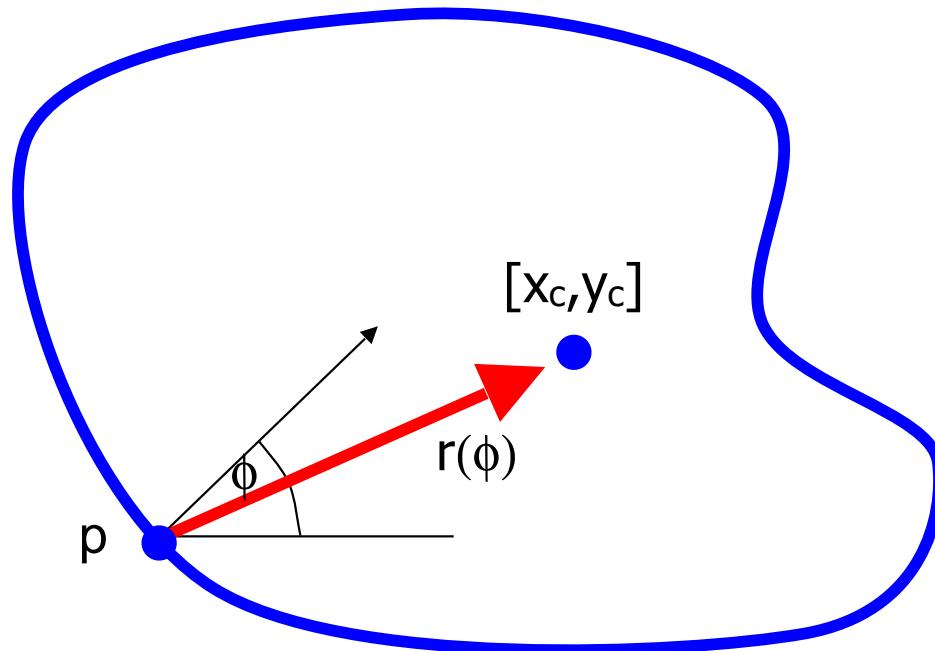


Finding a becher ...

... or a lake.



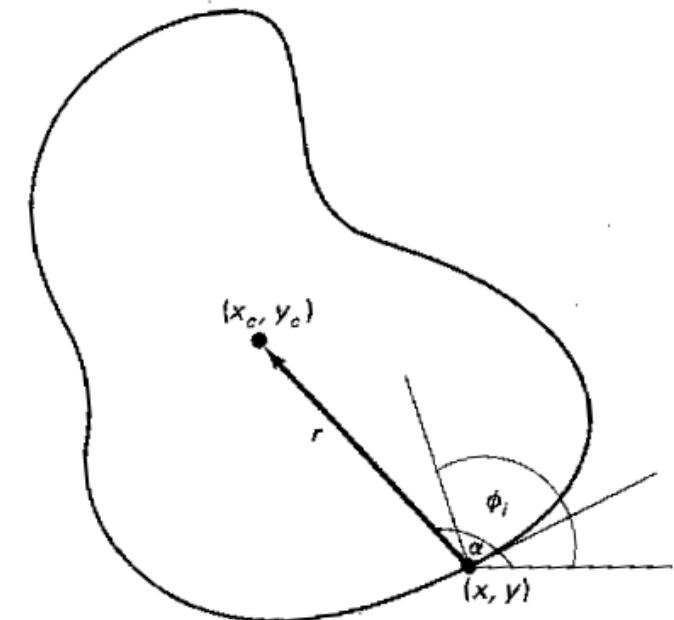
GENERALIZED HOUGH



- We want to find a shape defined by its boundary points in terms of the location of a reference point $[x_c, y_c]$.
- For every boundary point p , we can compute the displacement vector $r = [x_c, y_c] - p$ as a function of local gradient orientation ϕ .

R-TABLE

ϕ_1	$(r^1_1, \alpha^1_1), (r^1_2, \alpha^1_2), \dots, (r^1_{n1}, \alpha^1_{n1})$
ϕ_2	$(r^2_1, \alpha^2_1), (r^2_2, \alpha^2_2), \dots, (r^2_{n2}, \alpha^2_{n2})$
.	.
.	.
.	.
ϕ_m	$(r^m_1, \alpha^m_1), (r^m_2, \alpha^m_2), \dots, (r^m_{nm}, \alpha^m_{nm})$



$$x_c = x_i + r_i \cos(\alpha_i)$$

$$y_c = y_i + r_i \sin(\alpha_i)$$

Set of potential displacement vectors r, α given the boundary orientation ϕ .

--> Generalized template matching.

ALGORITHM

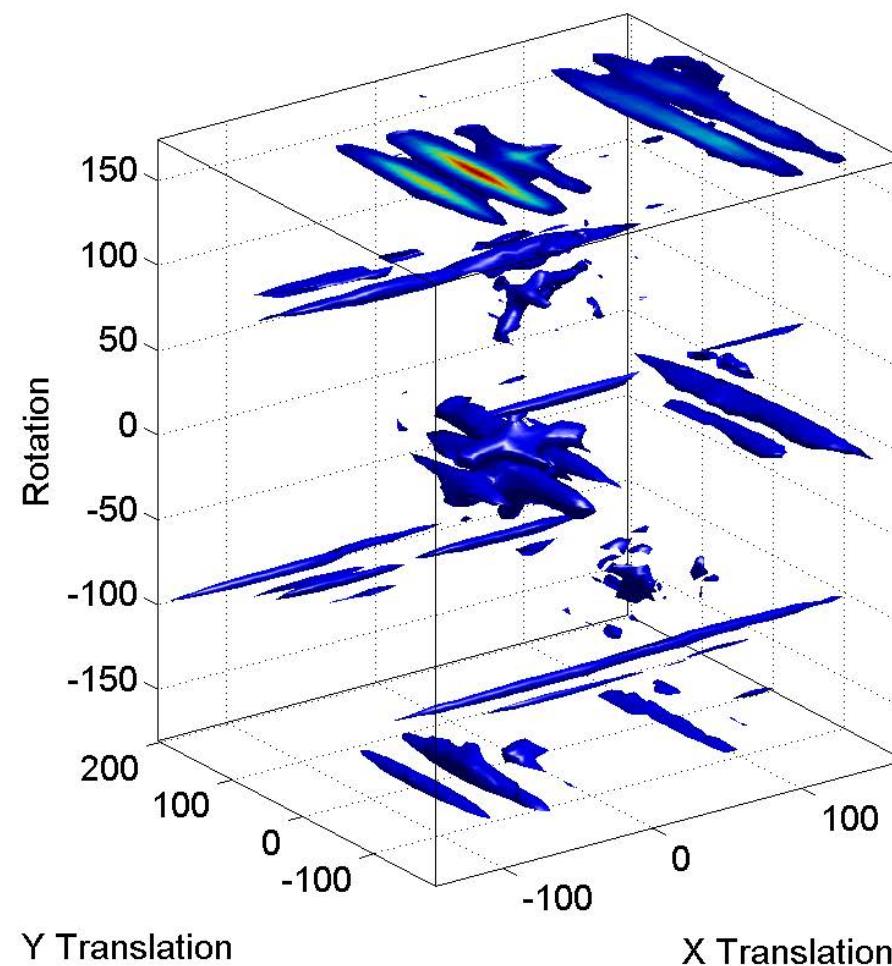


1. Make an R-table for the shape to be located.
2. Form an accumulator array of possible reference points initialized to zero.
3. For each edge point,
 - Compute the possible centers, that is, for each table entry, compute $x = x_e + r_\phi \cos(\theta(\phi))$
 $y = y_e + r_\phi \sin(\theta(\phi))$
 - Increment the accumulator array

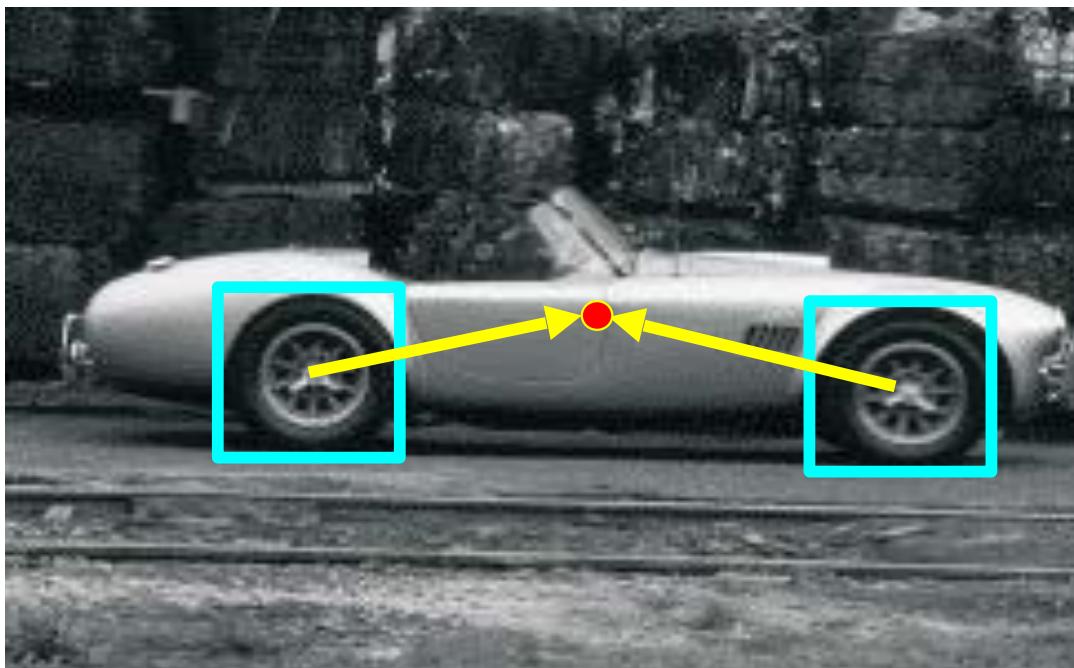
REAL-TIME HOUGH



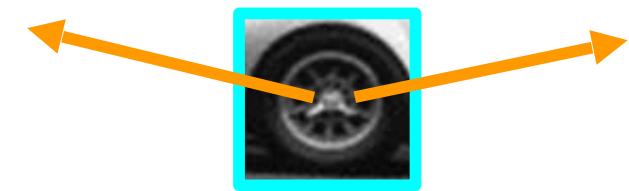
ACCUMULATOR



FROM DELINEATION TO DETECTION



Training image



Visual codeword with
displacement vectors

- Instead of indexing displacements by gradient orientation, index by “visual codeword”.

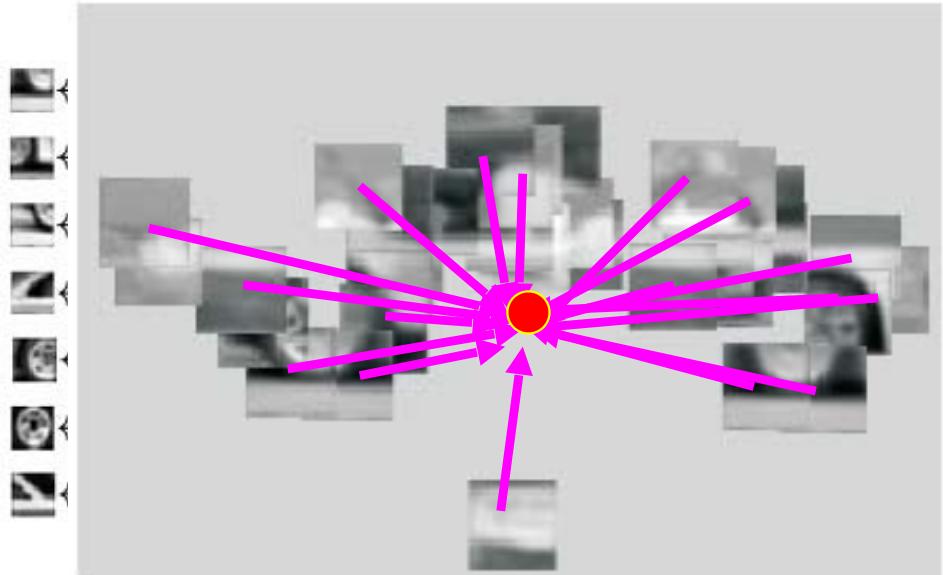
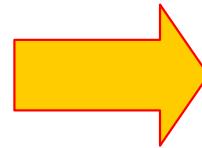
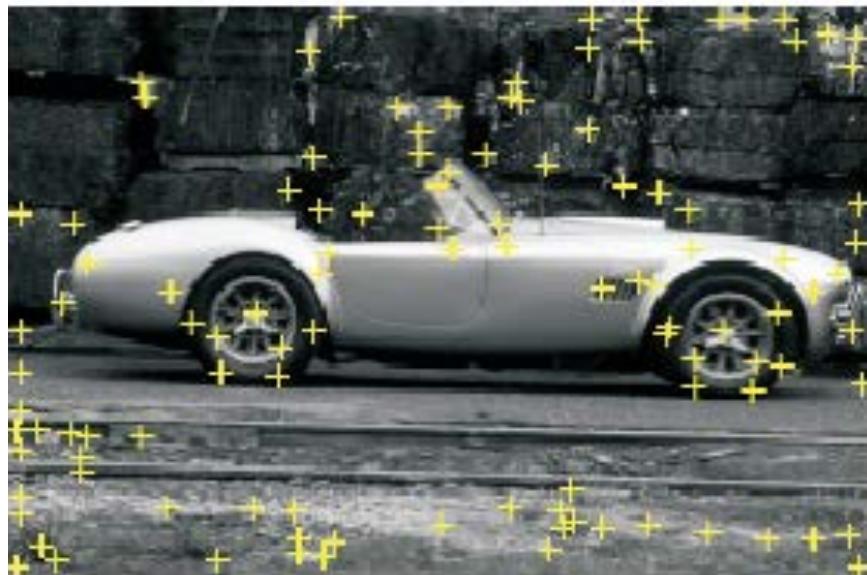
FROM DELINEATION TO DETECTION



Test image

- Instead of indexing displacements by gradient orientation, index by “visual codeword”.

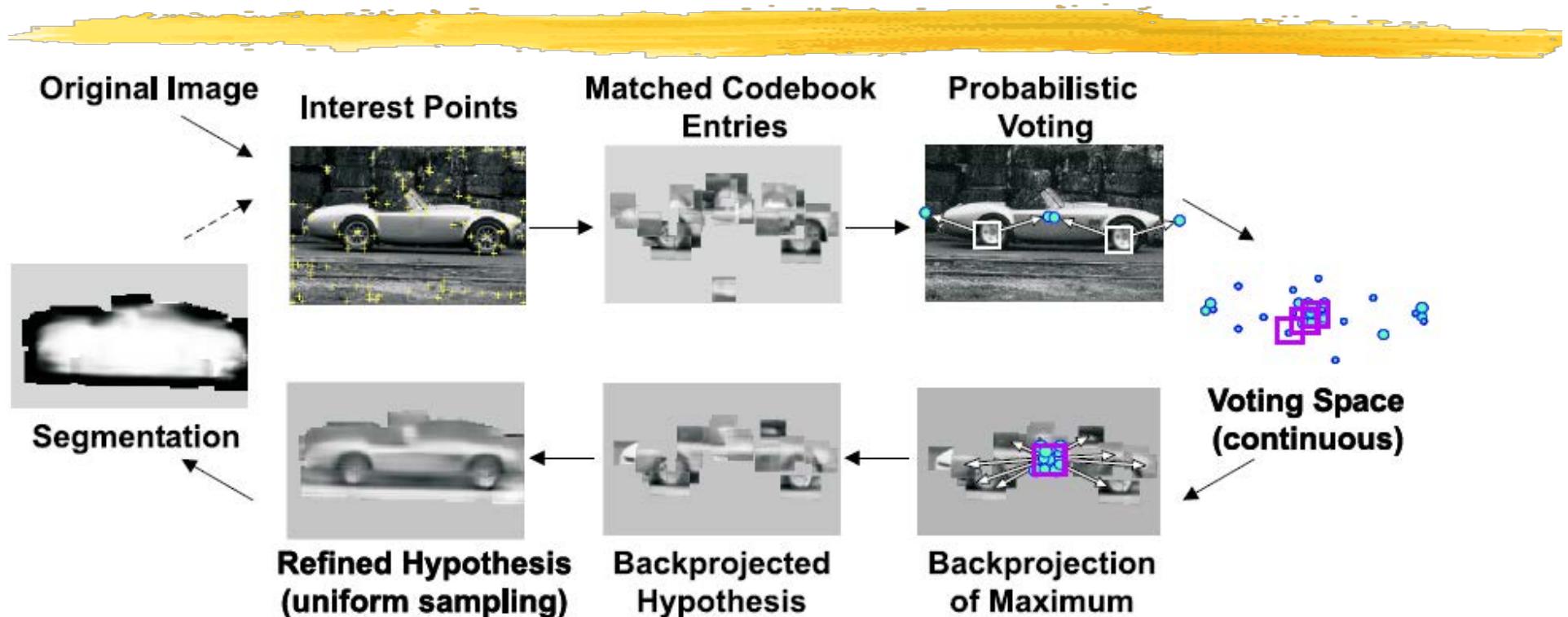
TRAINING



1. Build codebook of patches around extracted interest points using clustering.
2. Map the patch around each interest point to closest codebook entry.
3. For each codebook entry, store all positions it was found, relative to object center.

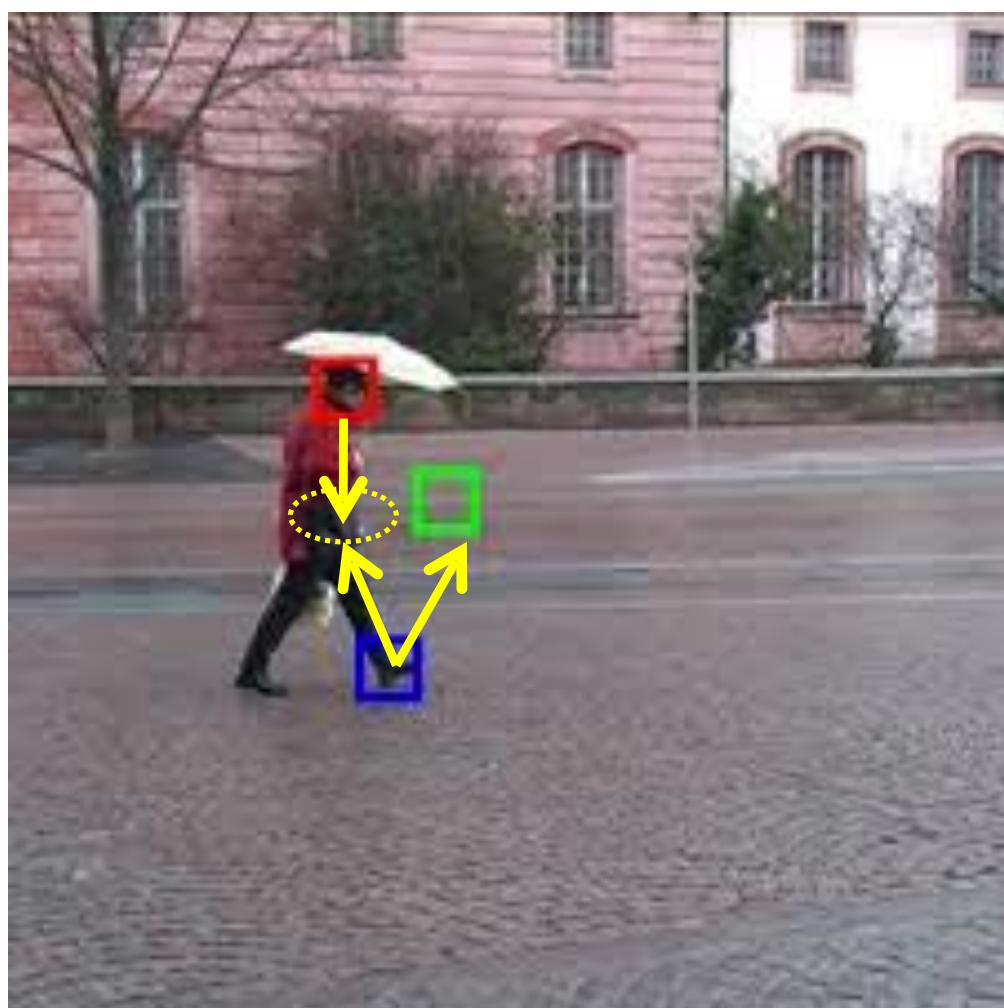
--> Build an R table.

TESTING



1. Given test image, extract patches, match to codebook entry.
2. Cast votes for possible positions of object center.
3. Search for maxima in voting space.
4. Extract weighted segmentation mask based on stored masks for the codebook occurrences.

PEDESTRIAN DETECTION



OCCLUSION HANDLING



LIMITATIONS



Computational cost grows exponentially with the number of model parameters:

- Only works for objects whose shape can be defined by a small number of parameters.
- Approach is robust but lacks flexibility.

TECHNIQUES



Semi-Automated Techniques:

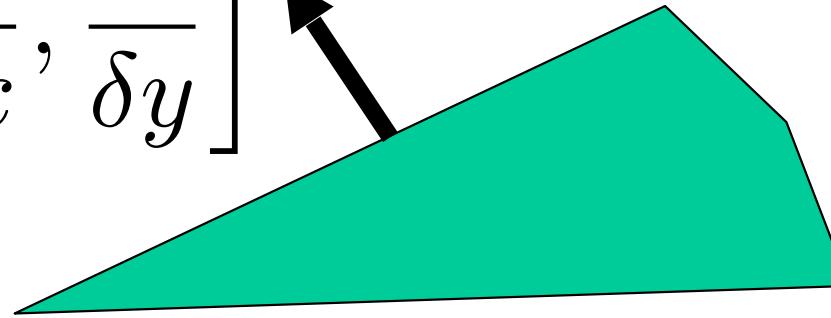
- Dynamic programming
- Deformable Models

Fully Automated Techniques:

- Hough transform
- Graph Based Approaches

MAGNITUDE AND ORIENTATION

$$\left[\frac{\delta I}{\delta x}, \frac{\delta I}{\delta y} \right]$$

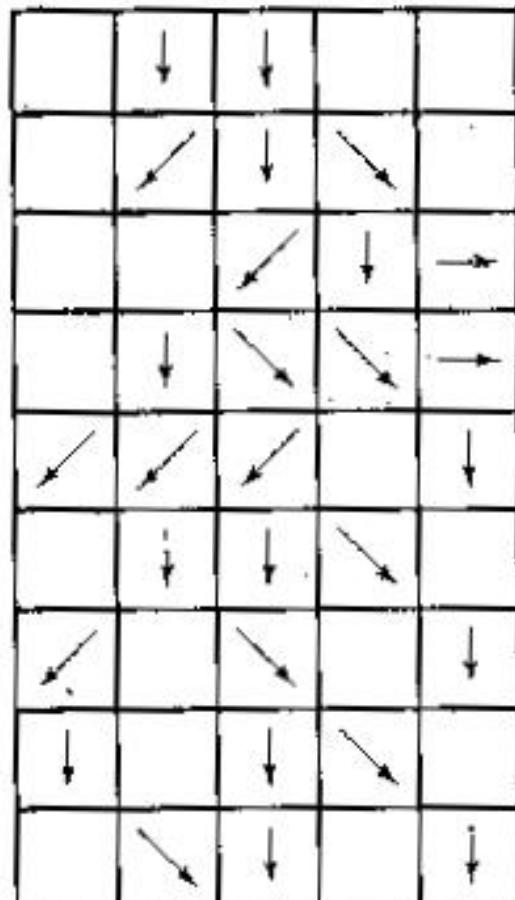


$$\text{Contrast: } G = \sqrt{\frac{\delta I^2}{\delta x} + \frac{\delta I^2}{\delta y}}$$

$$\text{Orientation: } \Theta = \arctan\left(\frac{\delta I}{\delta y}, \frac{\delta I}{\delta x}\right)$$

MINIMUM SPANNING TREE

Image modeled as a graph:



41



-> Generate minimal distance graph $O(N \log(N))$ algorithm)

DELINEATION 1998



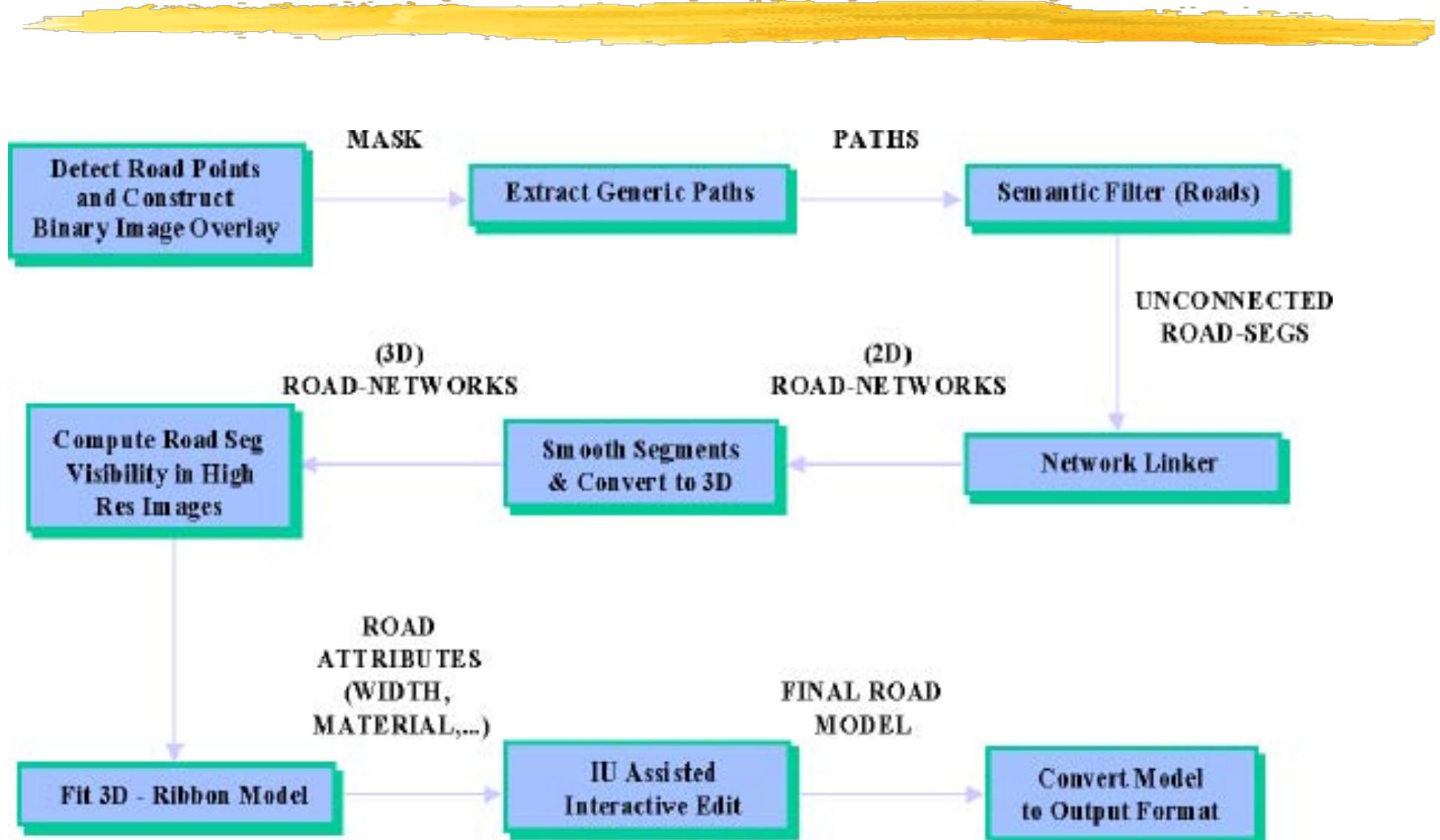
Detect road centerlines

Find generic paths

Apply semantic filter

Find road widths

FROM IMAGE TO ROADS

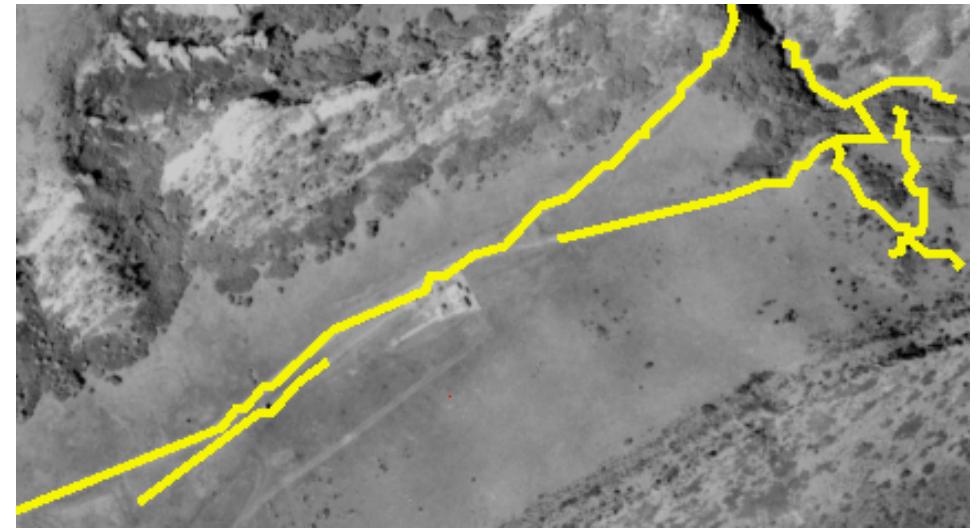


ROAD DELINEATION

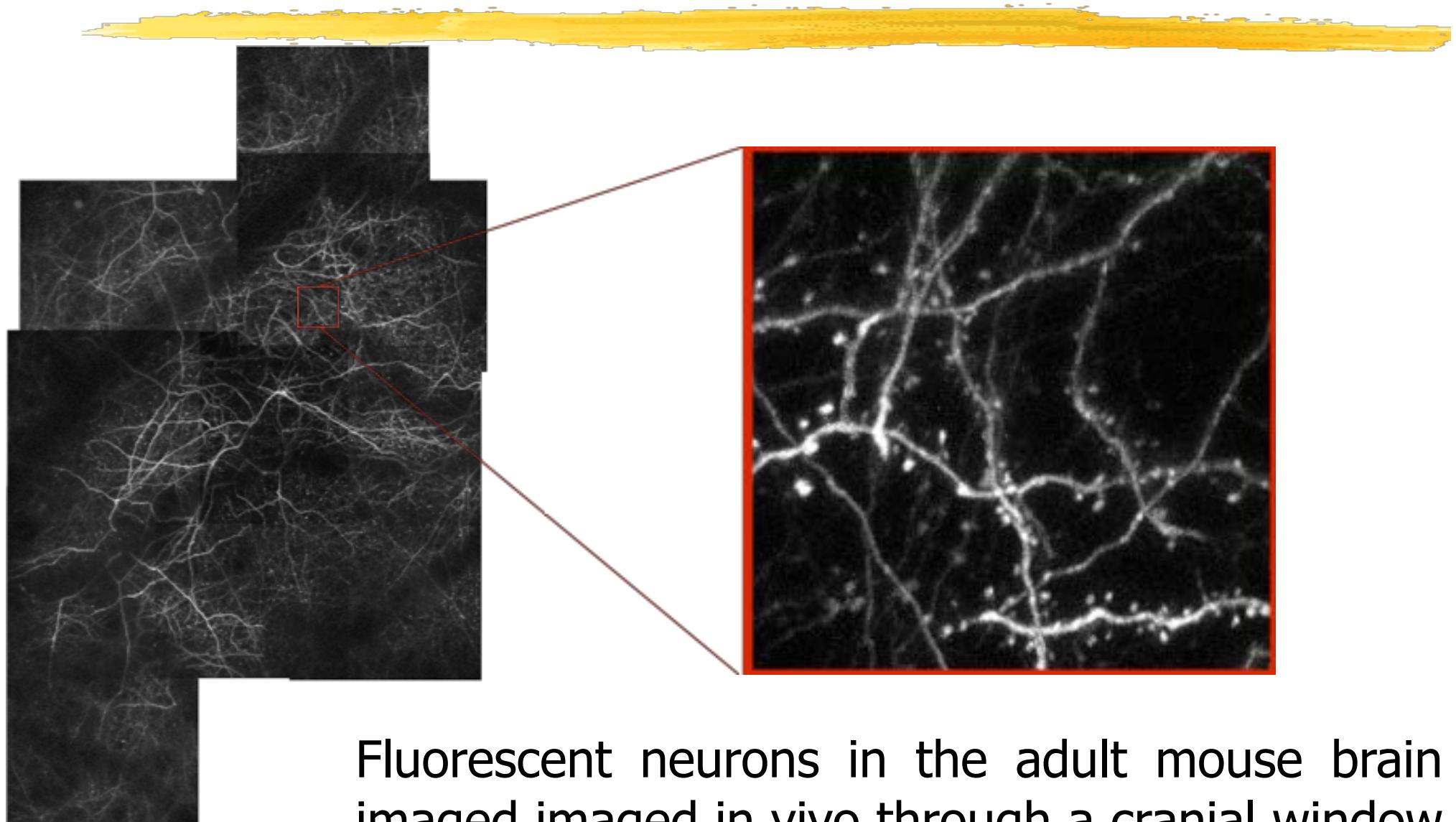


Fischler & Heller, 1998.

ROAD EDITING

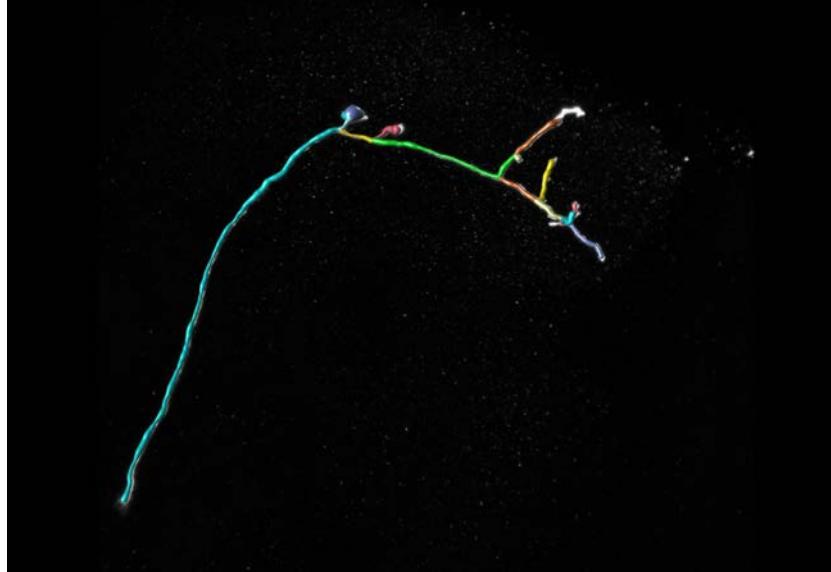
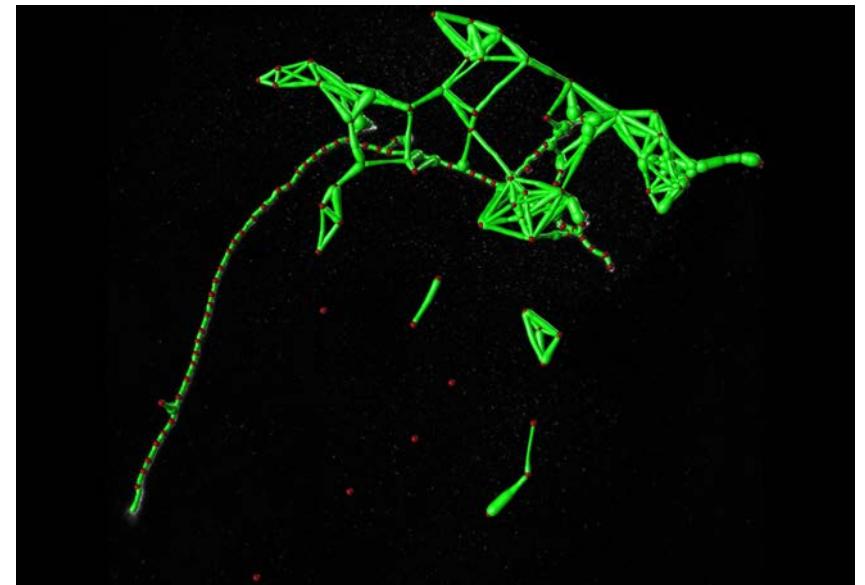
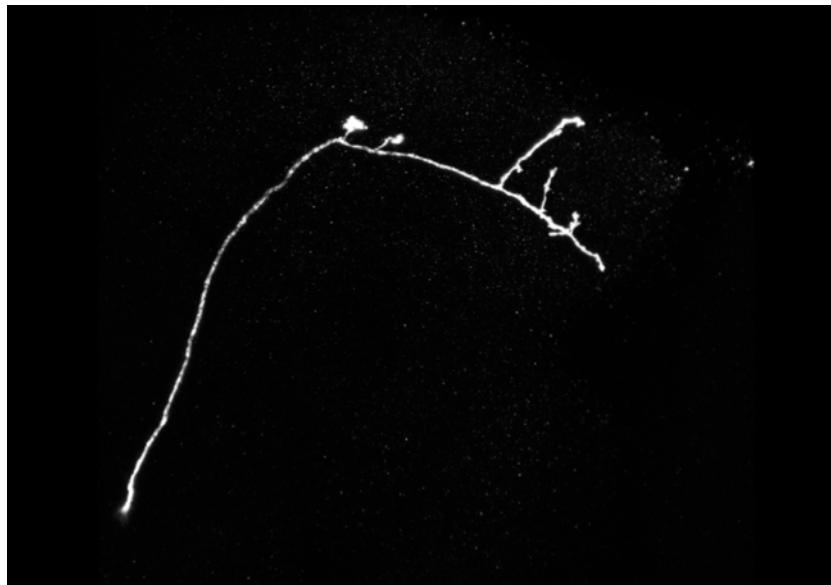


DENDRITES AND AXONS



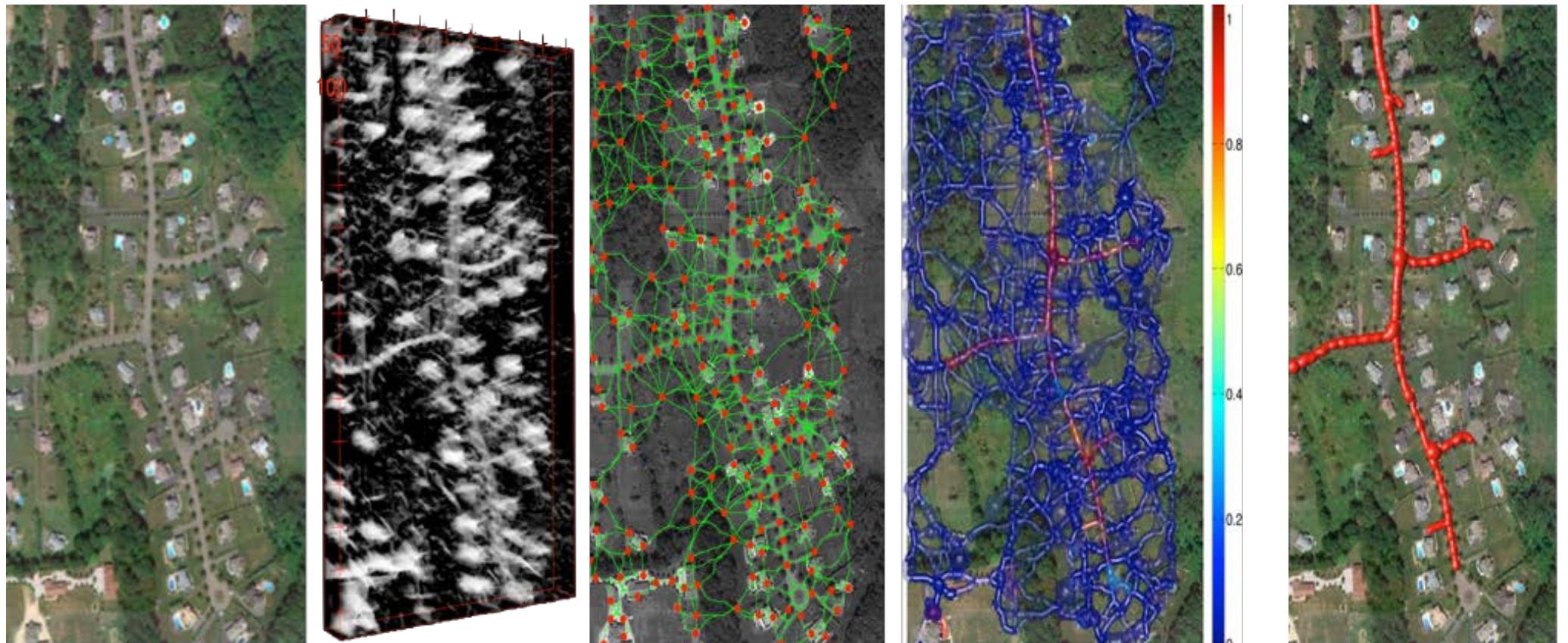
Fluorescent neurons in the adult mouse brain imaged *in vivo* through a cranial window using a 2-photon microscope.

DELINEATION IN 2011



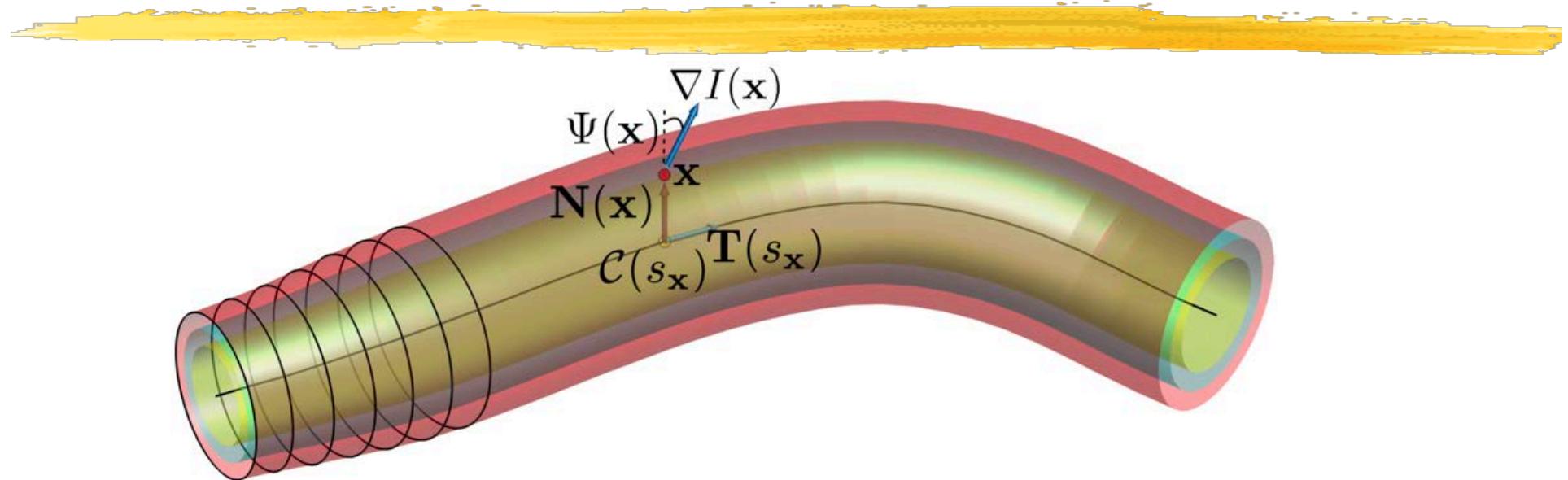
$$\begin{aligned}\mathbf{t}^* &= \underset{\mathbf{t} \in \mathcal{T}(G)}{\operatorname{argmax}} P(\mathbf{T} = \mathbf{t} | I) , \\ &= \underset{\mathbf{t} \in \mathcal{T}(G)}{\operatorname{argmax}} P(I | \mathbf{T} = \mathbf{t}) P(\mathbf{T} = \mathbf{t}) , \\ &= \underset{\mathbf{t} \in \mathcal{T}(G)}{\operatorname{argmin}} \sum_{e_{ij} \in G} \left[-\log \frac{P(T_{ij} = 1 | I_{ij})}{P(T_{ij} = 0 | I_{ij})} \right] t_{ij} \\ &\quad + \sum_{e_{ij}, e_{jk} \in G} f(e_{ij}, e_{jk}) t_{ij} t_{jk} .\end{aligned}$$

.. AND ROADS



→ Machine plays a crucial role to ensure that the **same algorithm works in different situations.**

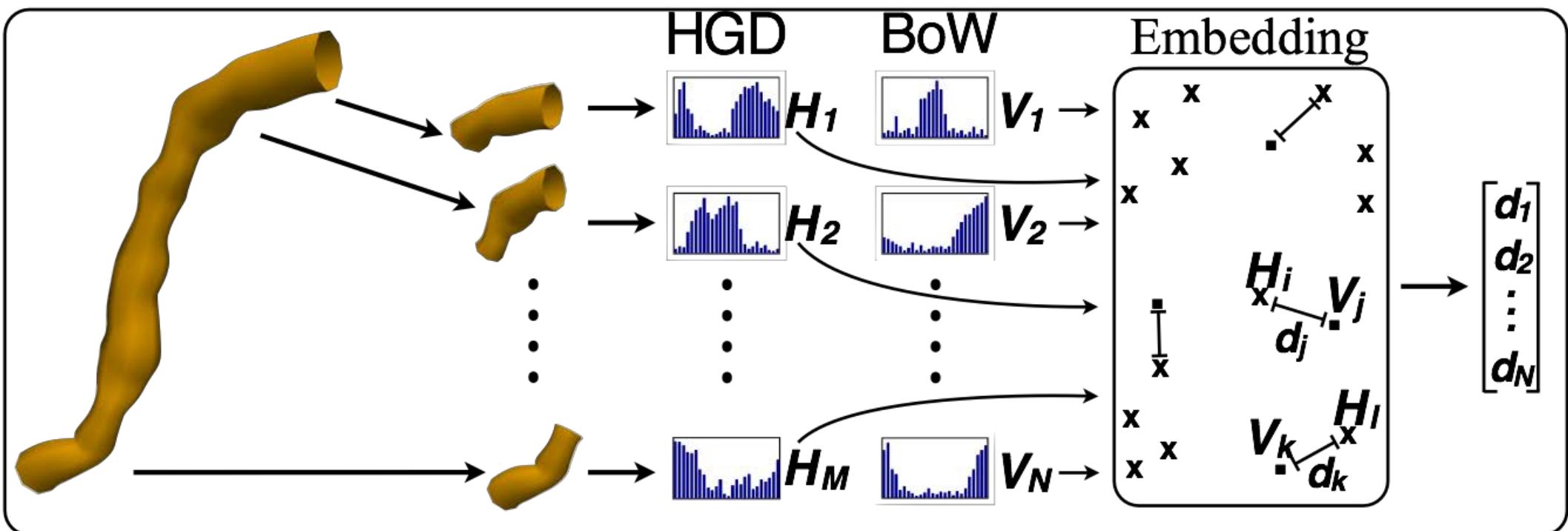
HISTOGRAM OF GRADIENT DEVIATIONS



$$\Psi(\mathbf{x}) = \begin{cases} \text{angle}(\nabla I(\mathbf{x}), \mathbf{N}(\mathbf{x})) , & \text{if } \|\mathbf{x} - \mathcal{C}(s_{\mathbf{x}})\| > \varepsilon \\ \text{angle}(\nabla I(\mathbf{x}), \Pi(\mathbf{x})) , & \text{otherwise,} \end{cases}$$

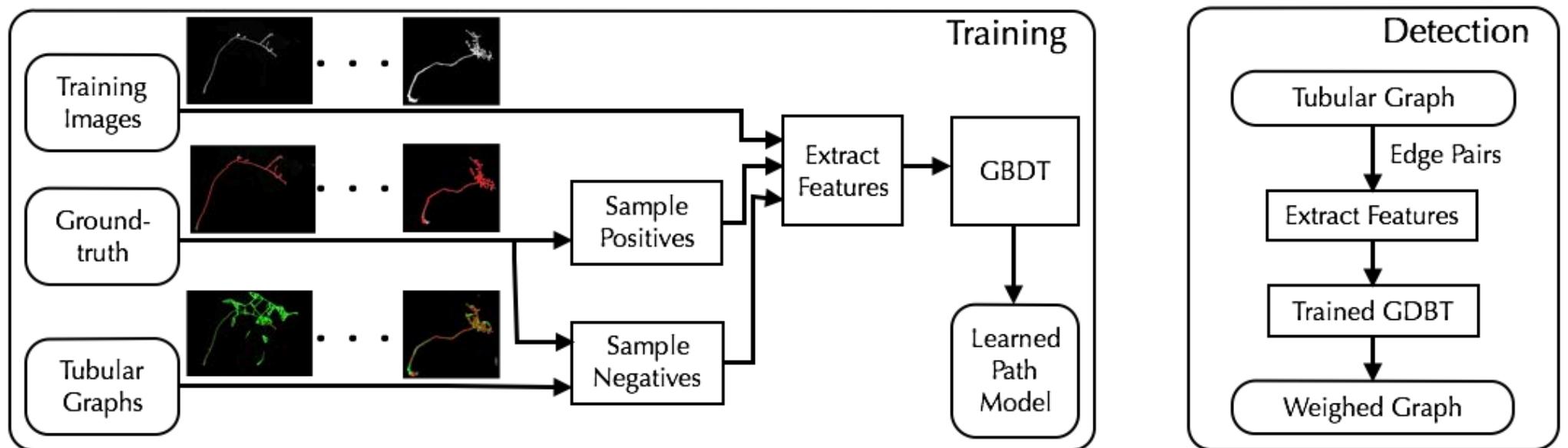
→ One histogram per radius interval plus four geometry features (curvature, tortuosity, . . .)

EMBEDDING

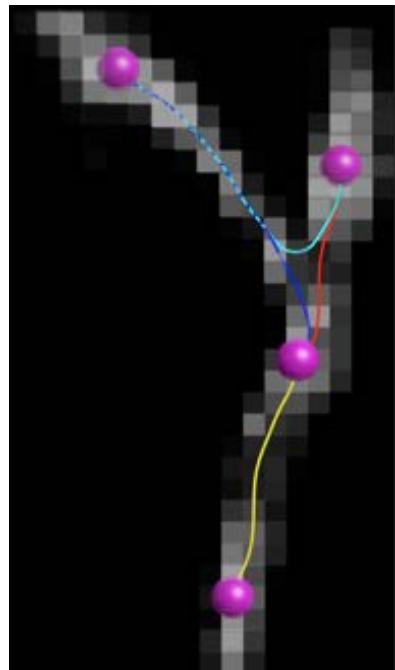


→ Same length feature vectors whatever the actual length of the path.

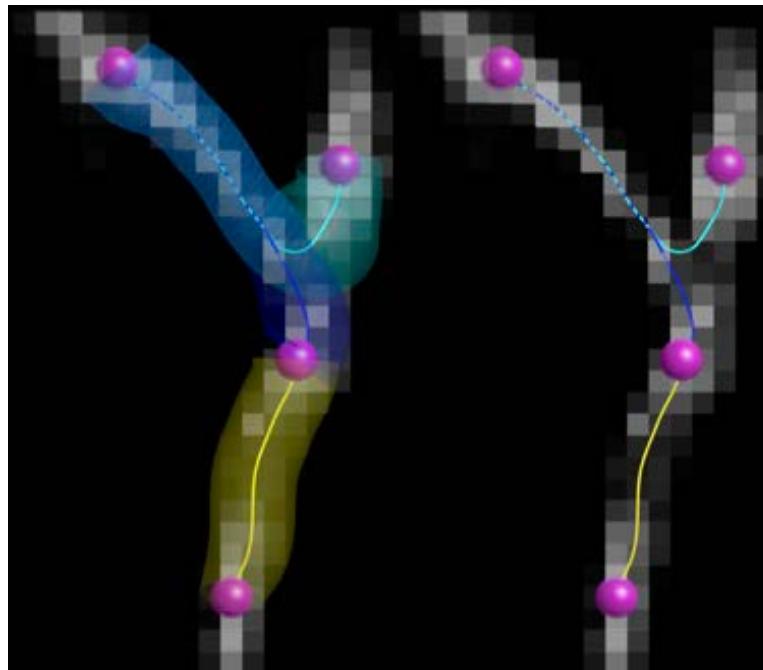
PATH CLASSIFICATION



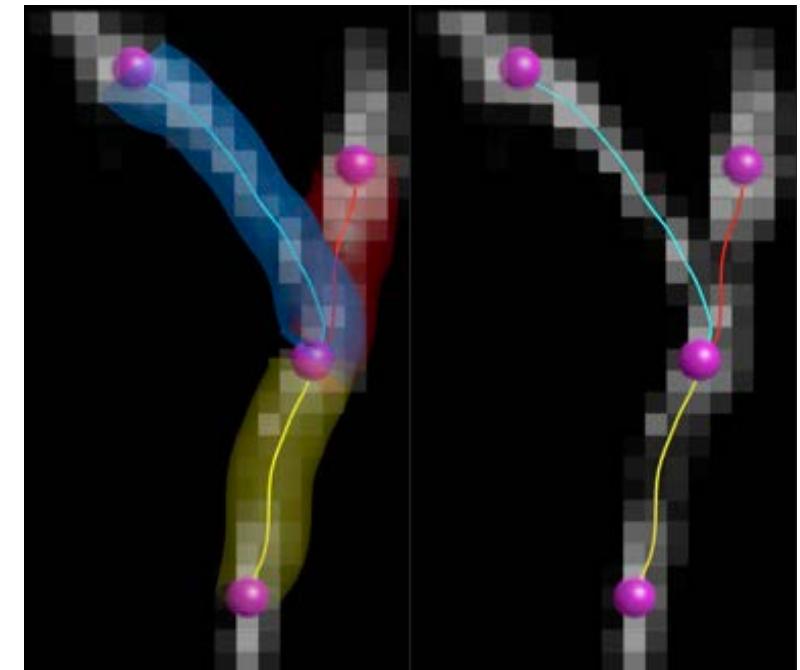
FINDING THE BEST TREE



Tubularity graph



Without edge pair term



With edge pair term

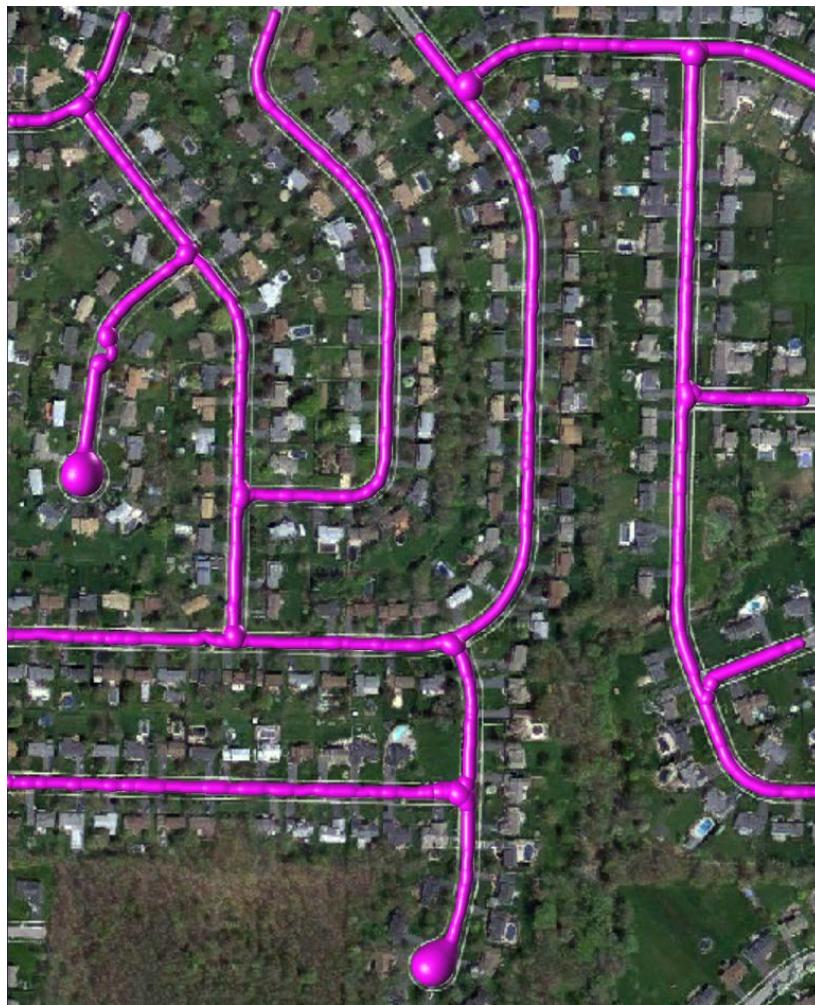
$$\begin{aligned} \mathbf{t}^* &= \underset{\mathbf{t} \in \mathcal{T}(G)}{\operatorname{argmax}} P(\mathbf{T} = \mathbf{t} | I) , \\ &= \underset{\mathbf{t} \in \mathcal{T}(G)}{\operatorname{argmax}} P(I | \mathbf{T} = \mathbf{t}) P(\mathbf{T} = \mathbf{t}) , \\ &= \underset{\mathbf{t} \in \mathcal{T}(G)}{\operatorname{argmin}} \sum_{e_{ij} \in G} c_{ij}^d t_{ij} \end{aligned}$$

QMIP FORMULATION

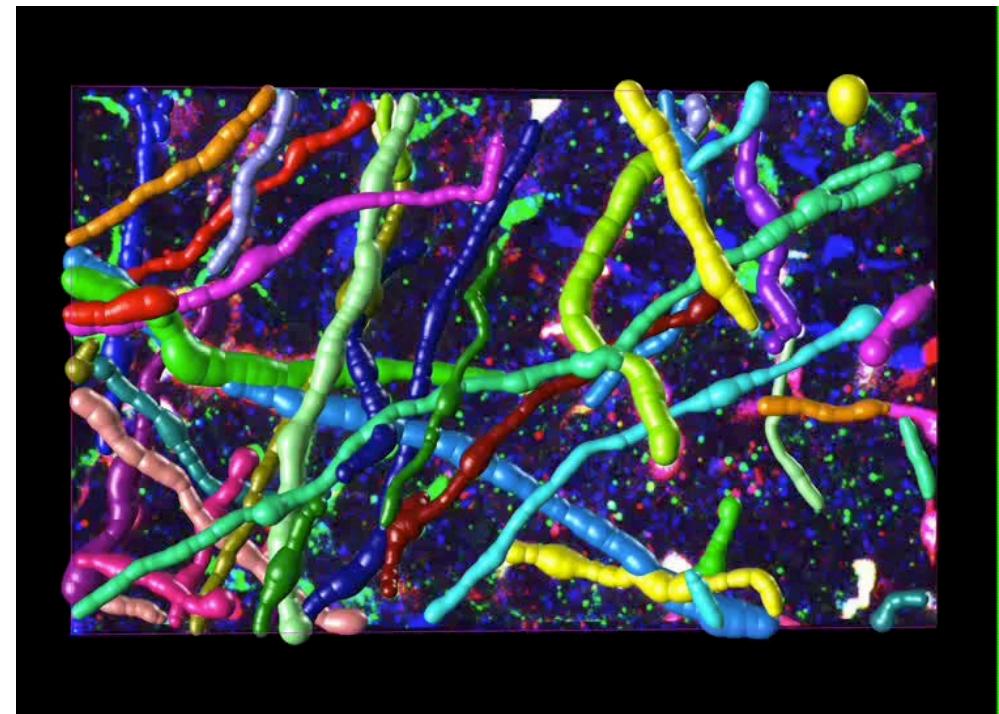
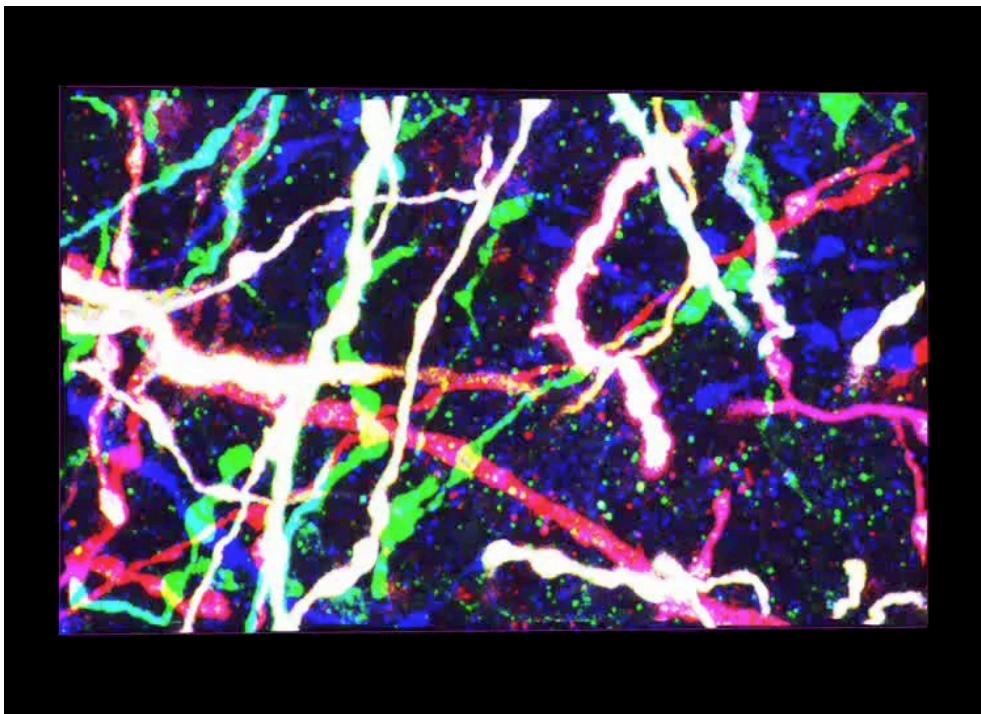
$$\begin{aligned} \min \quad & \sum_{e_{ij} \in E, e_{jk} \in E} c_{ijk} t_{ij} t_{jk} \\ \text{s.t.} \quad & \sum_{v_j \in V \setminus \{v_r\}} y_{rj}^l \leq 1, \quad \forall v_l \in V \setminus \{v_r\}, \\ & \sum_{v_j \in V \setminus \{v_k\}} y_{jk}^l \leq 1, \quad \forall v_l \in V \setminus \{v_r\}, \\ & \sum_{v_j \in V \setminus \{v_i, v_r\}} y_{ij}^l - \sum_{v_j \in V \setminus \{v_i, v_l\}} y_{ji}^l = 1, \quad \begin{array}{l} \forall v_k \in V \setminus \{v_r\}, \\ \forall v_i \in V \setminus \{v_r, v_k\}, \end{array} \\ & y_{ij}^l \leq t_{ij}, \quad \forall e_{ij} \in E, v_l \in V \setminus \{v_r, v_i, v_j\}, \\ & y_{il}^l = t_{il}, \quad \forall e_{il} \in E, \\ & y_{ij}^l \geq 0, \quad \forall e_{ij} \in E, v_l \in V \setminus \{v_r, v_i\}, \\ & t_{ij} \in \{0, 1\}, \quad \forall e_{ij} \in E. \end{aligned}$$

given the root note v_r .

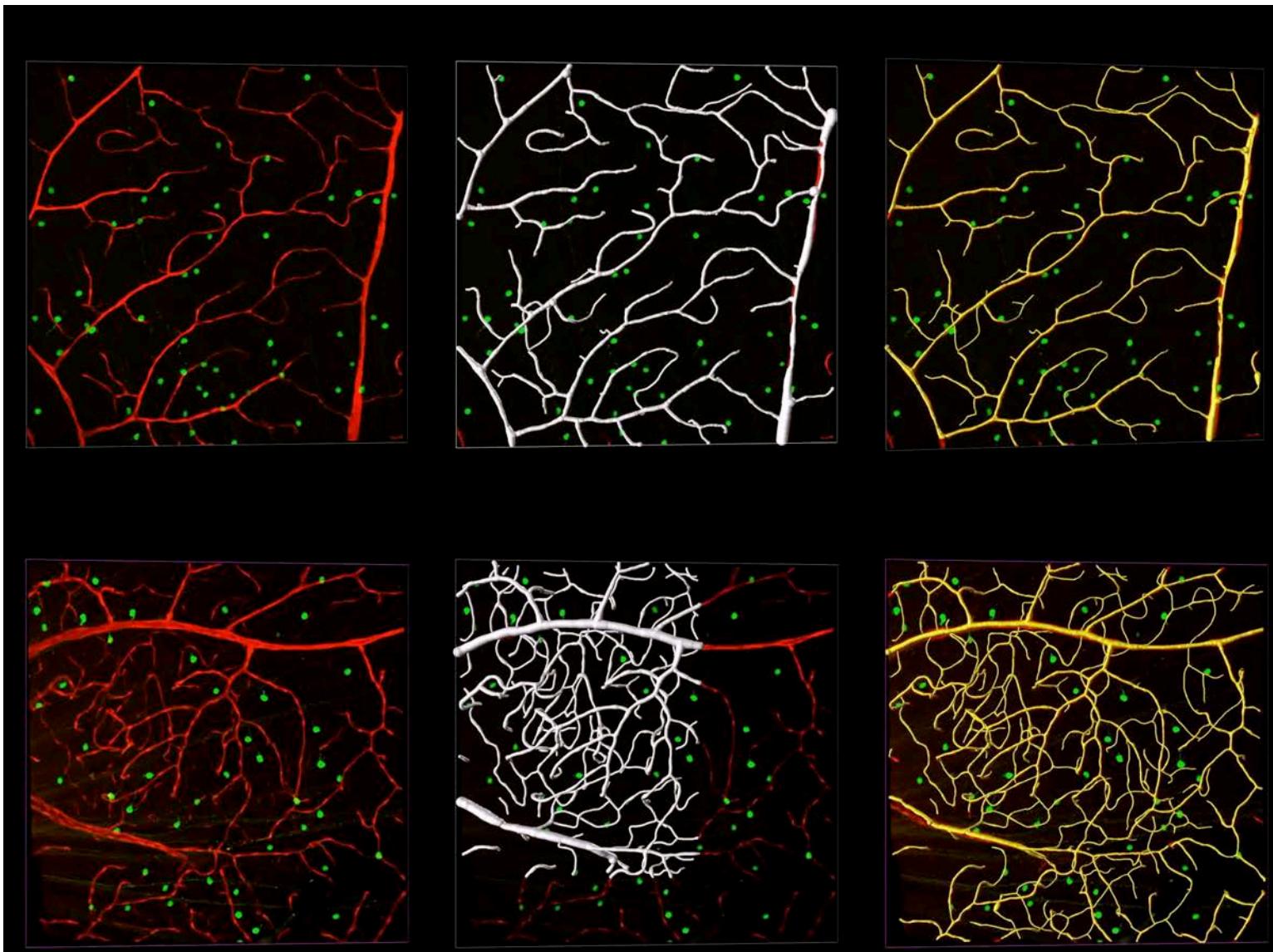
ROADS



BRAINBOW IMAGES



BLOOD VESSELS



DEEP TSUNAMI

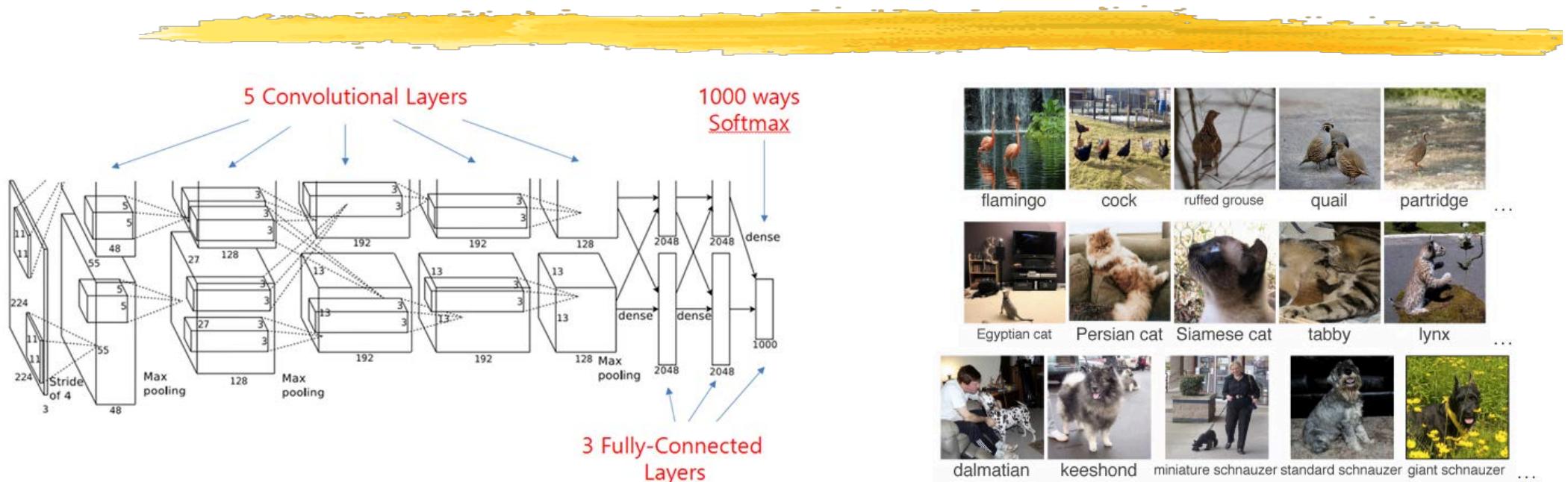
The New York Times

*Turing Award Won by Three
Pioneers in Artificial Intelligence*



From left, Yann LeCun, Geoffrey Hinton and Yoshua Bengio. The researchers worked on key developments for neural networks, which are reshaping how computer systems are built.

AlexNet (2012)



Task: Image classification

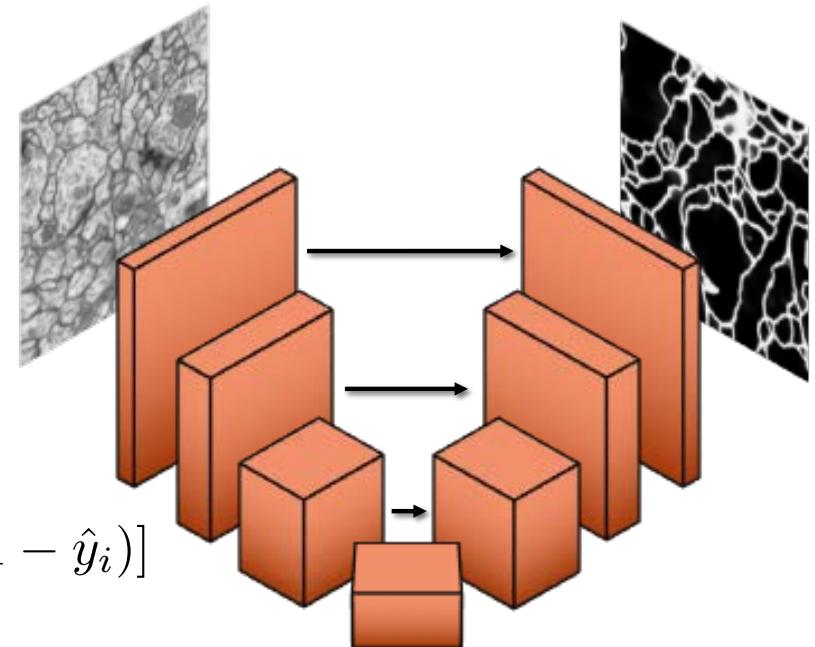
Training images: Large Scale Visual Recognition Challenge 2010

Training time: 2 weeks on 2 GPUs

Major Breakthrough: Training large networks has now been shown to be practical!!

DELINEATION 2018

Train Encoder-decoder U-Net architecture using binary cross-entropy



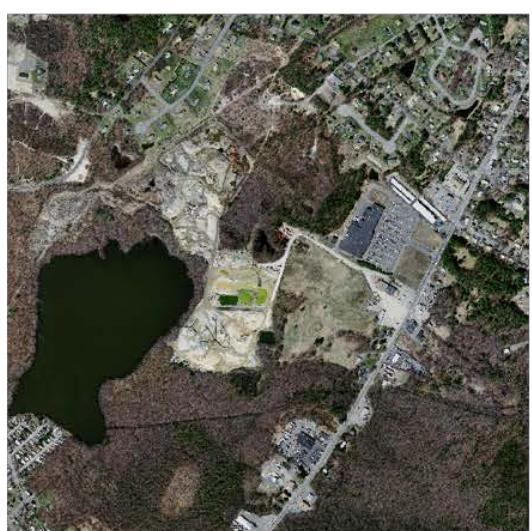
Minimize

$$L_{bce}(\mathbf{x}, \mathbf{y}; \mathbf{w}) = -\frac{1}{i} \sum_1^P [y_n \log(\hat{y}_i) + (1 - y_i) \log(1 - \hat{y}_i)]$$

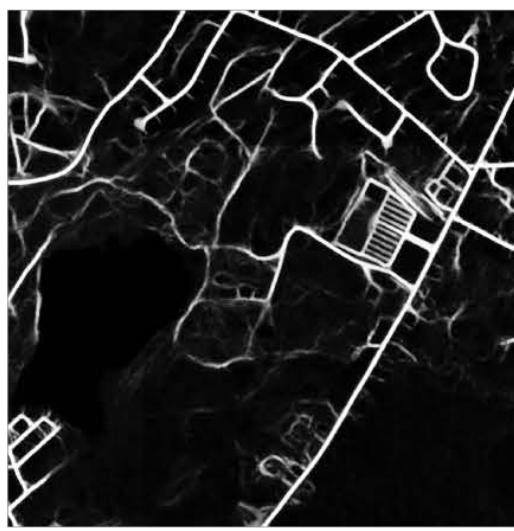
where

- $\hat{\mathbf{y}} = f_{\mathbf{w}}(\mathbf{x})$,
- \mathbf{x} in an input image,
- \mathbf{y} the corresponding ground truth.

NETWORK OUTPUT



Image

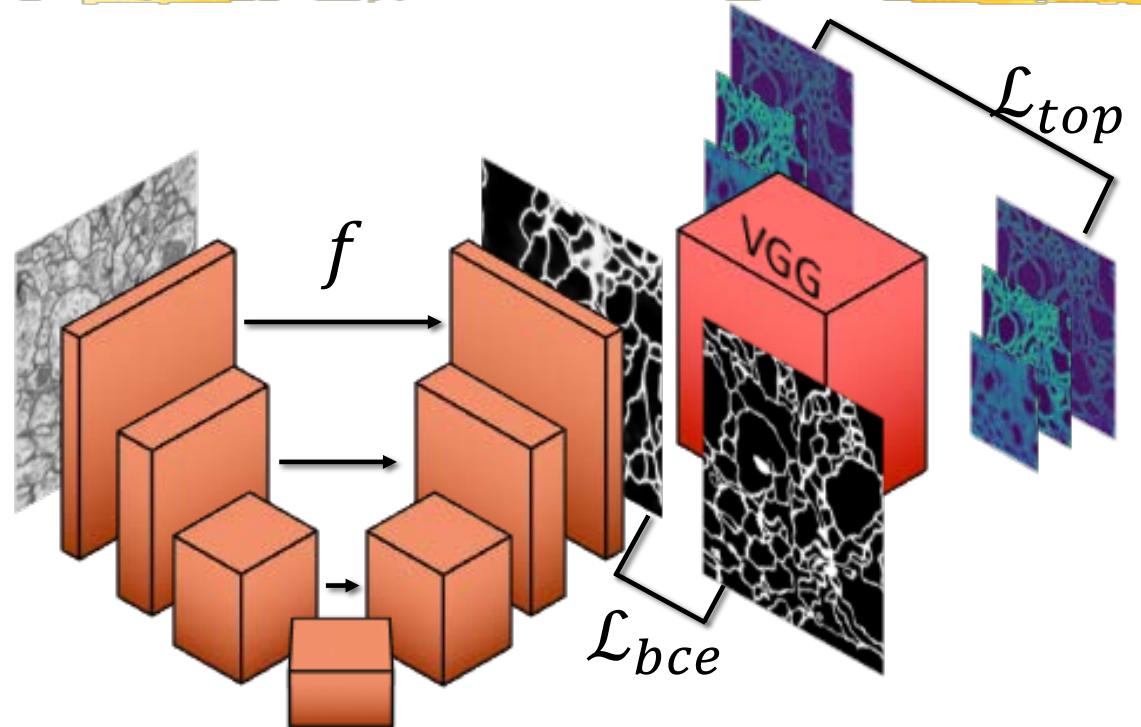


BCE Loss



Ground truth

ACCOUNTING FOR TOPOLOGY



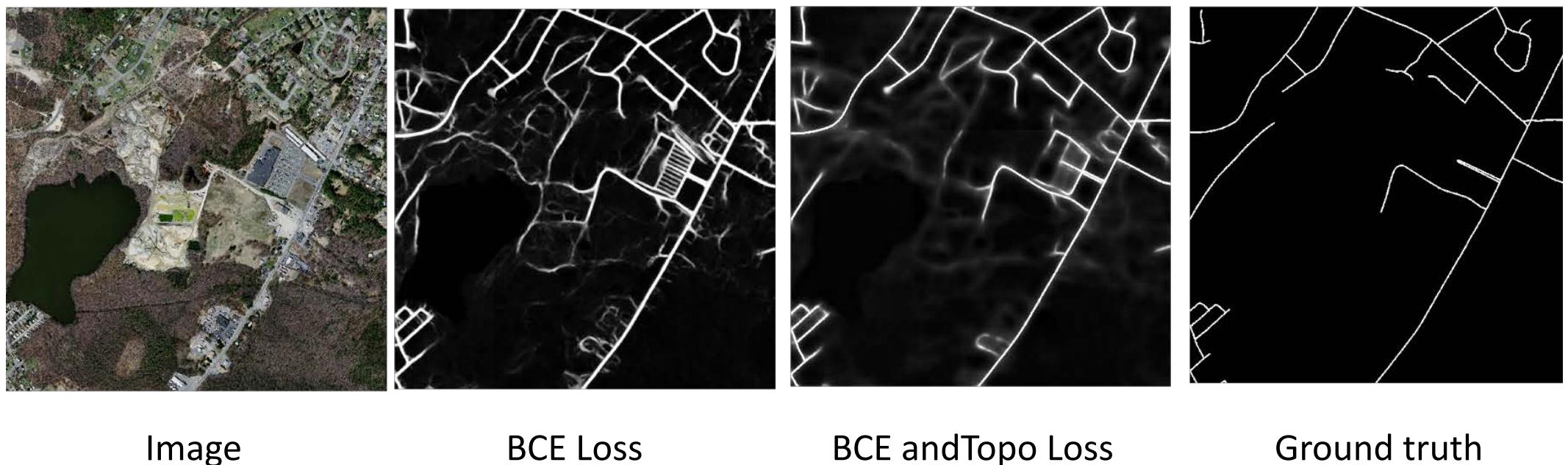
Minimize

$$L(\mathbf{x}, \mathbf{y}; \mathbf{w}) = L_{bce}(\mathbf{x}, \mathbf{y}; \mathbf{w}) + L_{top}(\mathbf{x}, \mathbf{y}; \mathbf{w}) ,$$

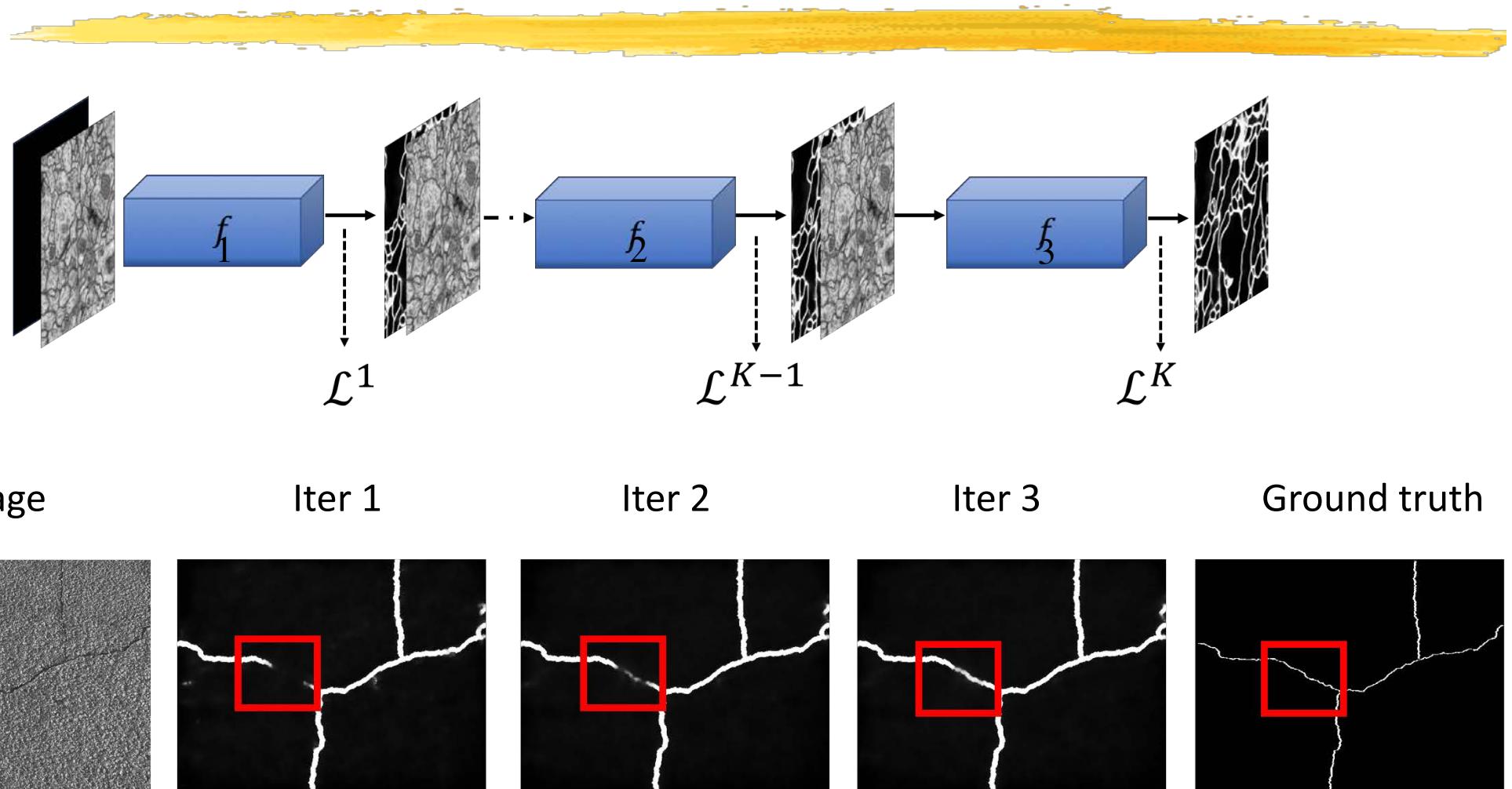
$$L_{top}(\mathbf{x}, \mathbf{y}; \mathbf{w}) \propto \sum_{n=1}^N \sum_{m=1}^{M_n} \|l_n^m(\mathbf{y}) - l_n^m(f(\mathbf{x}, \mathbf{w}))\|_2^2 ,$$

L_{top} : Sum of square differences between feature maps in a pretrained VGG.

ACCOUNTING FOR TOPOLOGY



ITERATIVE REFINEMENT



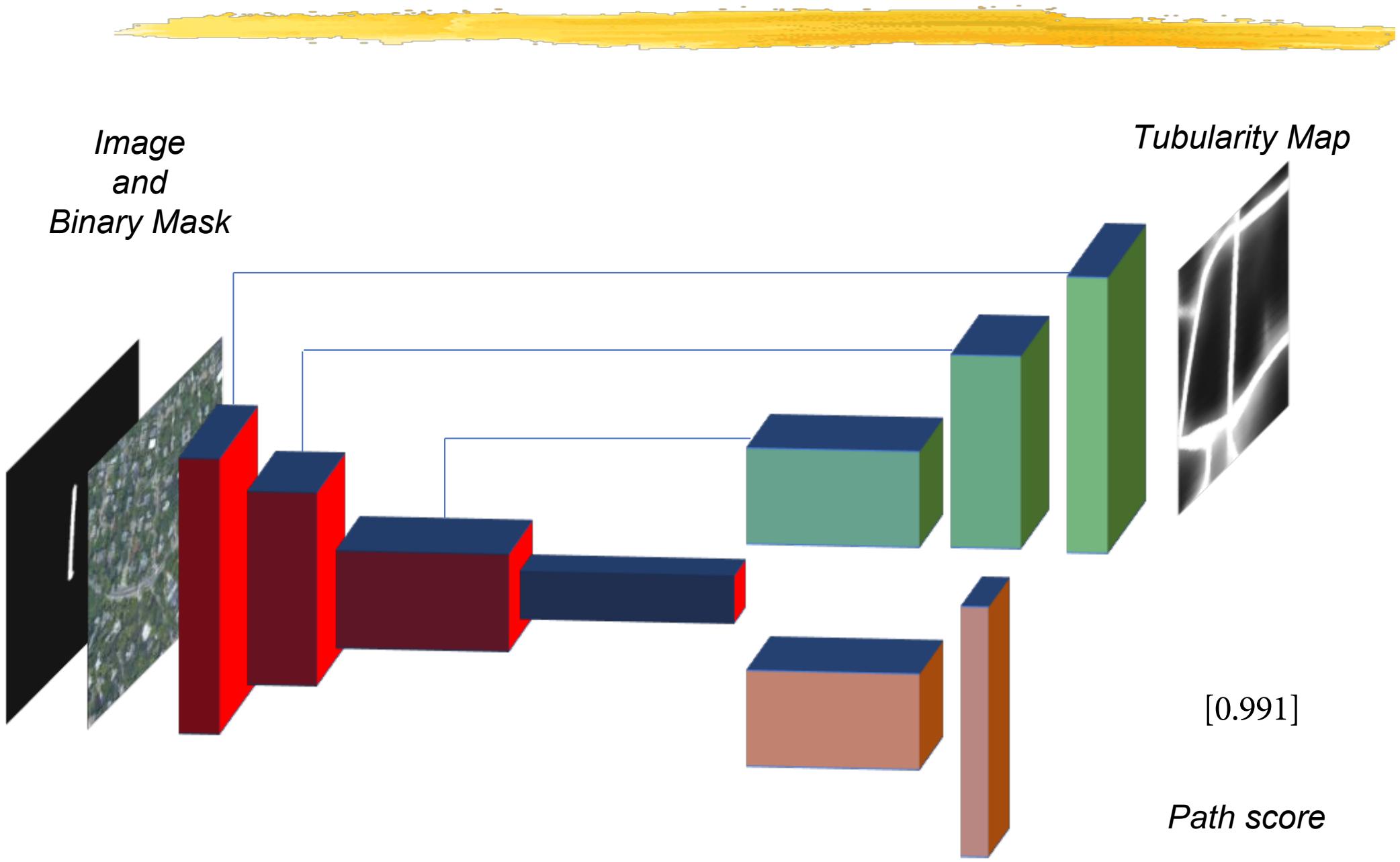
Use the same network to progressively refine the results
keeping the number of parameters constant

DELINEATION STEPS



1. Compute a probability map.
2. Sample and connect the samples.
3. Assign a weight to the paths.
4. Retain the best paths.

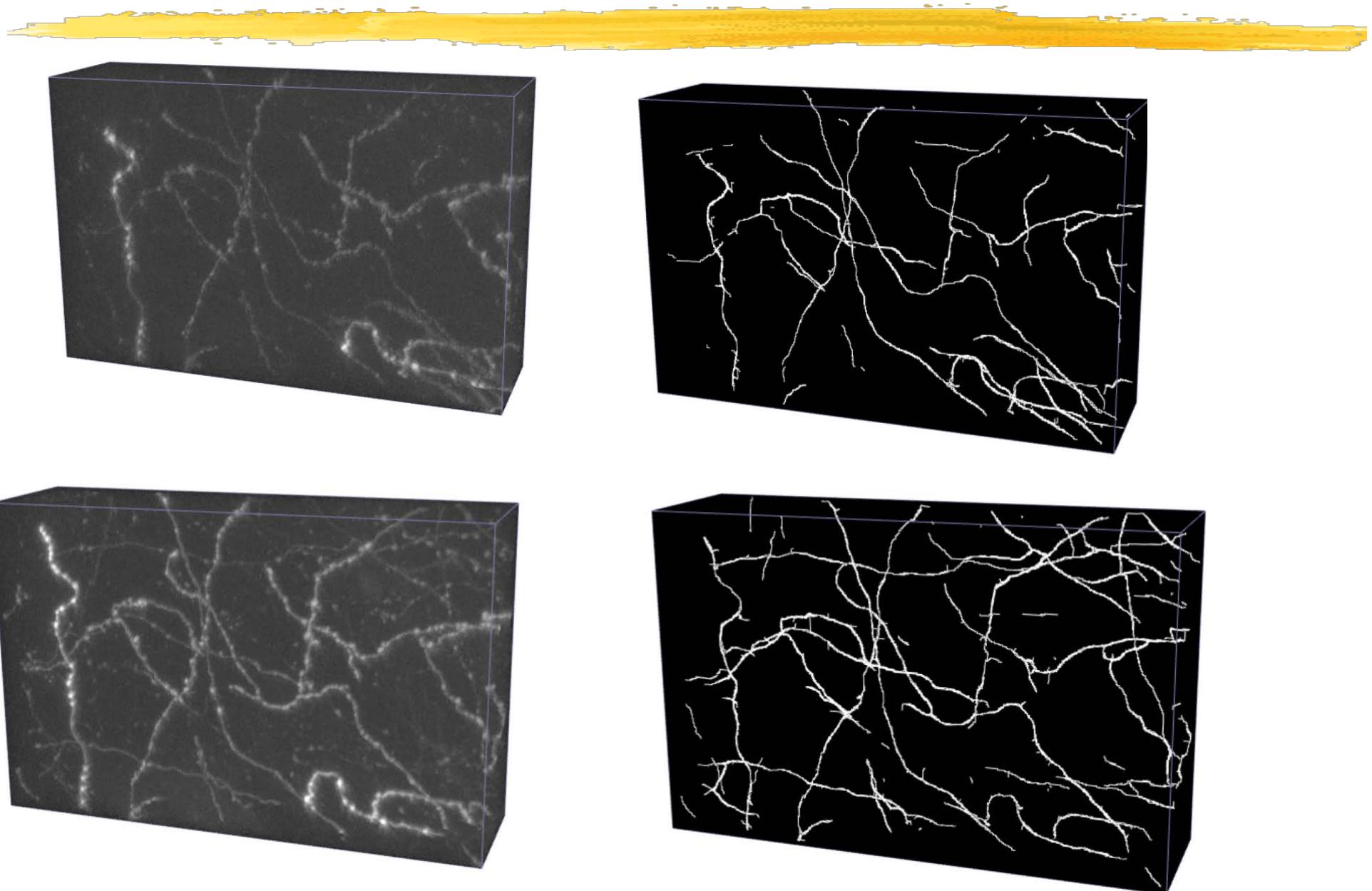
DUAL USE U-NET



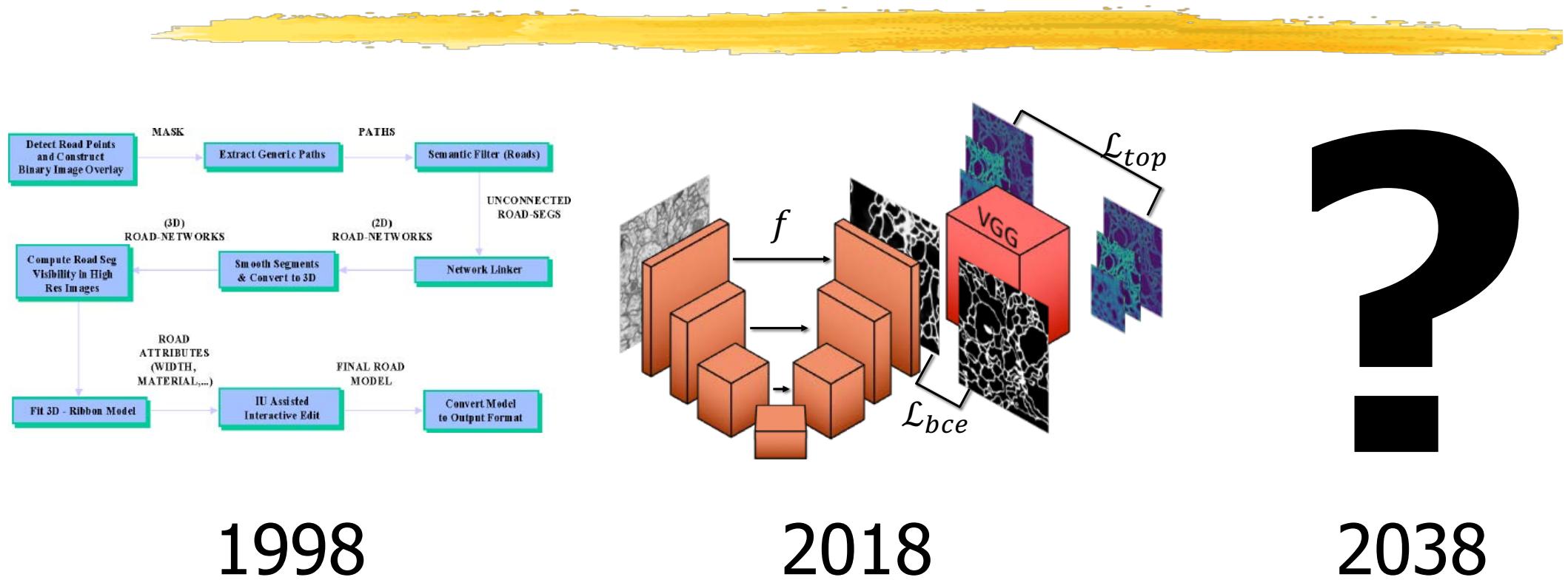
STREETS OF TORONTO



DENDRITES AND AXONS



1998 - 2038



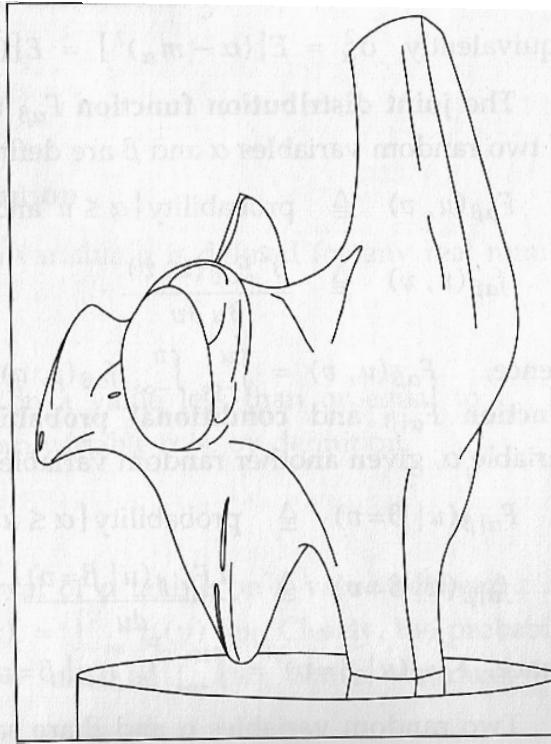
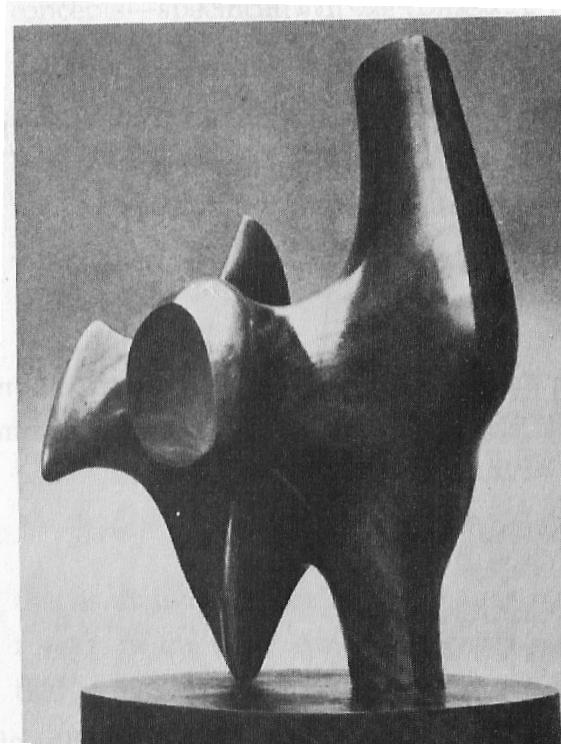
It is difficult to make predictions, especially about the future.
Sometimes attributed to Niels Bohr.

IN SHORT



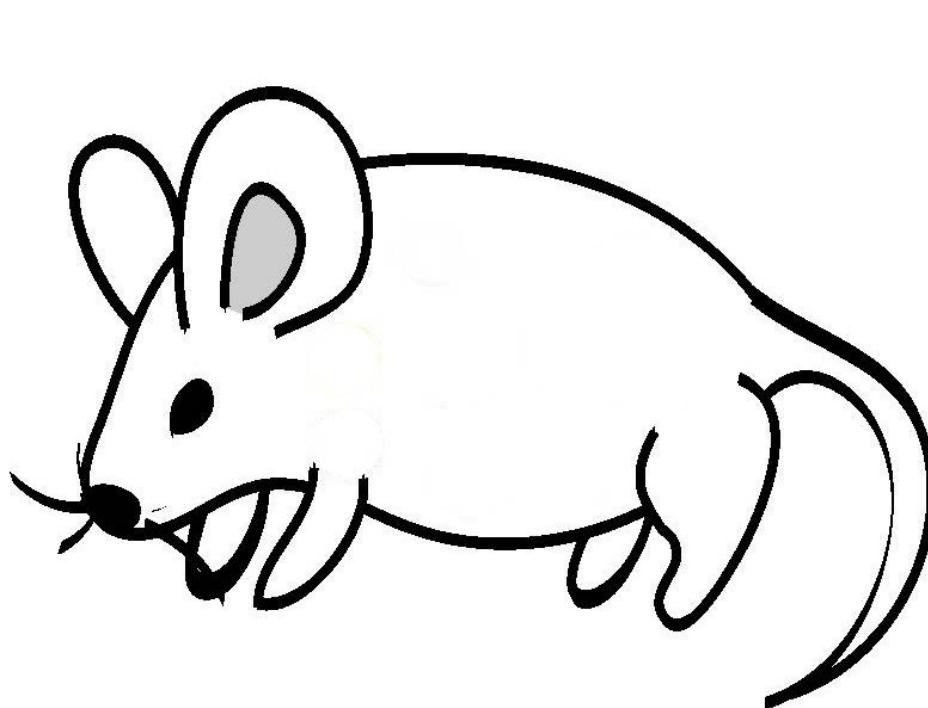
- Edge and image information is noisy.
 - Models are required to make sense of it.
- An appropriate combination of graph-based techniques, machine learning, and semi-automated tools is required.

EDGE DETECTION



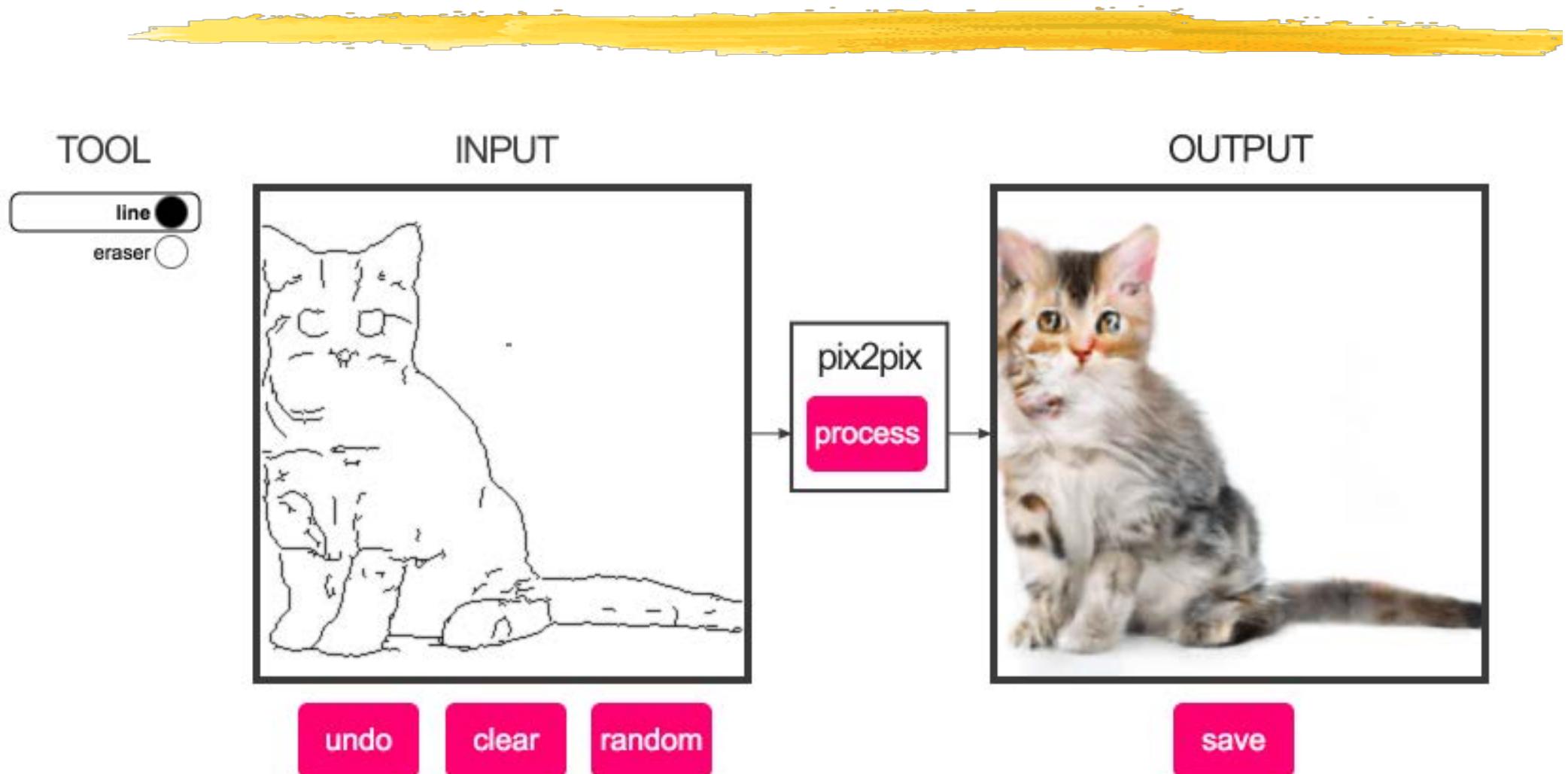
- What's an edge
- Image gradients
- Edge operators

LINE DRAWINGS



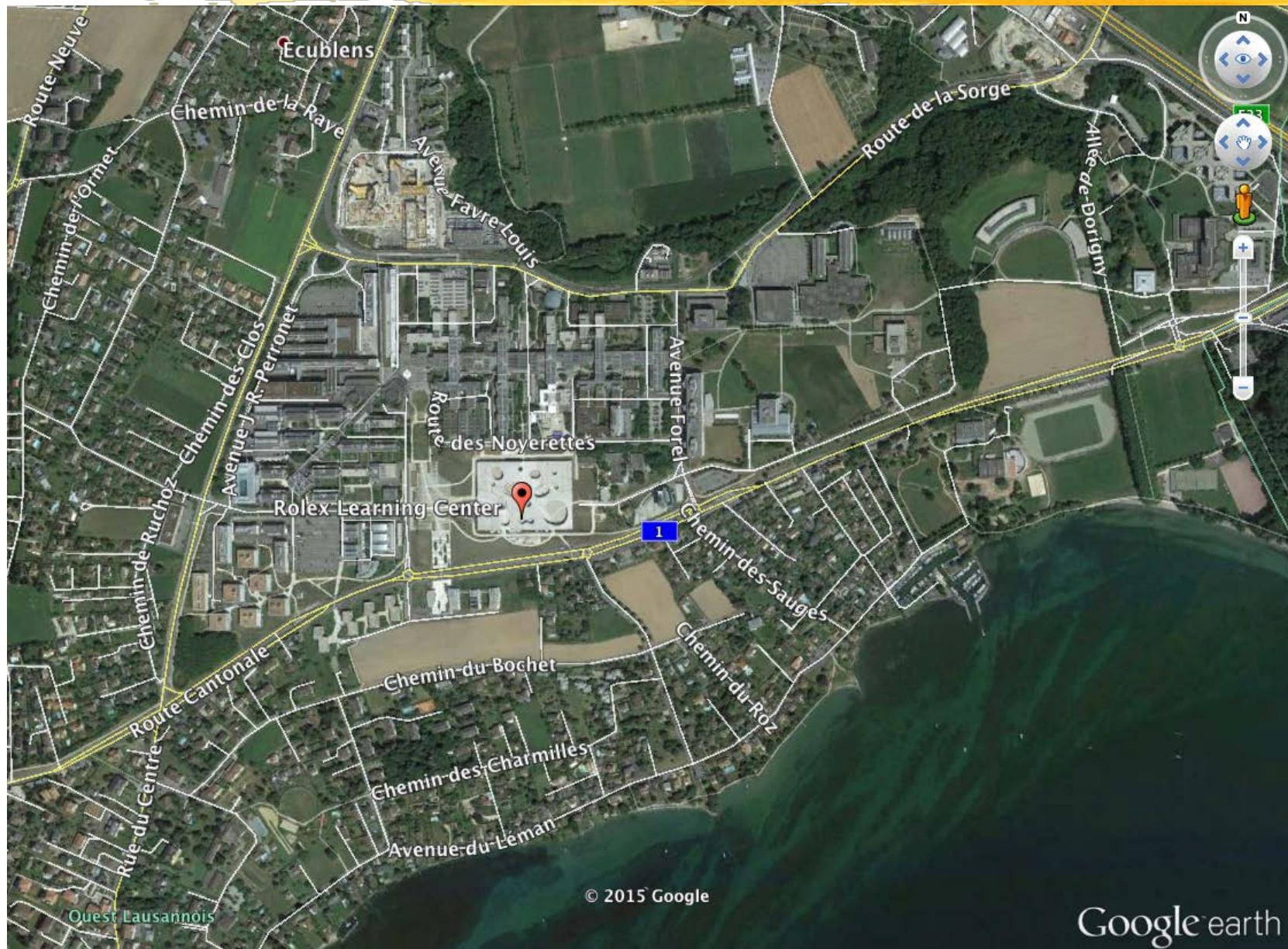
- Edges seem fundamental to human perception.
- They form a compressed version of the image.

FROM EDGES TO CATS



<https://affinelayer.com/pixsrv/>

MAPS AND OVERLAYS



© 2015 Google

Google earth

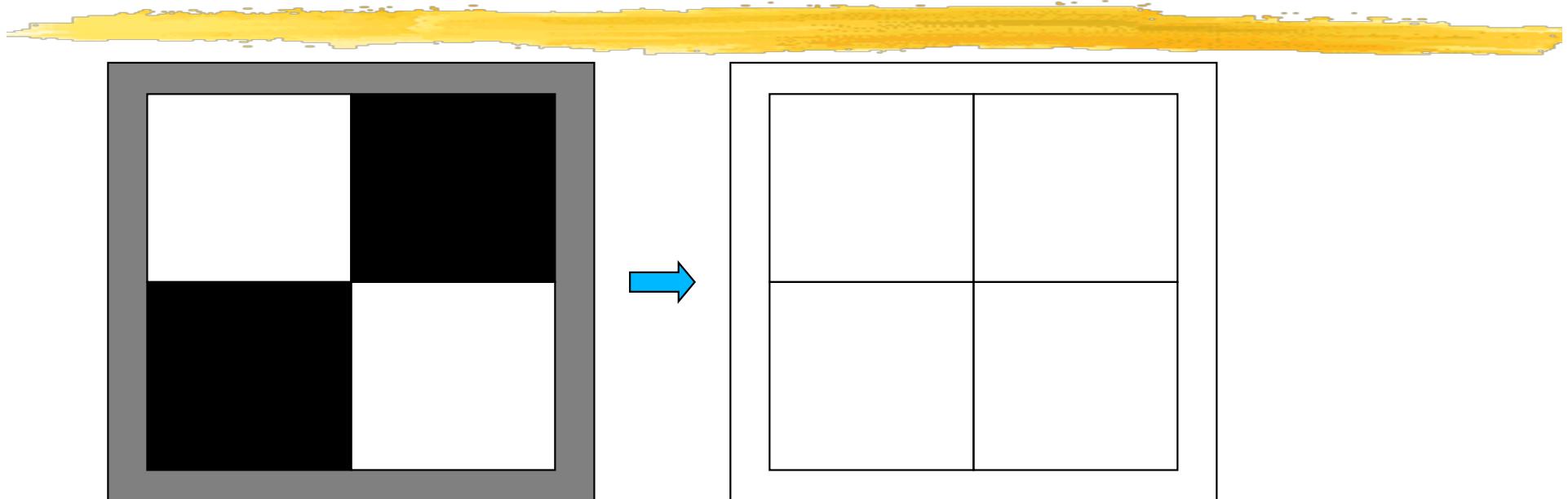
CORRIDOR



CORRIDOR



EDGES AND REGIONS



Edges:

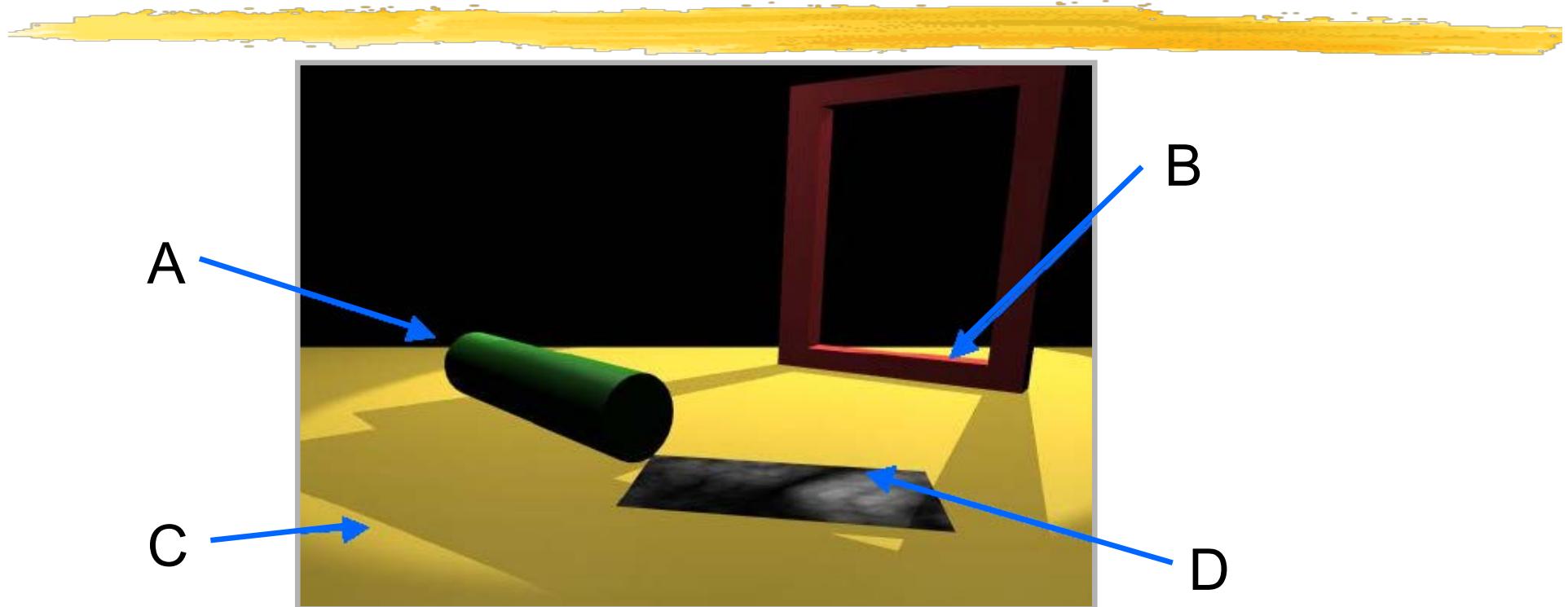
Boundary between bland image regions.

Regions:

Homogenous areas between edges.

→ Duality Edge/Regions.

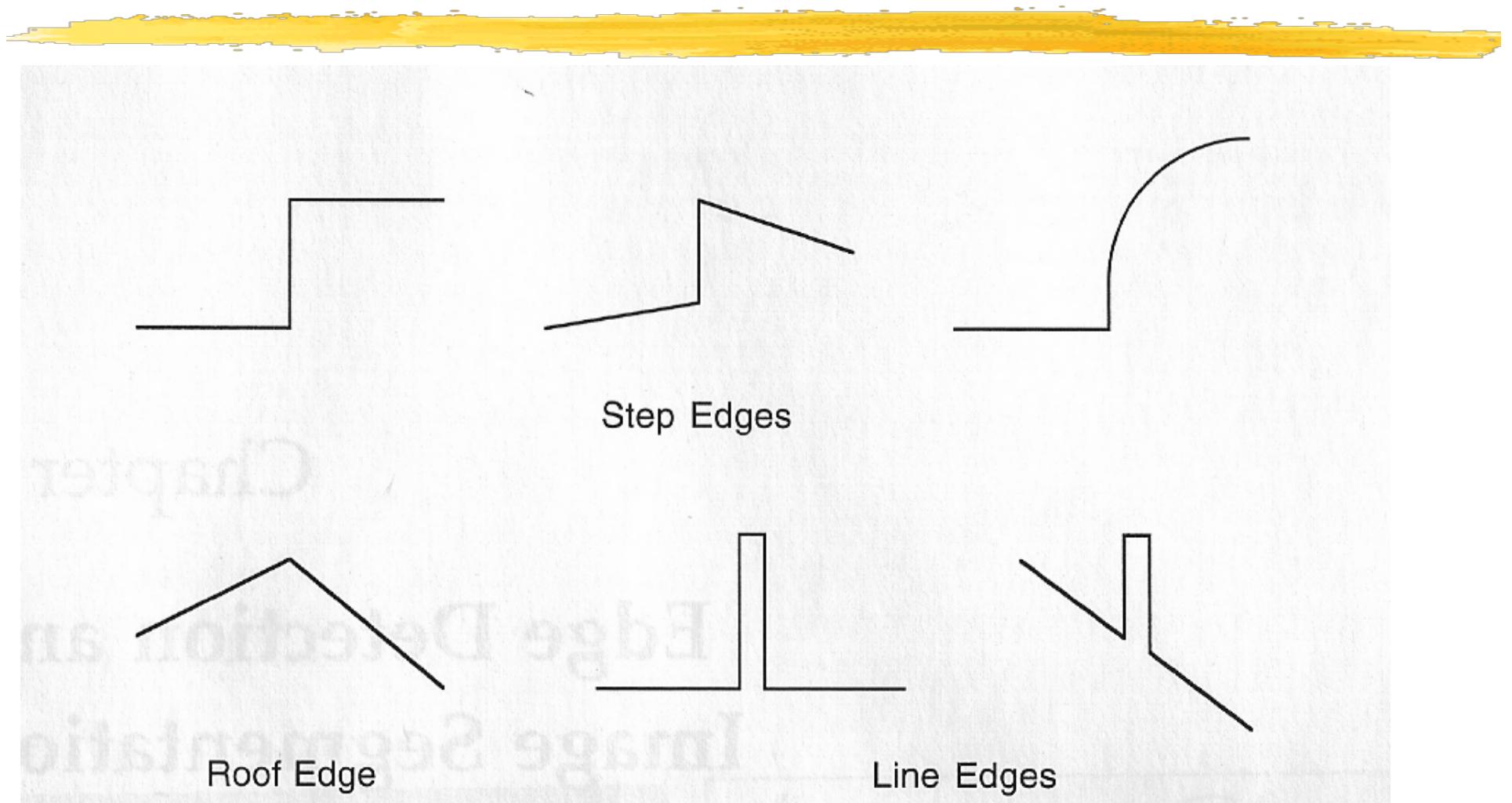
DISCONTINUITIES



- A. Depth discontinuity: Abrupt depth change in the world
- B. Surface normal discontinuity: Change in surface orientation
- C. Illumination discontinuity: Shadows, lighting changes
- D. Reflectance discontinuity: Surface properties, markings

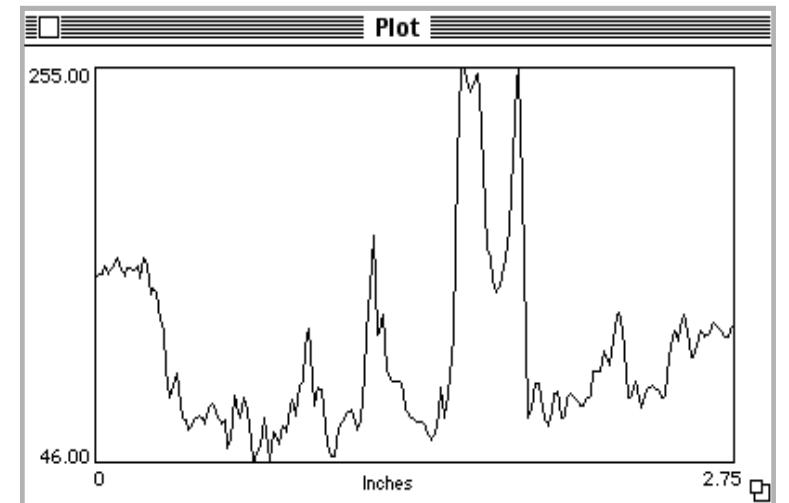
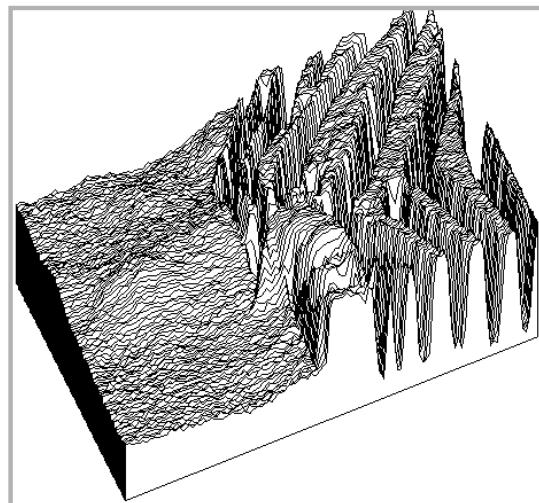
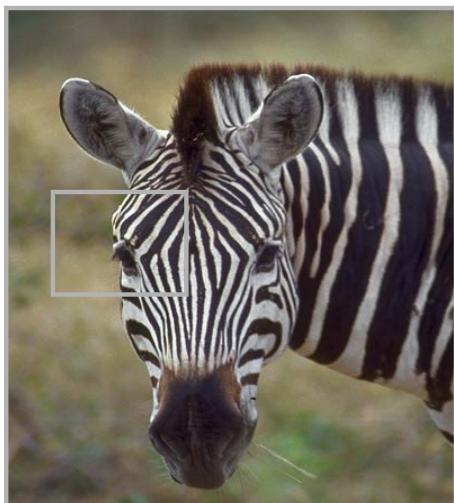
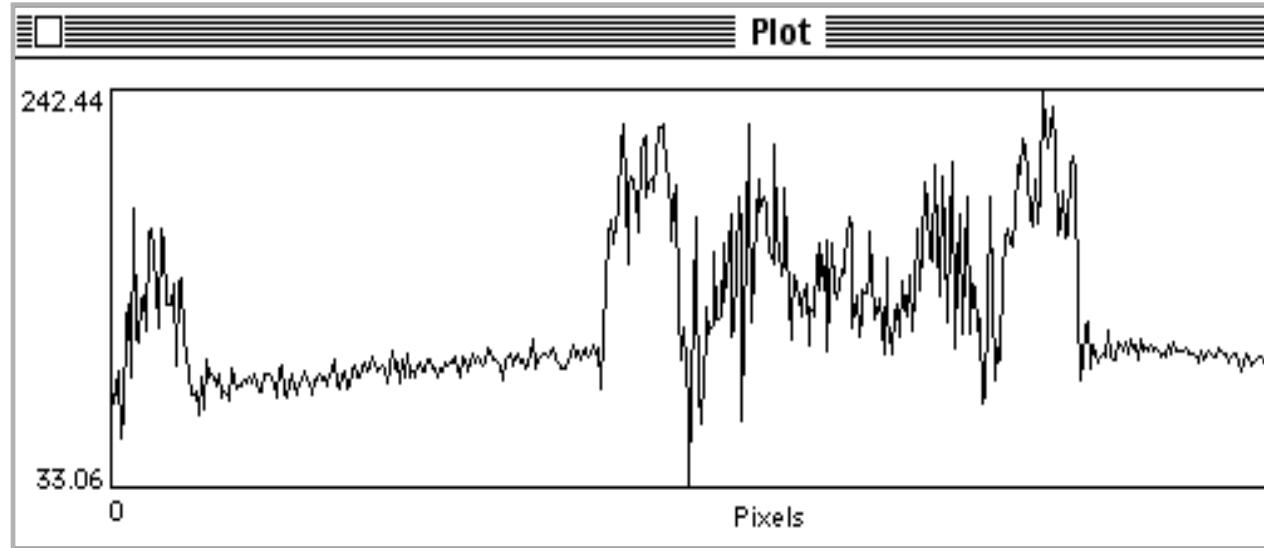
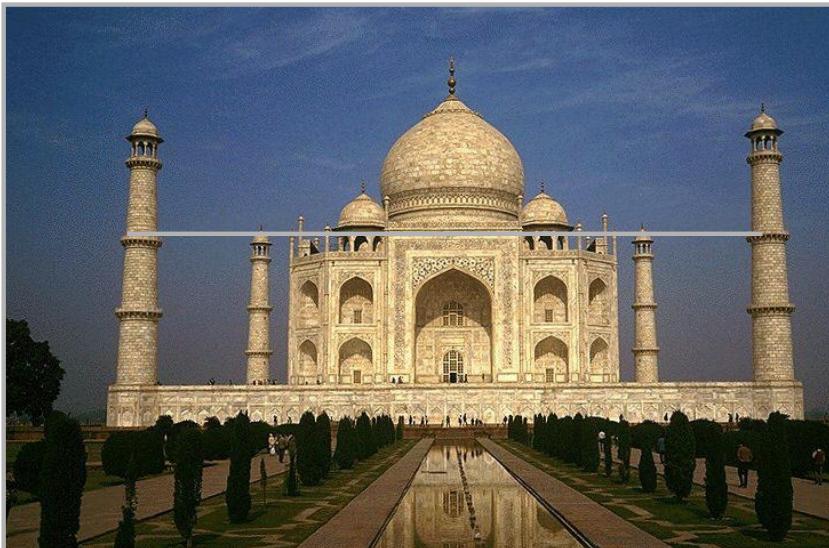
→ **Contrast:** Gray levels different on both sides

EDGE PROFILES

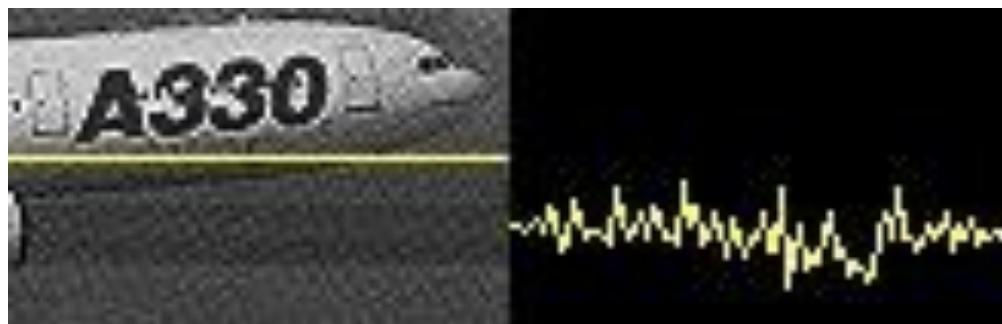
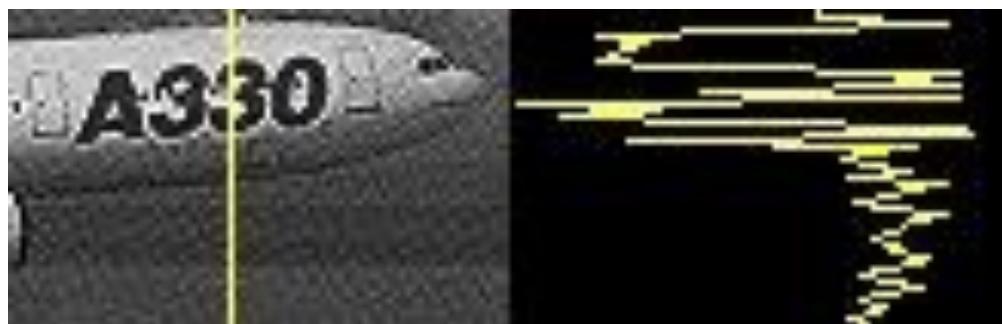


Edges are where a change occurs

REALITY

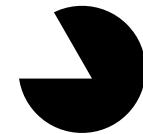


MORE REALITY



Very noisy signals
→ Much knowledge is required!!

ILLUSORY CONTOURS

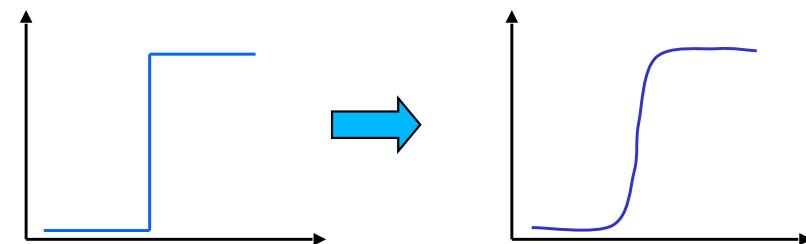


No closed contour, but

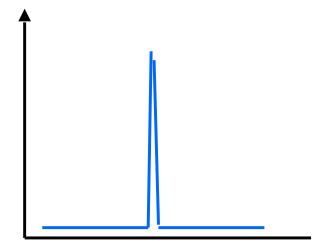
IDEAL STEP EDGE



$f(x) = \text{step edge}$

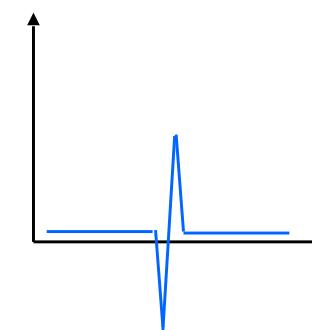


1st Derivative $f'(x)$



maximum

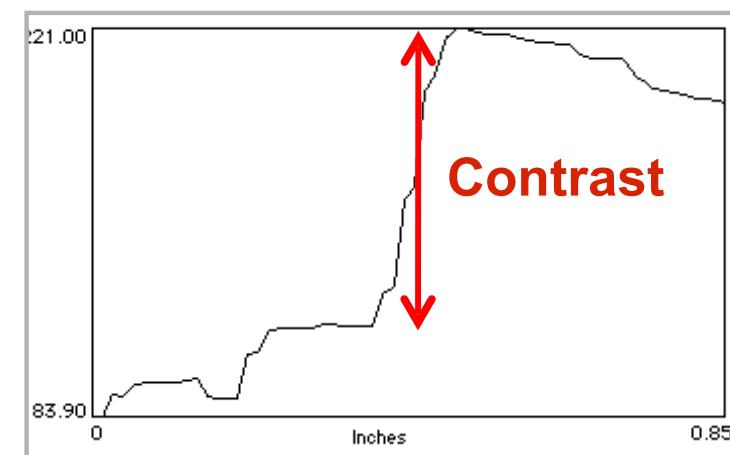
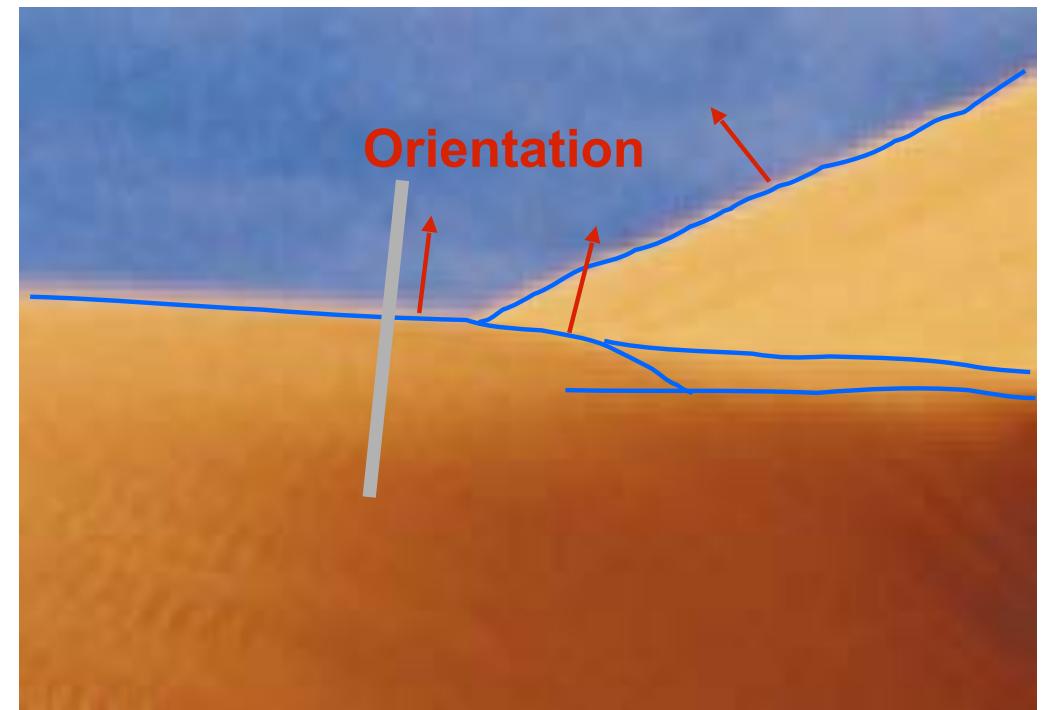
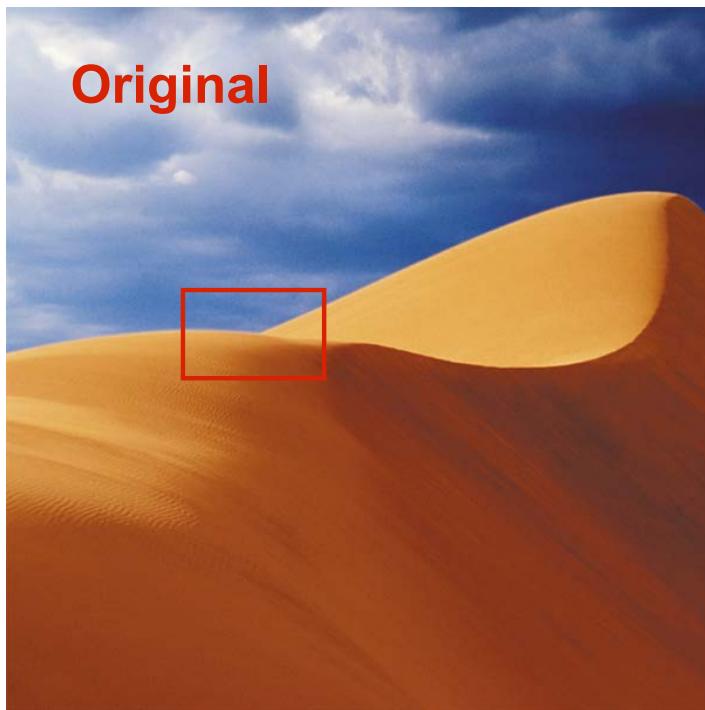
2nd Derivative $f''(x)$



zero crossing

Rapid change in image => High local gradient

EDGE PROPERTIES



EDGE DESCRIPTORS

Edge Normal:

- Unit vector in the direction of maximum intensity change

Edge Direction:

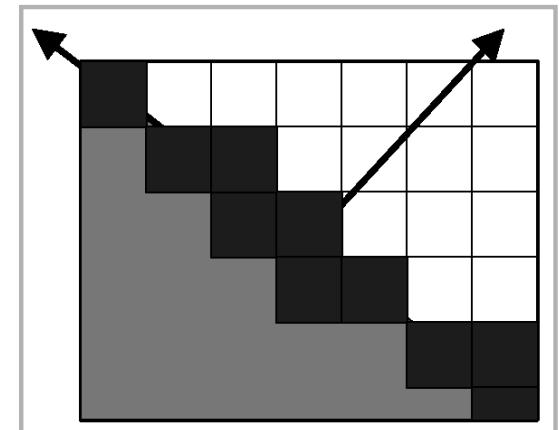
- Unit vector perpendicular to the edge normal

Edge position or center

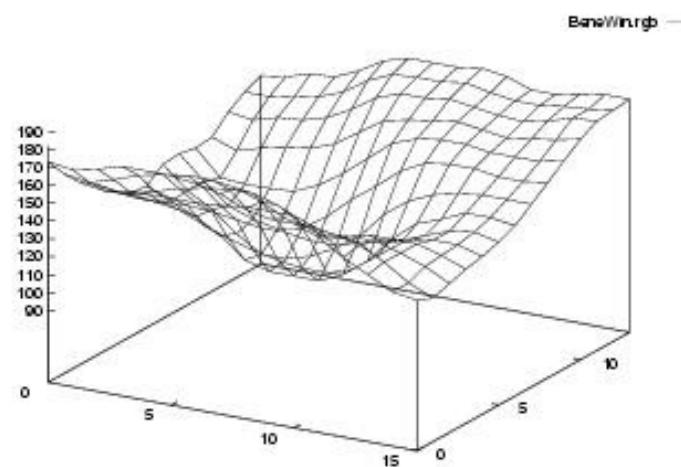
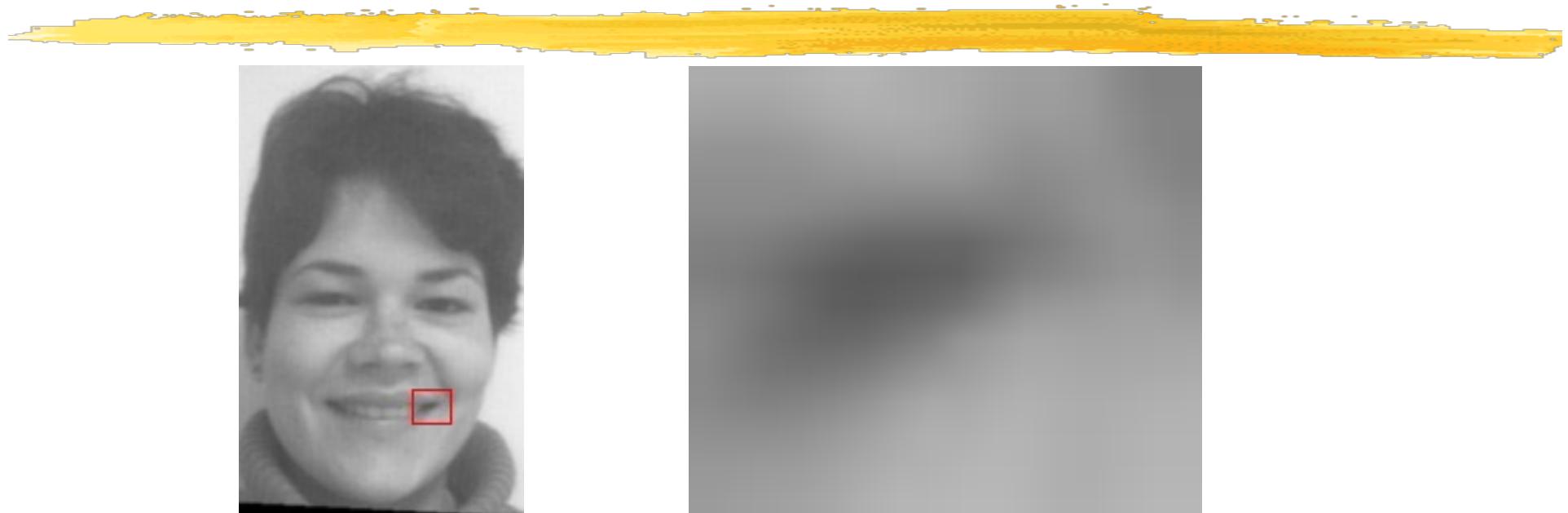
- Image location at which edge is located

Edge Strength

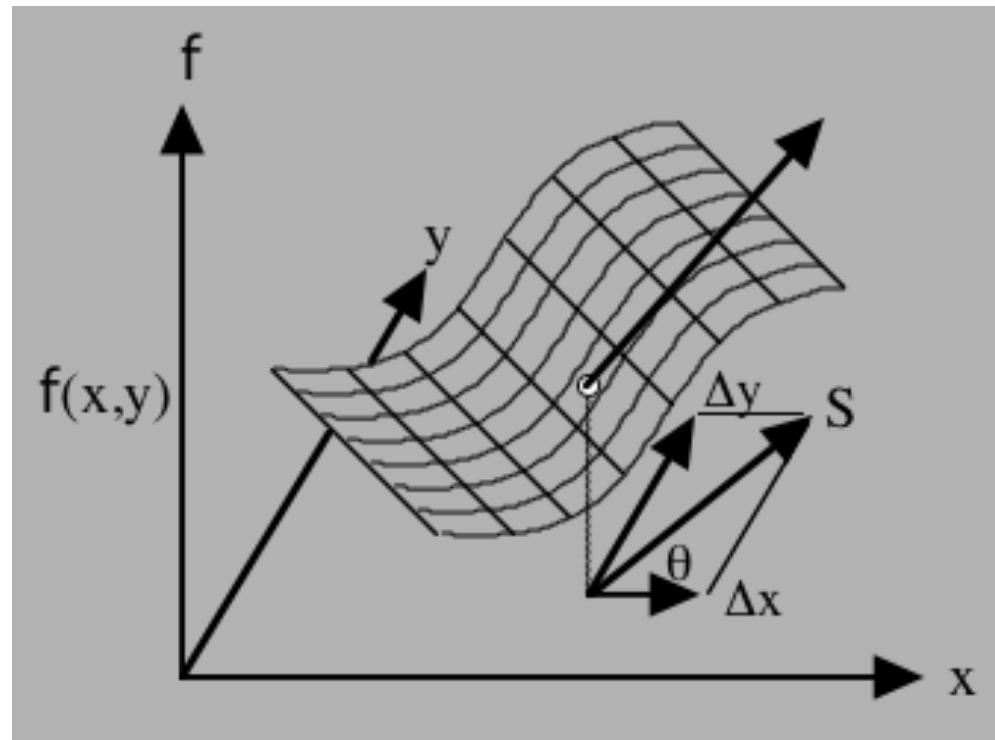
- Speed of intensity variation across the edge.



IMAGES AS 3-D SURFACES



GEOMETRIC INTERPRETATION



Since $I(x,y)$ is not a continuous function:

1. Locally fit a smooth surface.
2. Compute its derivatives.

IMAGE GRADIENT

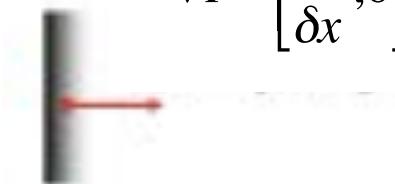


The gradient of an image

$$\nabla I = \left[\frac{\delta I}{\delta x}, \frac{\delta I}{\delta y} \right]$$

points in the direction of most rapid change in intensity.

$$\nabla I = \left[\frac{\delta I}{\delta x}, 0 \right]$$

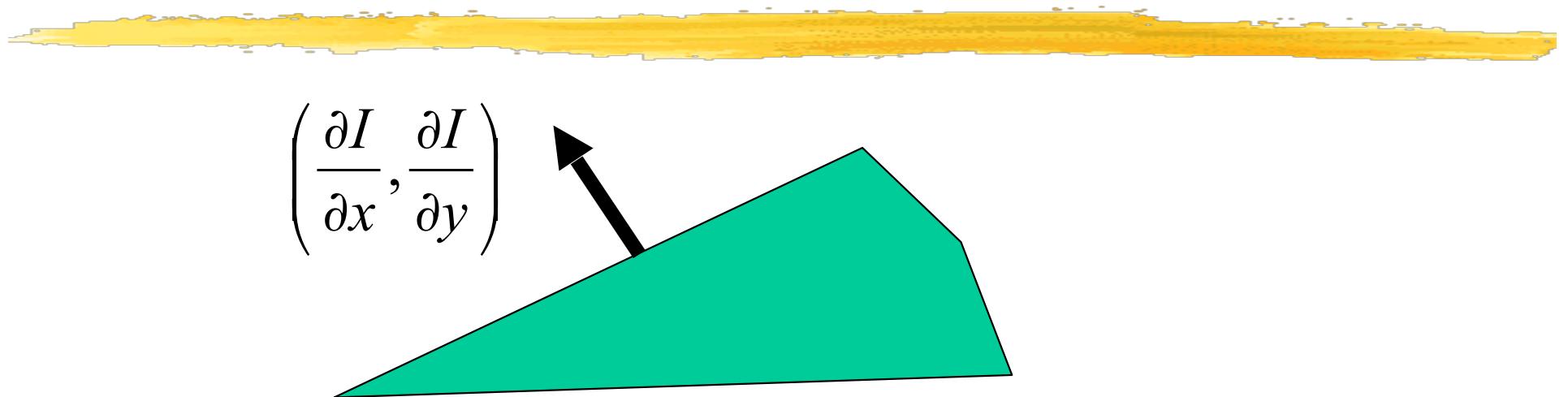


$$\nabla I = \left[\frac{\delta I}{\delta x}, \frac{\delta I}{\delta y} \right]$$

$$\nabla I = \left[0, \frac{\delta I}{\delta y} \right]$$



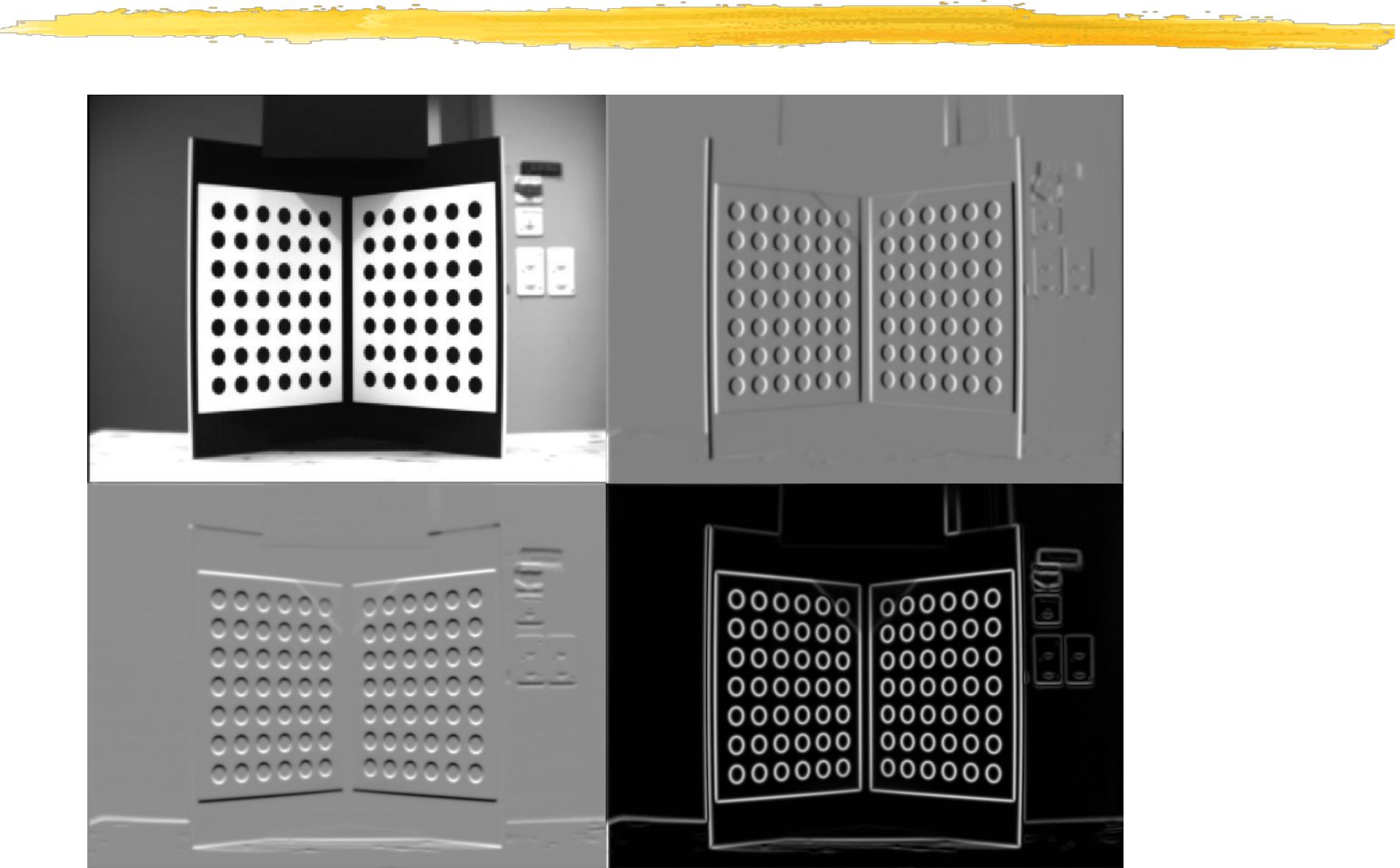
MAGNITUDE AND ORIENTATION



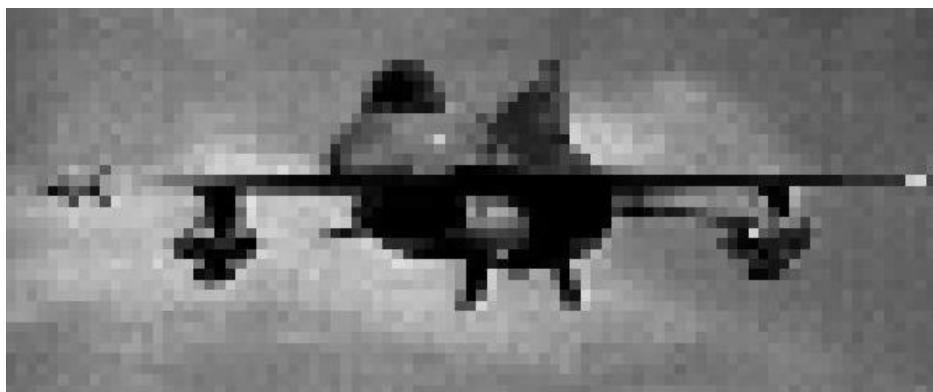
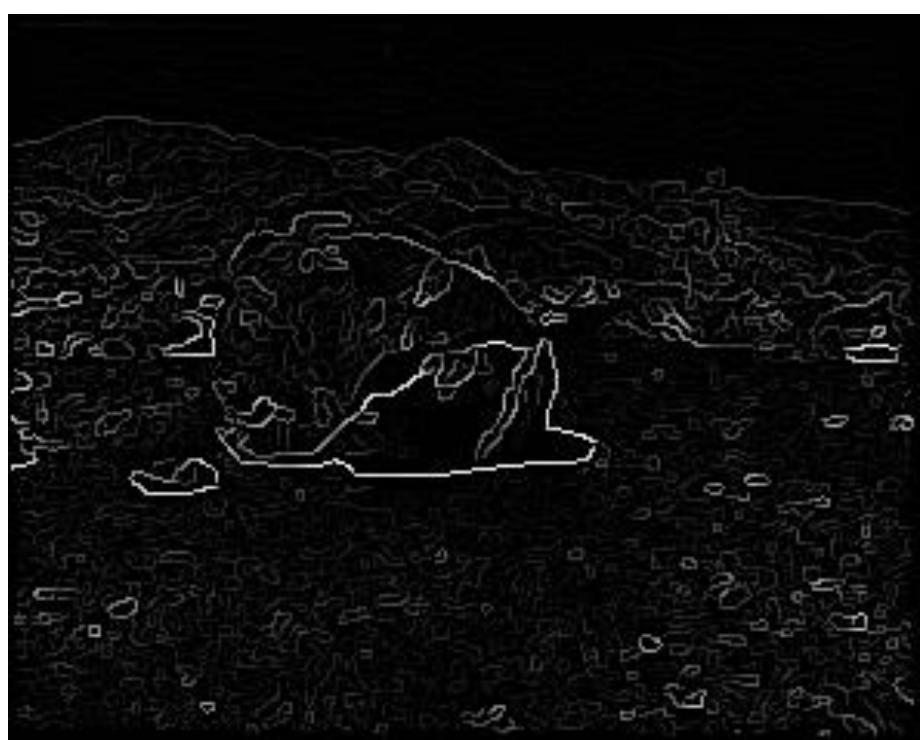
$$\text{Measure of contrast : } G = \sqrt{\frac{\partial I^2}{\partial x} + \frac{\partial I^2}{\partial y}}$$

$$\text{Edge orientation : } \theta = \arctan\left(\frac{\partial I}{\partial y}, \frac{\partial I}{\partial x}\right)$$

GRADIENT IMAGES



REAL IMAGES

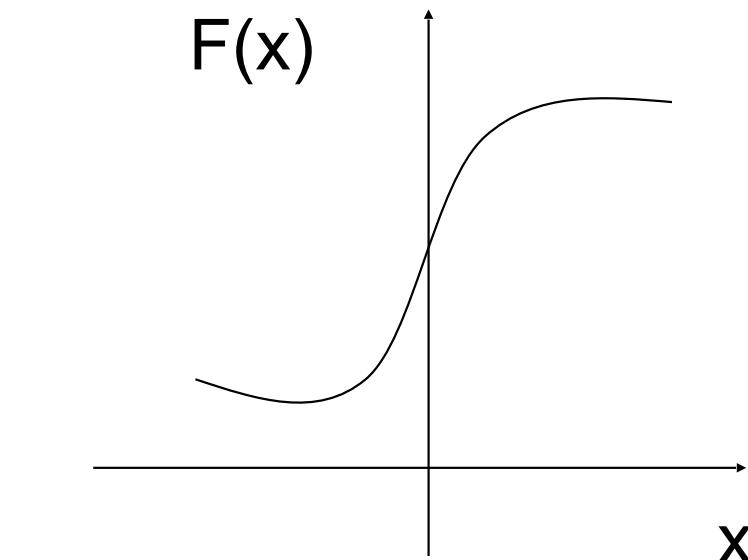


EDGE OPERATORS

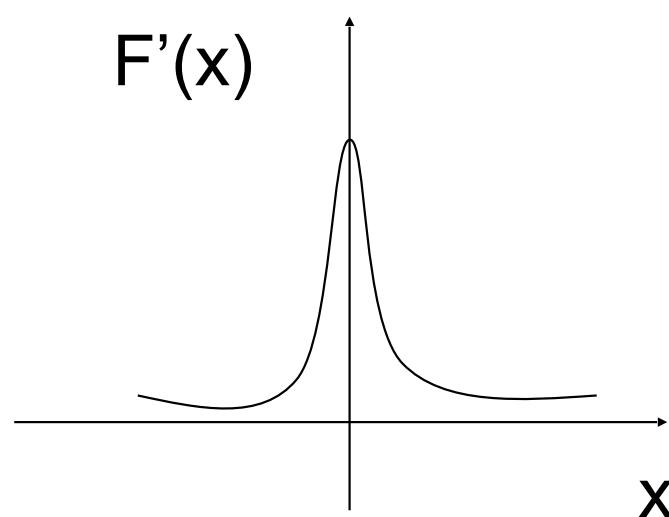


- Difference Operators
- Convolution Operators
- Parametric Matchers
- Trained Detectors
- Deep Nets

GRADIENT METHODS



Edge = Sharp variation

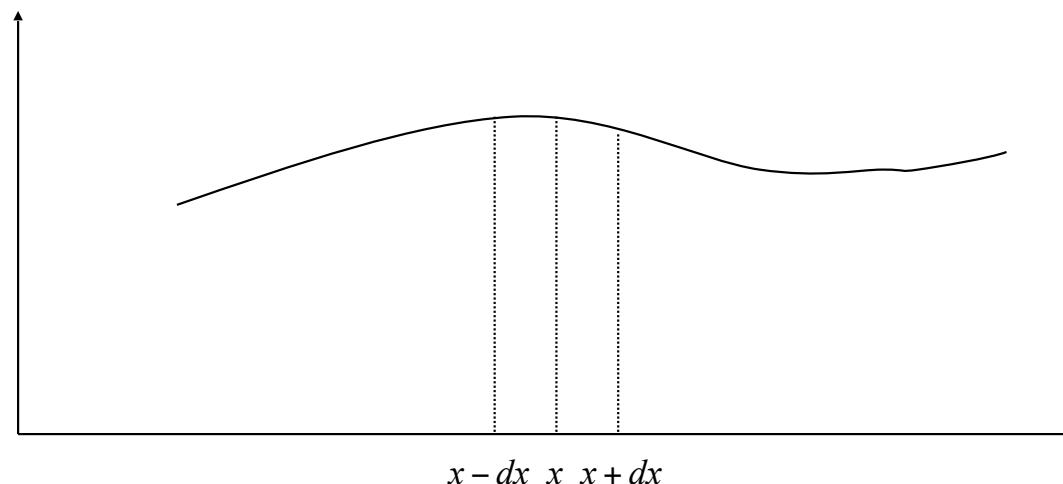


Large first derivative



1D FINITE DIFFERENCES

In one dimension:



$$\frac{df}{dx} \approx \frac{f(x + dx) - f(x)}{dx} \approx \frac{f(x + dx) - f(x - dx)}{2dx}$$

$$\frac{d^2f}{dx^2} \approx \frac{f(x + dx) - 2f(x) + f(x - dx)}{dx^2}$$

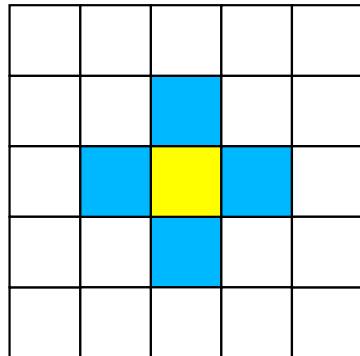
CODING 1D FINITE DIFFERENCES



Line stored as an array:

- `for i in range(n-1):
 q[i]=(p[i+1]-p[i])`
- `for i in range(1,n-1):
 q[i]=(p[i+1]-p[i-1])/2`
- `q=(p[2:]-p[:-2])/2`

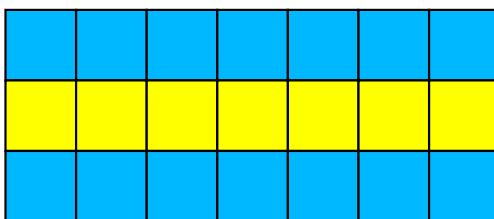
2D FINITE DIFFERENCES



$$\frac{\partial f}{\partial x} \approx \frac{f(x + dx, y) - f(x, y)}{dx} \approx \frac{f(x + dx, y) - f(x - dx, y)}{2dx}$$

$$\frac{\partial f}{\partial y} \approx \frac{f(x, y + dy) - f(x, y)}{dy} \approx \frac{f(x, y + dy) - f(x, y - dy)}{2dy}$$

CODING 2D FINITE DIFFERENCES



p →



Image stored as a 2D array:

- $dx = p[1,:,:] - p[:-1,:]$
 $dy = p[:,1:] - p[:,:-1]$
- $dx = (p[2,:,:] - p[:-2,:])/2$
 $dy = (p[:,2:] - p[:,:-2])/2$

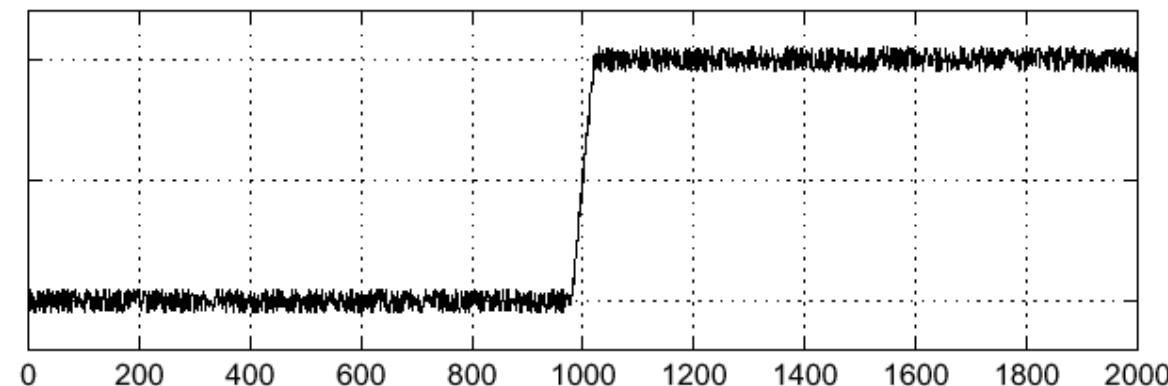
Image stored in raster format:

```
{  
    int i;  
    for(i=0;i<xdim;i++){  
        dx[i] = p[i+1] - p[i];  
        dy[i] = p[i+xdim]-p[i];  
    }  
}
```

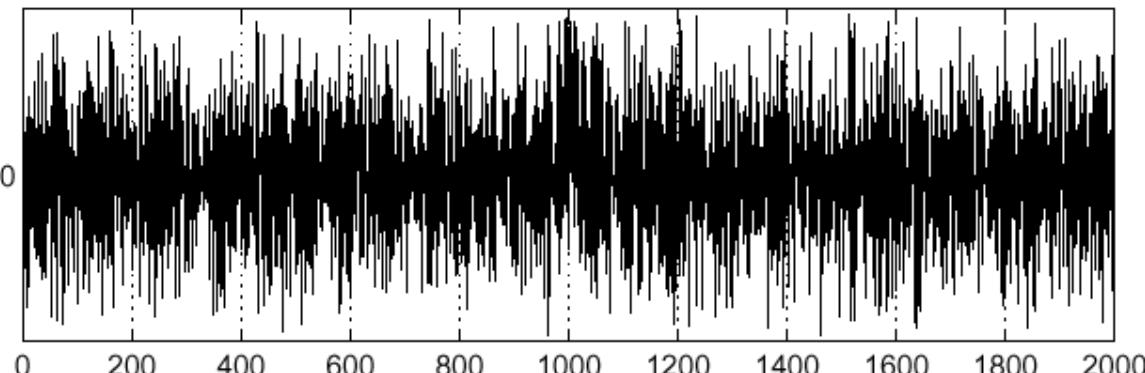
NOISE IN 1D



Consider a single row or column of the image:



$$\frac{d}{dx} f(x)_0$$



FOURIER INTERPRETATION



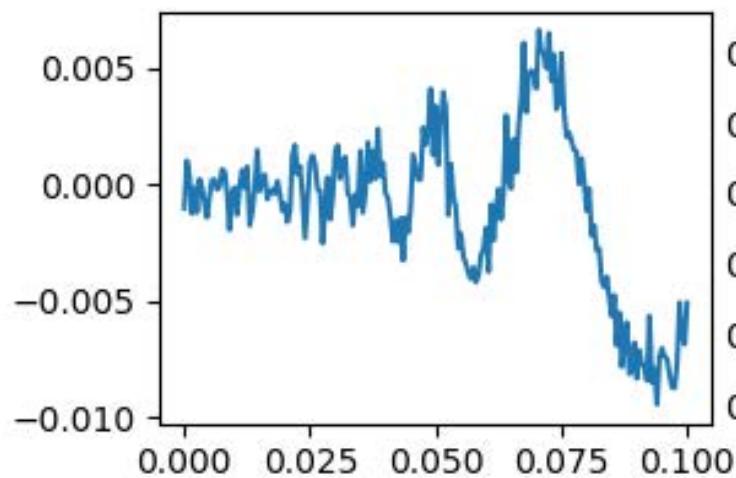
Function	Fourier Transform
$\frac{df}{dx}(x)$	$uF(u)$
$\frac{\delta f}{\delta x}(x, y)$	$uF(u, v)$
$\frac{\delta f}{\delta y}(x, y)$	$vF(u, v)$

→ Differentiating emphasizes high frequencies
and therefore noise!

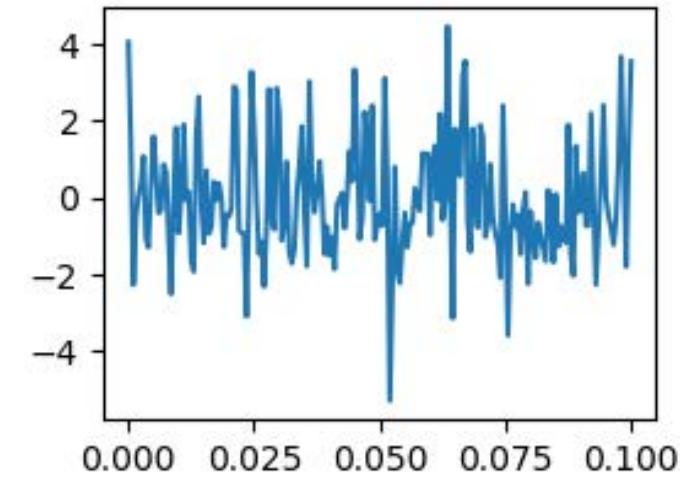
$$f(x) = x^2 \sin(1/x)$$



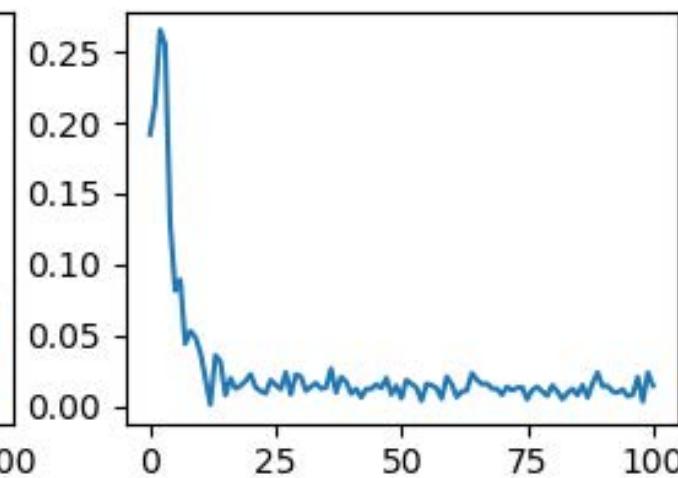
f



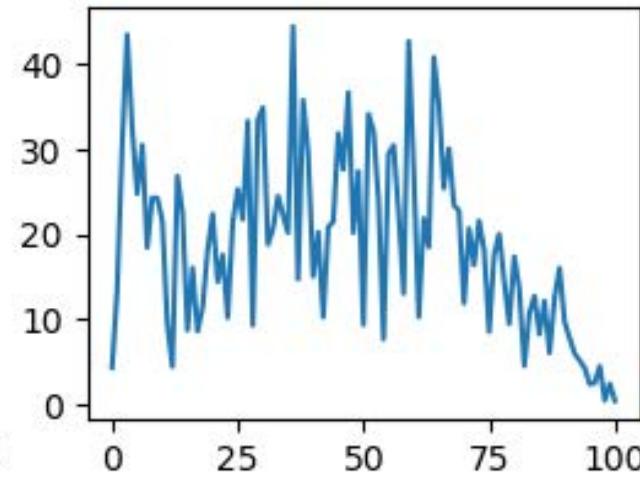
$\frac{df}{dx}$



F



uF



REMOVING NOISE



Problem:

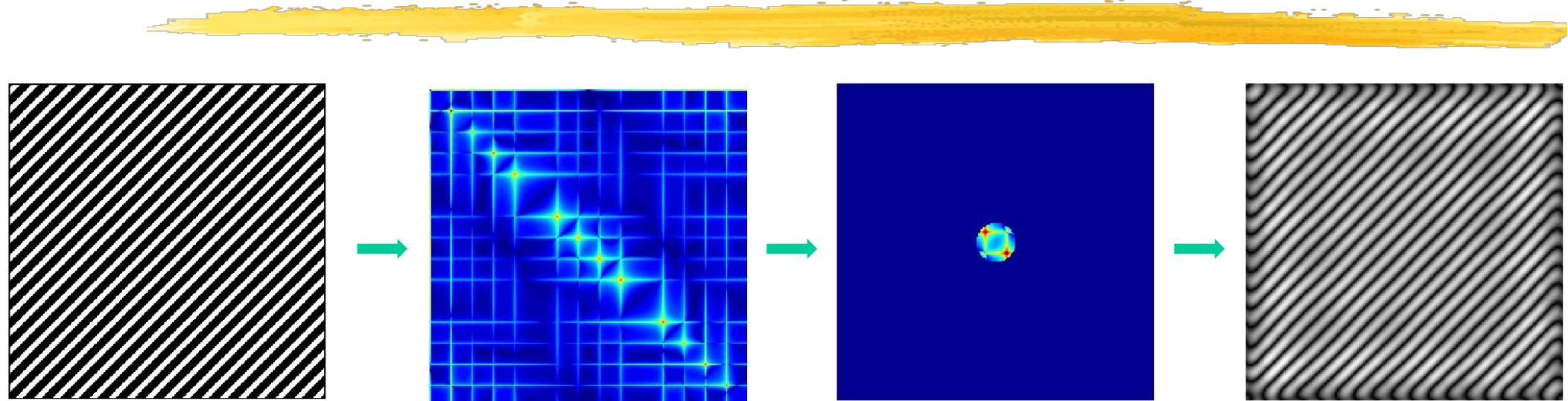
High frequencies lead to trouble with derivation.

Solution:

Suppress high frequencies by

- multiplying DFT of the signal with something that suppresses high frequencies.

DIAGONAL STRUCTURES



Rotated stripes:

- Dominant diagonal structures
- Discretization produces additional harmonics

Removing higher frequencies and reconstructing:

- Smoothed image

REMOVING NOISE



Problem:

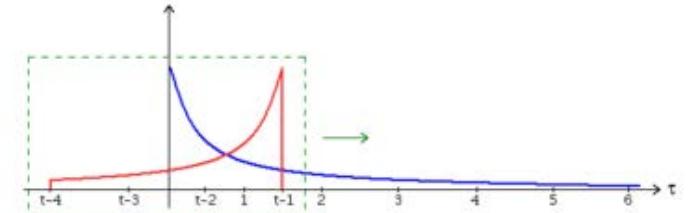
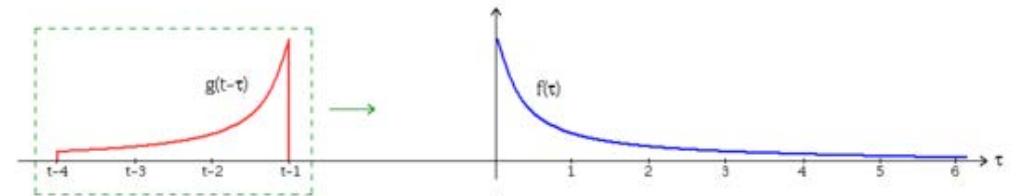
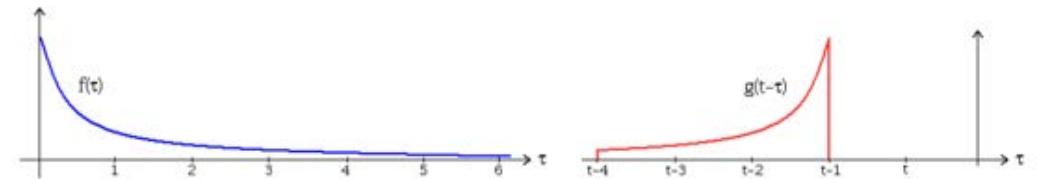
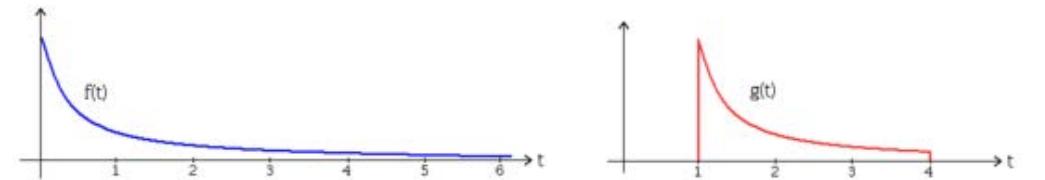
High frequencies lead to trouble with derivation.

Solution:

Suppress high frequencies by

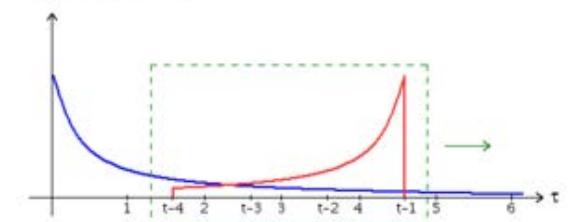
- multiplying DFT of the signal with something that suppresses high frequencies;
- convolving with a low-pass filter.

1D CONVOLUTION

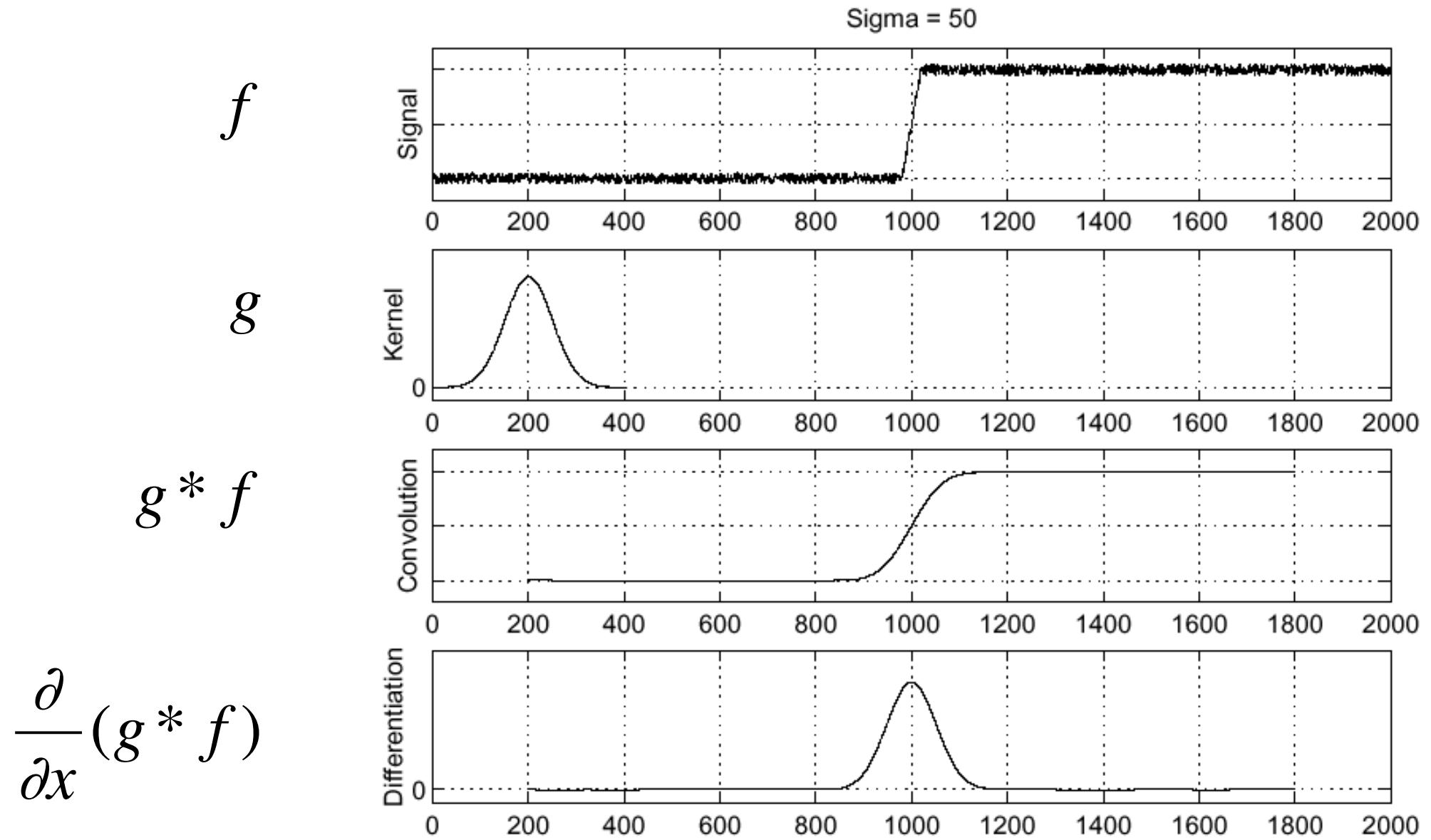


$$g * f(t) = \int_{\tau} g(t - \tau) f(\tau) d\tau$$

$$g * f(m) = \sum_n g(m - n) f(n)$$



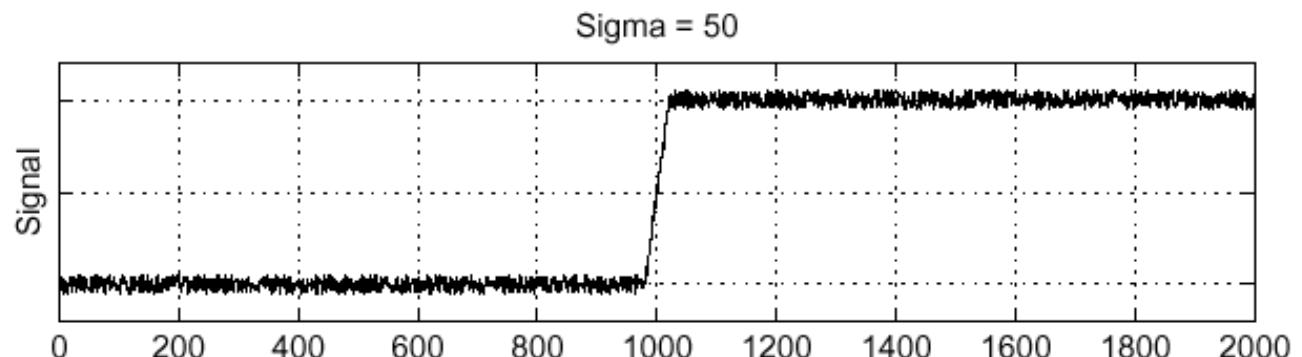
SOLUTION: SMOOTH FIRST



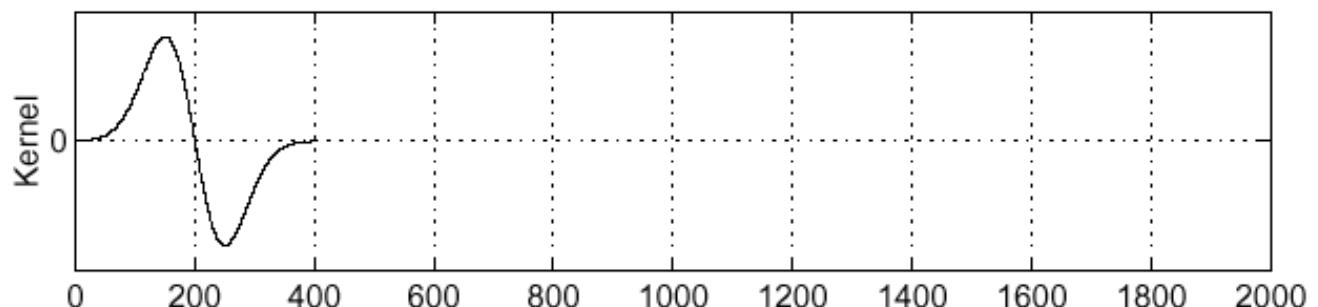
DERIVATIVE THEOREM OF CONVOLUTION



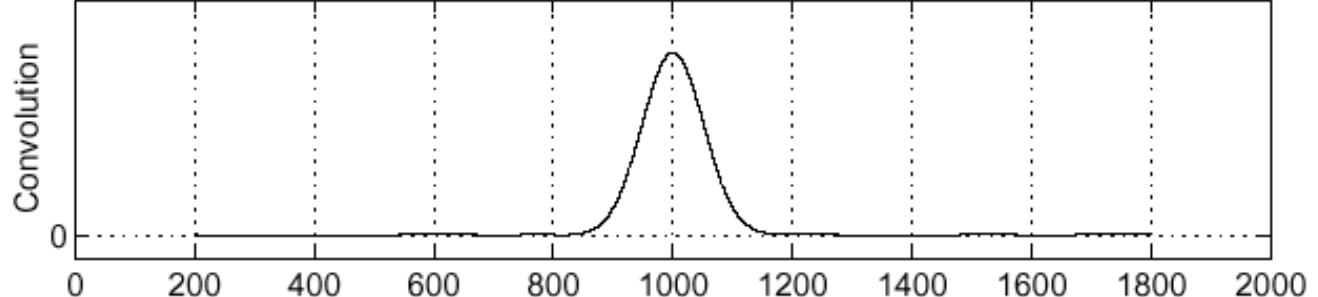
f



$\frac{\partial g}{\partial x}$



$$\frac{\partial}{\partial x} (g * f) = \frac{\partial g}{\partial x} * f$$



--> Faster because dg/dx can be precomputed.

1D AND 2D CONVOLUTION

Continuous case:

$$m \bullet f(x) = \int_u m(u) f(x - u) du$$

$$m \bullet f(x, y) = \iint_{u \ v} m(u, v) f(x - u, y - v) du dv$$

Discrete case:

$$m \bullet f(x) = \sum_{i=-w}^w m(i) f(x - i)$$

$$m \bullet f(x, y) = \sum_{i=-w}^w \sum_{j=-h}^h m(i, j) f(x - i, y - j)$$

CONVOLUTION IN C

Naive C implementation:

```
static double g[][]={{{-1.0,-2.0,-1.0},{0.0,0.0,0.0},{1.0,2.0,1.0}};  
{  
    for(i=i0;i<N;i++)  
        for(j=j0;j<N;j++){  
            q[i][j]=0;  
            for(a=a0;a<W;a++)  
                for(b=b0;b<W,b++)  
                    q[i][j]+=g[a][b]*p[i-a][j-b];  
        }  
}
```

Computational complexity:

- N^2W^2 multiplications for a $N \times N$ image and a $W \times W$ mask.
- Lots of memory access

→ Slow, but can be sped up when the filters are separable.

DIFFERENTIATION AS CONVOLUTION

$$[-1,1] \rightarrow \frac{\partial f}{\partial x} \approx \frac{f(x+dx, y) - f(x, y)}{dx}$$

$$[-0.5, 0, 0.5] \rightarrow \frac{\partial f}{\partial x} \approx \frac{f(x+dx, y) - f(x-dx, y)}{2dx}$$

$$\begin{bmatrix} -1 \\ 1 \end{bmatrix} \rightarrow \frac{\partial f}{\partial y} \approx \frac{f(x, y+dy) - f(x, y)}{dy}$$

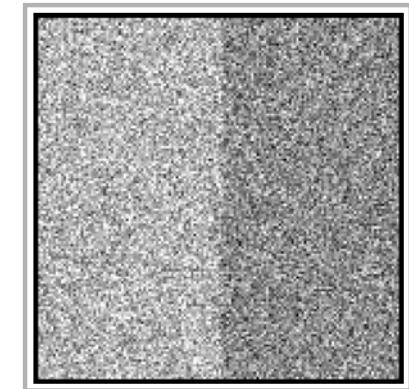
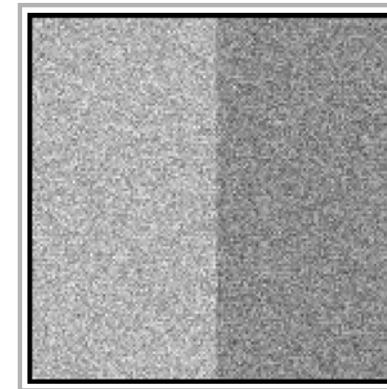
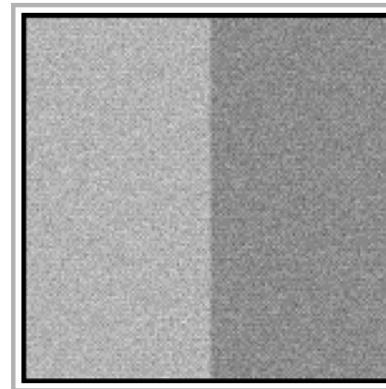
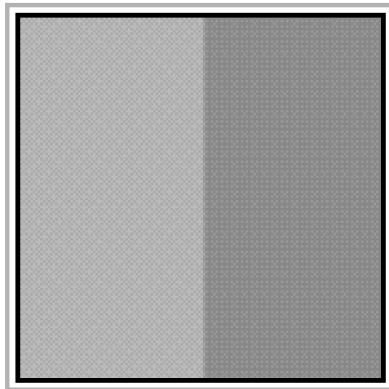
$$\begin{bmatrix} -0.5 \\ 0 \\ 0.5 \end{bmatrix} \rightarrow \frac{\partial f}{\partial y} \approx \frac{f(x, y+dy) - f(x, y-dy)}{2dy}$$

NOISE IN 2D

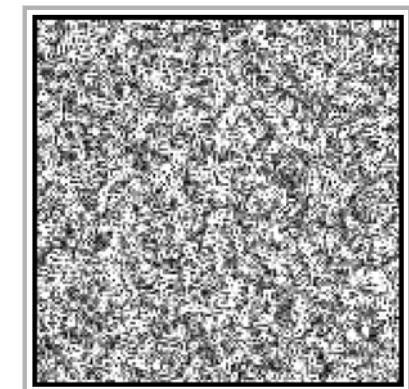
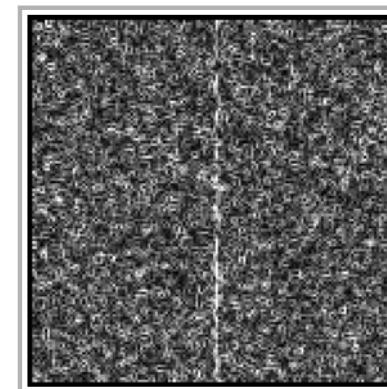
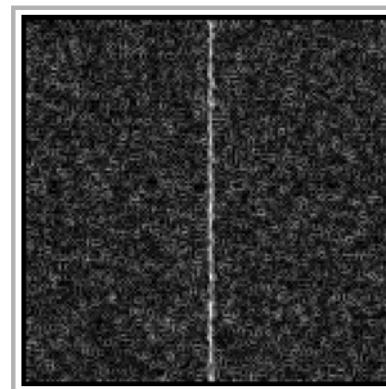
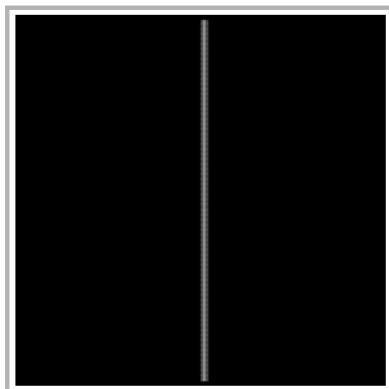


Ideal step edge

Step edge + noise

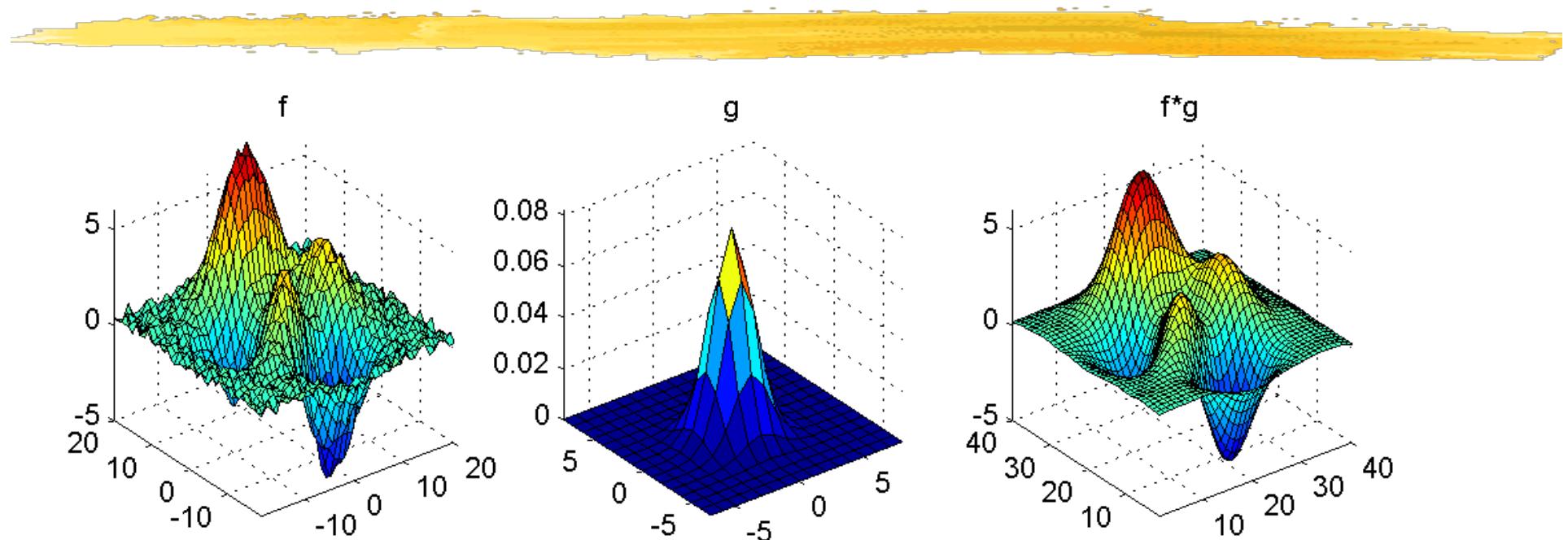


Increasing noise



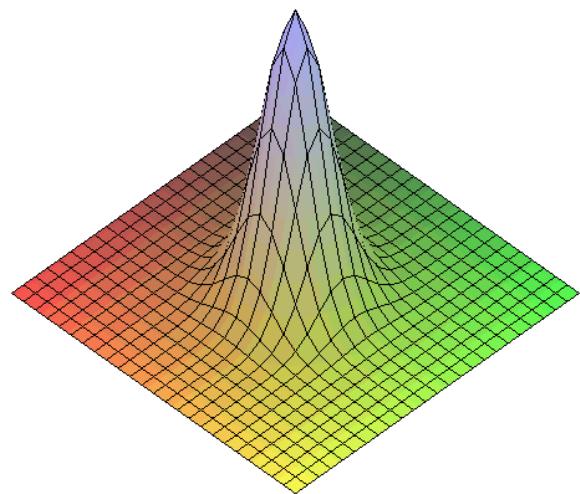
→ Use wider masks to add an element of smoothing

GAUSSIAN SMOOTHING

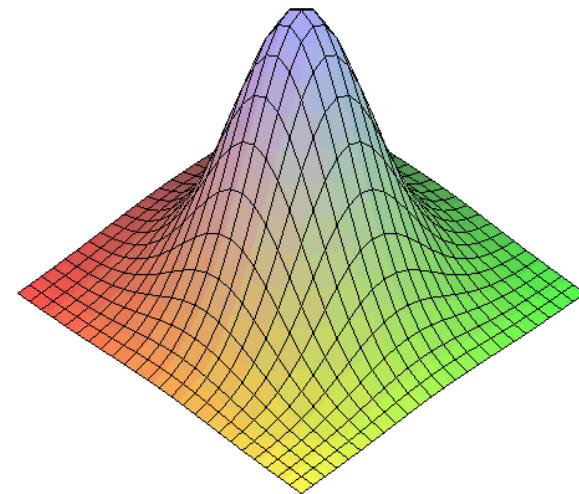


- Eliminates high frequency noise.
- Is fast because the kernel is
 - small,
 - separable.

GAUSSIAN MASKS



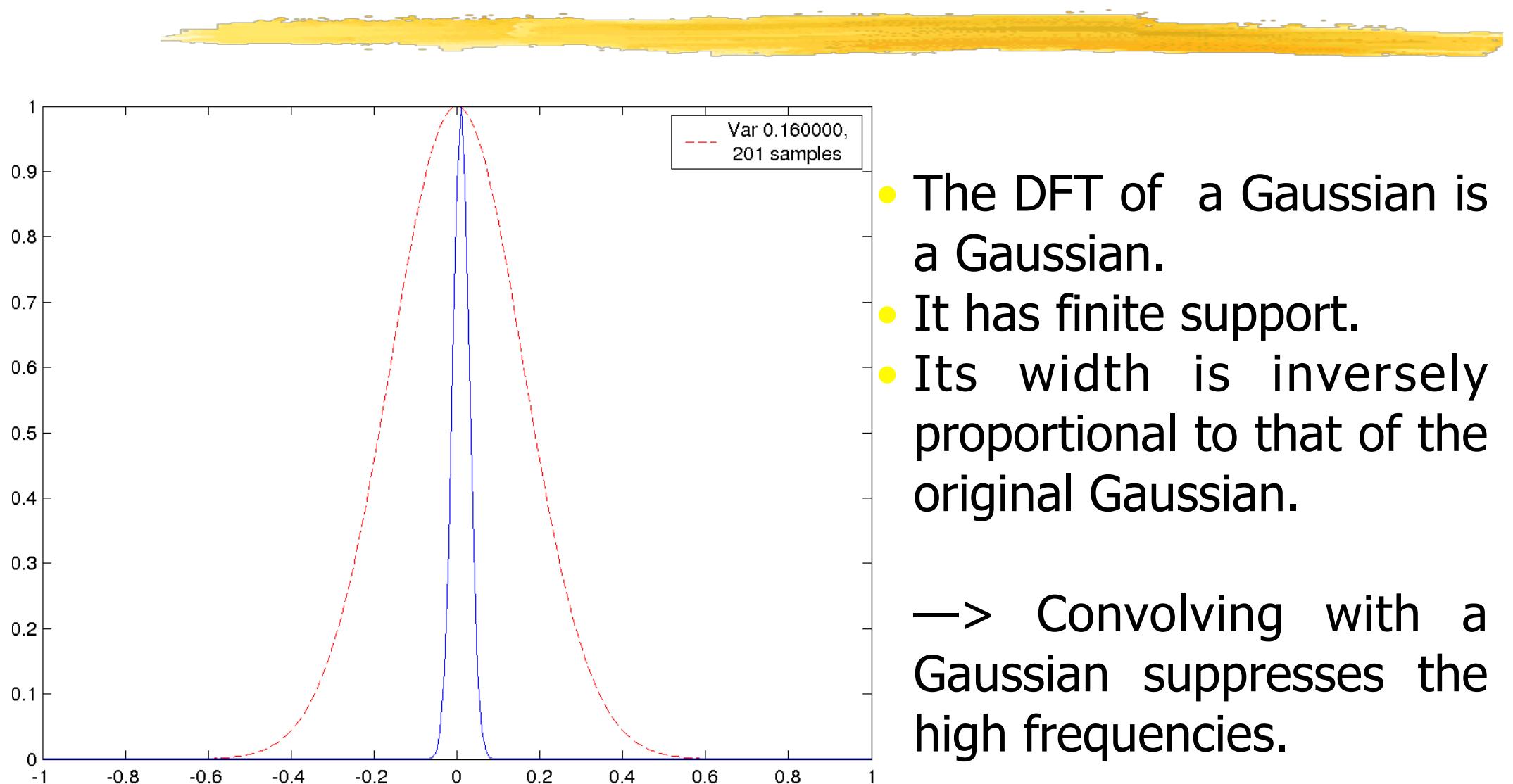
$$\sigma = 1$$



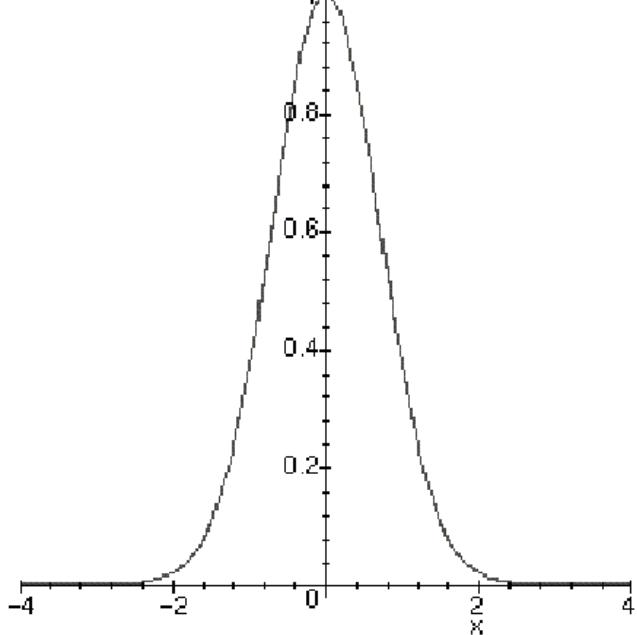
$$\sigma = 2$$

$$g_2(x, y) = \frac{1}{2\pi\sigma^2} \exp(-(x^2 + y^2)/2\sigma^2)$$

FOURIER TRANSFORM



SEPARABILITY

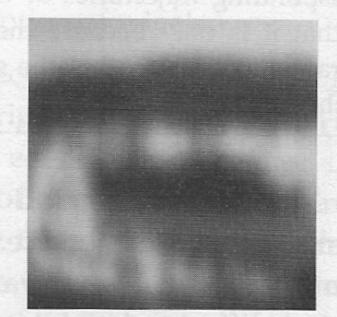
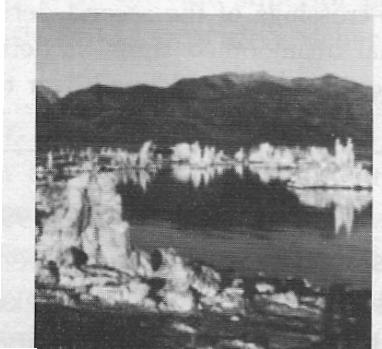
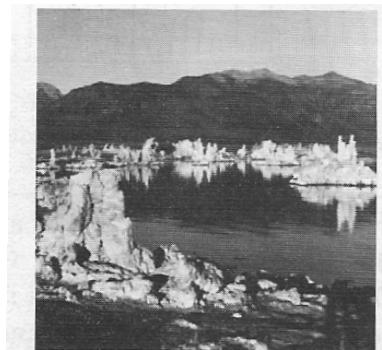
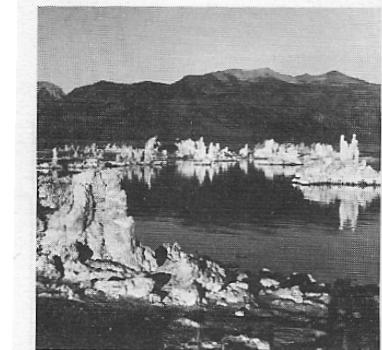


$$g_1(x) = \frac{1}{\sqrt{\pi}\sigma} \exp(-x^2/\sigma^2)$$

$$g_2(x, y) = g_1(x)g_1(y)$$

$$\begin{aligned} \iint_{uv} g_2(u, v) f(x-u, y-v) du dv &= \int_u g_1(u) \left(\int_v g_1(v) f(x-u, y-v) dv \right) du \\ &= \int_v g_1(v) \left(\int_u g_1(u) f(x-u, y-v) du \right) dv \end{aligned}$$

SMOOTHED IMAGES



GAUSSIAN DERIVATIVES

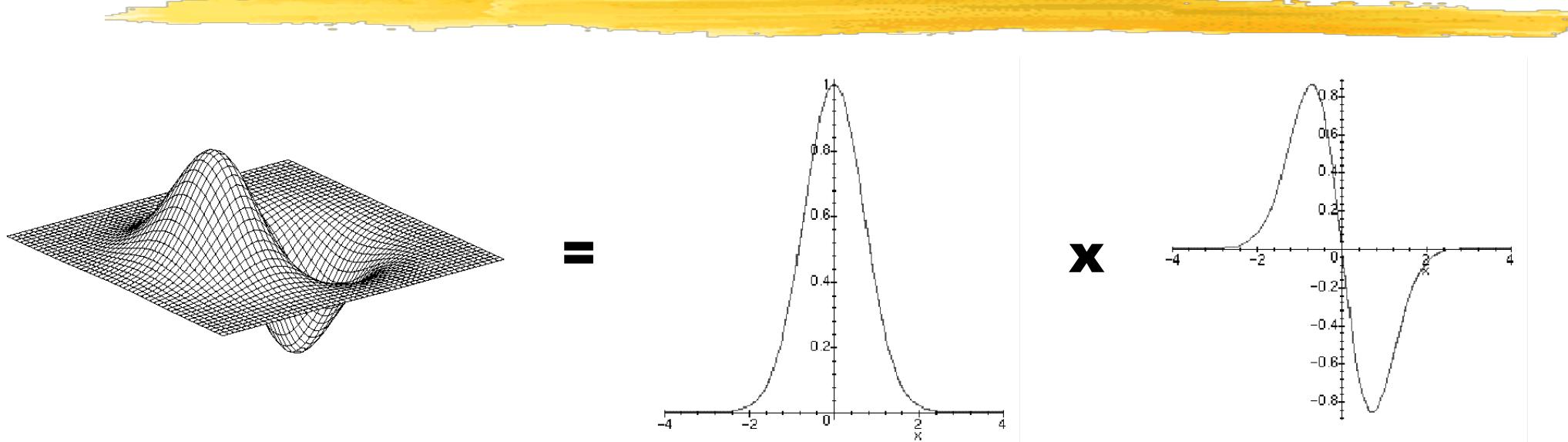
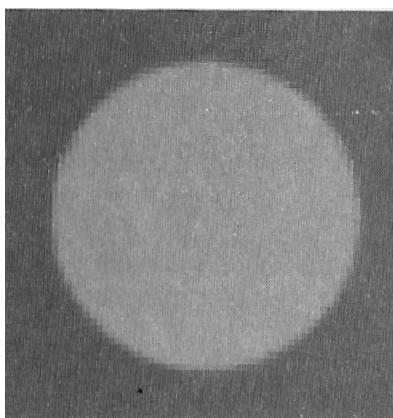


Image derivatives computed by convolving
with the derivative of a Gaussian:

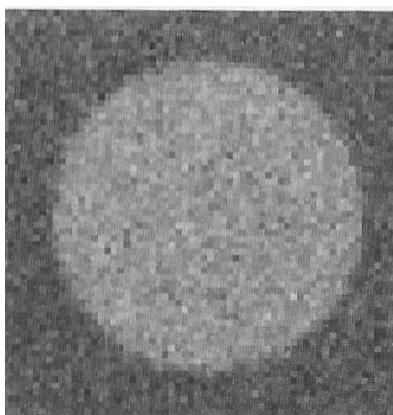
$$\frac{\partial}{\partial x} \iint_{u,v} g_2(u,v) f(x-u, y-v) du dv = \int_u g_1'(u) \left(\int_v g_1(v) f(x-u, y-v) dv \right) du$$

$$\frac{\partial}{\partial y} \iint_{u,v} g_2(u,v) f(x-u, y-v) du dv = \int_v g_1'(v) \left(\int_u g_1(u) f(x-u, y-v) du \right) dv$$

INCREASING SIGMA

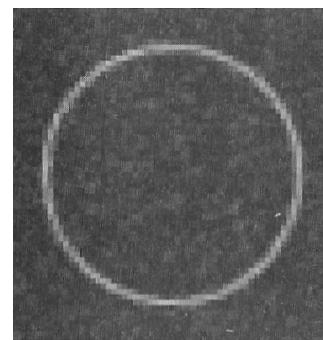


No Noise

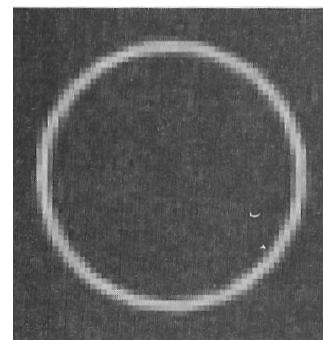


Noise Added

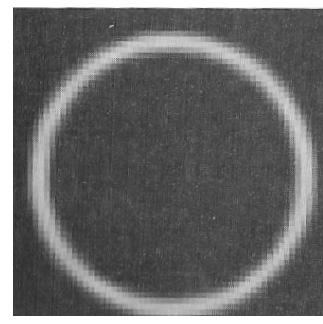
$\sigma=1$



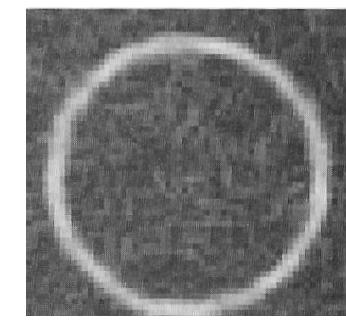
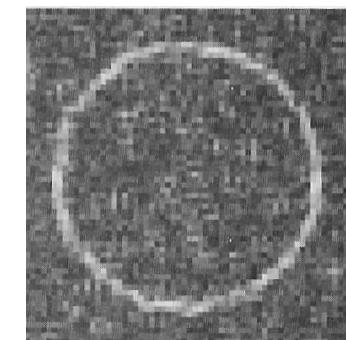
$\sigma=2$



$\sigma=4$



No Noise



Noise Added

GAUSSIAN MASKS



Sigma=1:

g : 0.000070 0.010332 0.207532 0.564131 0.207532 0.010332 0.000070

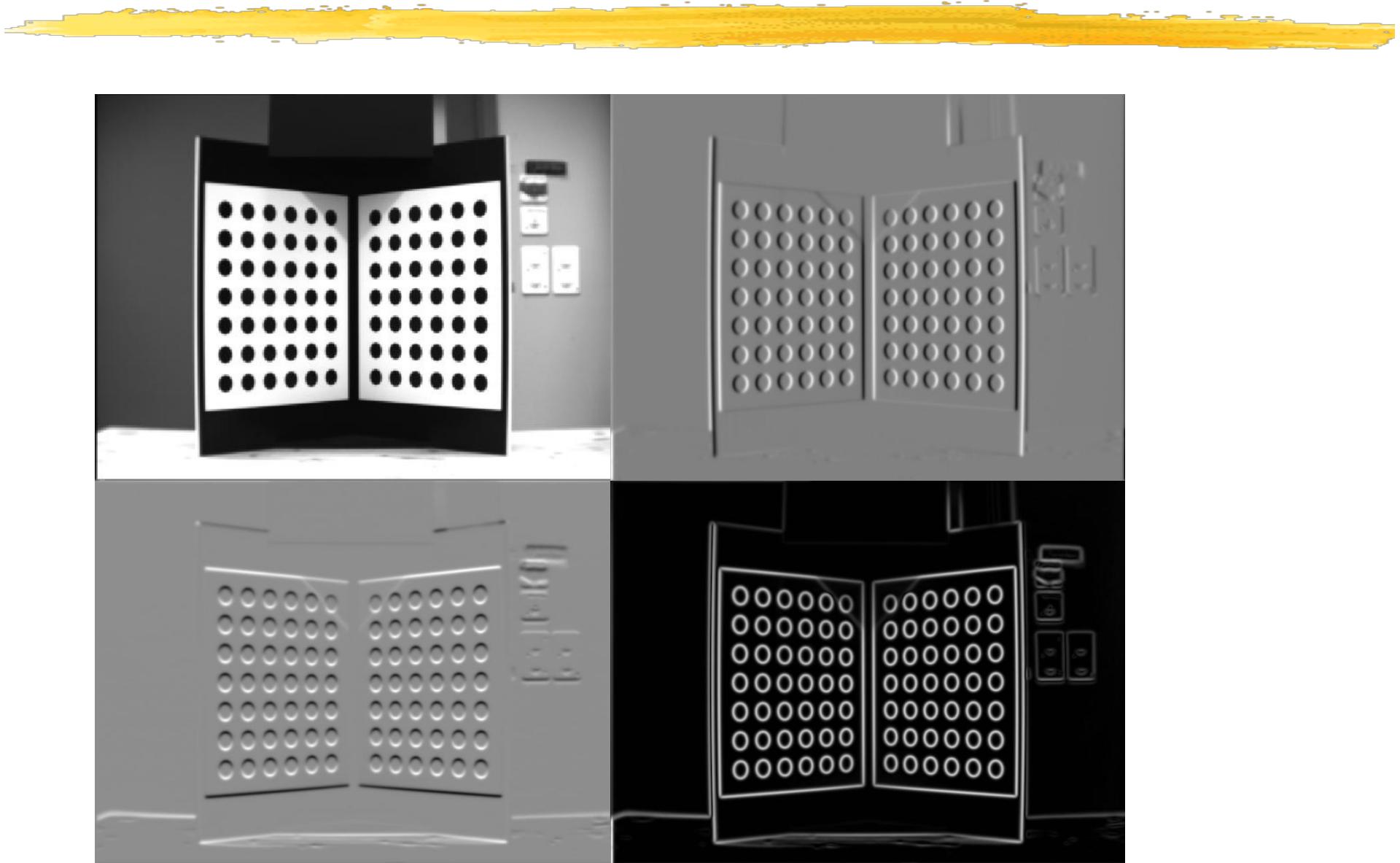
g' : 0.000418 0.041330 0.415065 0.000000 -0.415065 -0.041330 -0.000418

Sigma=2:

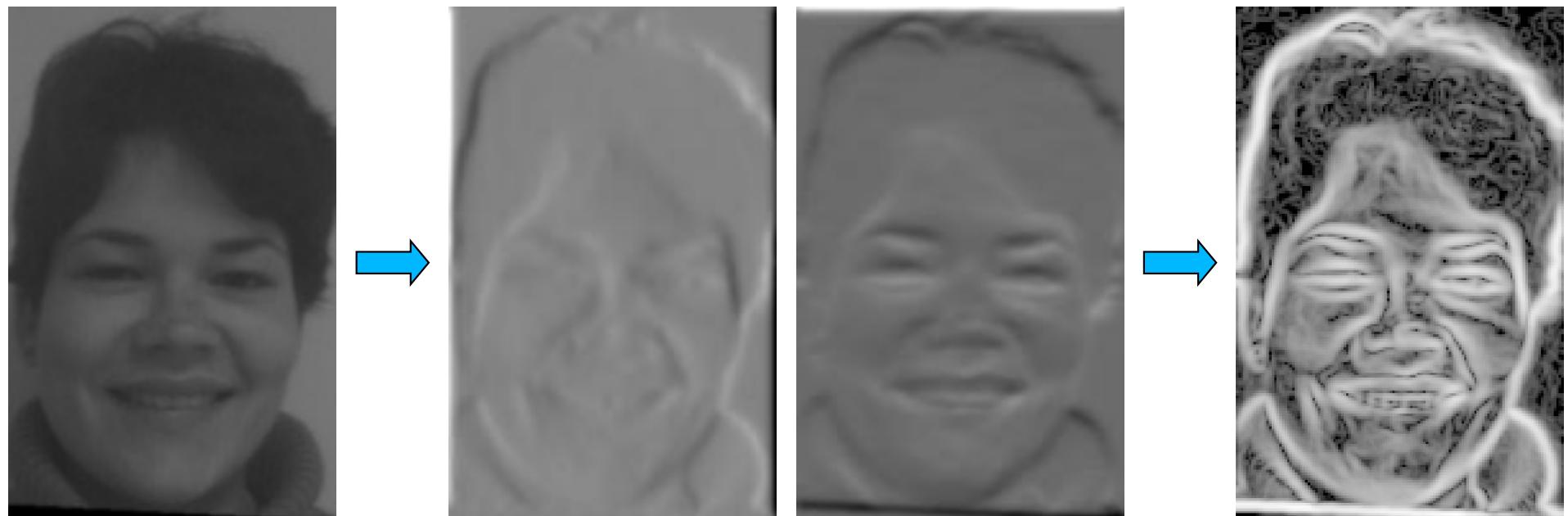
g : 0.005167 0.029735 0.103784 0.219712 0.282115 0.219712 0.103784 0.029735 0.005167

g' : 0.010334 0.044602 0.103784 0.109856 0.000000 -0.109856 -0.103784 -0.044602 -0.010334

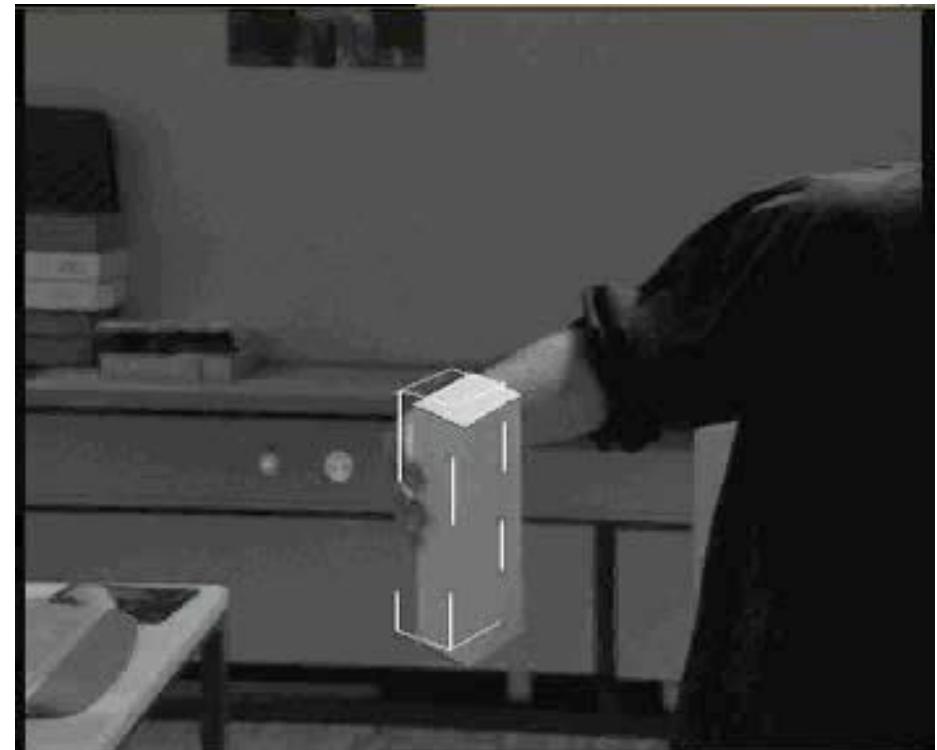
DERIVATIVE IMAGES



DERIVATIVE IMAGES

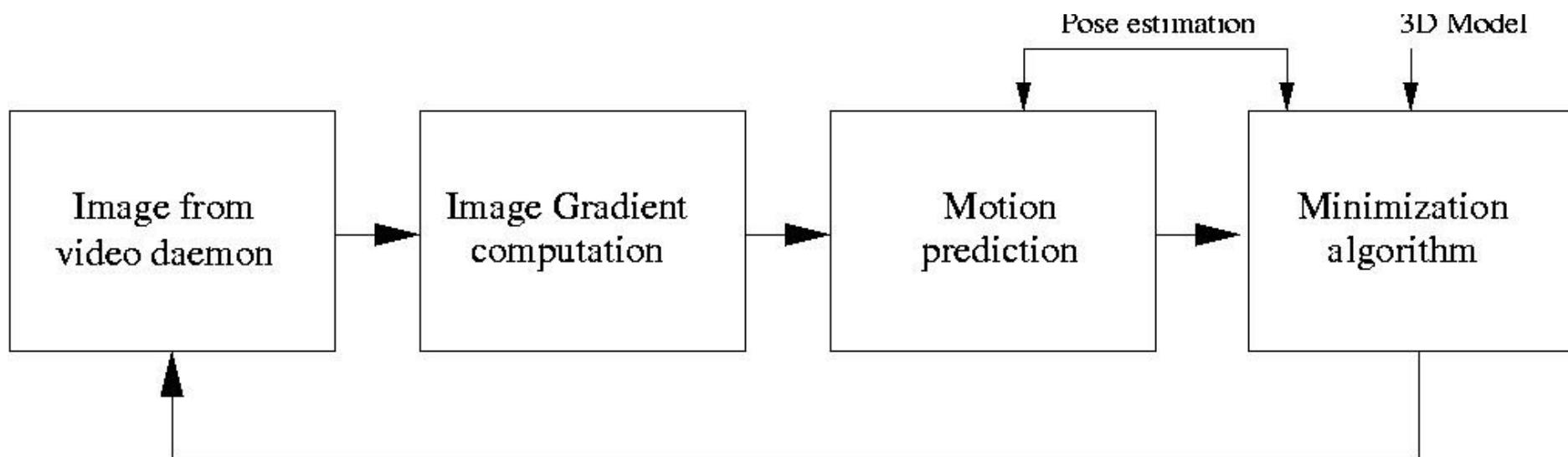
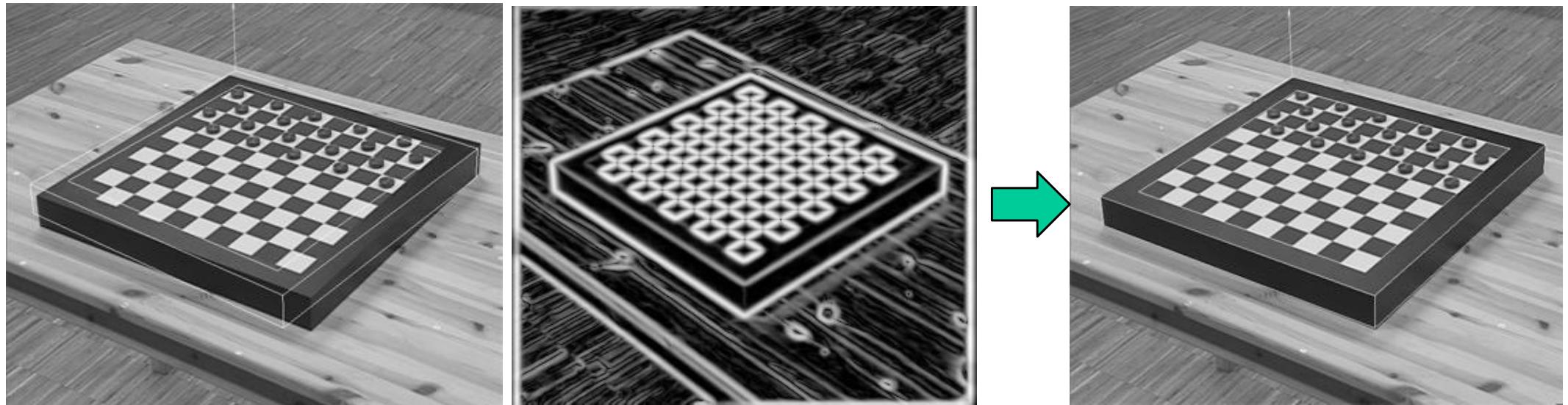


GRADIENT-BASED TRACKING



Maximize edge-strength along projection of the 3—D wireframe.

GRADIENT MAXIMIZATION

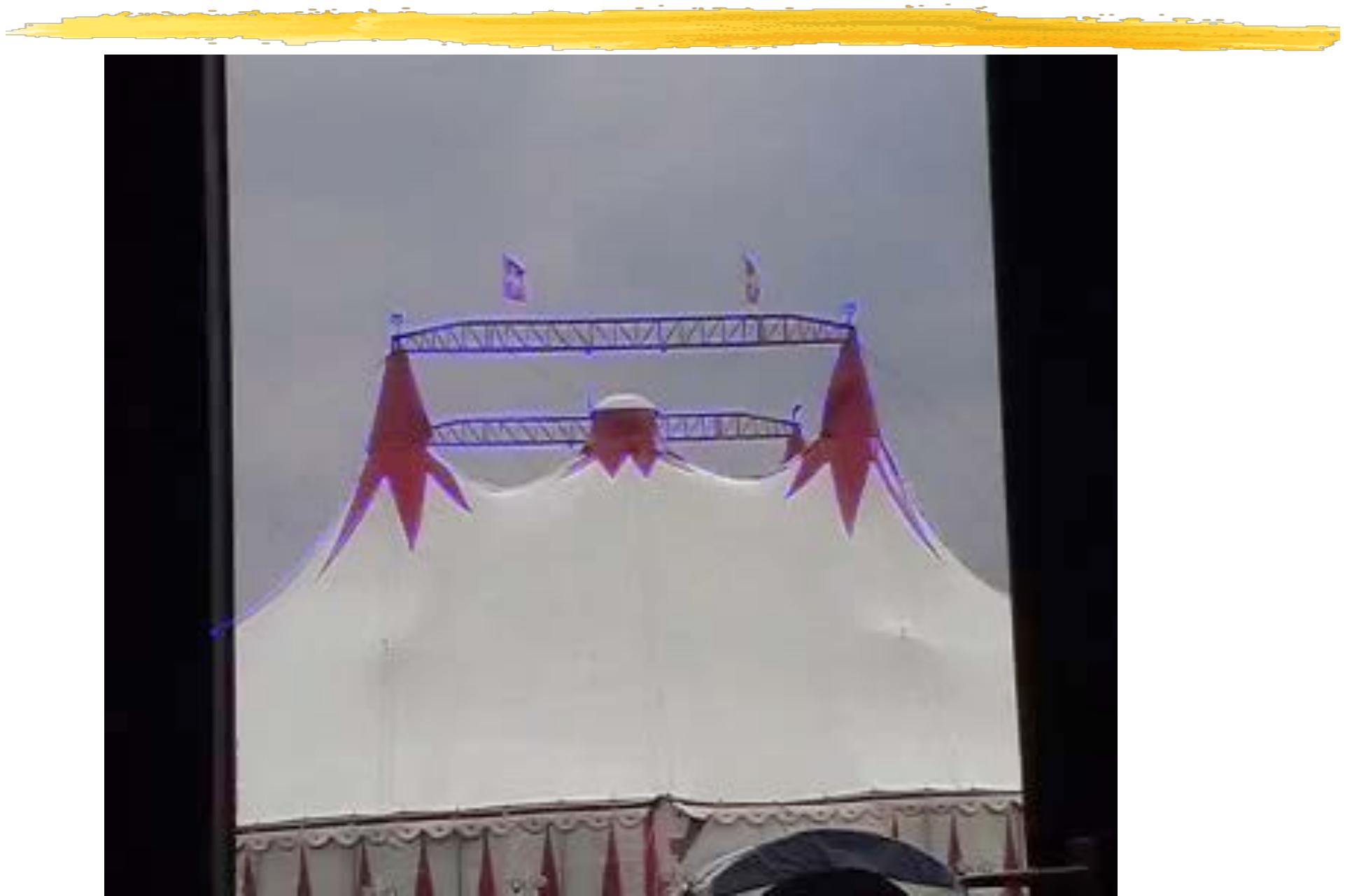


WING DEFORMATION

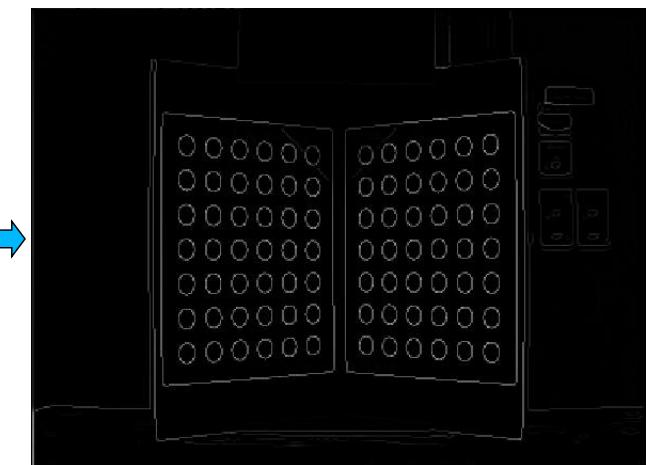
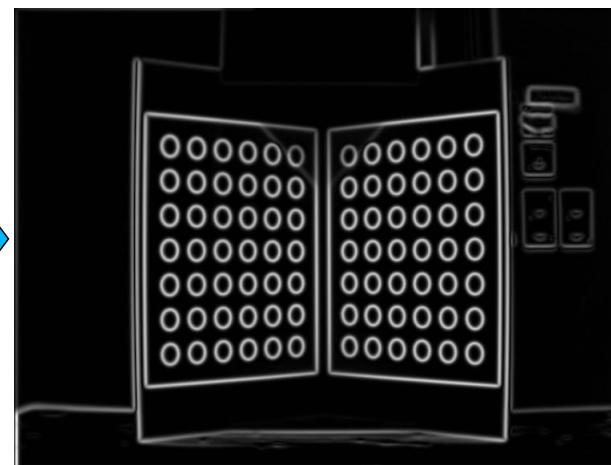
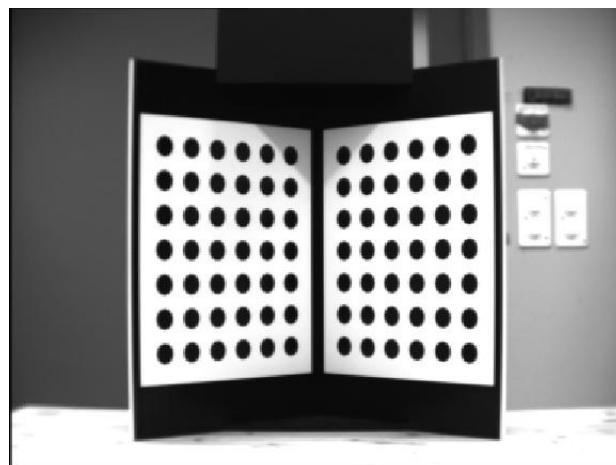


- Measure true behavior in flight.
- Validate computational models.

REAL-TIME TRACKING



CANNY EDGE DETECTOR



CANNY EDGE DETECTOR



Convolution

- Gradient strength
- Gradient direction

Thresholding

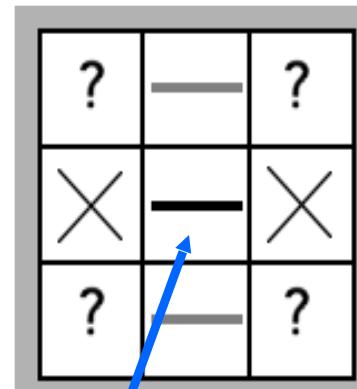
Non Maxima Suppression

Hysteresis Thresholding

NON-MAXIMA SUPPRESSION

In parallel, at each pixel in edge image, use window to select as a function of edge orientation:

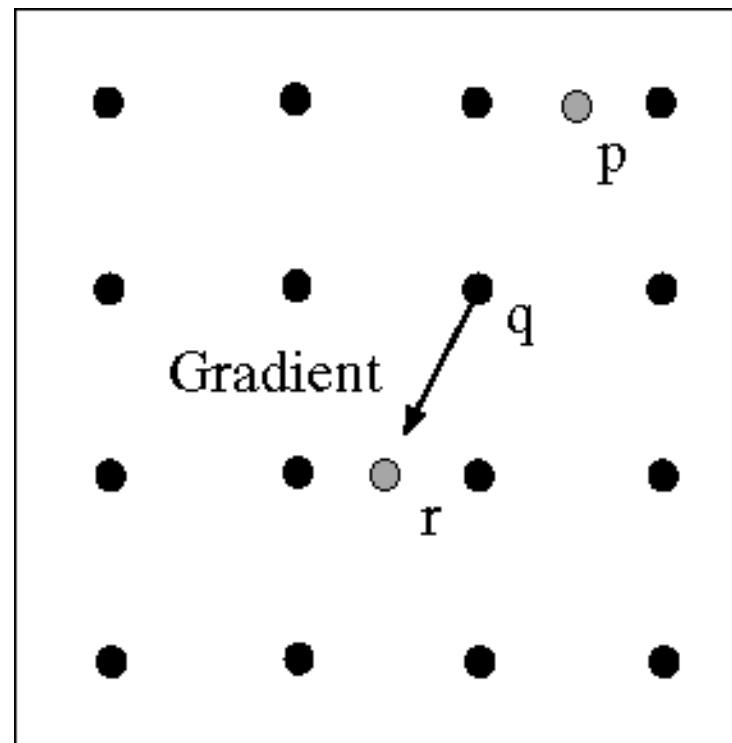
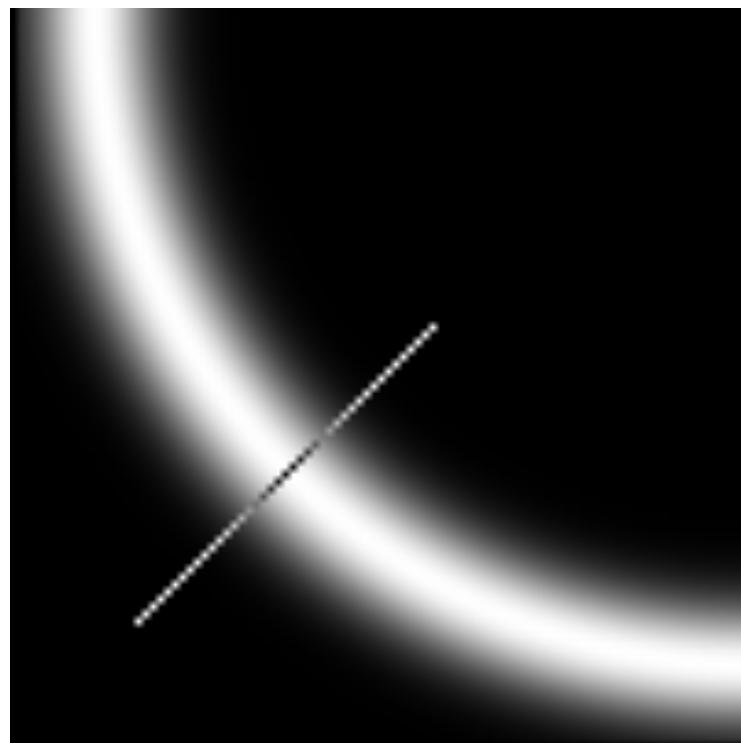
Window W



Central Edge

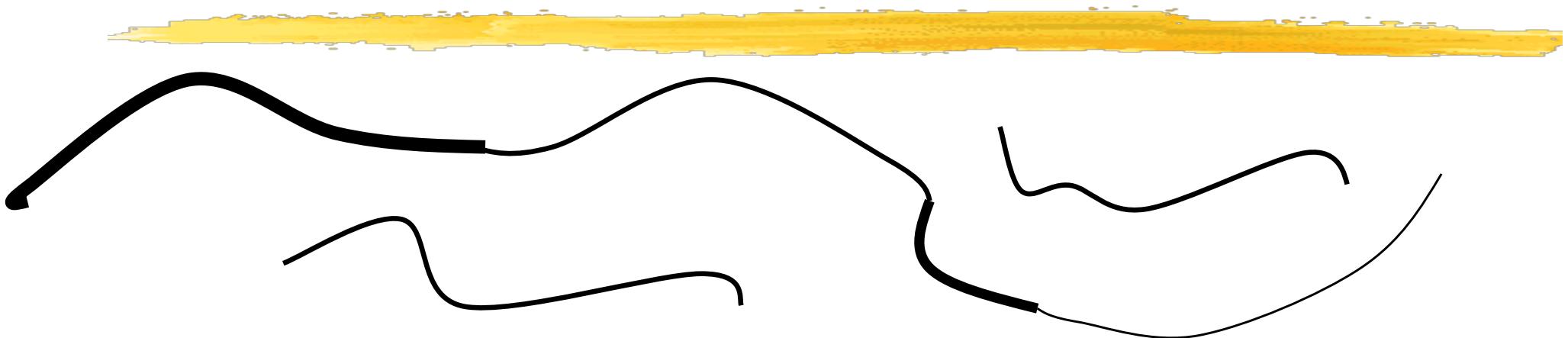
- Definitely consider these edges
- X Do not consider these edges
- ? Maybe consider them, depending on algorithm

NON-MAXIMA SUPPRESSION



Check if pixel is local maximum along gradient direction,
which requires checking interpolated pixels p and r .

HYSTERESIS THRESHOLDING



Algorithm takes two thresholds: high & low

- A pixel with edge strength above high threshold is an edge.
- Any pixel with edge strength below low threshold is not.
- Any pixel above the low threshold and next to an edge is an edge.

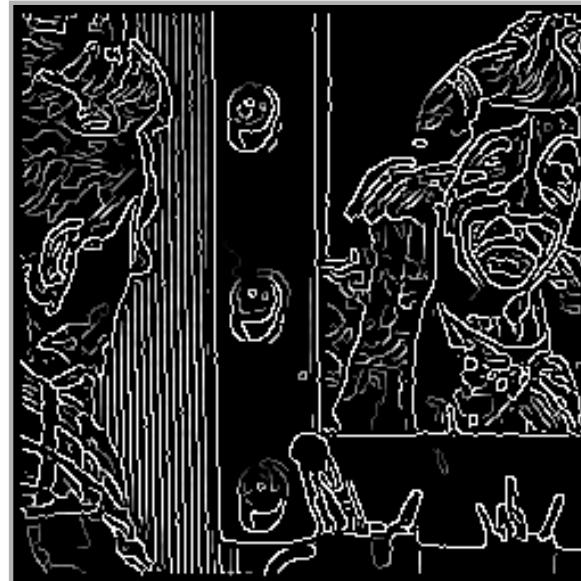
Iteratively label edges

- Edges grow out from 'strong edges'
- Iterate until no change in image.

CANNY RESULTS



$\sigma=1$, $T_2=255$, $T_1=220$



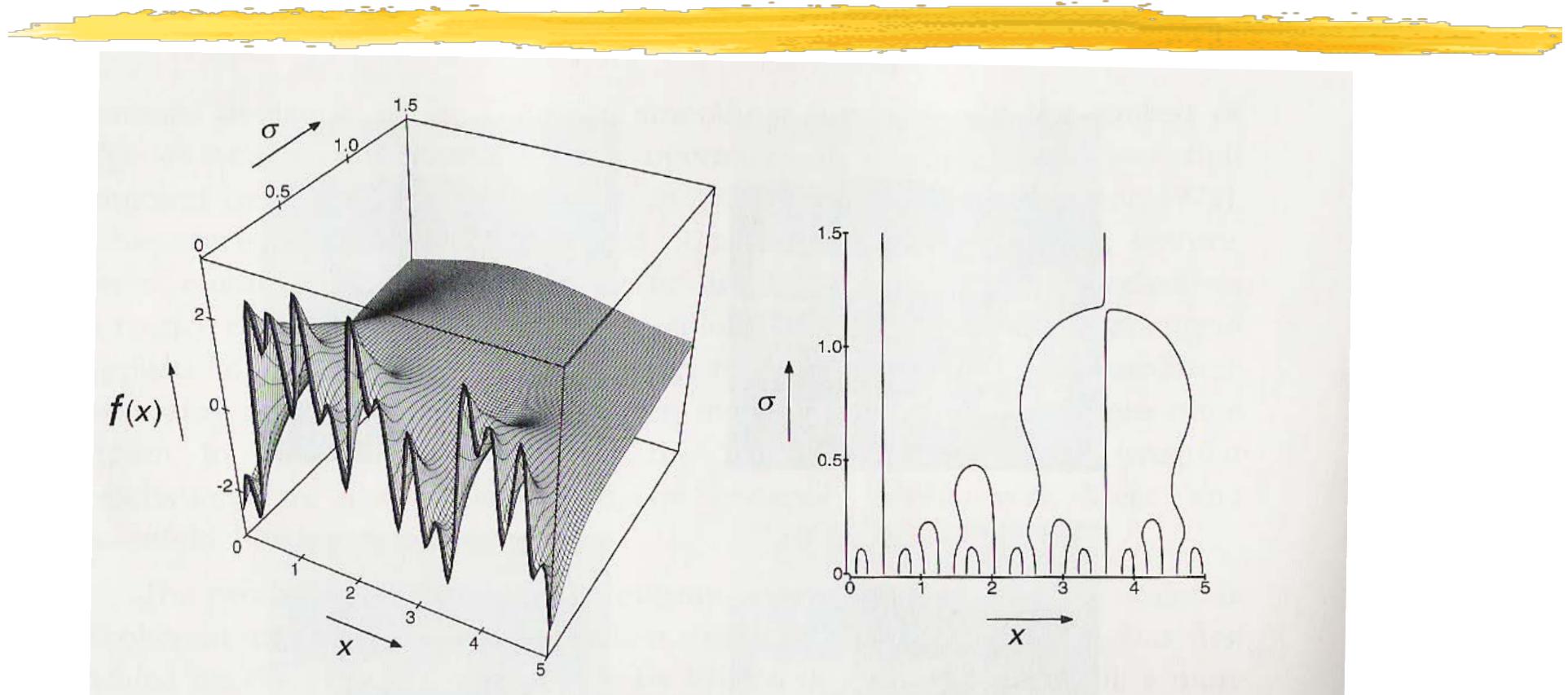
$\sigma=1$, $T_2=128$, $T_1=1$



$\sigma=2$, $T_2=128$, $T_1=1$

M. Heath, S. Sarkar, T. Sanocki, and K.W. Bowyer, "A Robust Visual Method for Assessing the Relative Performance of Edge-Detection Algorithms" IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 19, No. 12, December 1997, pp. 1338-1359.

SCALE SPACE



Increasing scale (σ) removes details but never adds new ones:

- Edge position may shift.
- Two edges may merge.
- An edge may **not** split into two.

MULTIPLE SCALES



$\sigma = 1$



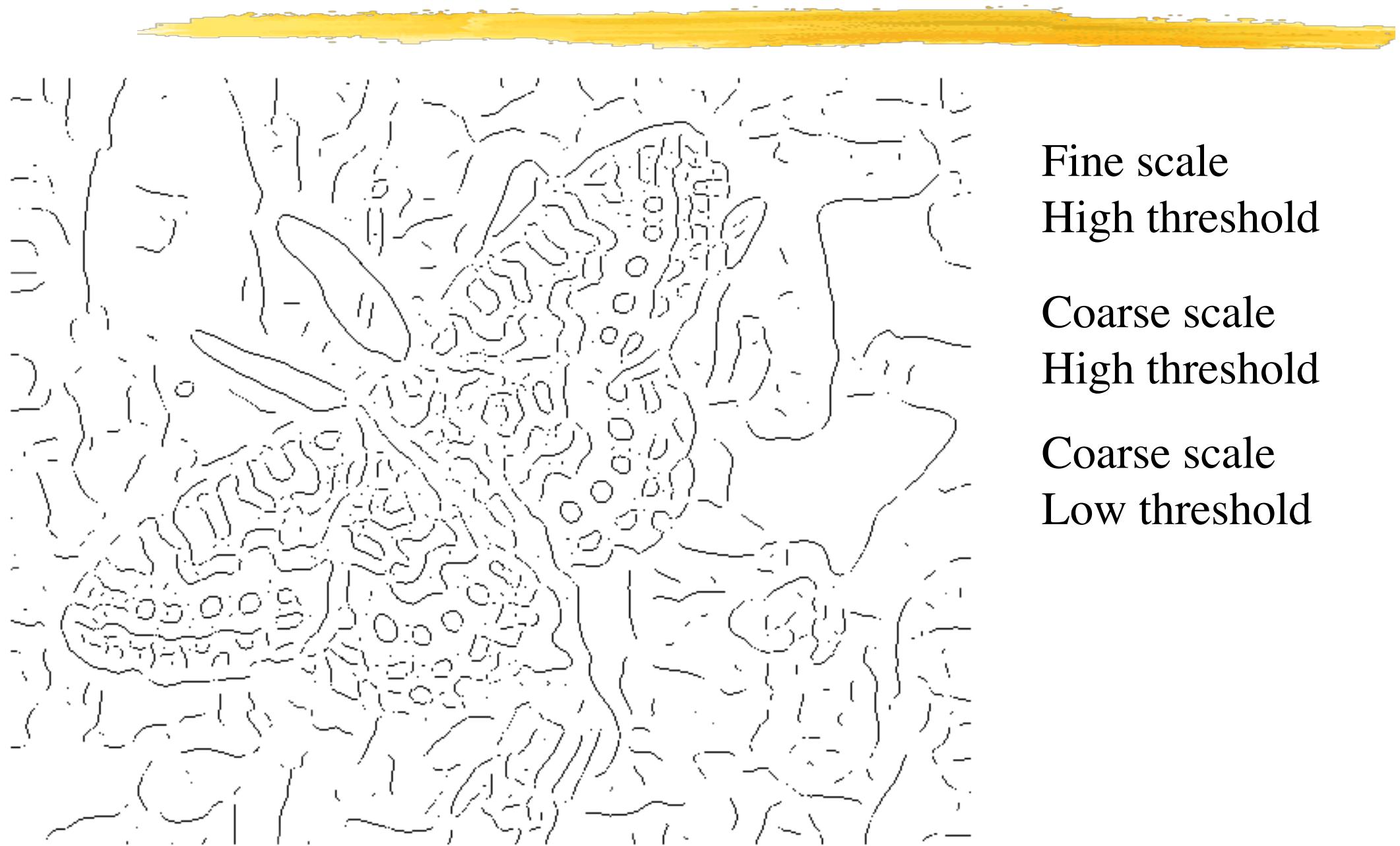
$\sigma = 2$



$\sigma = 4$

→ Choosing the right scale is a difficult semantic problem.

SCALE vs THRESHOLD

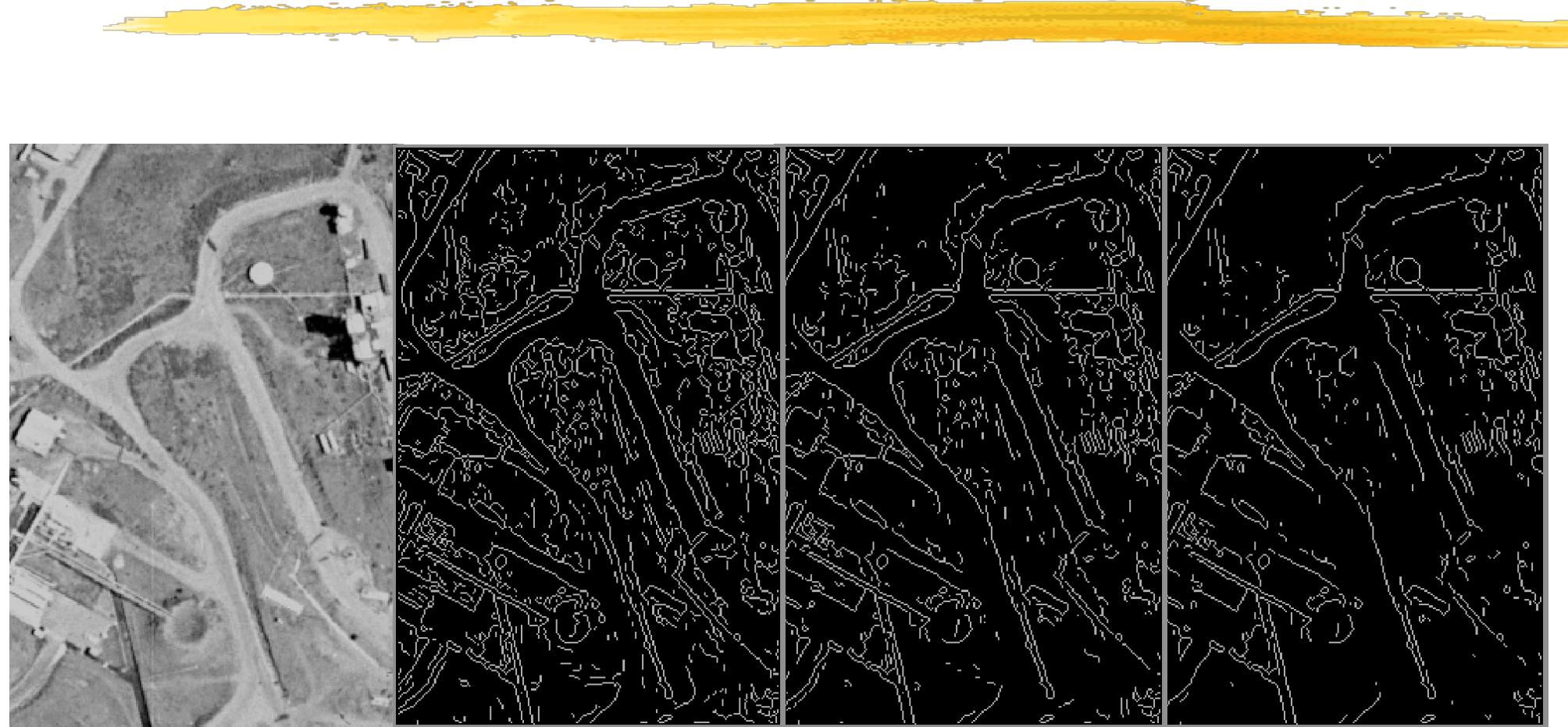


Fine scale
High threshold

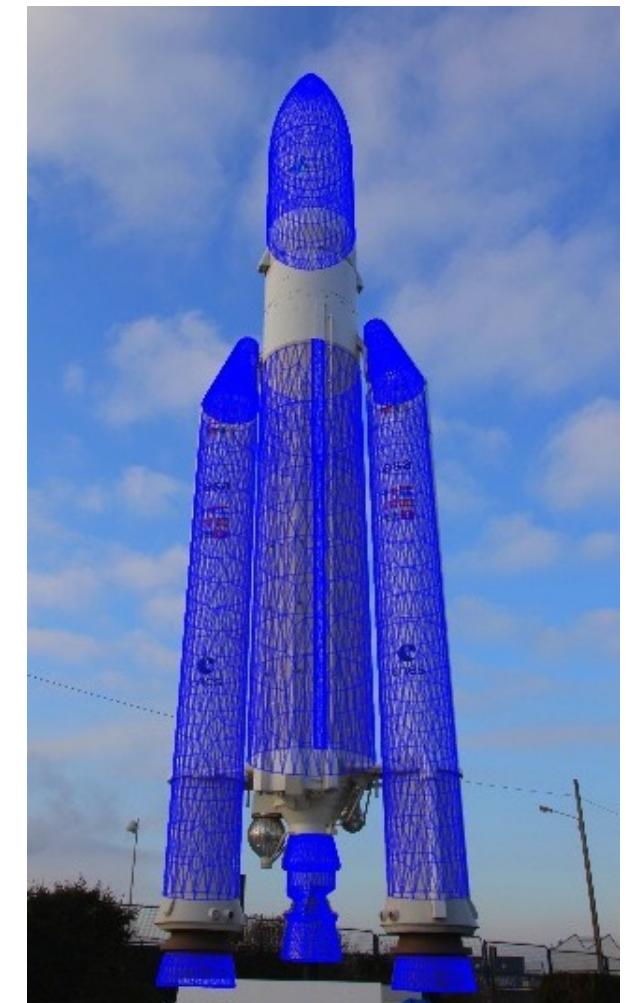
Coarse scale
High threshold

Coarse scale
Low threshold

ROAD IMAGE



TRACKING ARIANE



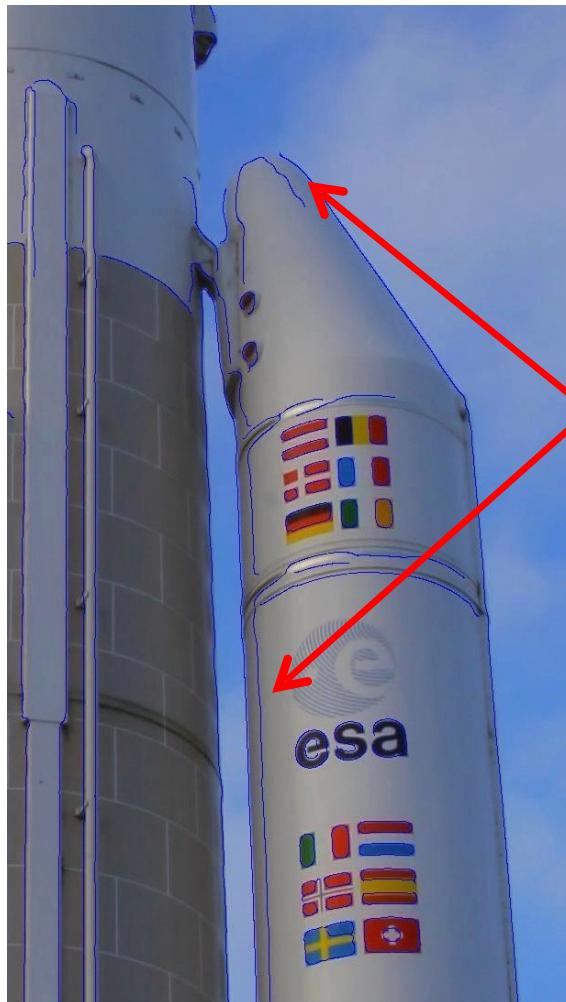
RAPID TRACKER



Given an initial pose estimate:

- Estimate the occluding contours.
- Find closest edge points in the normal direction.
- Re-estimate pose to minimize distances in the least squares sense.
- Iterate until convergence.

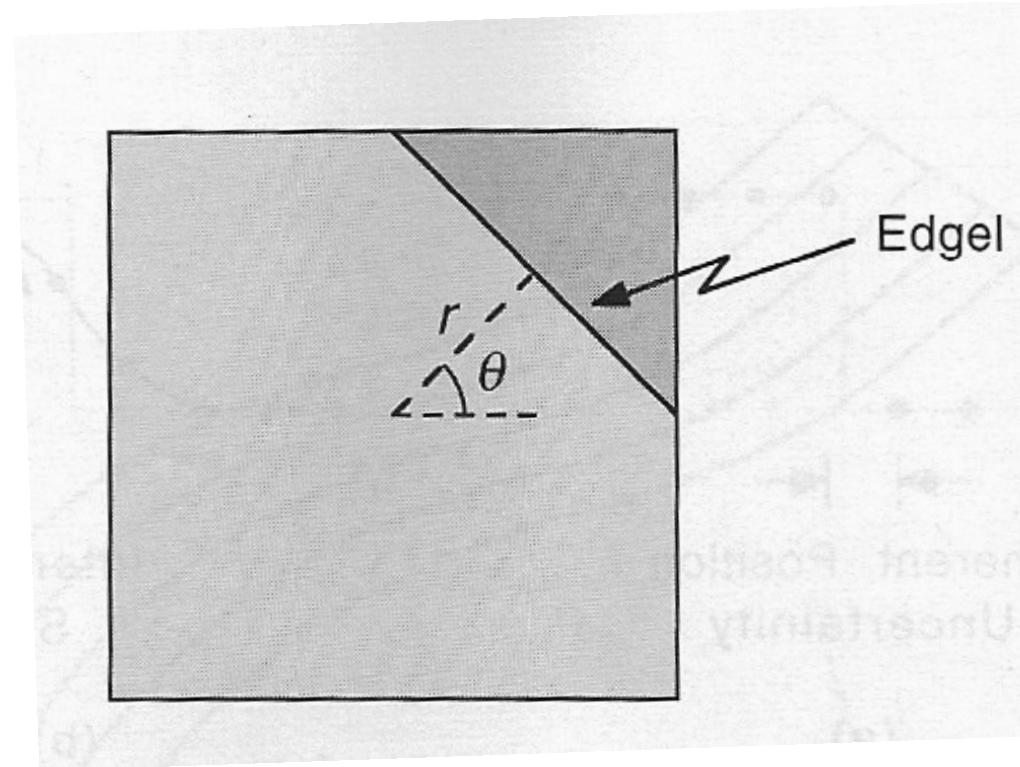
STEEP SMOOTH SHADING



- Rapidly varying gray levels.
- Large gradients.

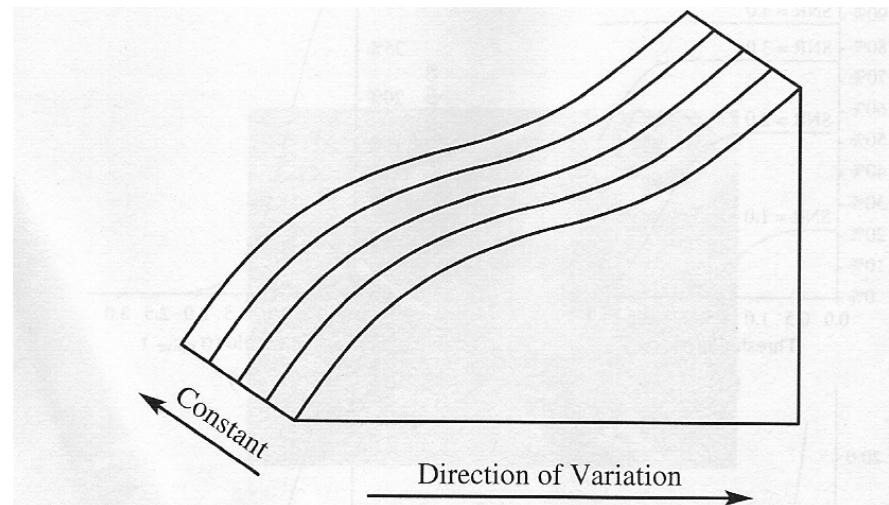
→ Shading can produce spurious edges.

PARAMETRIC MODEL MATCHERS



→ 4 parameters model to be fit in the least squares sense.

SURFACE FITTING



1. Estimate the edgel direction by fitting a cubic surface.
2. Fit a 1-D surface in the direction of the edgel
 - Step shaped surface,
 - Quadratic polynomial.
3. Declare an edge if step shape better than quadratic.

TEXTURE BOUNDARIES

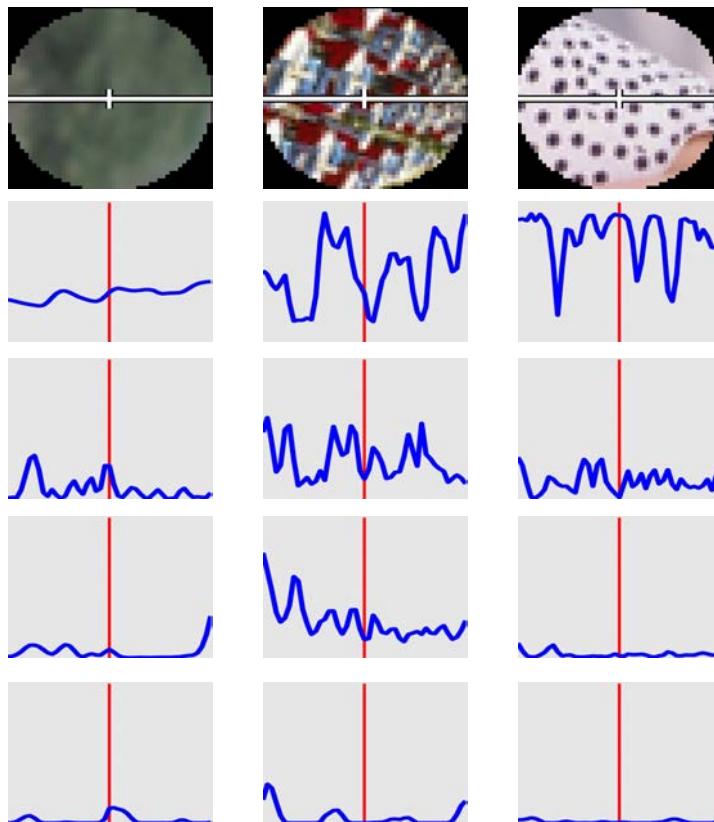


- Not all image contours are characterized by strong contrast.
- Sometimes, textural changes are just as significant.

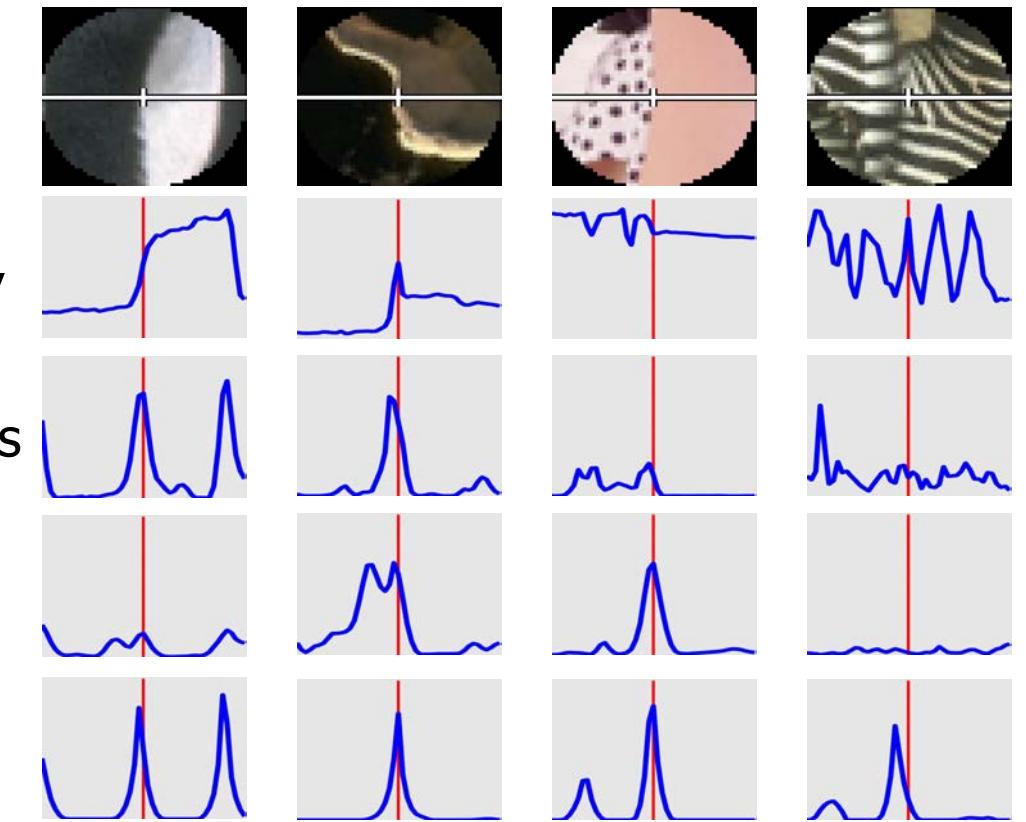
DIFFERENT BOUNDARY TYPES



Non-boundaries



Boundaries

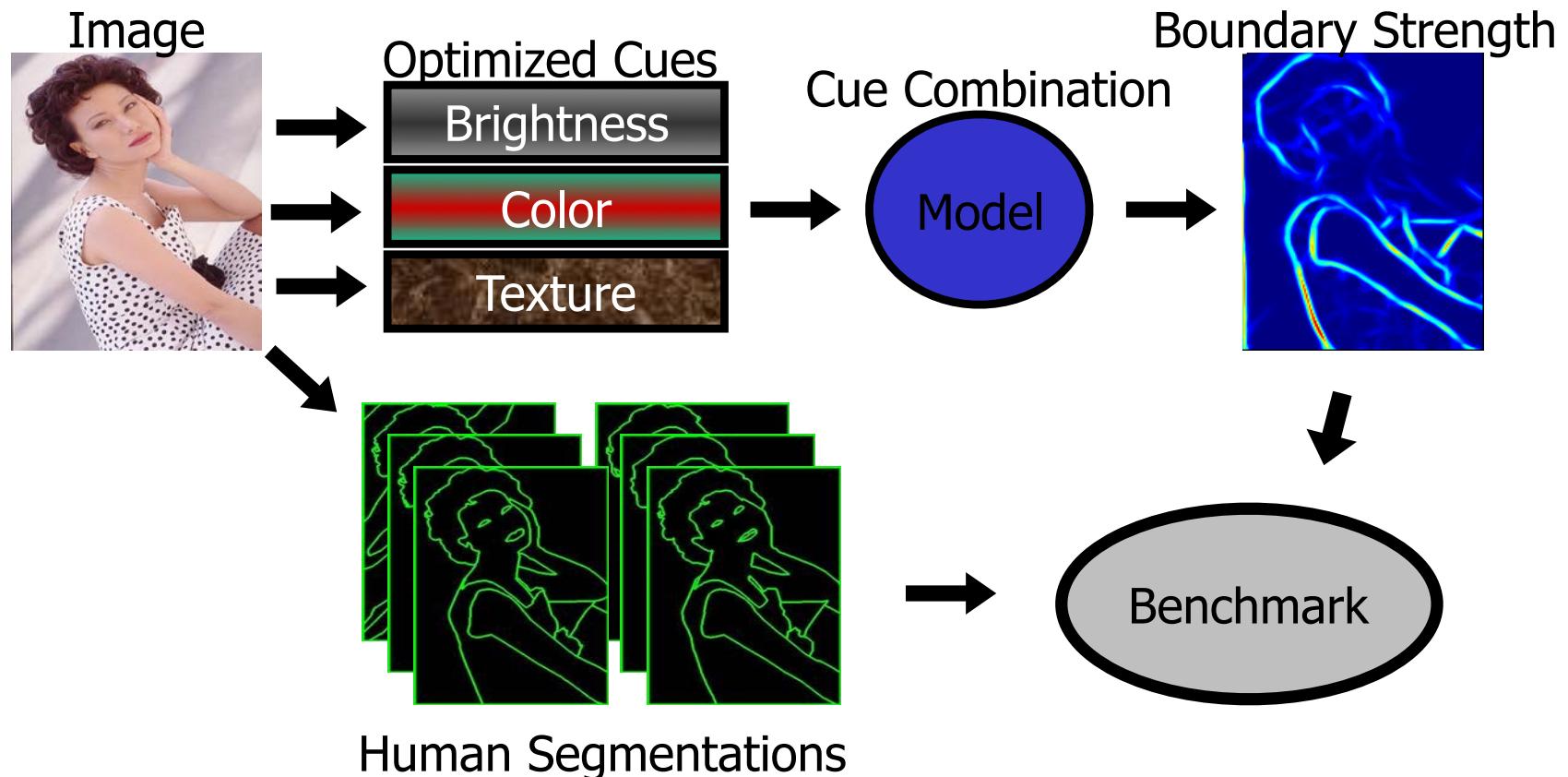


TRAINING DATABASE



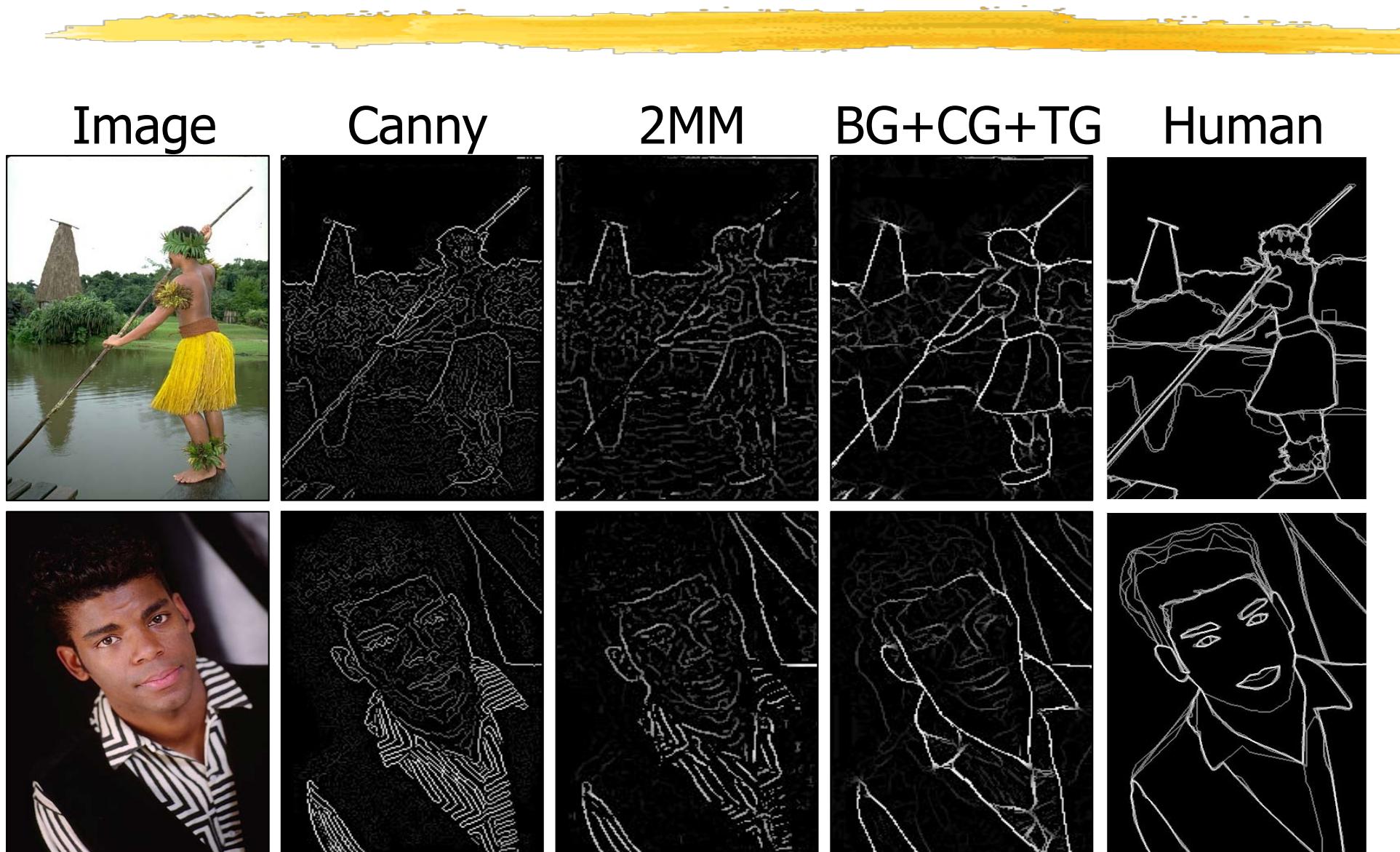
1000 images with 5 to 10 segmentations each.

MACHINE LEARNING

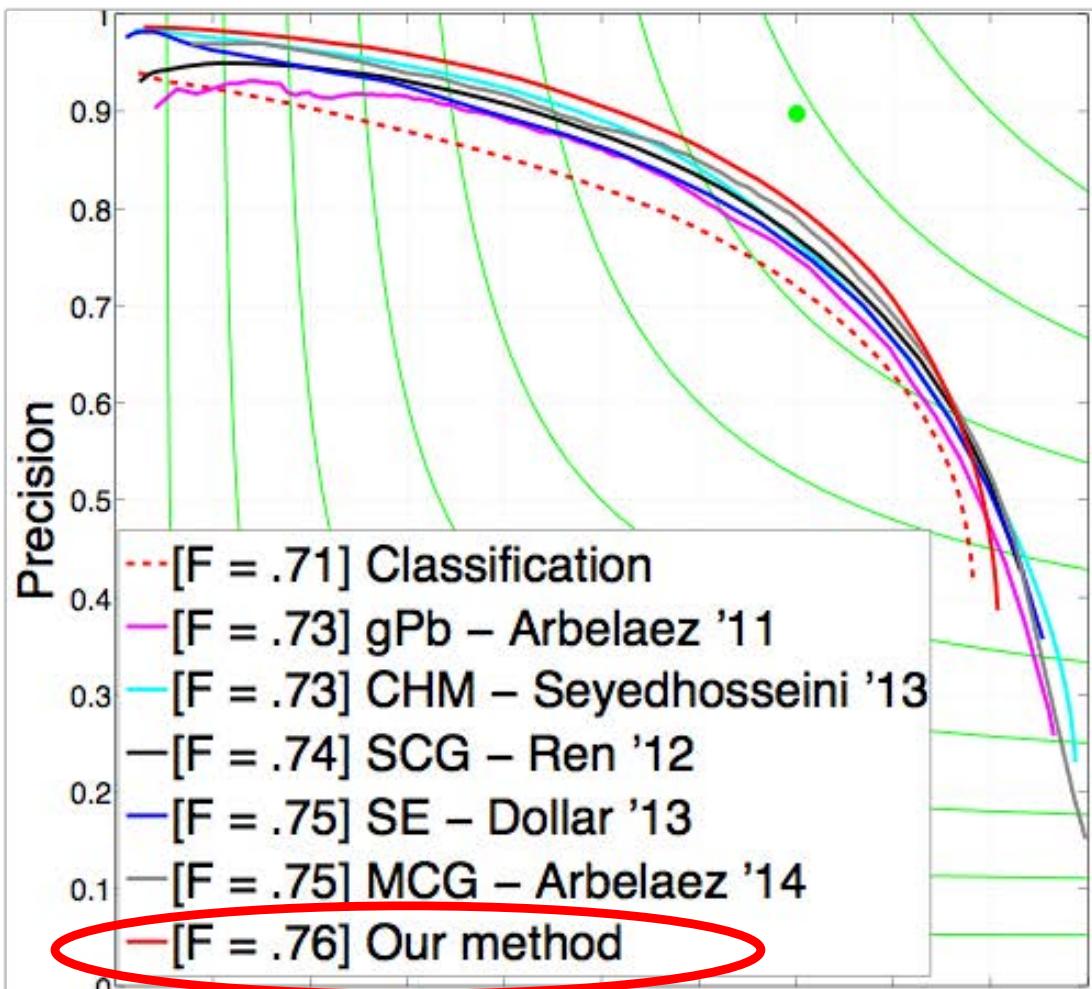


Learn the probability of being a boundary pixel on the basis of a set of features.

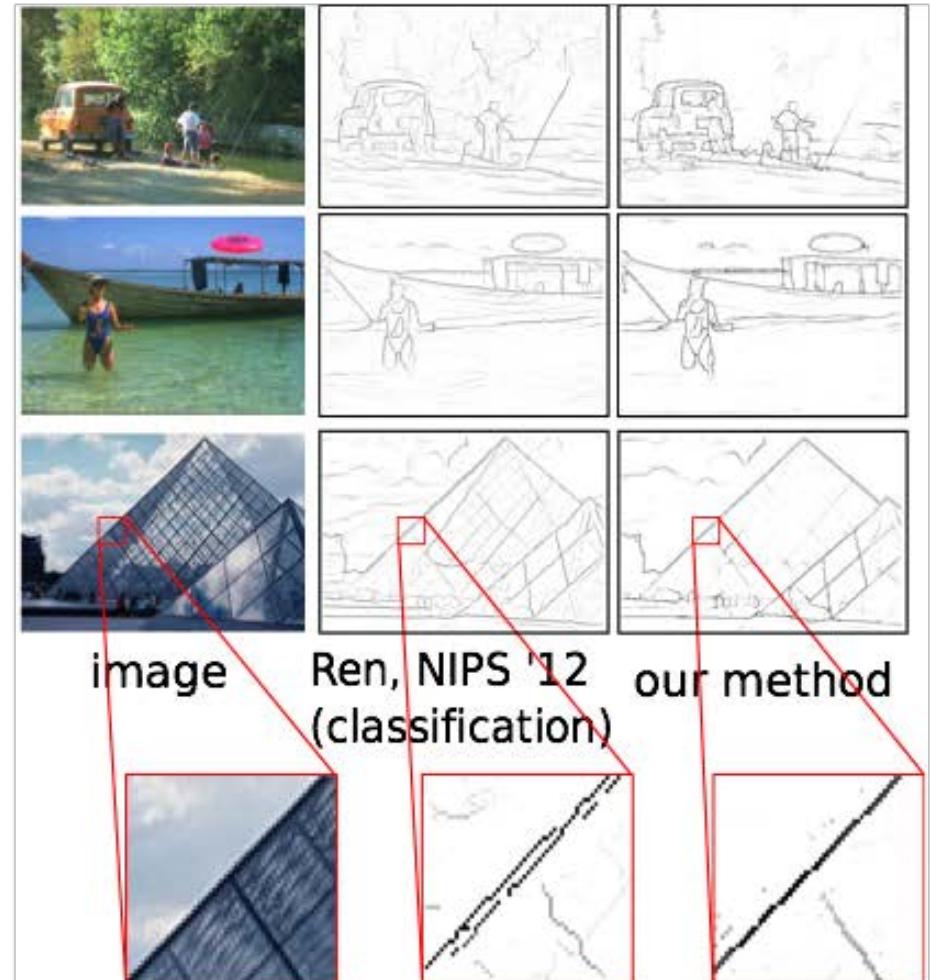
RESULTS



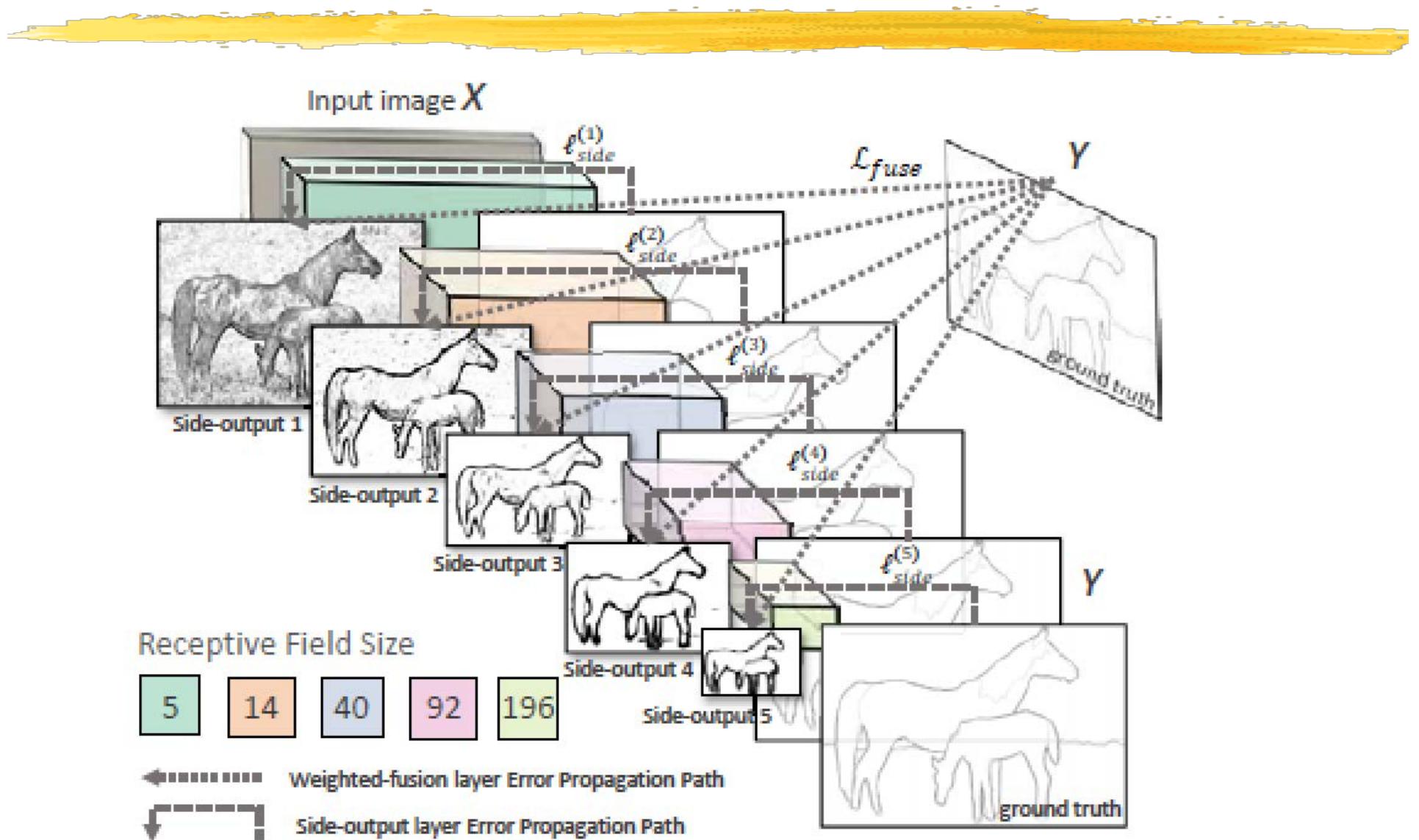
CLASSIFICATION vs REGRESSION



Yes!



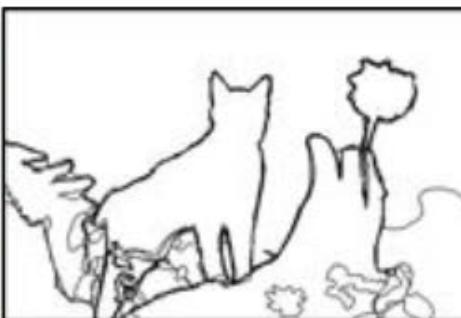
DEEP LEARNING



DEEP LEARNING VS CANNY



(a) original image



(b) ground truth



(c) HED: output



(d) HED: side output 2



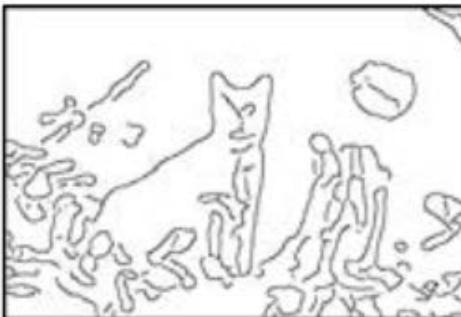
(e) HED: side output 3



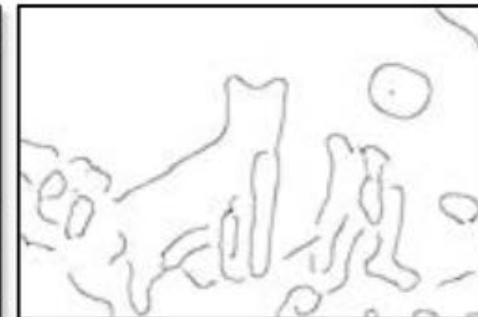
(f) HED: side output 4



(g) Canny: $\sigma = 2$



(h) Canny: $\sigma = 4$

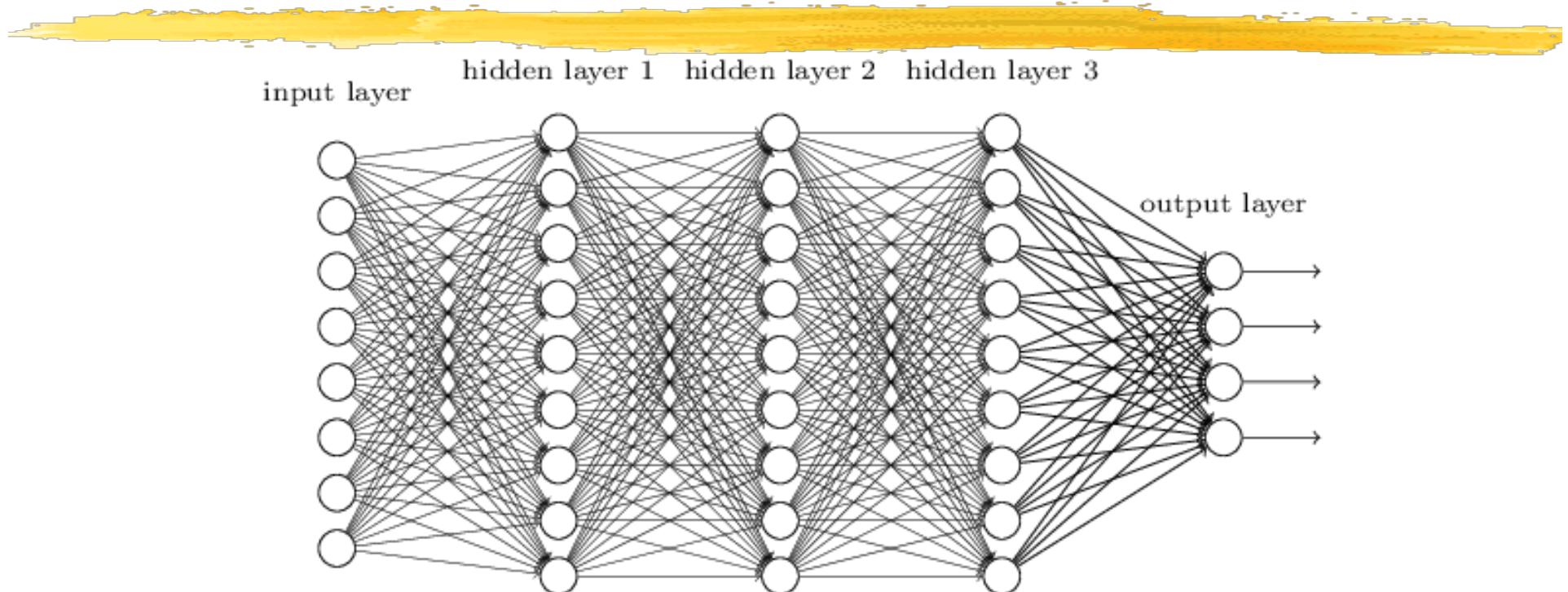


(i) Canny: $\sigma = 8$

DEEPER LEARNING

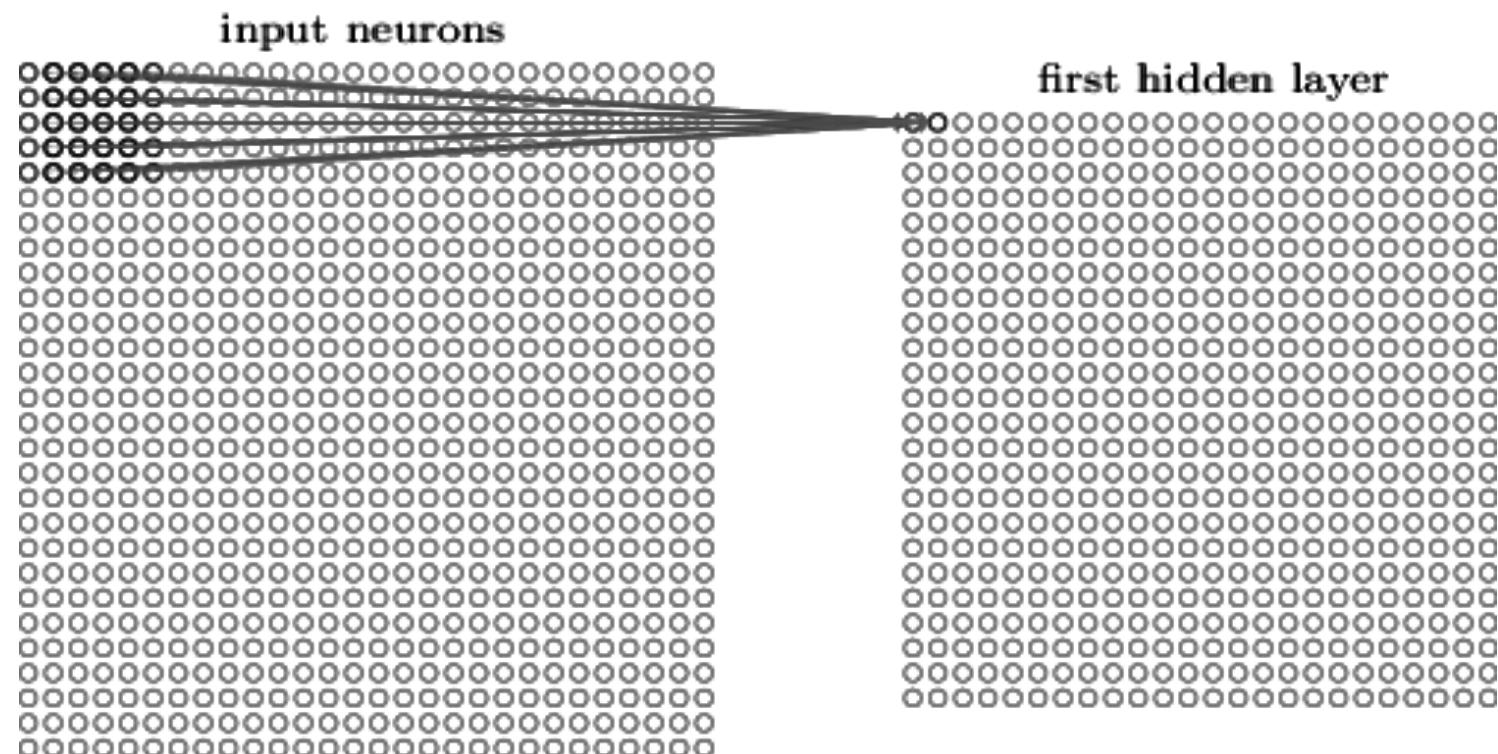


DEEP LEARNING



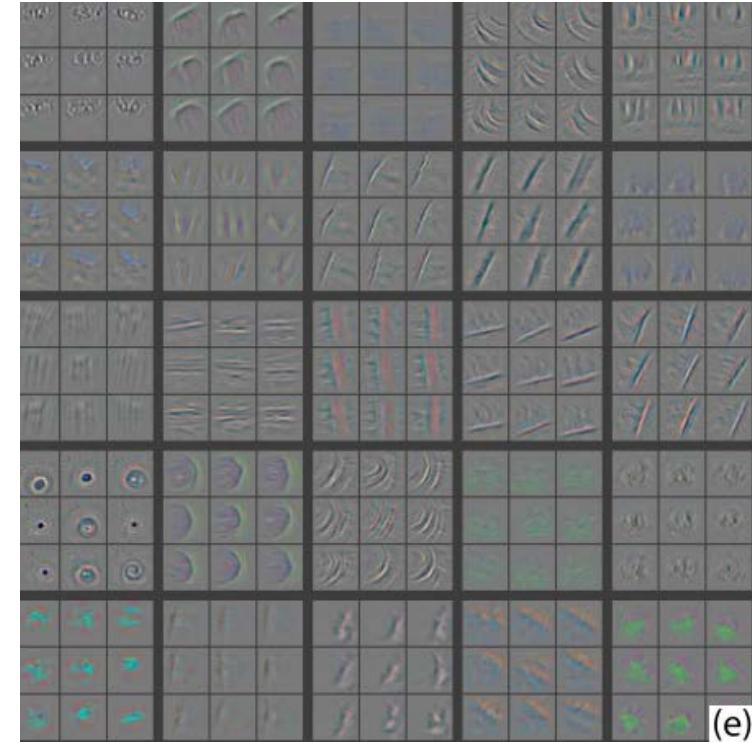
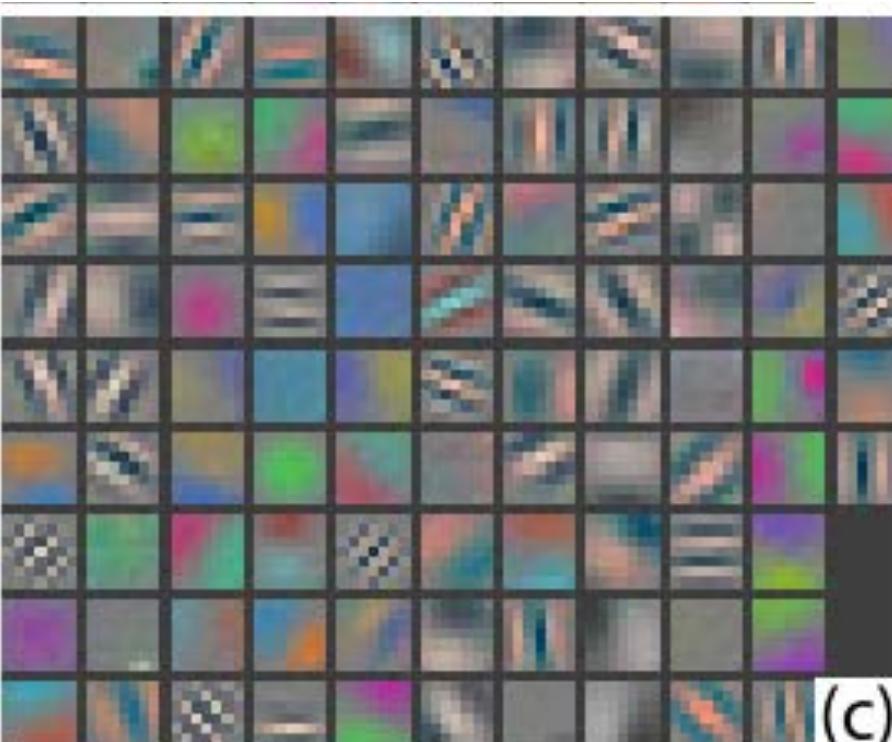
- Store in each node a function of the linear combination of the activations of all nodes in the previous layers.
- The network can be trained to produce a desired output given a specific input by learning the linear combination weights.

CONVOLUTIONAL LAYER



$$\sigma \left(b + \sum_{x=0}^{n_x} \sum_{y=0}^{n_y} w_{i,j} a_{i+x,j+y} \right)$$

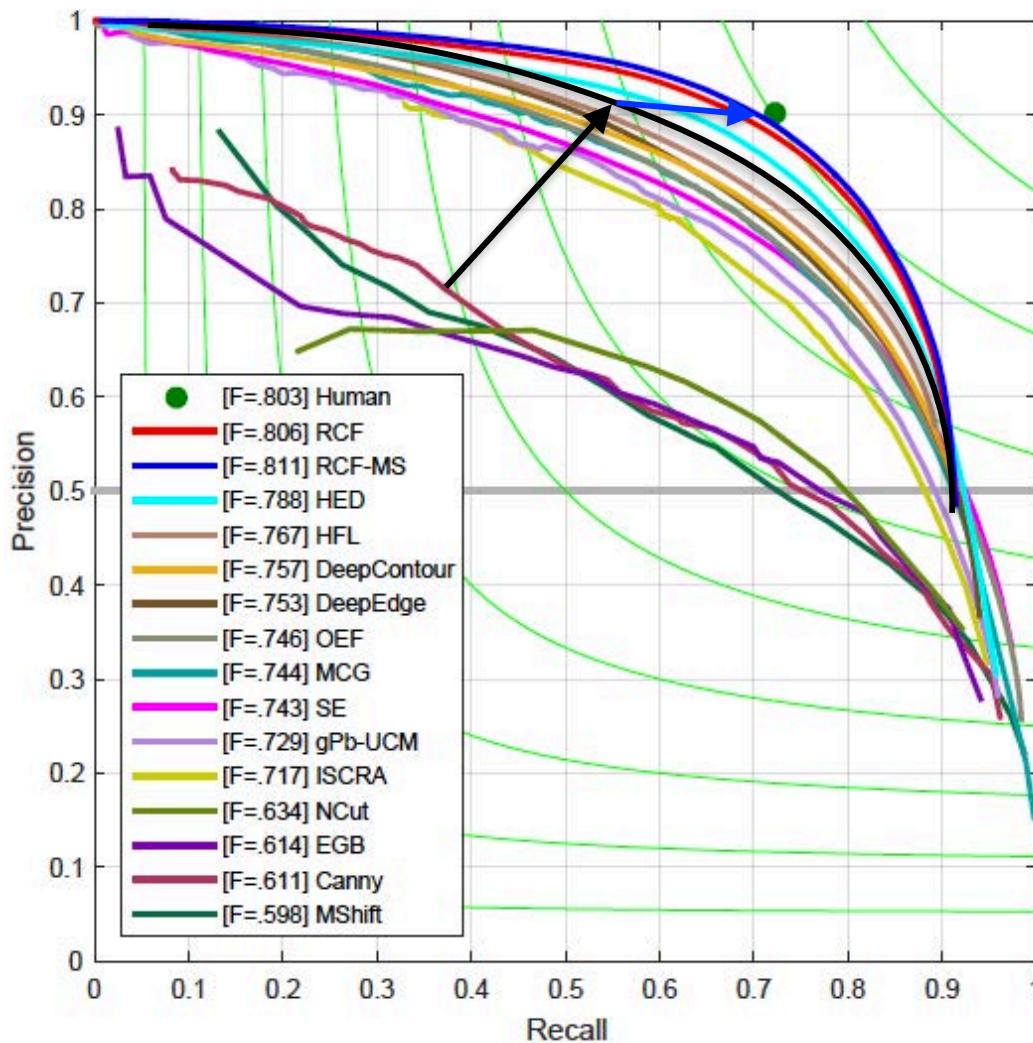
A PARTIAL EXPLANATION?



First and second layer features of a Convolutional Neural Net:

- They can be understood as performing multiscale filtering.
- The weights and thresholds are chosen by the optimization procedure.

50 YEARS OF EDGE DETECTION

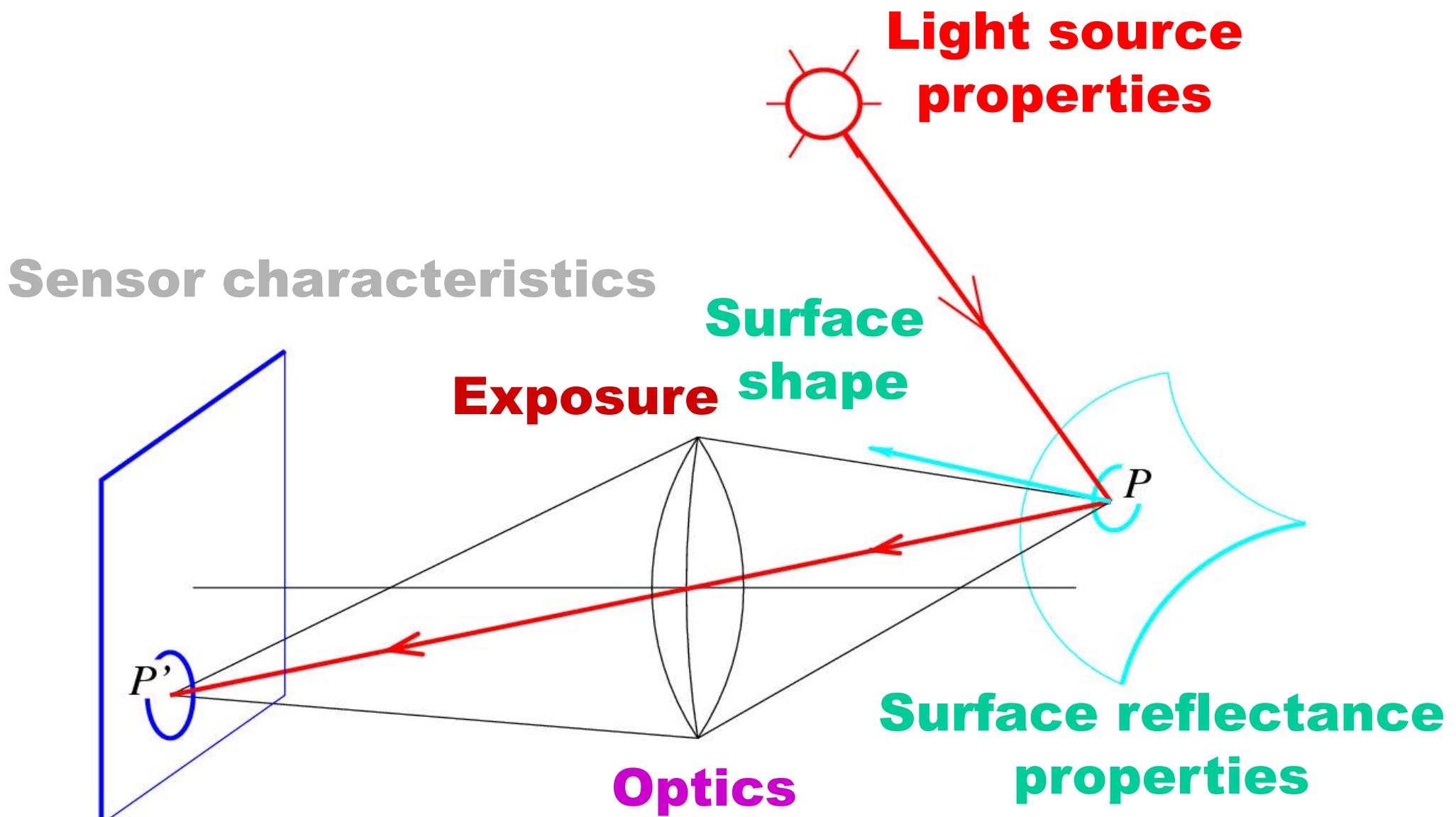


- Convolution operators respond to steep smooth shading.
- Parametric matchers tend to reject non ideal edges.
- Arbitrary thresholds and scale sizes are required.
- Learning-based methods need exhaustive databases.
- There still is work to go from contours to objects.

Canny, PAMI'86 → Sironi et al. PAMI'15

Sironi et al. PAMI'15 → Liu et al., CVPR'17

IMAGE FORMATION



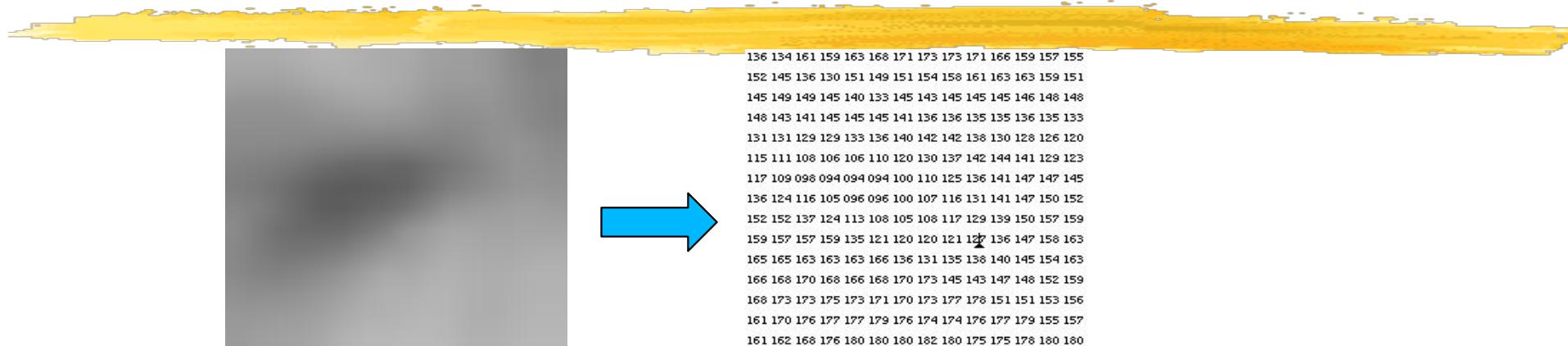
ANALOG IMAGES



An image can be understood as a 2D light intensity function $f(x,y)$ where:

- x and y are spatial coordinates
 - $f(x, y)$ is proportional to the brightness or gray value of the image at that point.
- Cannot be stored as such on a digital computer.

DIGITAL IMAGES



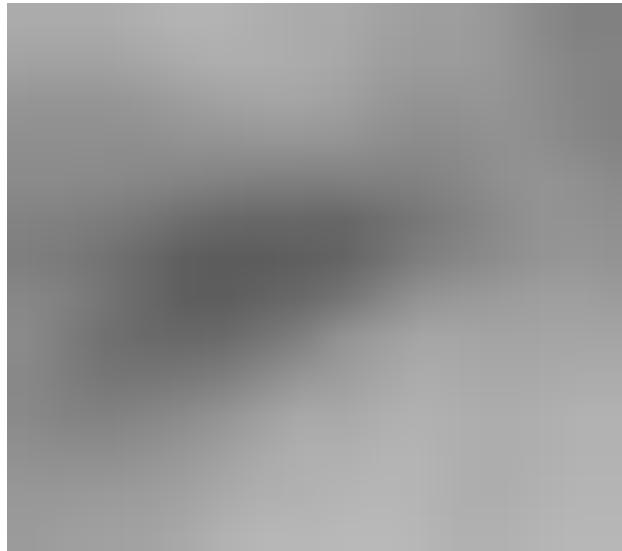
136 134 161 159 163 168 171 173 173 171 166 159 157 155
152 145 136 130 151 149 151 154 158 161 163 163 159 151
145 149 149 145 140 133 145 143 145 145 146 146 148 146
148 143 141 145 145 145 141 136 136 135 135 136 135 133
131 131 129 129 133 138 140 142 142 138 130 128 126 120
115 111 108 106 106 110 120 130 137 142 144 141 129 123
117 109 098 094 094 094 100 110 125 136 141 147 147 145
136 124 116 105 096 096 100 107 116 131 141 147 150 152
152 152 137 124 113 108 105 108 117 129 139 150 157 159
159 157 157 159 135 121 120 120 121 127 136 147 158 163
165 165 163 163 163 166 136 131 135 138 140 145 154 163
166 168 170 168 166 168 170 173 145 143 147 148 152 159
168 173 173 175 173 171 170 173 177 178 151 151 153 156
161 170 176 177 177 179 176 174 174 176 177 179 155 157
161 162 168 176 180 180 180 182 180 175 175 178 180 180

A digitized image is one in which:

- Spatial and grayscale values have been made discrete.
- Intensities measured across a regularly spaced grid in x and y directions are sampled to
 - 8 bits (256 values) per point for black and white,
 - 3x8 bits per point for color images.

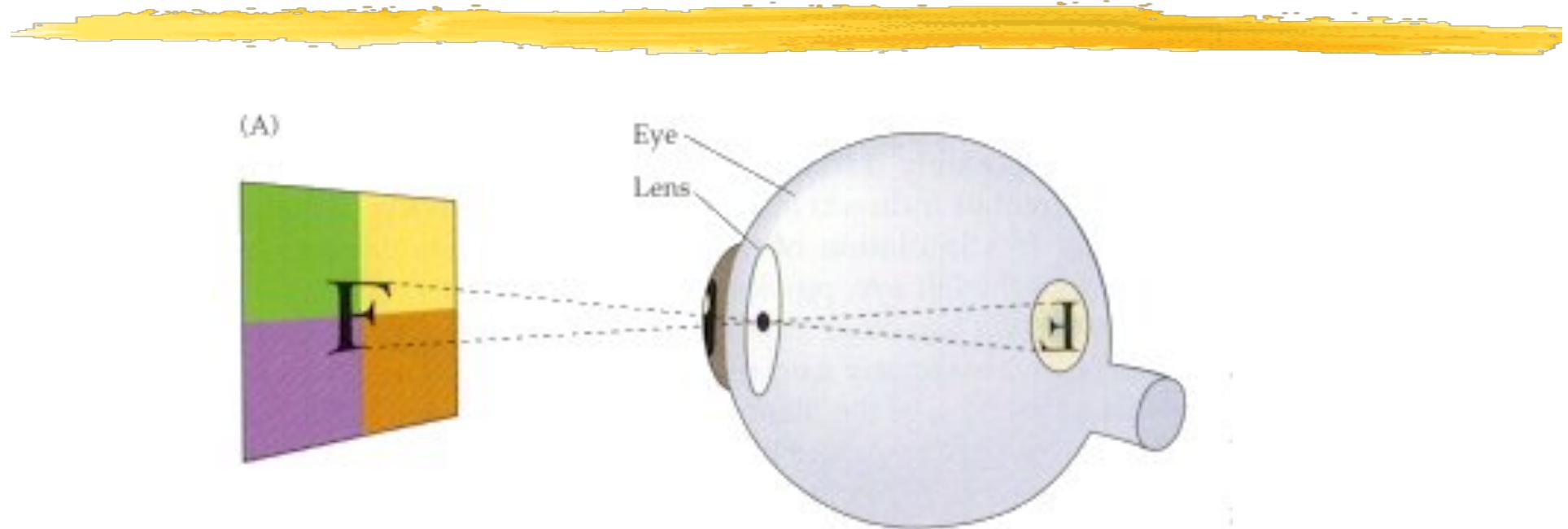
They are stored as a two dimensional arrays of gray-level values. The array elements are called pixels and identified by their x, y coordinates.

PIXELS



136 134 161 159 163 168 171 173 173 171 166 159 157 155
152 145 136 130 151 149 151 154 158 161 163 163 159 151
145 149 149 145 140 133 145 143 145 145 146 146 148 148
148 143 141 145 145 141 136 136 135 135 136 135 133
131 131 129 129 133 136 140 142 142 138 130 128 126 120
115 111 108 106 106 110 120 130 137 142 144 141 129 123
117 109 098 094 094 094 100 110 125 136 141 147 147 145
136 124 116 105 096 096 100 107 116 131 141 147 150 152
152 152 137 124 113 108 105 108 117 129 139 150 157 159
159 157 157 159 135 121 120 120 121 127 136 147 158 163
165 165 163 163 163 166 136 131 135 138 140 145 154 163
166 168 170 168 166 168 170 173 145 143 147 148 152 159
168 173 173 175 173 171 170 173 177 178 151 151 153 156
161 170 176 177 177 179 176 174 174 176 177 179 155 157
161 162 166 176 180 180 180 182 180 175 175 178 180 180

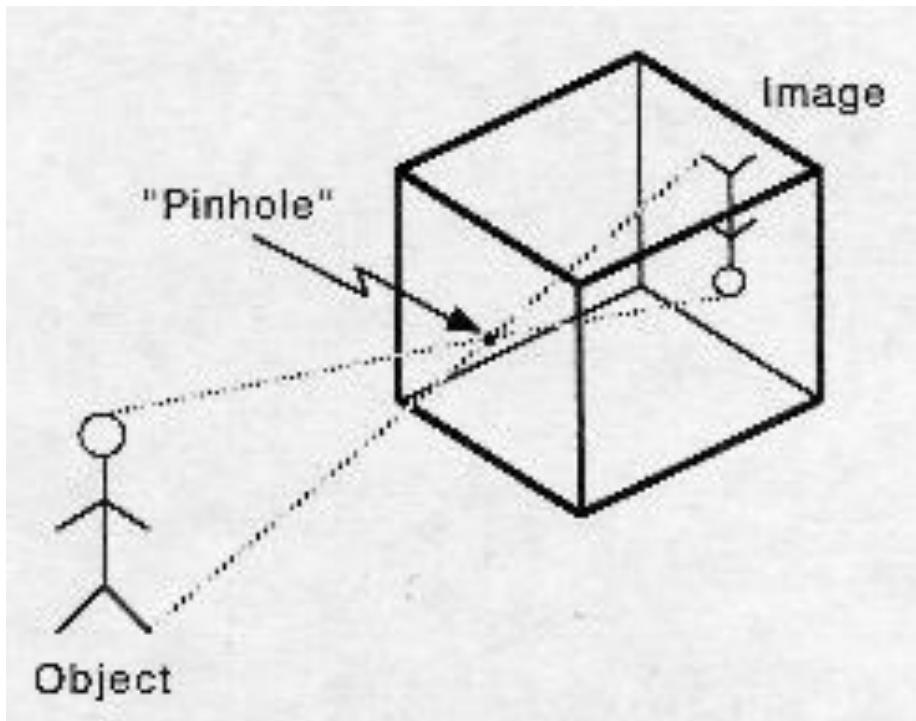
IMAGE FORMATION



Projection from surfaces to 2-D sensor.

- Where: Geometry
- How bright: Radiometry
- Stored how: Sensing

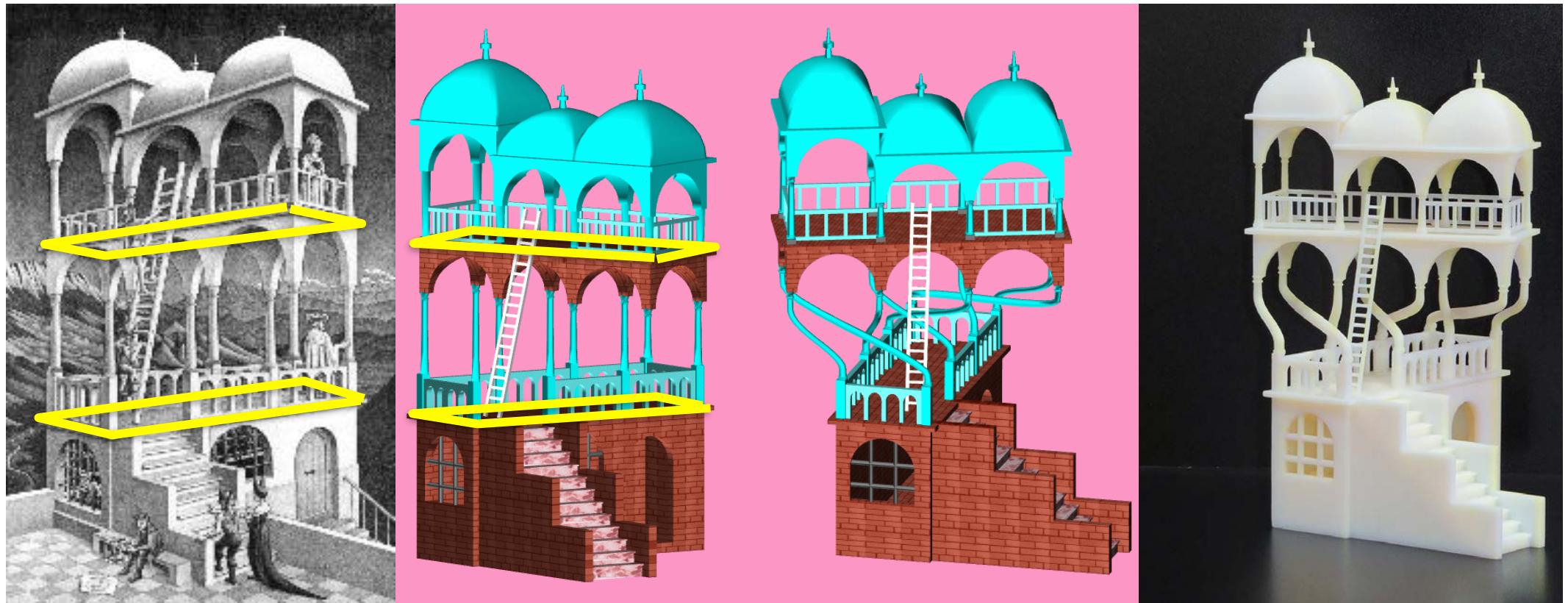
PINHOLE CAMERA MODEL



Idealized model of the perspective projection:

- All rays go through a hole and form a pencil of lines.
- The hole acts as a ray selector that allows an inverted image to form.

ESCHER'S BELVEDERE



M. C. Escher

Impossible

Possible

Done

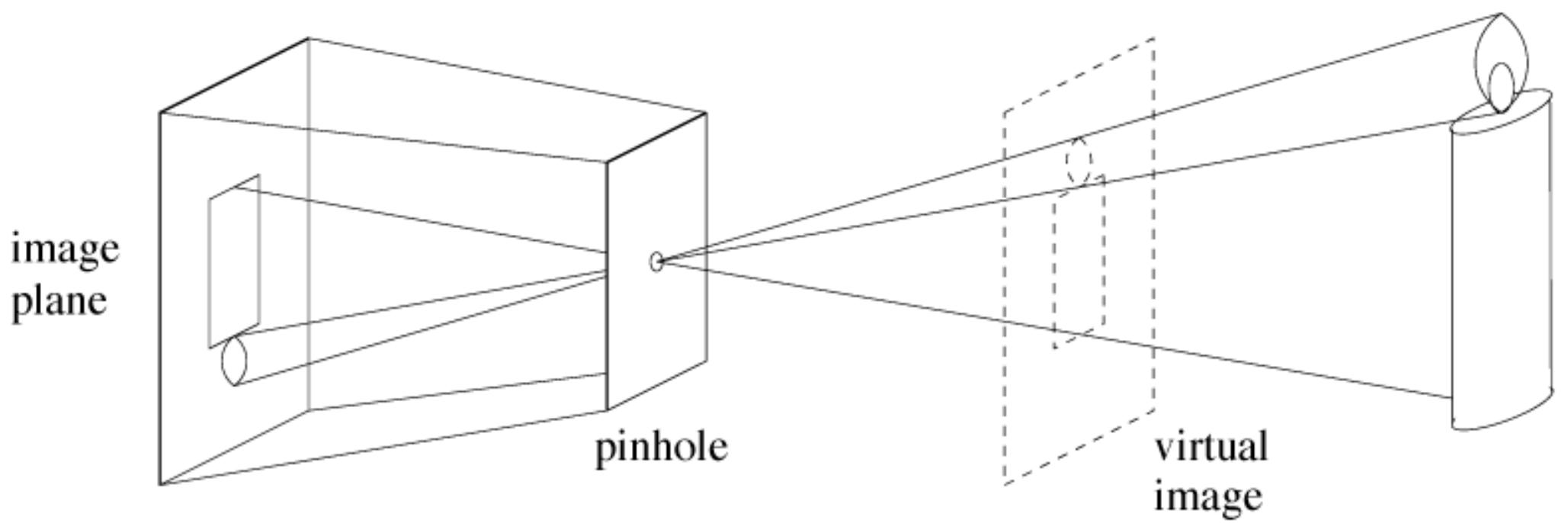
Gershon Helber

<http://www.cs.technion.ac.il/~gershon/>

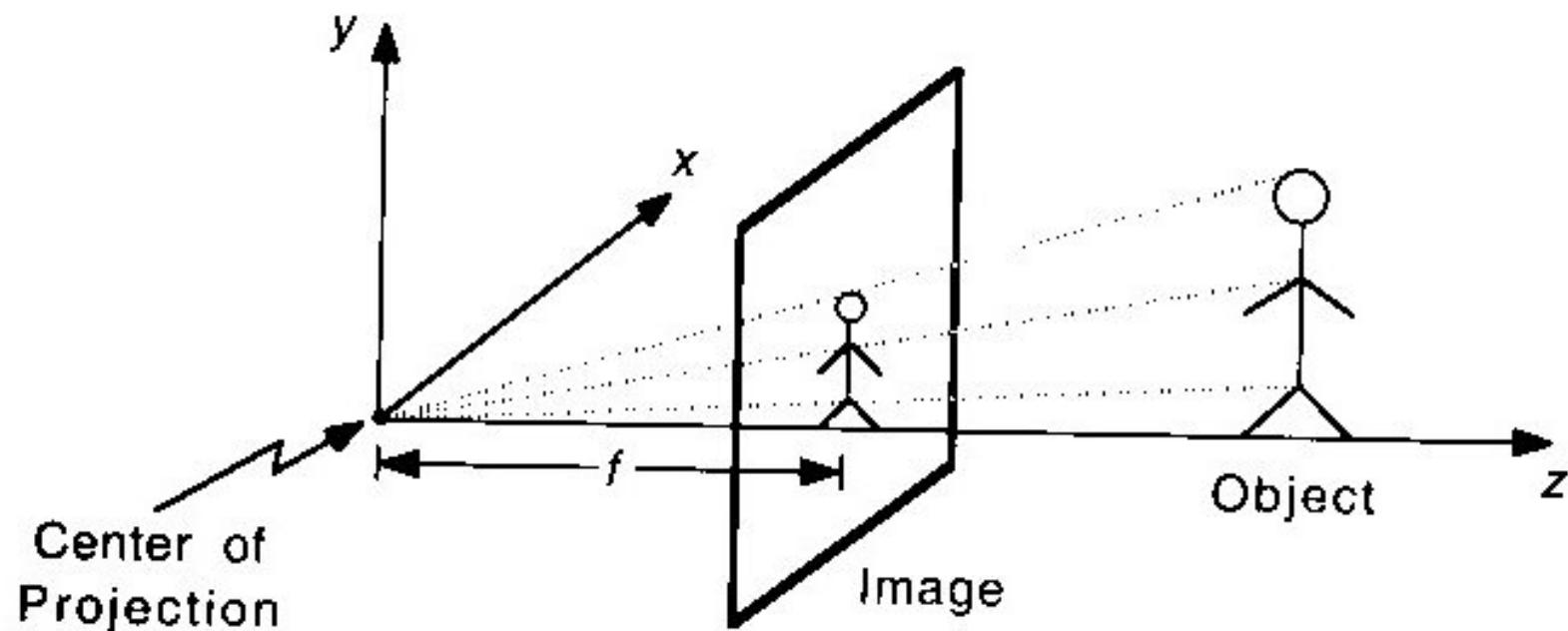
MAGNET LIKE SLOPES



VIRTUAL IMAGE



CAMERA GEOMETRY



Pinhole geometry without image reversal

COORDINATE SYSTEMS



World, Camera, Image Coordinate Systems

World Coordinate System:

$$(X_w, Y_w, Z_w)$$

Camera Coordinate System:

$$(X_c, Y_c, Z_c)$$

Image Coordinate System:

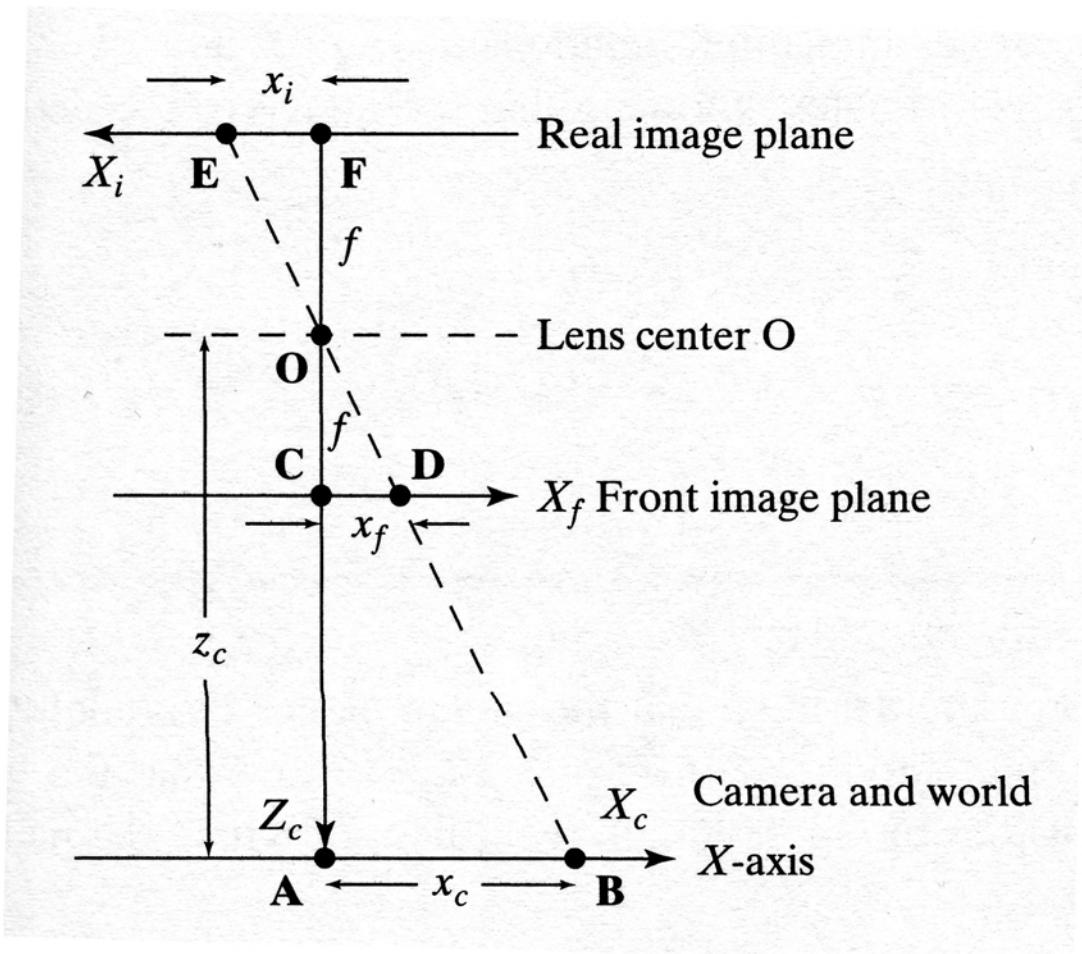
$$(X_i, Y_i, Z_i)$$

CAMERA COORDINATE SYSTEM



- The center of the projection coincides with the origin of the world.
- The camera axis (optical axis) is aligned with the world's z-axis.
- To avoid image inversion, the image plane is in front of the center of projection.

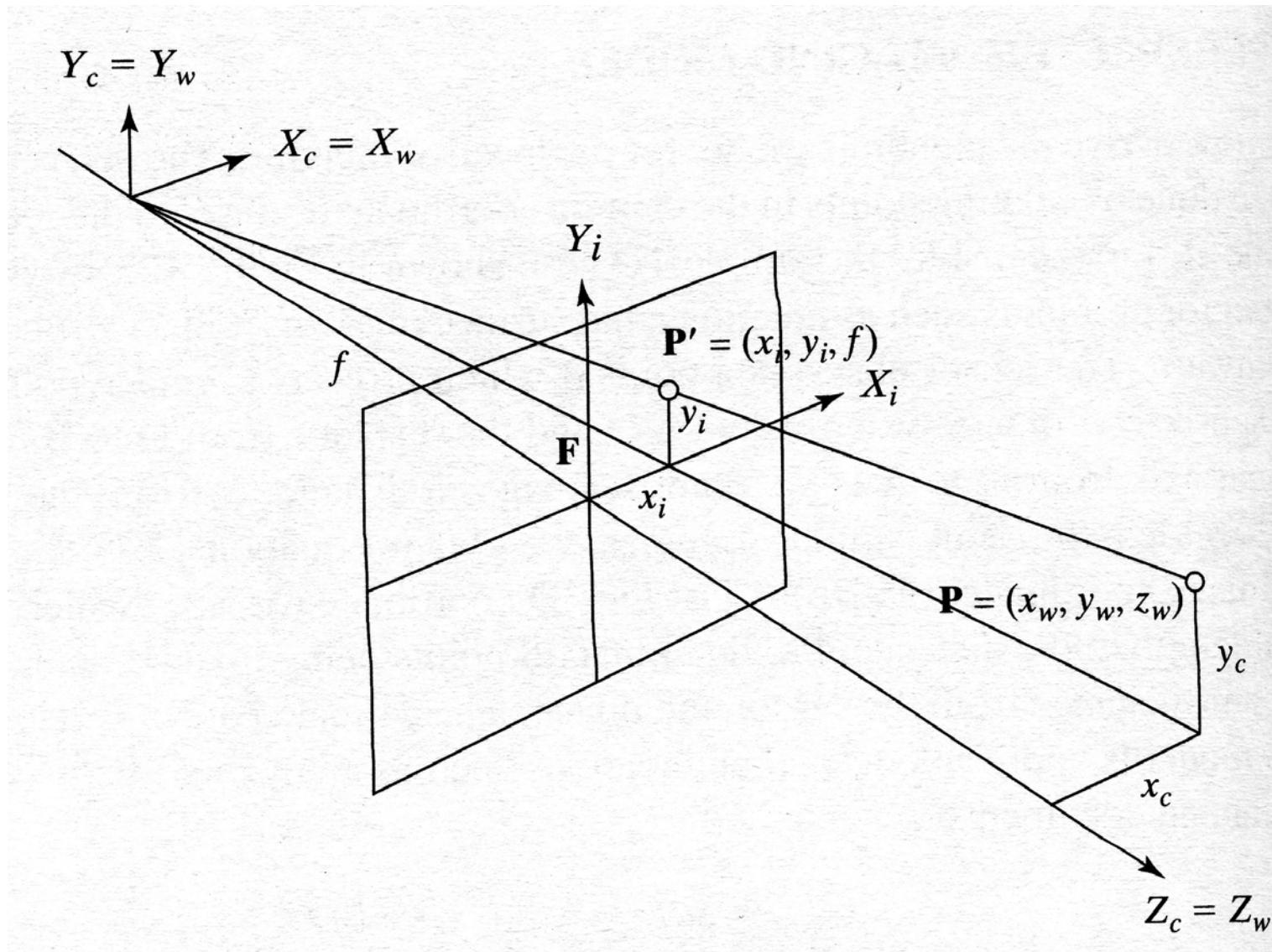
1D IMAGE



$$X_f / f = X_c / Z_c$$

$$X_f = f \frac{X_c}{Z_c}$$

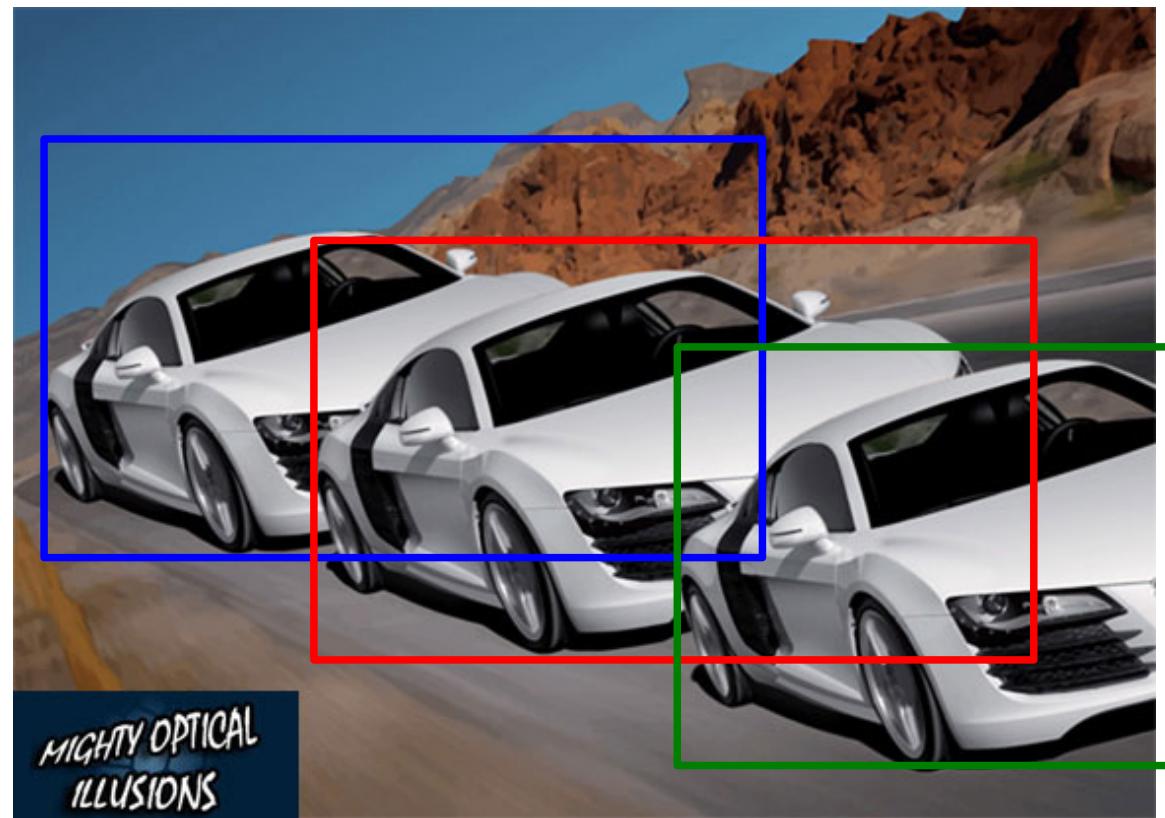
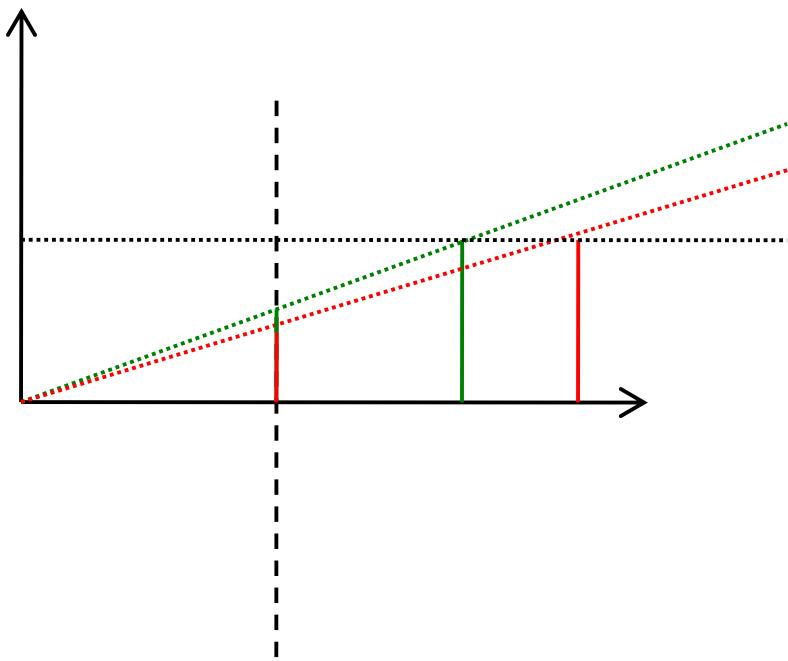
2D IMAGE



$$X_i = f \frac{X_c}{Z_c}$$

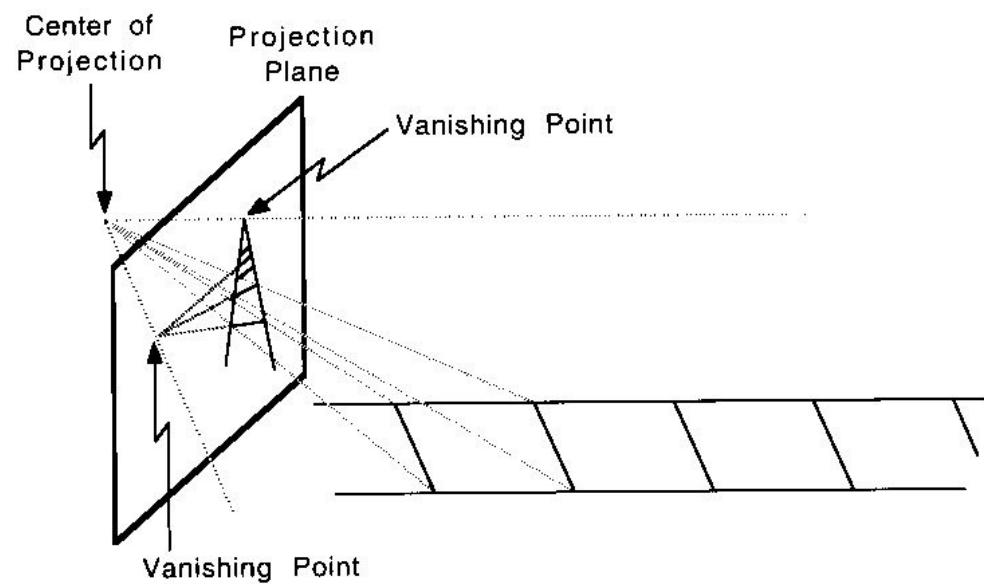
$$Y_i = f \frac{Y_c}{Z_c}$$

DISTANT OBJECTS APPEAR SMALLER



MIGHTY OPTICAL
ILLUSIONS

PARALLEL LINES MEET

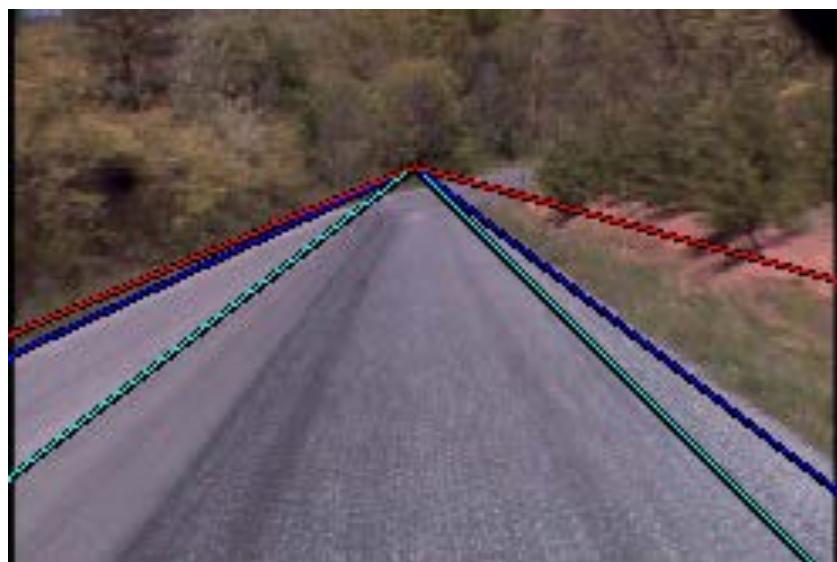
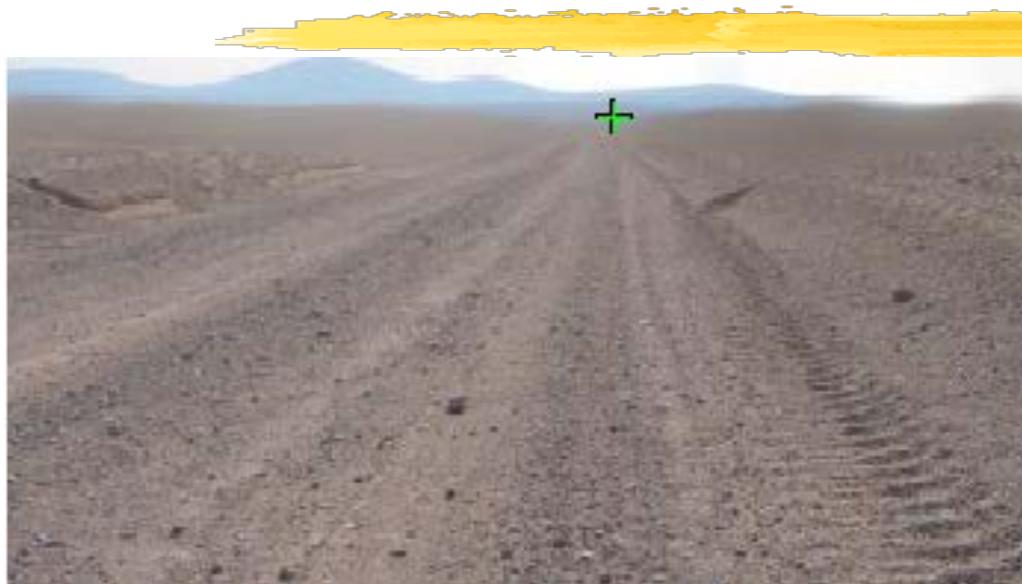


VANISHING POINTS

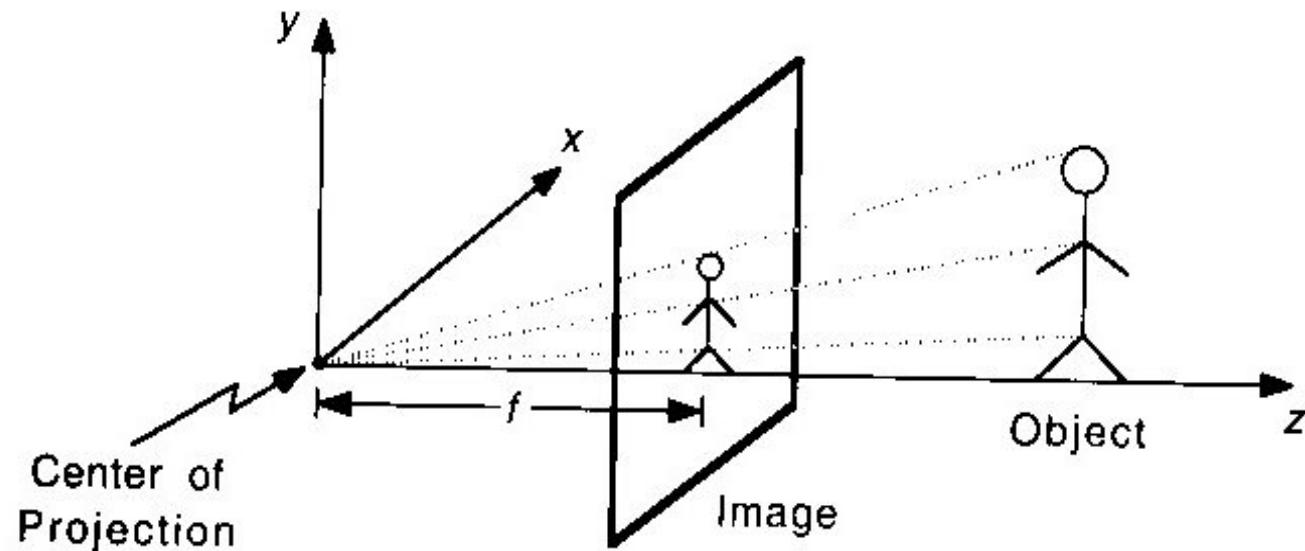


- The projections of parallel lines all meet at one point, called the vanishing point.
- As focal length and distance to camera increase, the image remains the same size but perspective effects diminish.

ROAD FOLLOWING



PROJECTION IS NON LINEAR



$$u = X_i = f \frac{X_c}{Z_c}$$

$$v = Y_i = f \frac{Y_c}{Z_c}$$

→ Reformulate it as a linear operation.

HOMOGENEOUS COORDINATES



Homogeneous representation of 2D point:

$\mathbf{x} = (x_1, x_2, x_3)$ represents $(x_1/x_3, x_2/x_3)$

Homogeneous representation of 3D point:

$\mathbf{X} = (x_1, x_2, x_3, x_4)$ represents $(x_1/x_4, x_2/x_4, x_3/x_4)$

→ Projections become linear transformations.

SIMPLE PROJECTION MATRIX

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix}$$

with $X_i = \frac{x}{z} = f \frac{X_c}{Z_c}$ and $Y_i = \frac{y}{z} = f \frac{Y_c}{Z_c}$

$$= \begin{bmatrix} f & 0 & 0 \\ 0 & f & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix}$$

INTRINSIC AND EXTRINSIC PARAMETERS

- Camera may not be at the origin, looking down the z-axis
 - Extrinsic parameters
- One unit in camera coordinates may not be the same as one unit in world coordinates
 - Intrinsic parameters

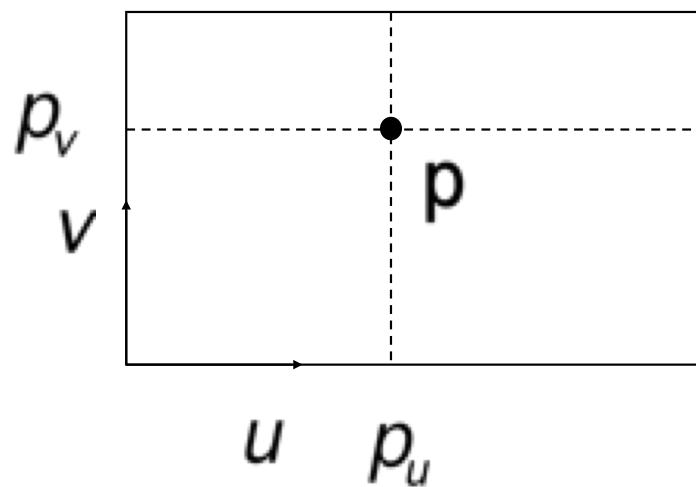
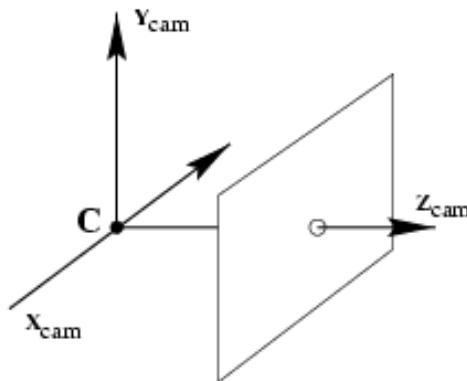
$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} \text{Matrix of} \\ \text{intrinsic parameters} \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \text{Matrix of} \\ \text{extrinsic parameters} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

LINEAR CAMERA MODEL

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} \text{Matrix of} \\ \text{intrinsic parameters} \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \text{Matrix of} \\ \text{extrinsic parameters} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$
$$= \mathbf{K} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \mathbf{Rt} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

where \mathbf{K} is a 3×3 matrix and \mathbf{Rt} a 4×4 matrix.

PRINCIPAL POINT



$$\begin{aligned} u &= X_i + p_u = fX/Z + p_u \\ v &= Y_i + p_v = fY/Z + p_v \\ \mathbf{K} &= \begin{bmatrix} f & 0 & p_u \\ 0 & f & p_v \\ 0 & 0 & 1 \end{bmatrix} \end{aligned}$$

INHOMOGENEOUS SCALING

$$u = \alpha_u X_i + p_u = \alpha_u X/Z + p_u$$

$$v = \alpha_v Y_i + p_v = \alpha_v Y/Z + p_v$$

$$\mathbf{K} = \begin{bmatrix} \alpha_u & 0 & p_u \\ 0 & \alpha_v & p_v \\ 0 & 0 & 1 \end{bmatrix}$$

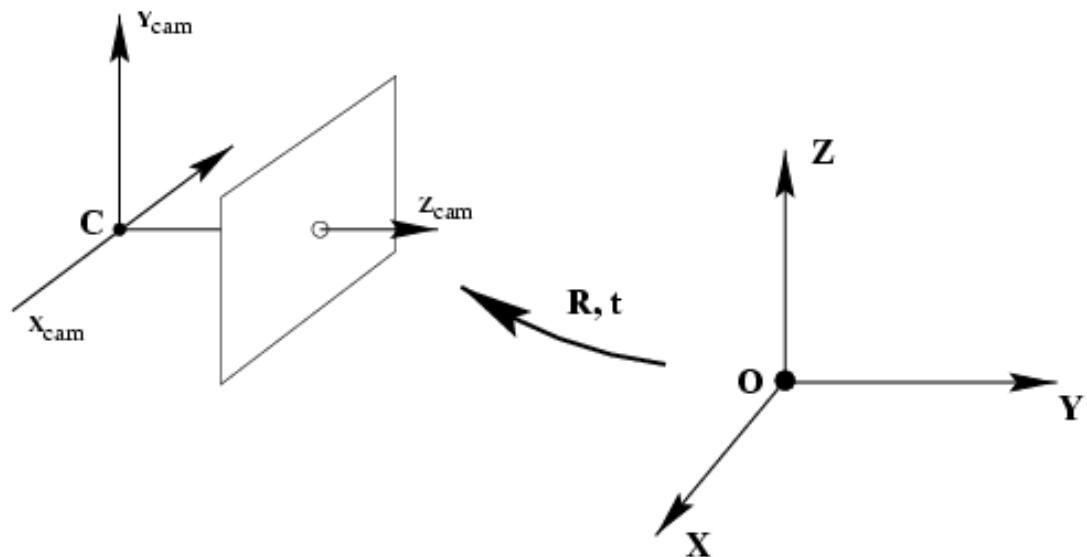
The pixels are not necessarily square.

AXIS SKEW


$$\mathbf{K} = \begin{bmatrix} \alpha_u & s & p_u \\ 0 & \alpha_v & p_v \\ 0 & 0 & 1 \end{bmatrix}$$

s encodes the non-orthogonality of the u and v directions. Very close to zero in modern cameras.

ROTATION / TRANSLATION



$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \end{bmatrix} = \mathbf{R} \left(\begin{bmatrix} X_w \\ Y_w \\ Z_w \end{bmatrix} - \tilde{\mathbf{C}} \right) \text{ with } \mathbf{R}^t \mathbf{R} = \mathbf{I}$$

$$\begin{bmatrix} X_c \\ Y_c \\ Z_c \\ 1 \end{bmatrix} = \begin{bmatrix} \mathbf{R} & \mathbf{T} \\ 0 & 1 \end{bmatrix} \begin{bmatrix} X_w \\ Y_w \\ Z_w \\ 1 \end{bmatrix} \text{ with } \mathbf{T} = -\mathbf{R}\tilde{\mathbf{C}}$$

→ Rotations and translations also expressed in terms of matrix multiplications in projective space.

FULL PROJECTION MATRIX

$$\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} a_u & s & p_u \\ 0 & a_v & p_v \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{R} & \mathbf{T} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

with $\mathbf{T} = -\mathbf{R}\tilde{\mathbf{C}}$ and $\mathbf{R}^t\mathbf{R} = \mathbf{I}$

$$= \begin{bmatrix} p_{11} & p_{12} & p_{13} & p_{14} \\ p_{21} & p_{22} & p_{23} & p_{24} \\ p_{31} & p_{32} & p_{33} & 1 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}$$

$$\mathbf{x} = \mathbf{K}[\mathbf{R} \mid -\mathbf{R}\tilde{\mathbf{C}}]\mathbf{X} = \mathbf{KR}[\mathbf{I} \mid -\tilde{\mathbf{C}}]\mathbf{X} = [\mathbf{M} \mid \mathbf{m}]\mathbf{X} = \mathbf{PX}$$

CAMERA CALIBRATION



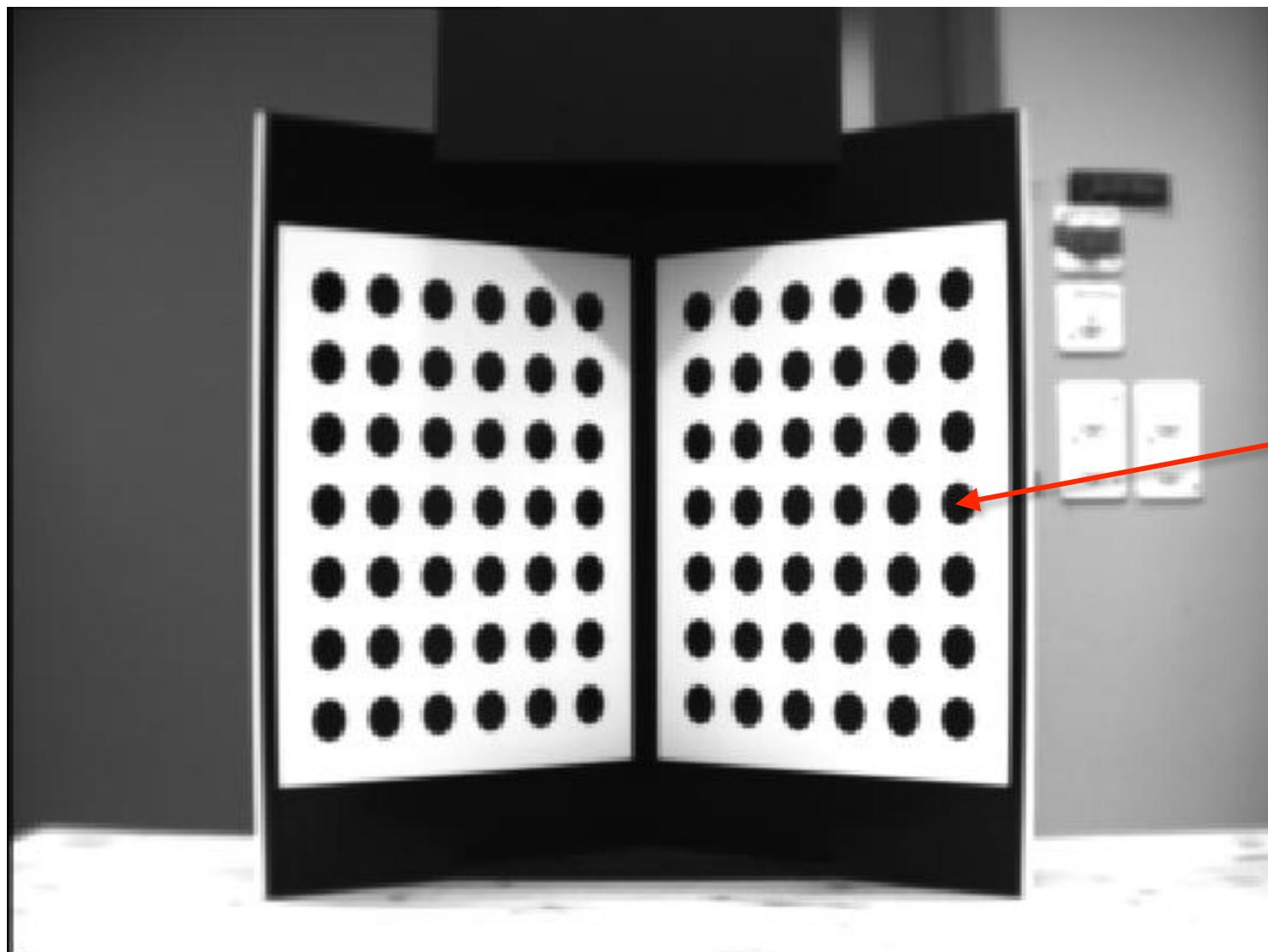
Internal Parameters:

- Horizontal and vertical scaling (2)
- Principal points (2)
- Skew of the axis (1)

External Parameters:

- Rotations (3)
 - Translations (3)
- **11 free parameters.**

CALIBRATION GRID



$$x_i = P x_i$$

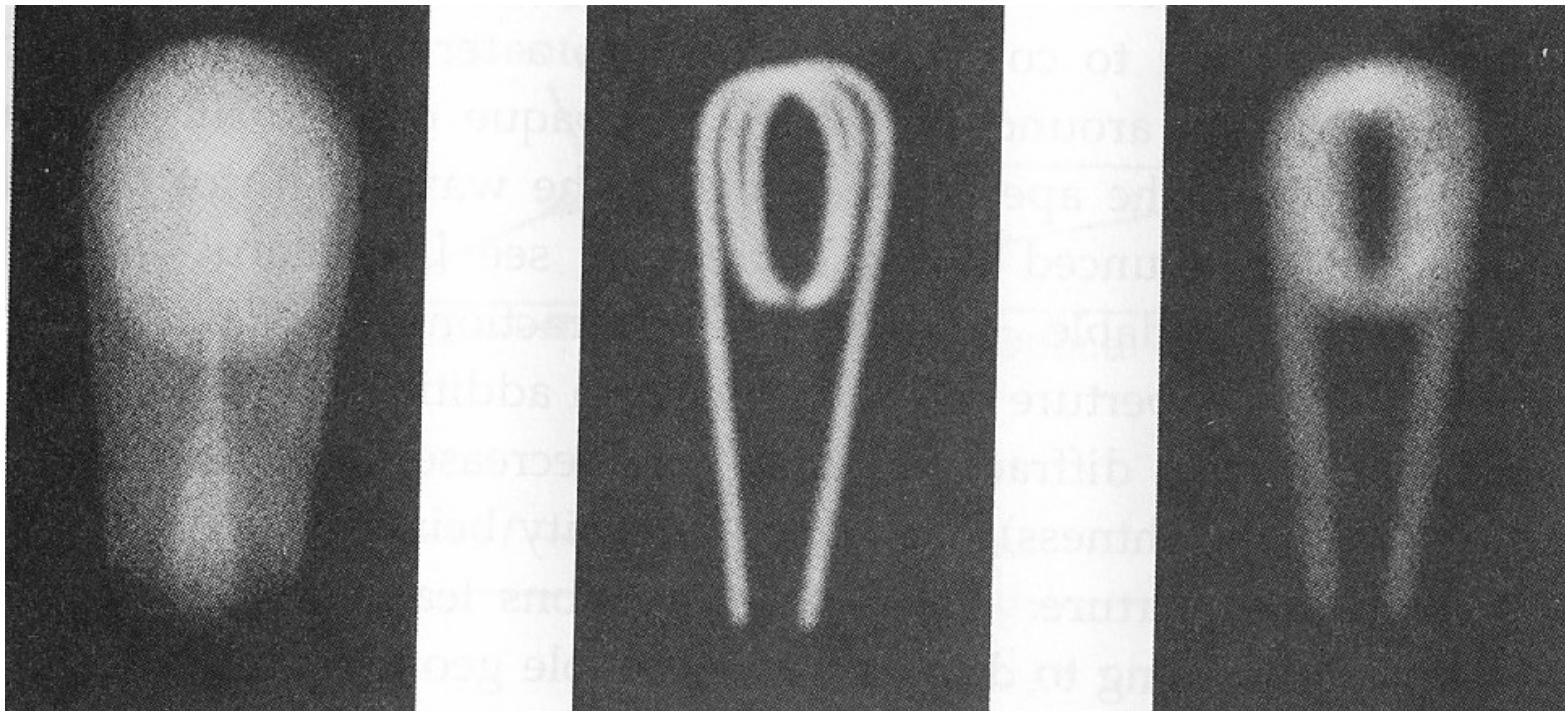
PARAMETERS OF PROJECTION MATRIX



For all i , $\mathbf{x}_i = \mathbf{P}\mathbf{X}_i$

- Number of measurements required:
 - 11 degrees of freedom.
 - 2 constraints per correspondence.
- Direct linear transform:
 - Minimal solution for 6 correspondences
 - Over-constrained solutions by imposing
$$\|\mathbf{P}\| = 1 \text{ or } P_{34} = 1$$
- Non linear optimization.

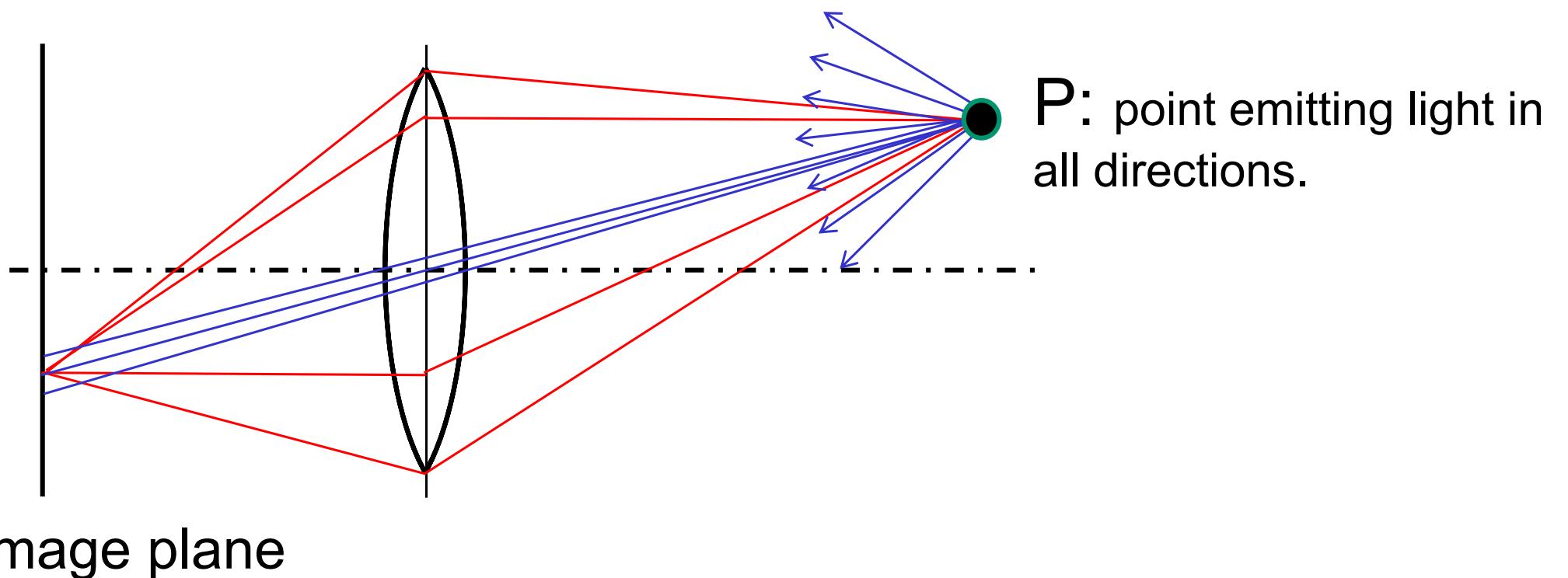
LIMITATIONS



Idealization because the hole cannot be infinitely small

- Image would be infinitely dim
 - Diffraction effects
- Use of Lenses.

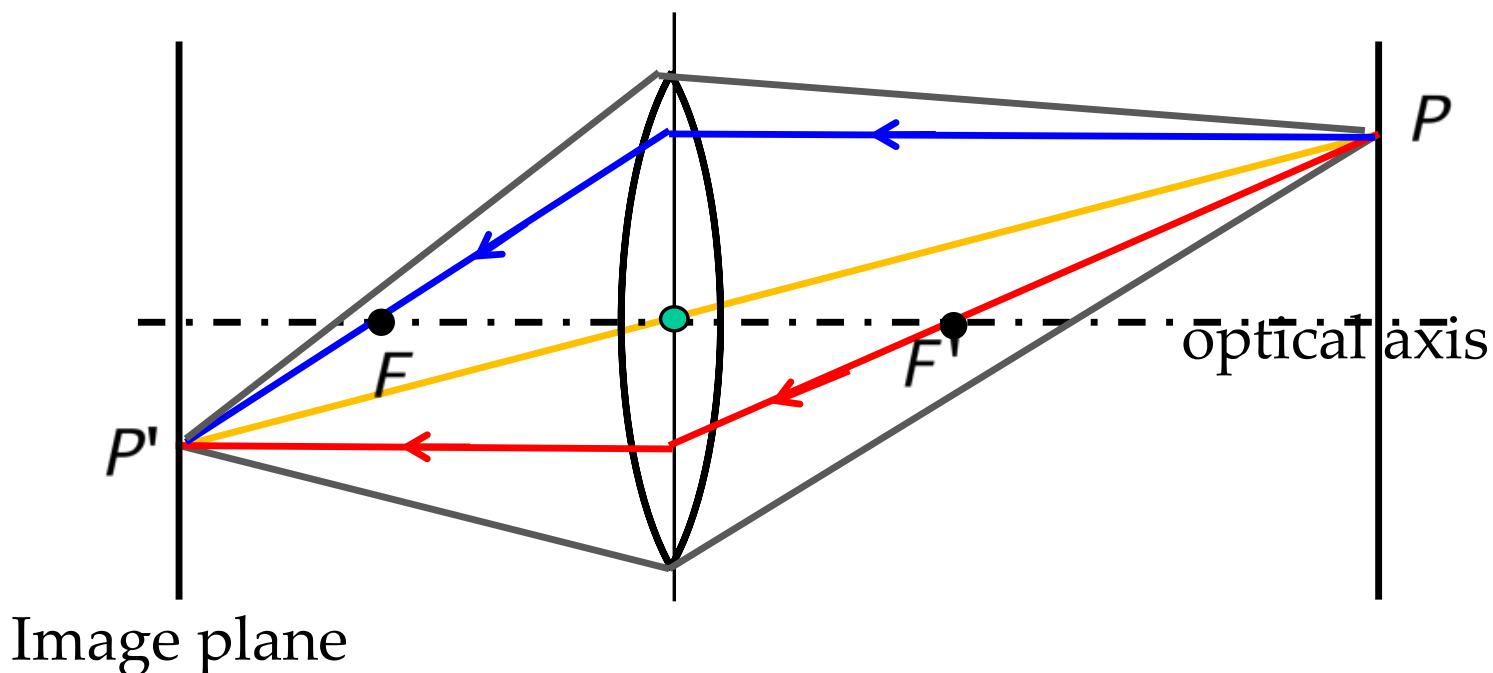
IMAGING WITH A LENS



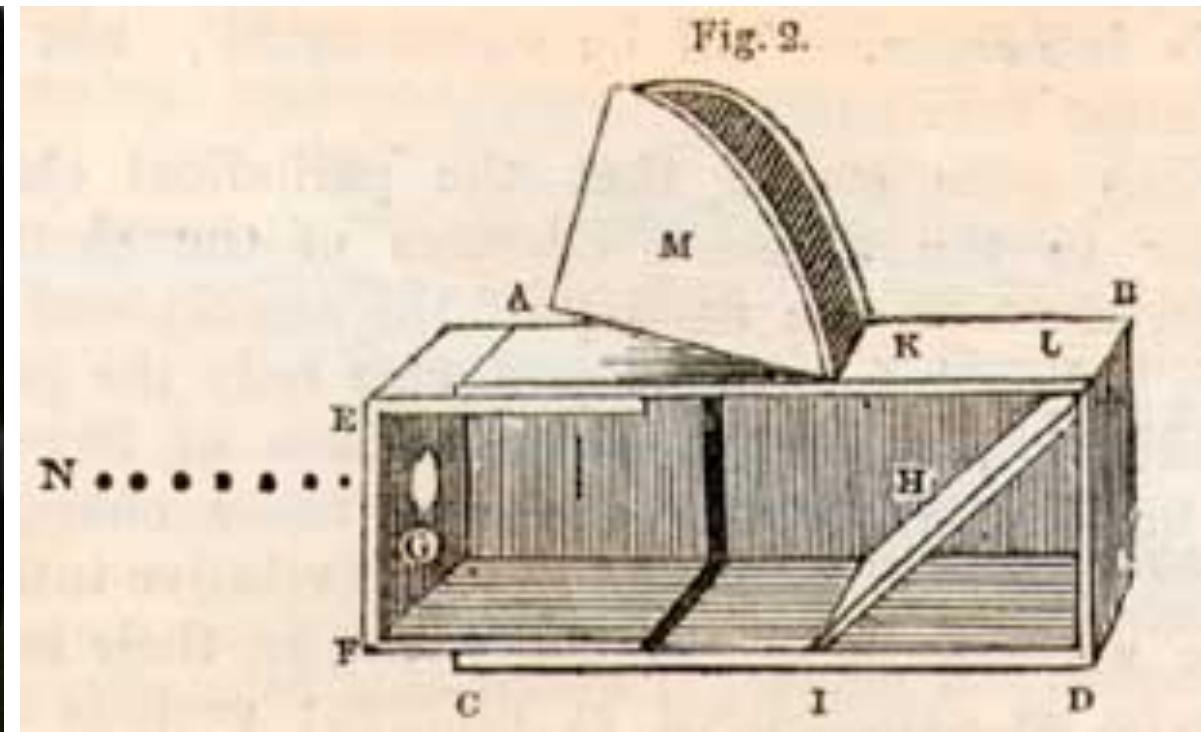
An ideal lens realizes the same projection as a pinhole but **gathers much more light!**

THIN LENS PROPERTIES

- An incident ray which passes through the center of the lens will in effect continue in the same direction that it had when it entered the lens.
- Any incident ray traveling parallel to the optical axis, will refract and travel through the focal point on the opposite side of the lens.
- Any incident ray traveling through the focal point on the way to the lens will be refracted and travel parallel to the principal axis.
- All rays emanating from P and entering the lens will converge at P'

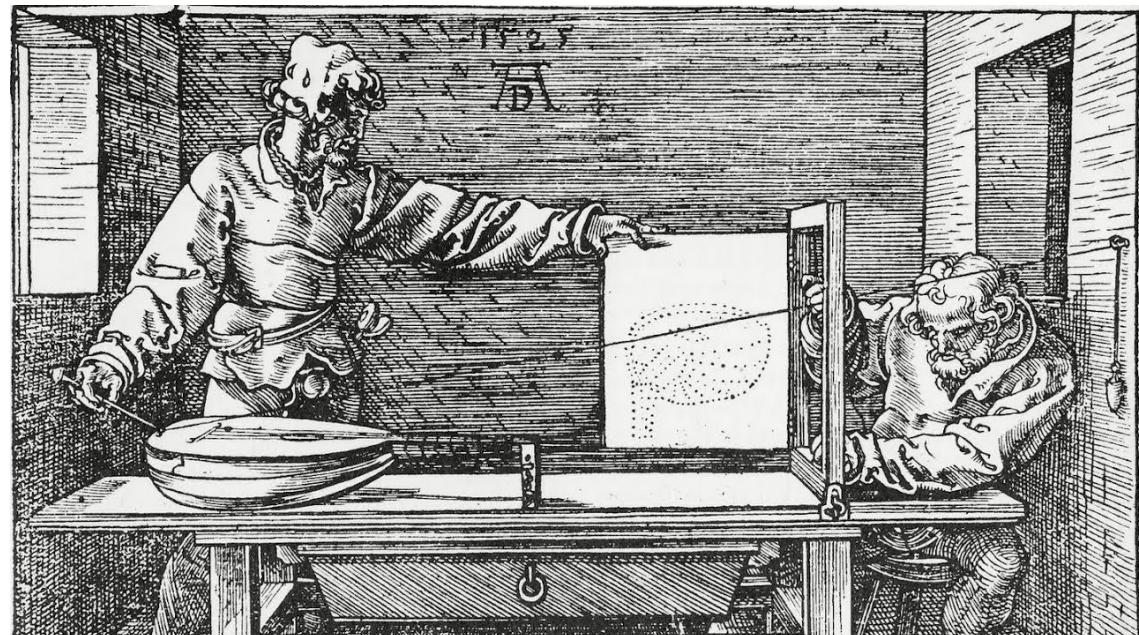


CAMERA OBSCURA



- Used by painters since the Renaissance to produce perspective projections.
- Direct ancestors to the first film cameras.

DURER 1471-1528



He clearly knew all about
the perspective transform!

SHIFTING PERSPECTIVE

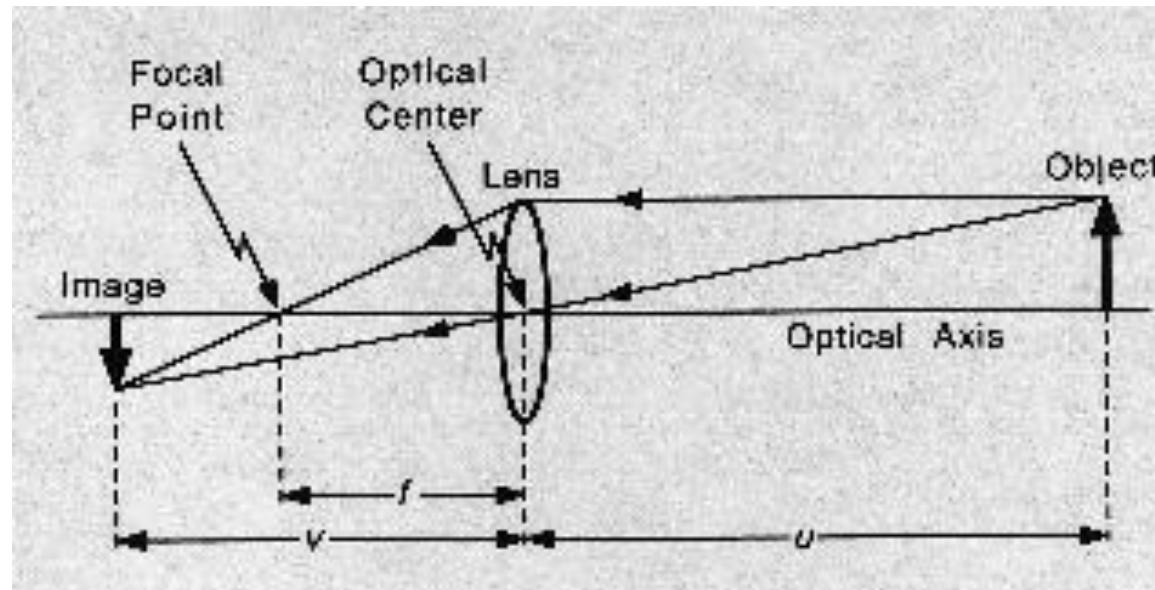


China, 8th century:

- The focal point moves from one part of the image to the other.
- The characters are always seen at eye-level as the picture is unrolled.

Buddha cutting his hair, 8th c.

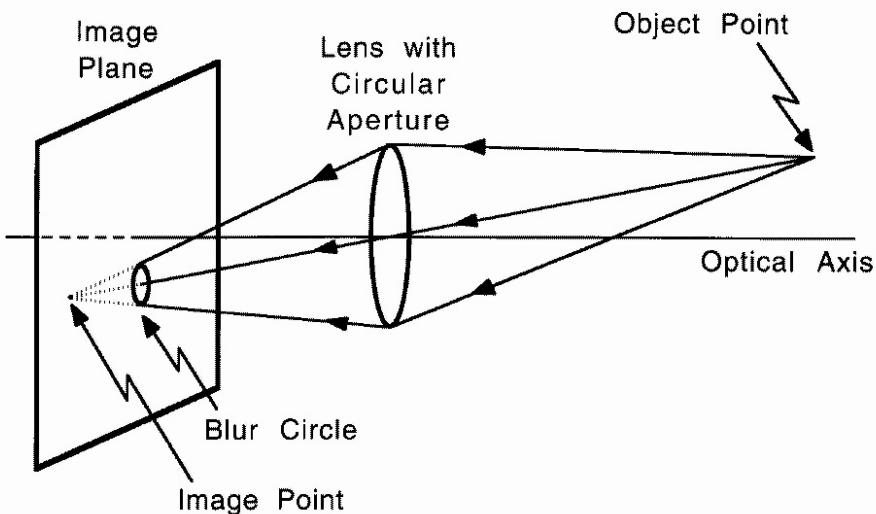
THIN LENS EQUATION



$$\cancel{\frac{1}{u} + \frac{1}{v} = \frac{1}{f}}$$

→ Lens with focal distance f equivalent to pinhole camera with similar focal distance but larger aperture.

DEPTH OF FIELD vs APERTURE



Large Aperture:

- Large blur circles
- Shallow depth of field

Small Aperture:

- Low intensity
- Long exposure time

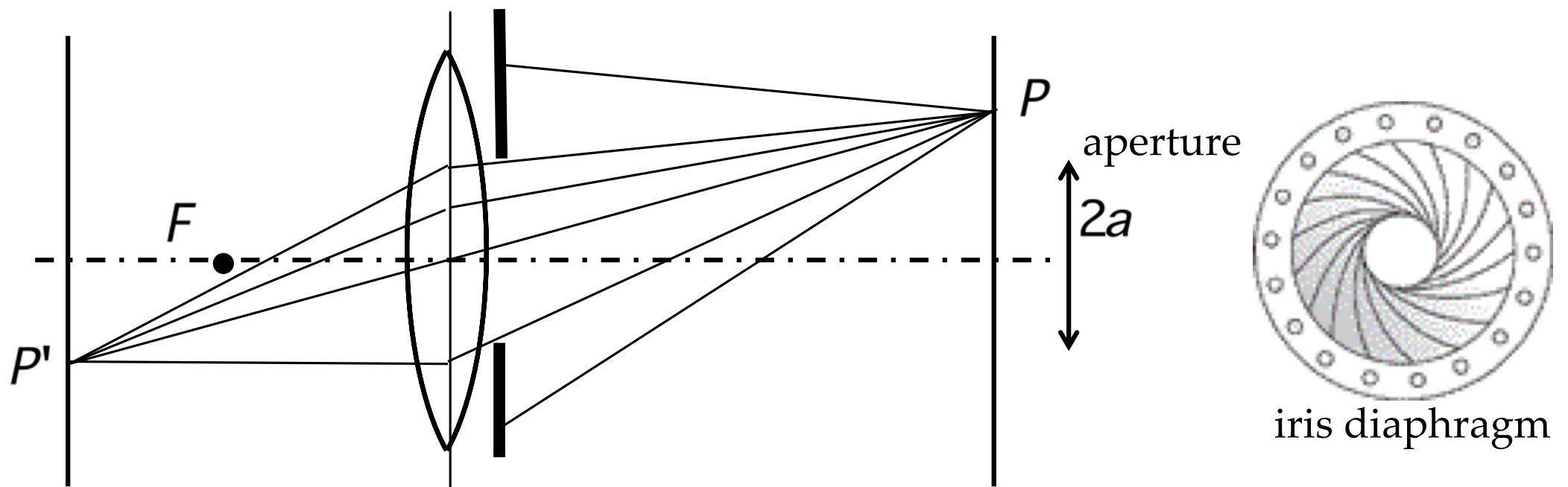
DEPTH OF FIELD

- Range of object distances ($d-d'$) over which the image is sufficiently well focused.
- Range for which blur circle is less than the resolution of the sensor.



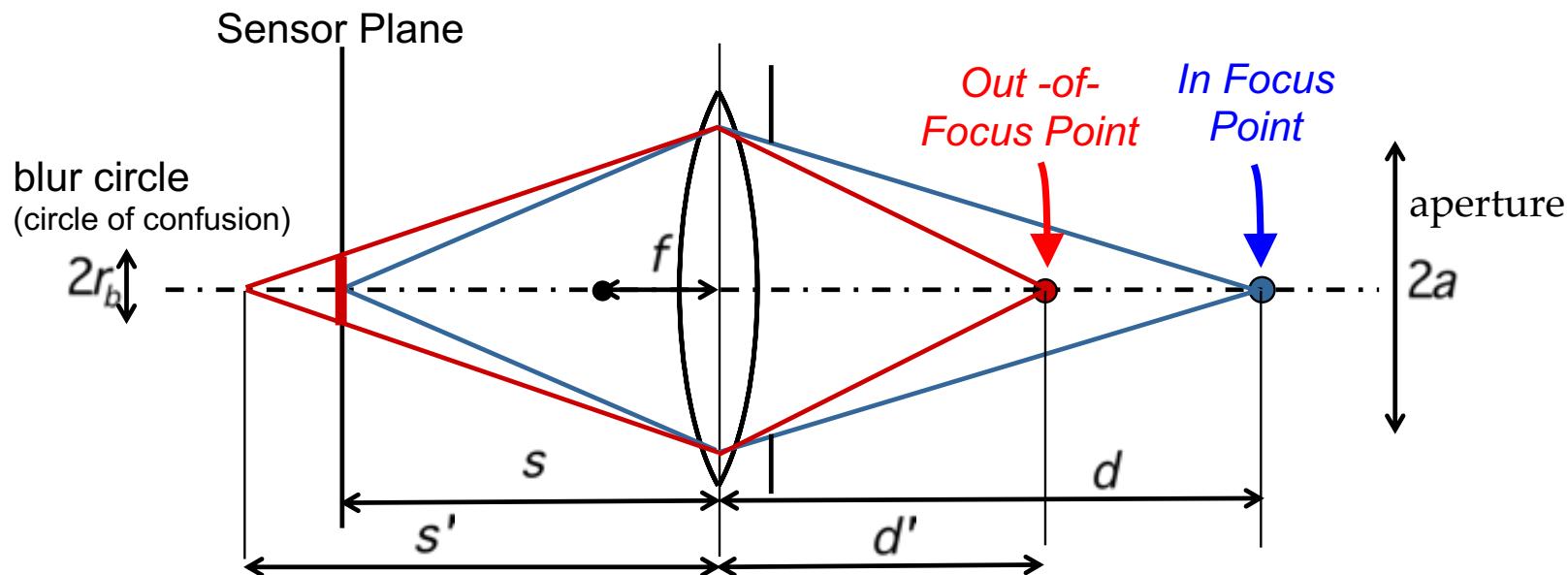
Small focal length → Large depth of field.

APERTURE



Diameter d of the lens that is exposed to light.

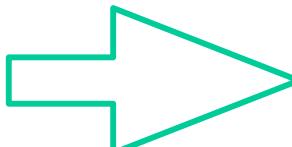
BLUR CIRCLE



- Simple geometry:

$$r_b = \frac{a}{s'} |s' - s|$$

- Thin lens equation:



$$(s' - s) = \frac{f}{(d' - f)} \frac{f}{(d - f)} (d - d')$$

$$\frac{1}{d} + \frac{1}{s} = \frac{1}{f} \Rightarrow s = \frac{df}{d-f}$$

$$\frac{1}{d'} + \frac{1}{s'} = \frac{1}{f} \Rightarrow s' = \frac{d'f}{d'-f}$$

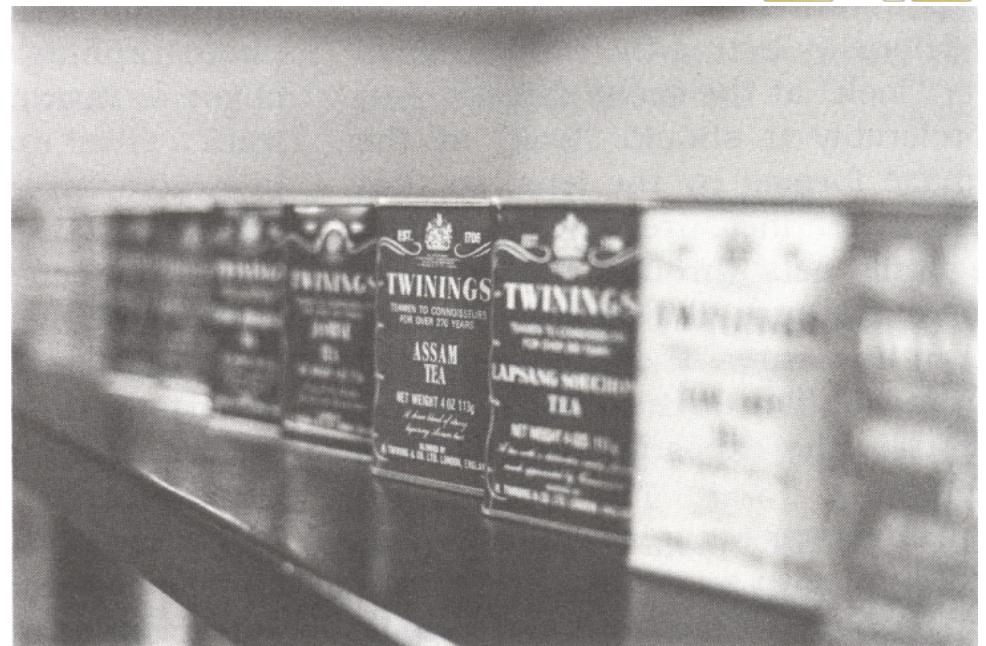
CHANGING APERTURE



f/11

1/30sec

Small aperture, long exposure.



f/2.8

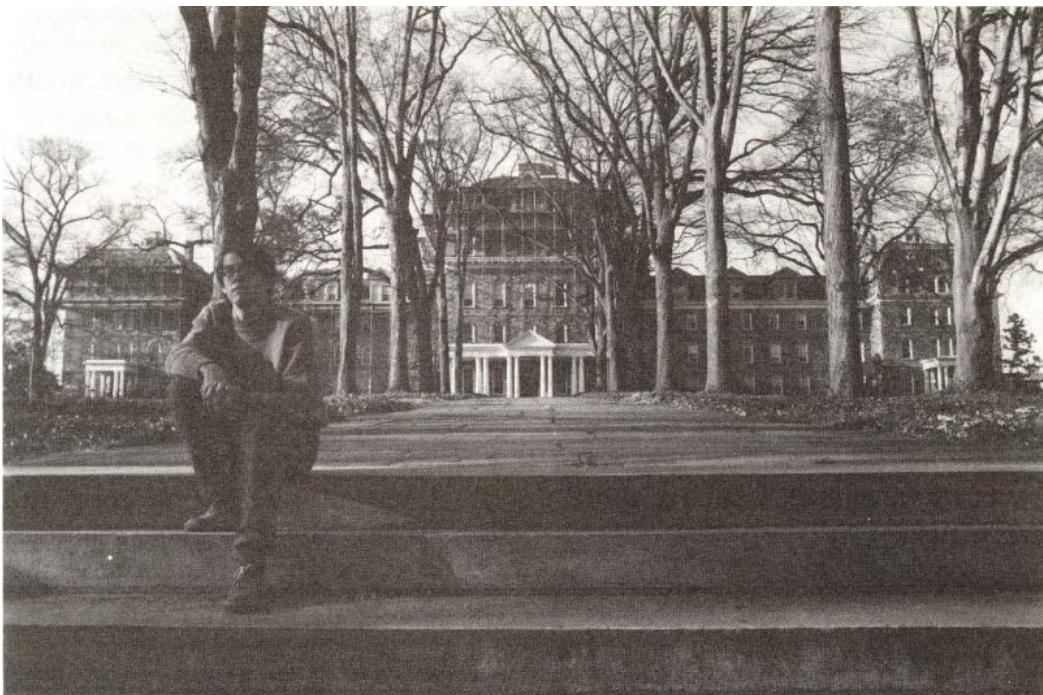
1/500sec

Large aperture, short exposure.

$$r_b = \frac{a}{s'} \left| \frac{f^2}{(d' - f)(d - f)} \right|$$

Small $a \rightarrow$ Small r_b

CHANGING FOCAL LENGTH



Wide field of view
(small f)

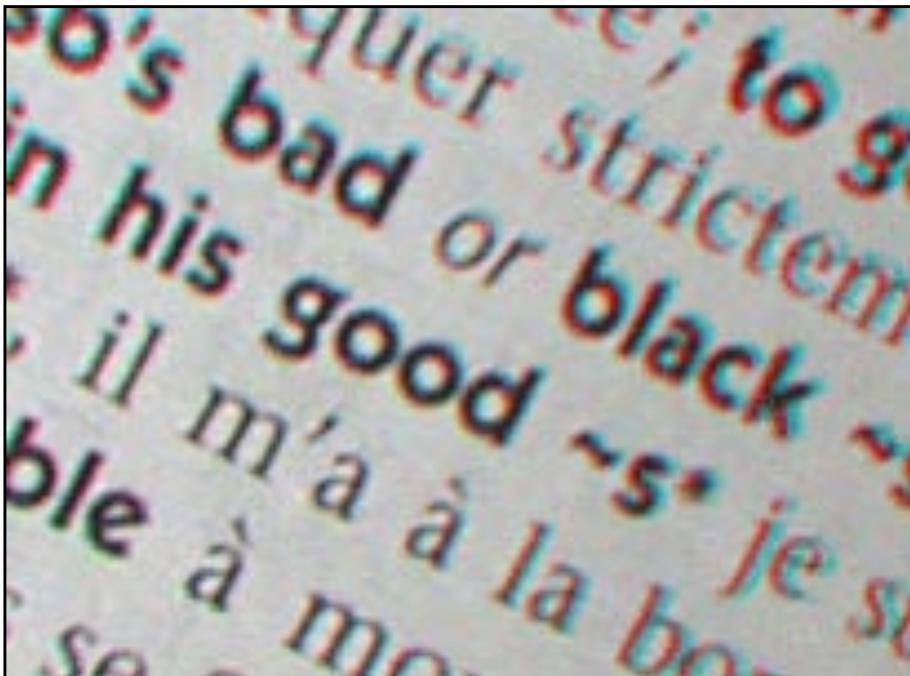


Narrow field of view
(large f)

$$r_b = \frac{a}{s'} \left| \frac{f^2}{(d' - f)(d - f)} \right|$$

Small $f \rightarrow$ Small r_b

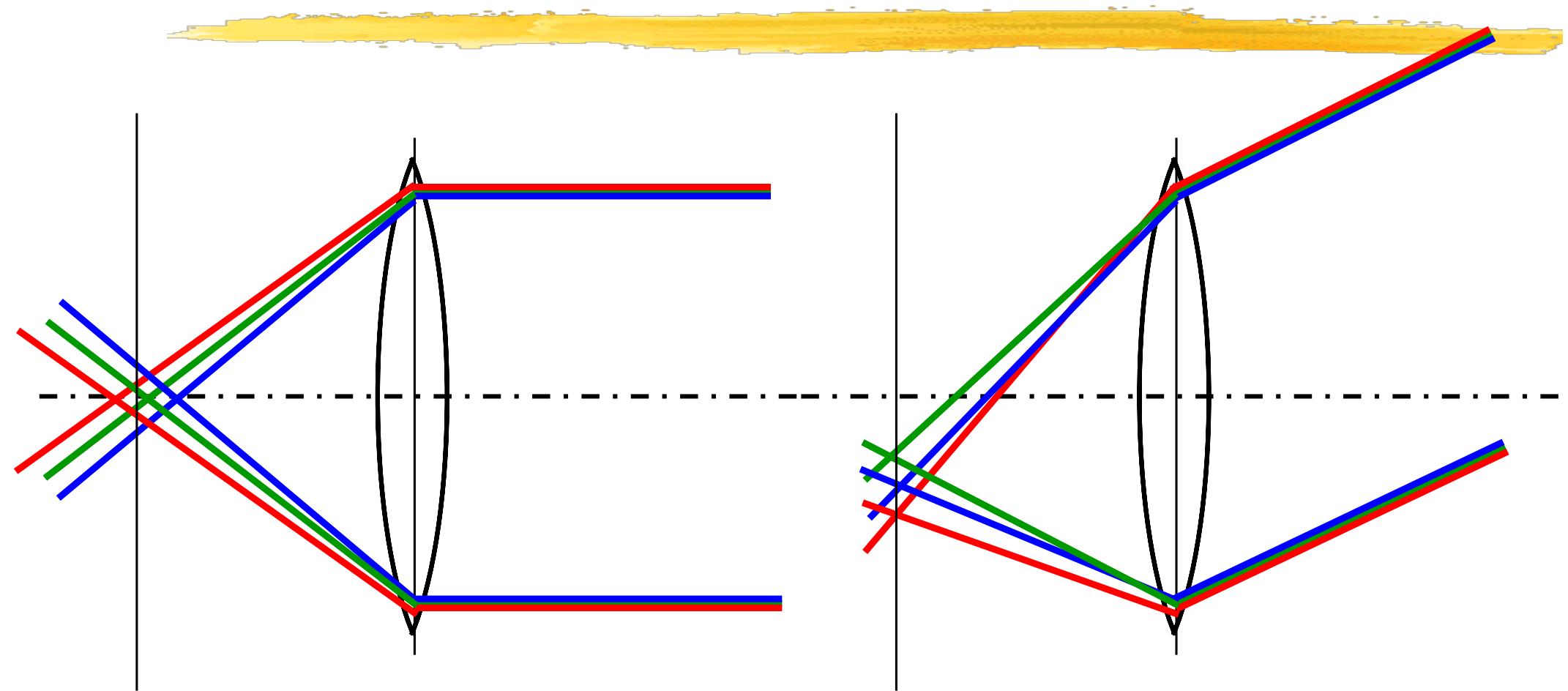
DISTORTIONS



The lens is not exactly a “thin lens:”

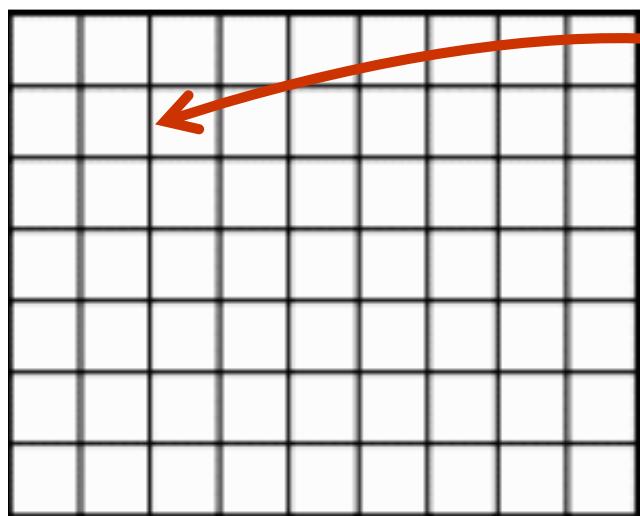
- Different wave lengths refracted differently
- Barrel Distortion

CHROMATIC ABERRATION

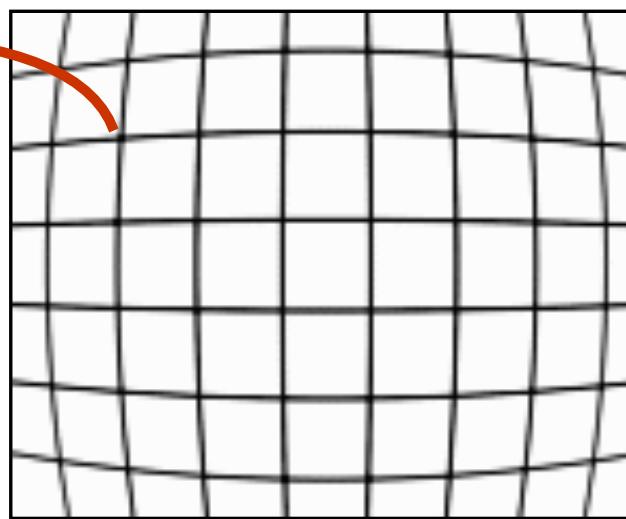


Different wavelengths are refracted differently.

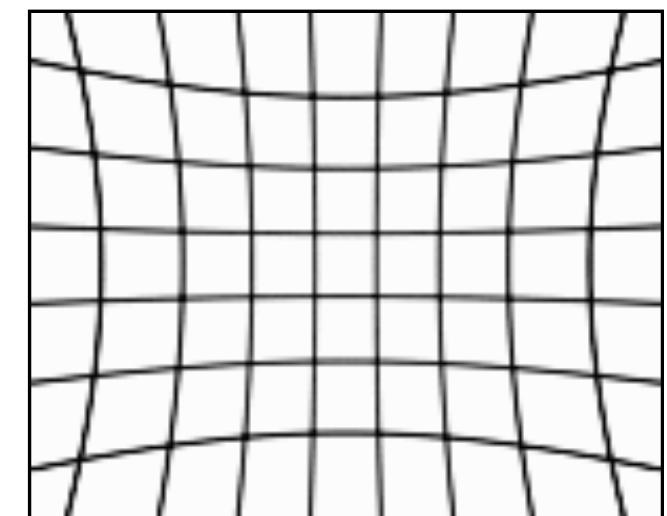
RADIAL LENS DISTORTIONS



No Distortion



Barrel Distortion

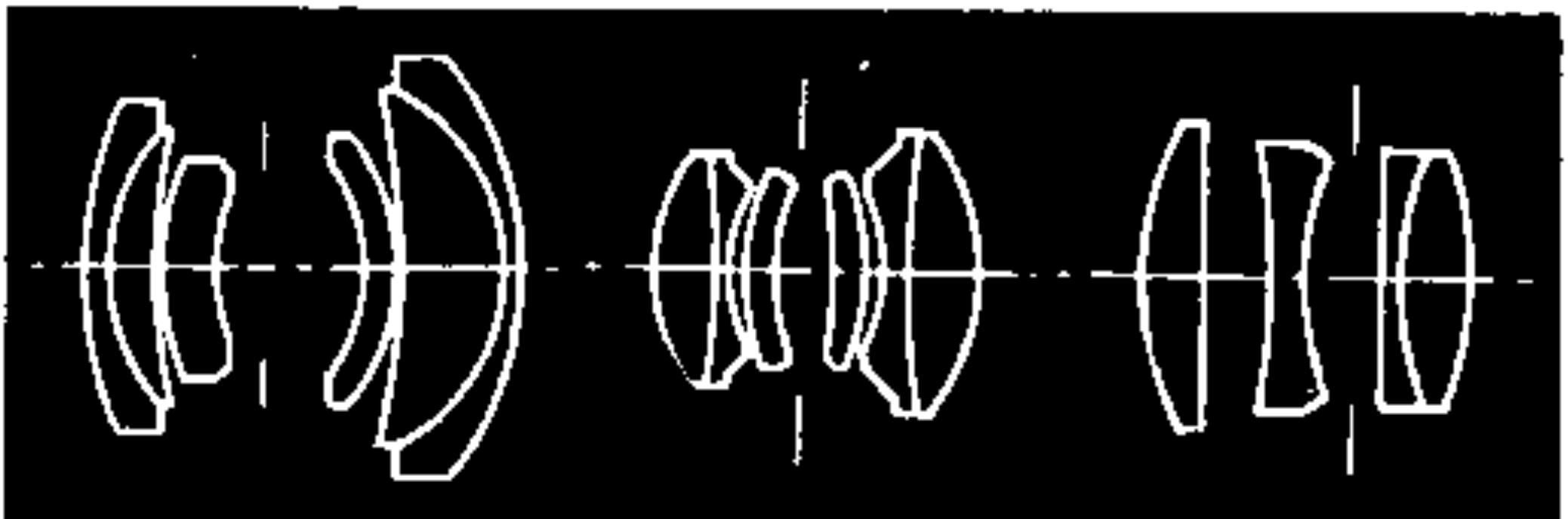


Pincushion Distortion

Radial distance from Image Center:

$$r_u = r_d + k_1 r_d^3$$

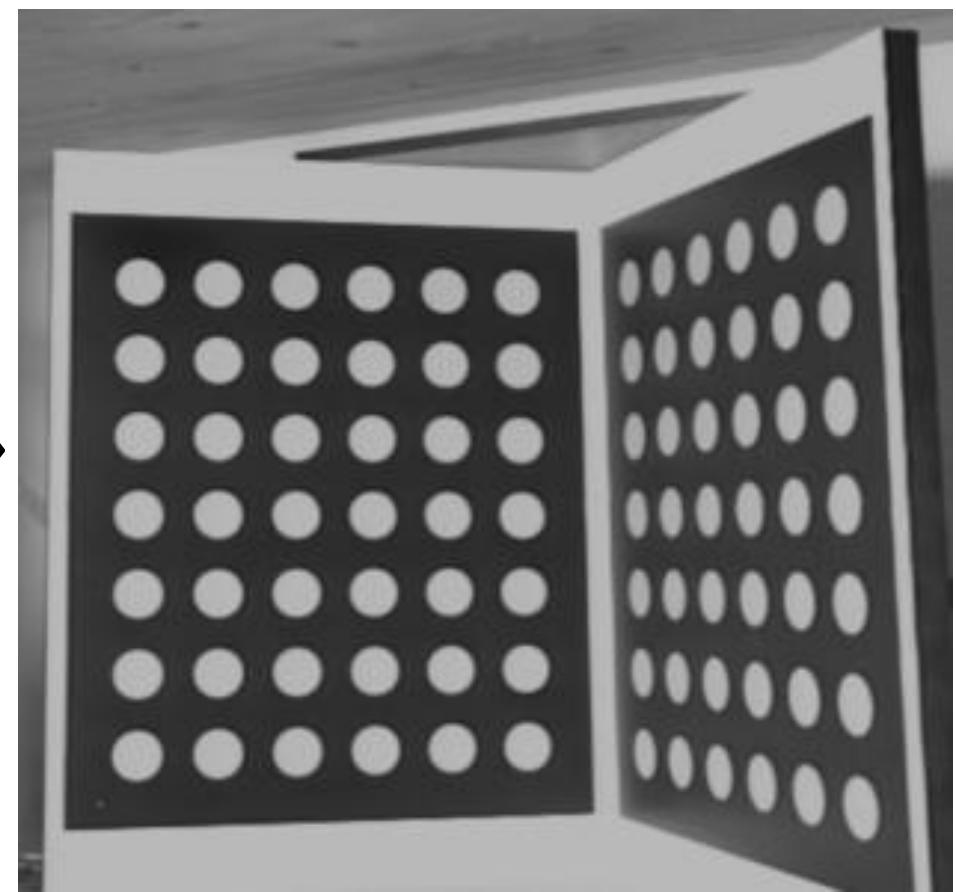
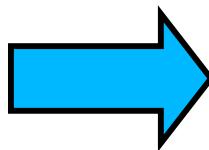
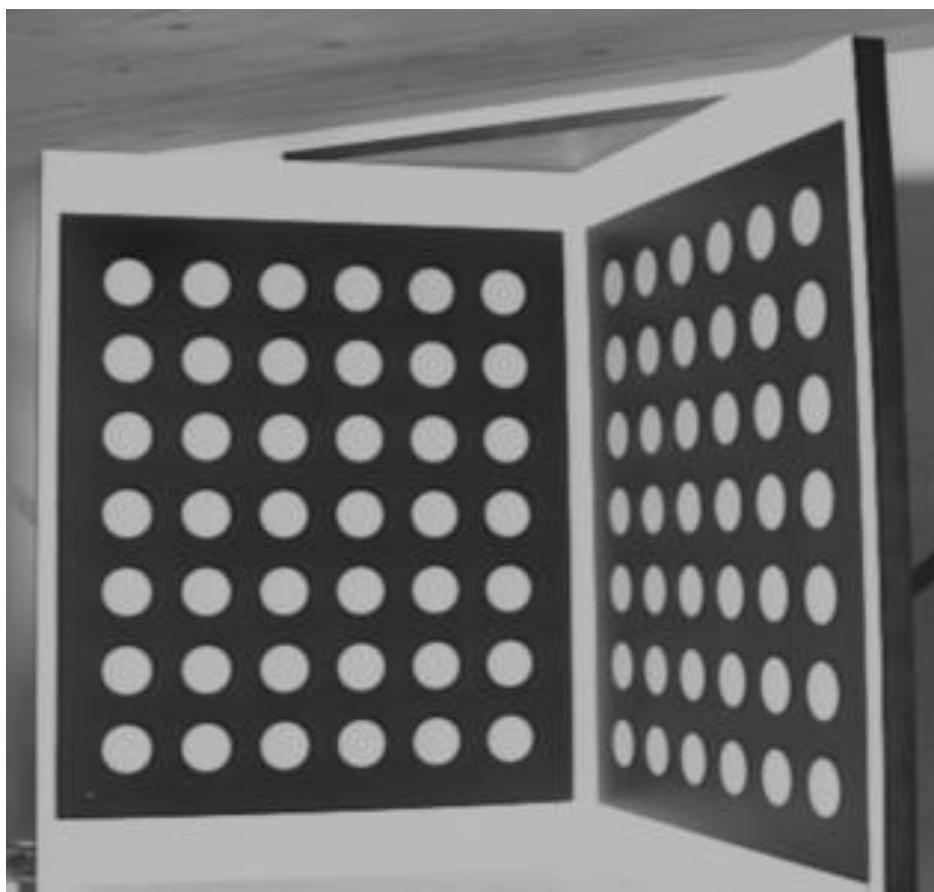
LENS SYSTEMS



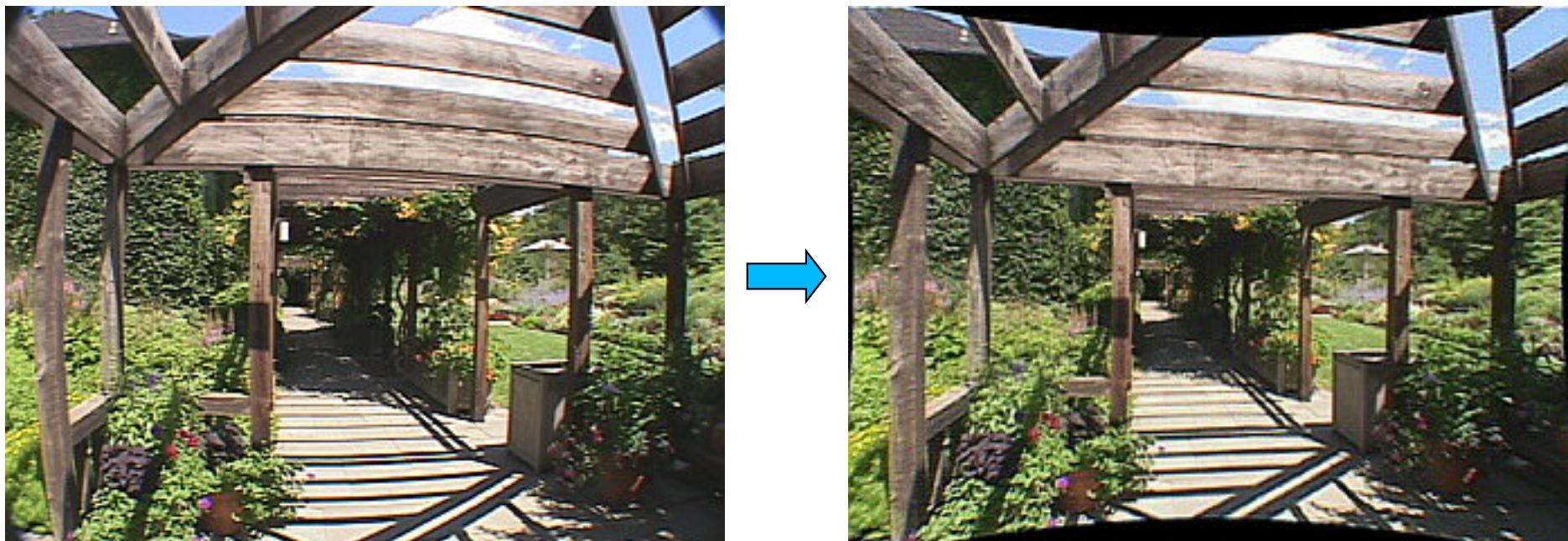
Aberrations can be minimized by aligning several lenses with well chosen

- Shapes,
- Refraction indices.

UNDISTORTING



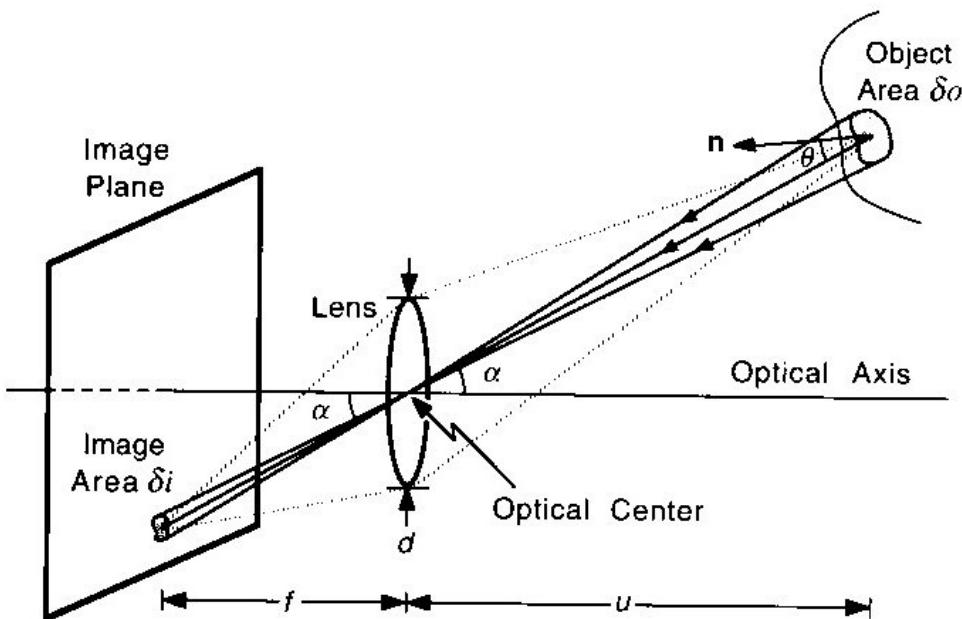
UNDISTORTING



Once the image is undistorted, the camera projection can be formulated as a projective transform.

- The pinhole camera model applies.

RADIOMETRY



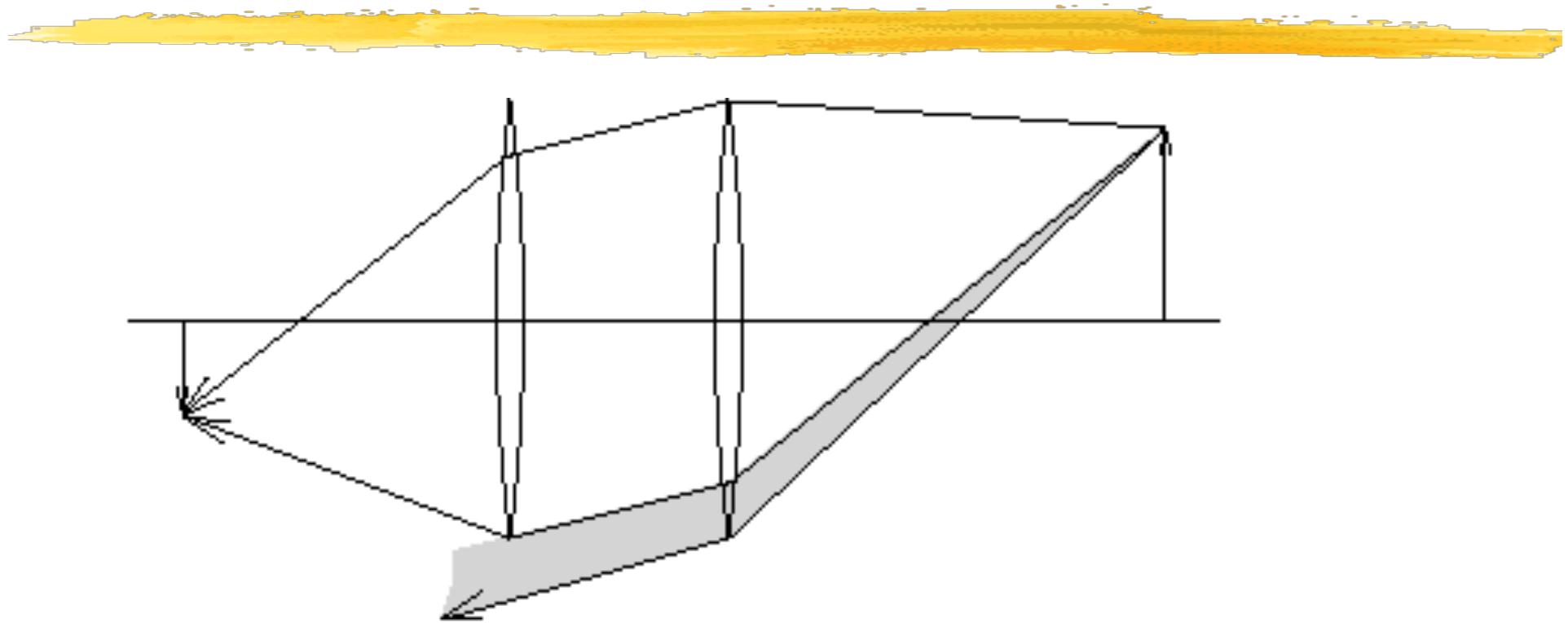
Scene Radiance: Amount of light radiation from a surface point (Watt / m² / Steradian)

Image Irradiance: Amount of light incident at the image of the surface point. (Watt / m²)

Fundamental Radiometric Equation:

$$\text{Irr} = \frac{\pi}{4} \left(\frac{d}{f}\right)^2 \cos^4(a) \text{Rad}$$

VIGNETTING

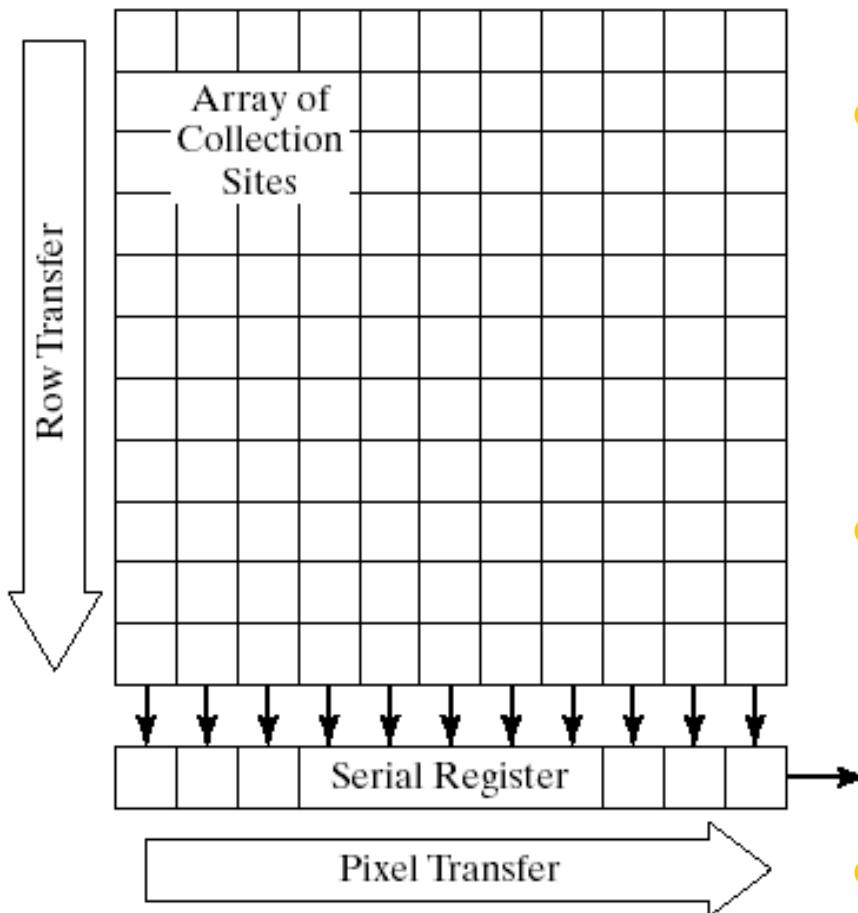


Images can get darker towards their edges because some of the light does not go through all the lenses.

DE VIGNETTING

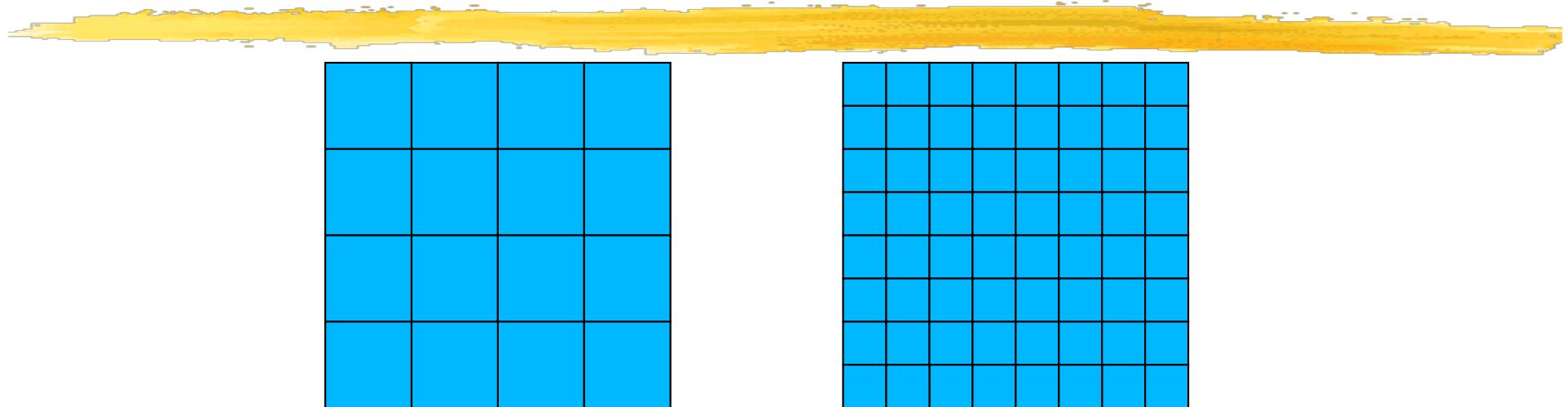


SENSOR ARRAY



- Photons free up electrons that are then captured by a potential well.
- Charges are transferred row by row wise to a register.
- Pixel values are read from the register.

SENSING



Conversion of the “optical image” into an “electrical image”:

$$E(x, y) = \int_{t_0}^{t_1} \int_0^{\Lambda} \text{Irr}(x, y, t, \lambda) s(\lambda) dt d\lambda$$

$$I(m, n) = \text{Quantize}\left(\int_{x_0}^{x_1} \int_{y_0}^{y_1} E(x, y) dx dy\right)$$

→ Quantization in

- Time
- Space

IN SHORT



- Camera geometry can be modeled in terms of the pinhole camera model, which is linear in projective space.
- Image radiance is roughly proportional to surface radiance and the two can be used interchangeably for our purposes.