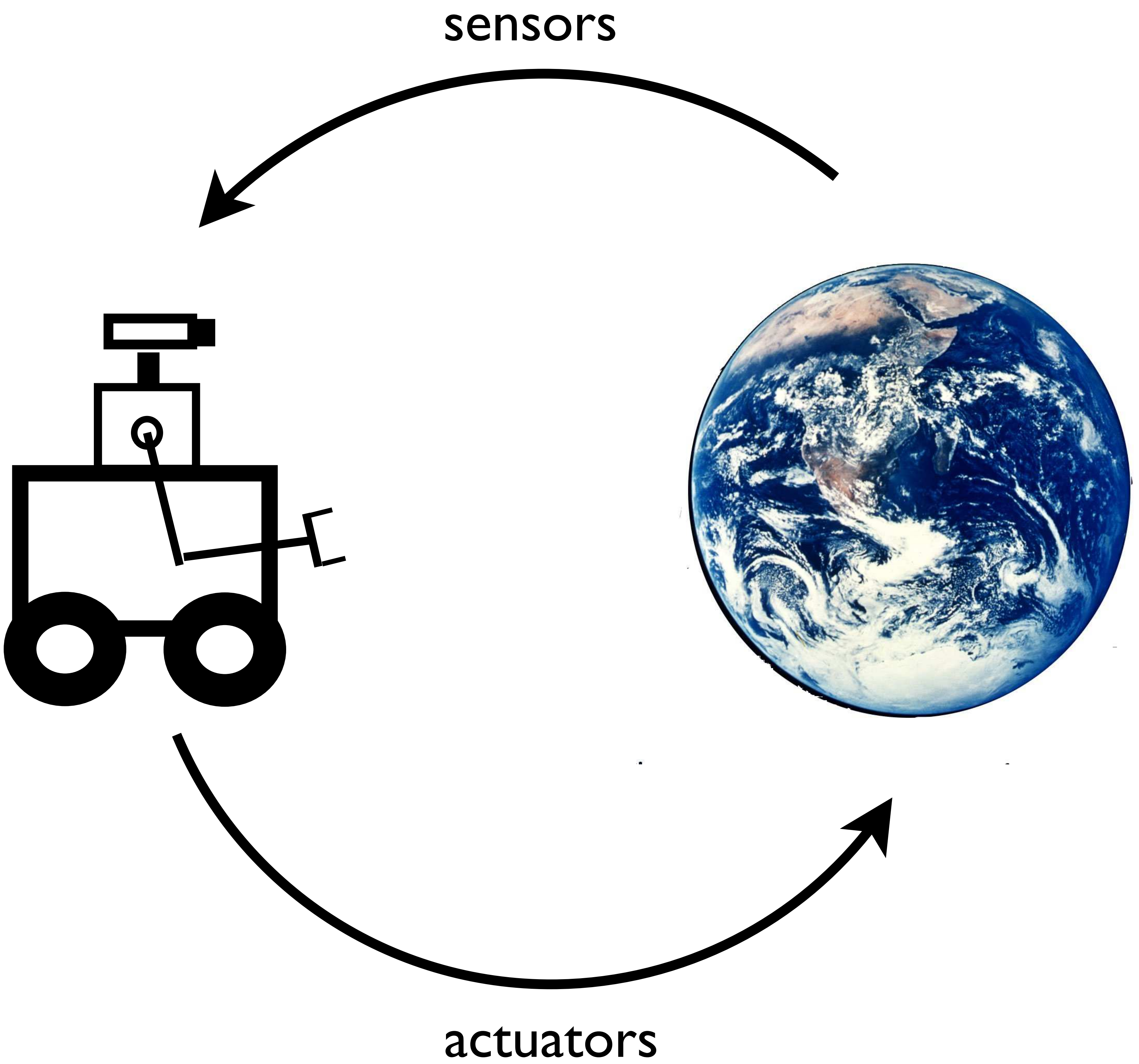
# Computer Vision

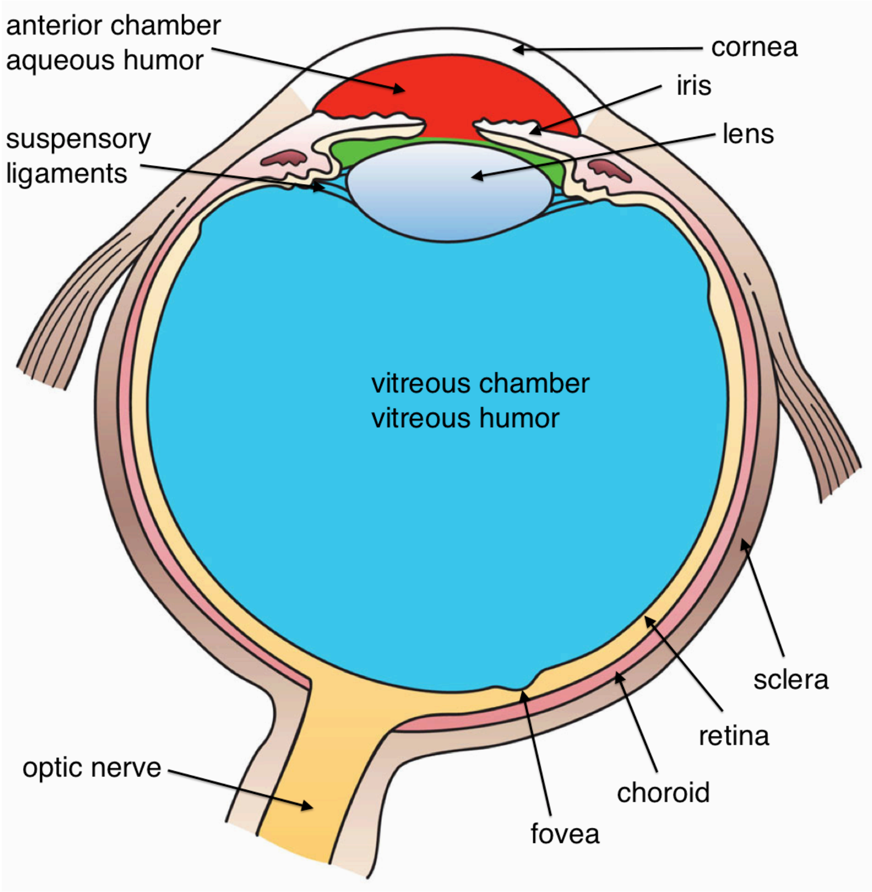
George Konidaris
gdk@cs.brown.edu

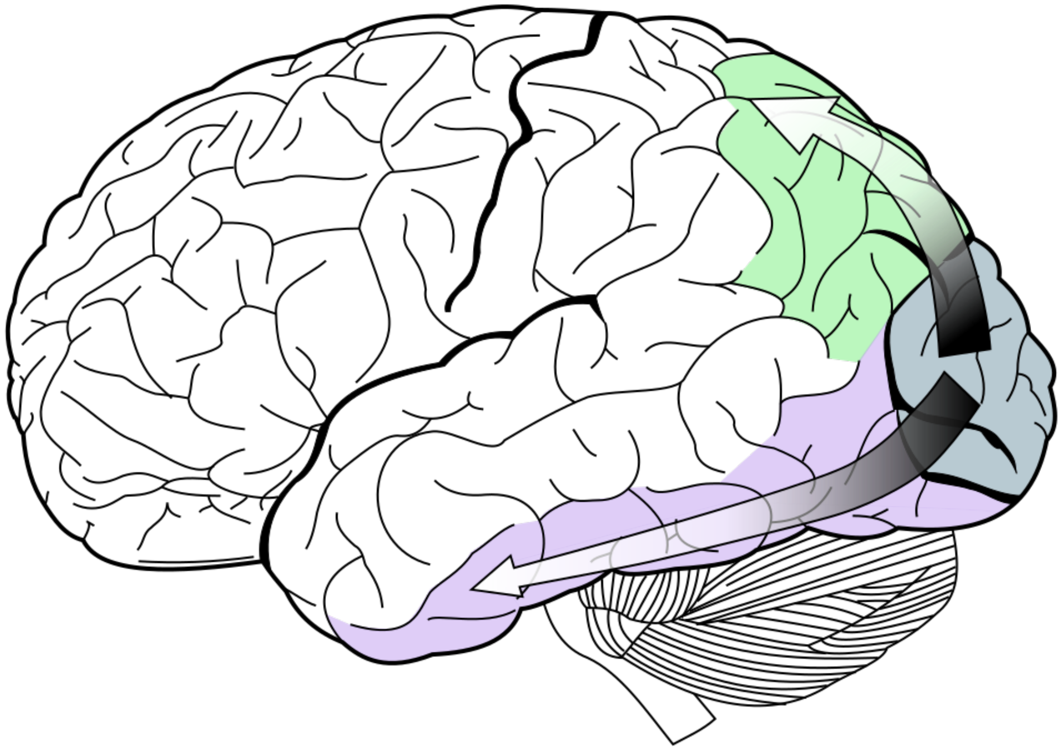
**Fall 2021**

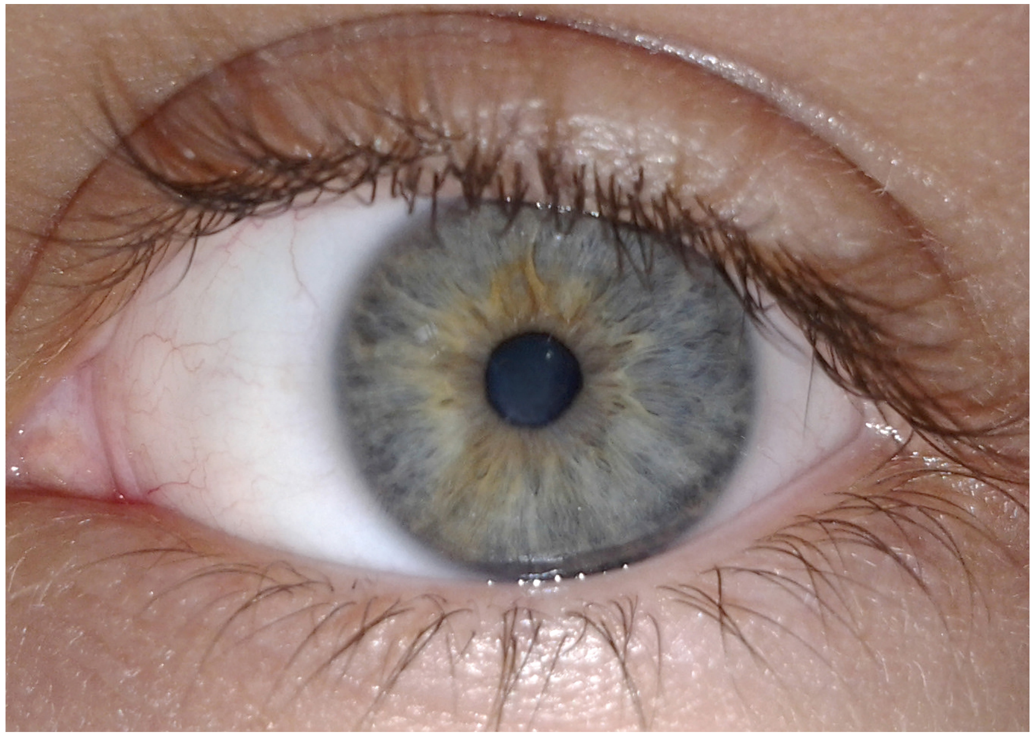


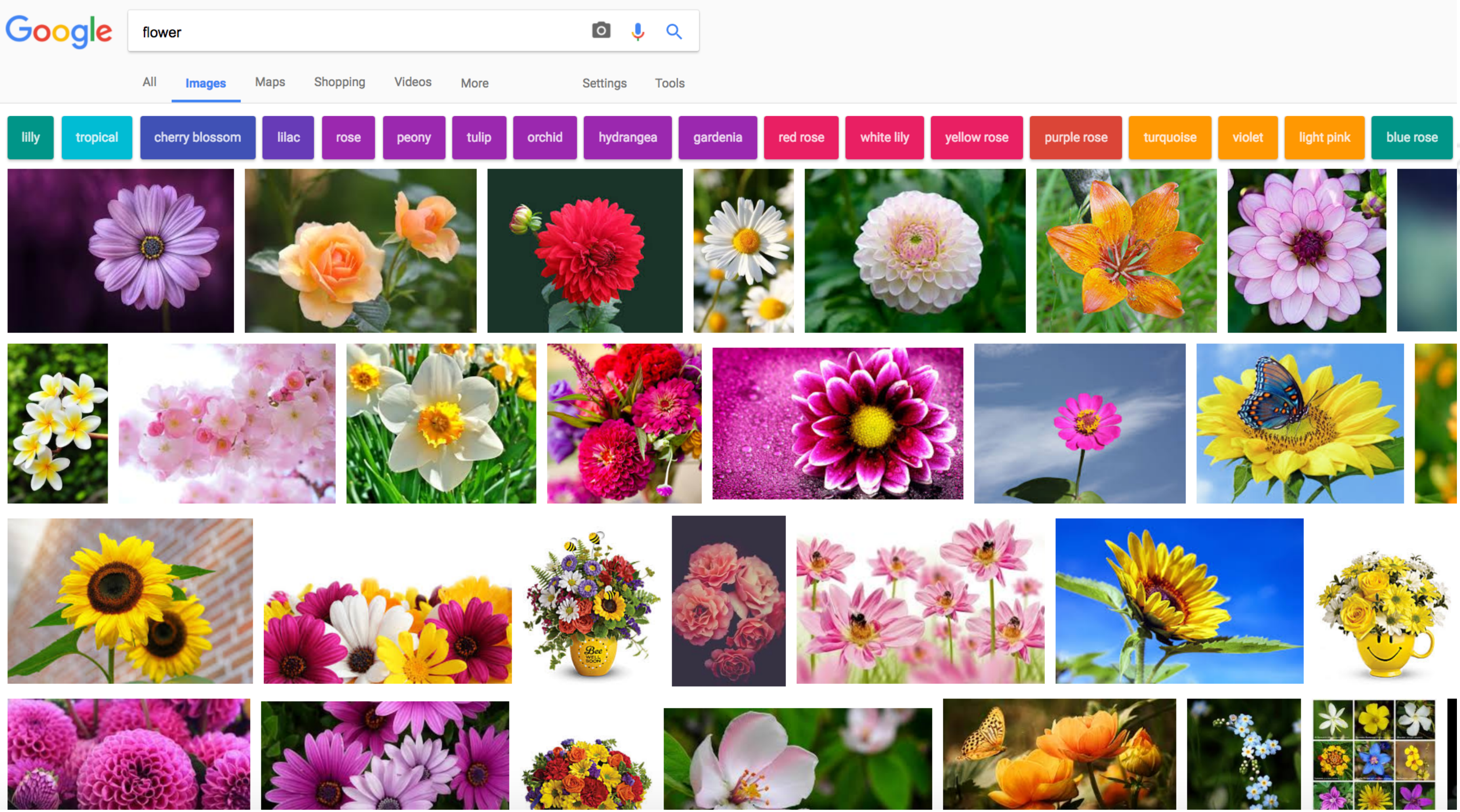
sensors

actuators


anterior chamber
aqueous humor
cornea
iris
lens
suspensory
ligaments
vitreous chamber
vitreous humor
sclera
retina
choroid
fovea
optic nerve



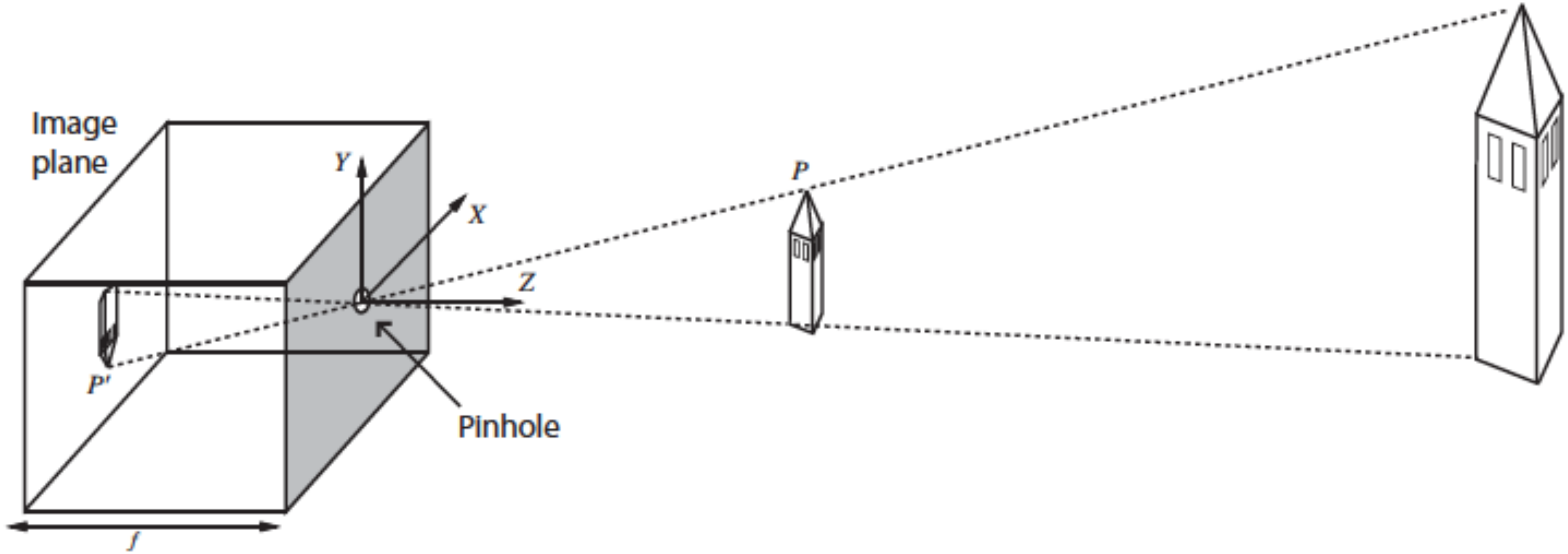




flower

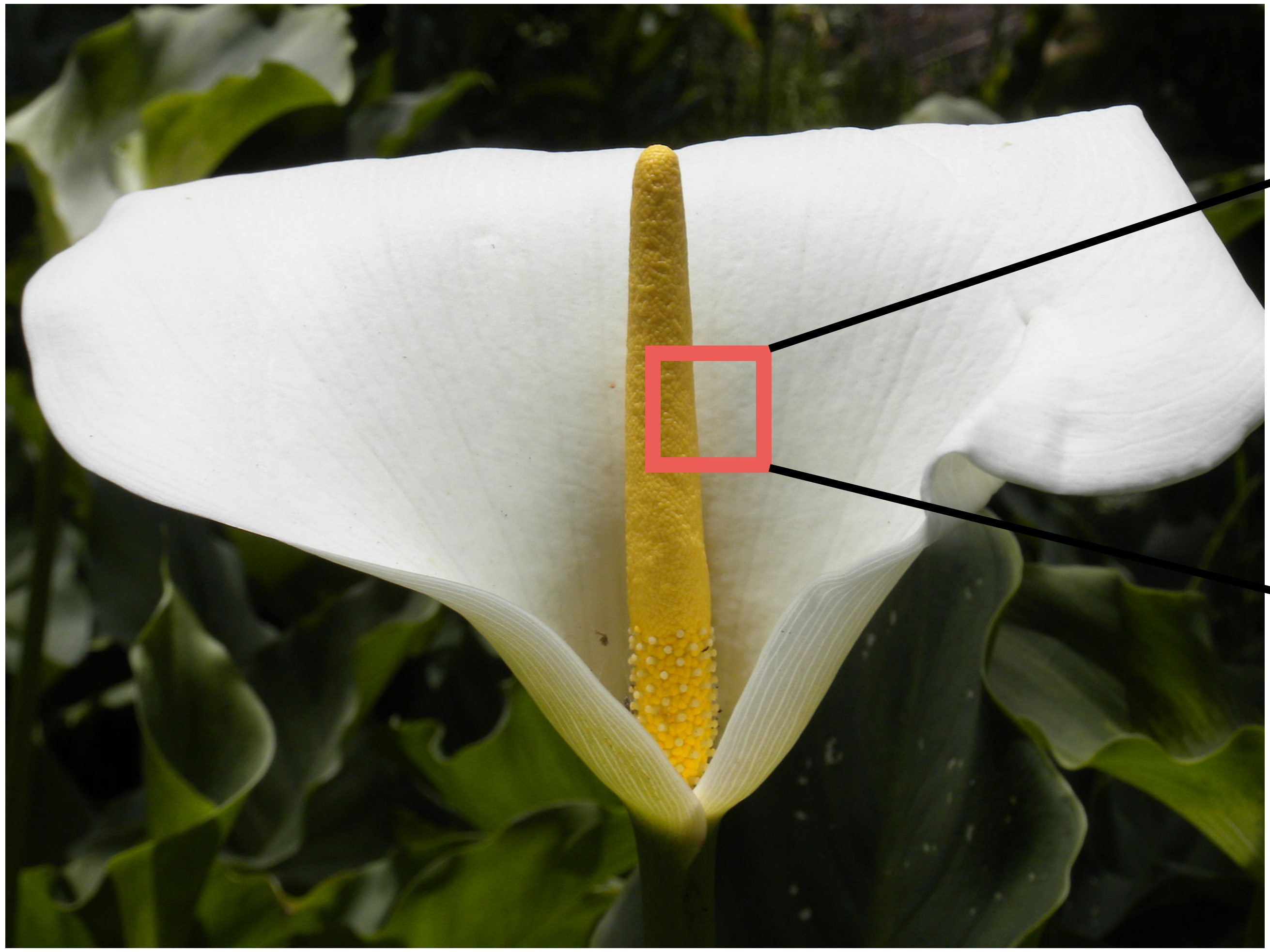
Google
All
Images
Maps
Shopping
Videos
More
Settings
Tools
lilly
tropical
cherry blossom
lilac
rose
peony
tulip
orchid
hydrangea
gardenia
red rose
white lily
yellow rose
purple rose
turquoise
violet
light pink
blue rose
Bee WELL SOON

# Image Capture



[R&N]

# What’s an Image?



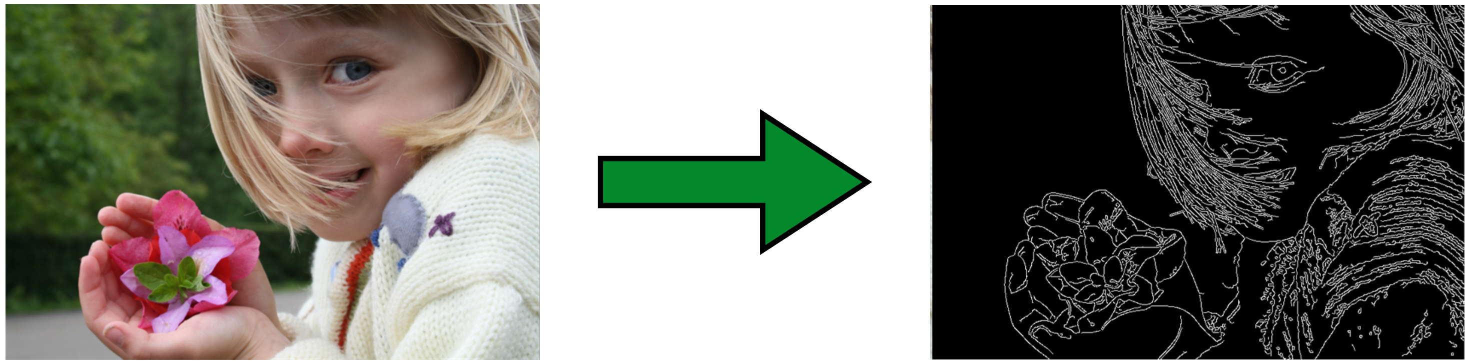
. . .

| | | | | |
|---|---|---|---|---|
| | 131,122,240 | 131,126,224 | 231,222,240 | |
| | 91,112,226 | 91,116,211 | 246,236,243 | |
| . . . | 84, 91,220 | 141,122,216 | 251,244,241 | . . . |
| | 136,132,210 | 112,134,234 | 235,235,240 | |
| | 126,134,220 | 108,101,224 | 254,241,246 | |

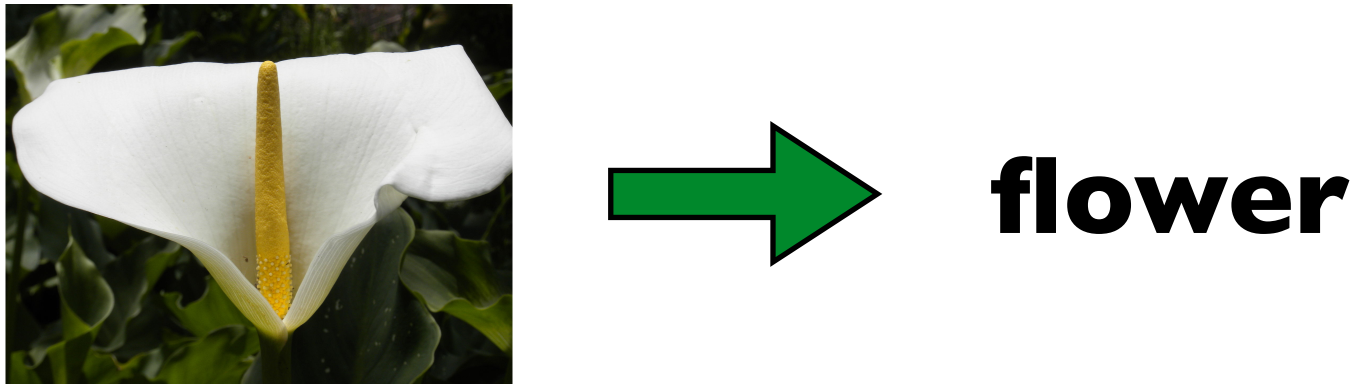
. . .

# Computer Vision

## Image preprocessing

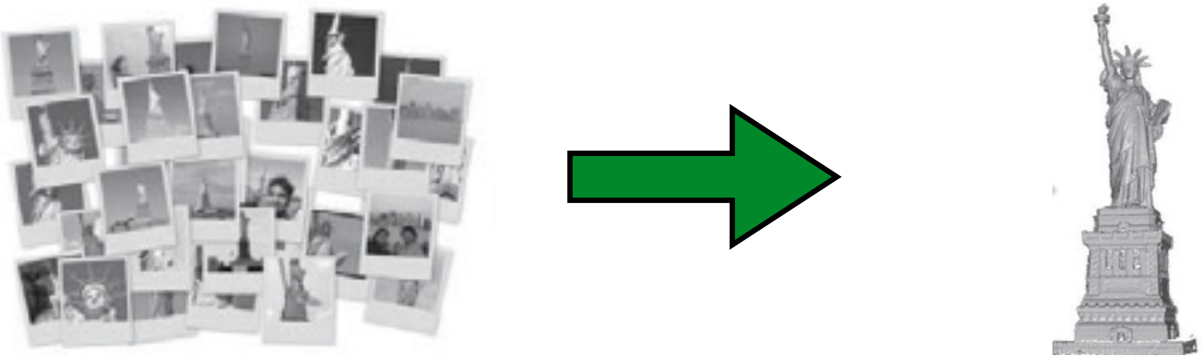


## Recognition



flower

## Reconstruction



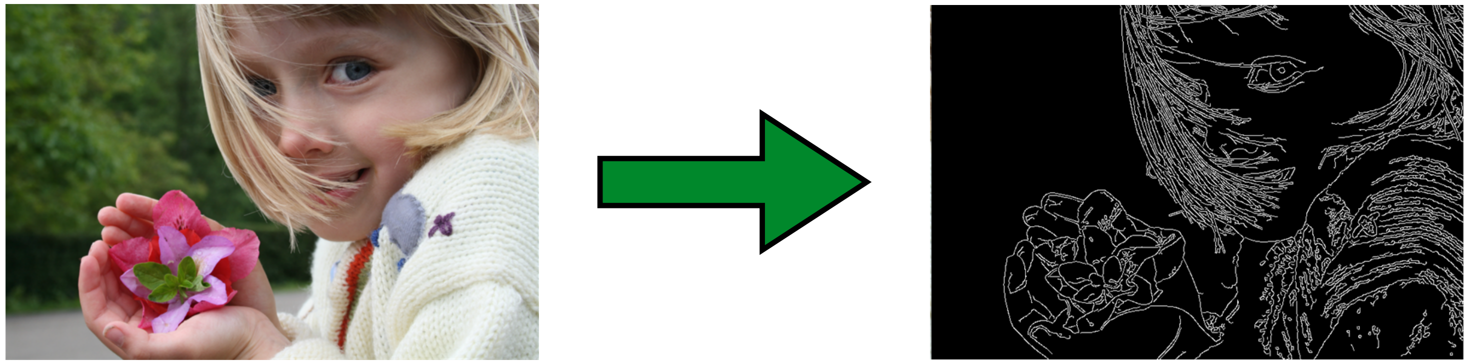
[R&N]

# Image Preprocessing

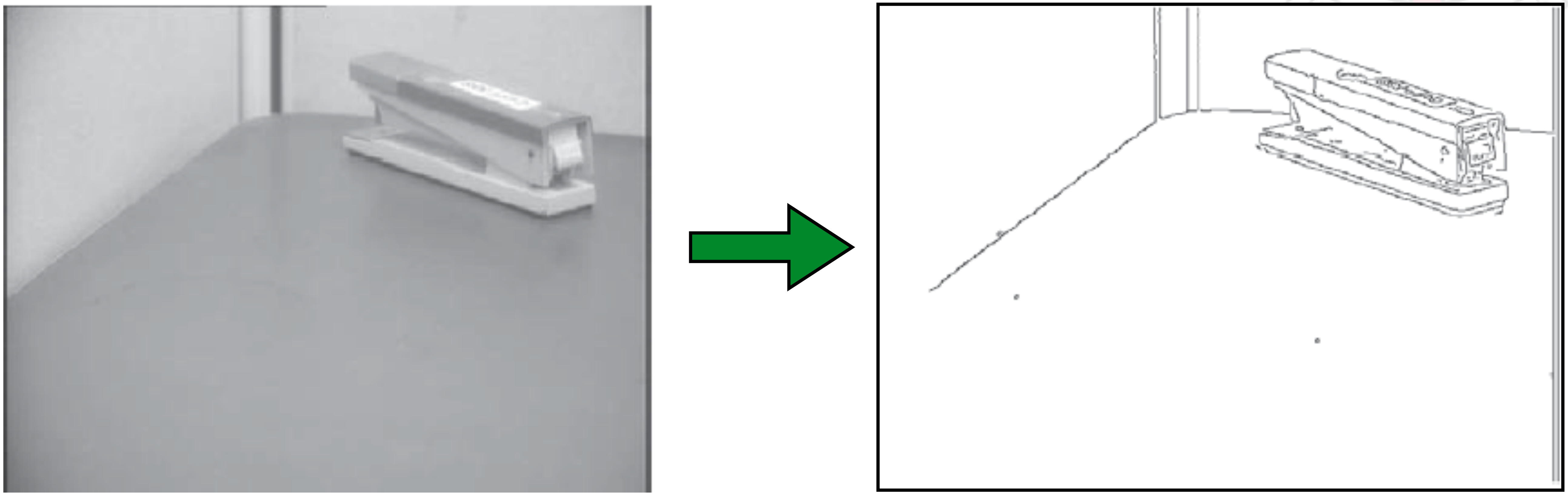
*Collection of methods*

Typically:

- Low-level
- Repetitive
- Local
- Easy to parallelize
- Serve as input to later processing



# Edge Detection



[R&N]

# What's an Edge?

*"Edges are straight lines or curves in the image space across which there is a "significant" change in image brightness."*

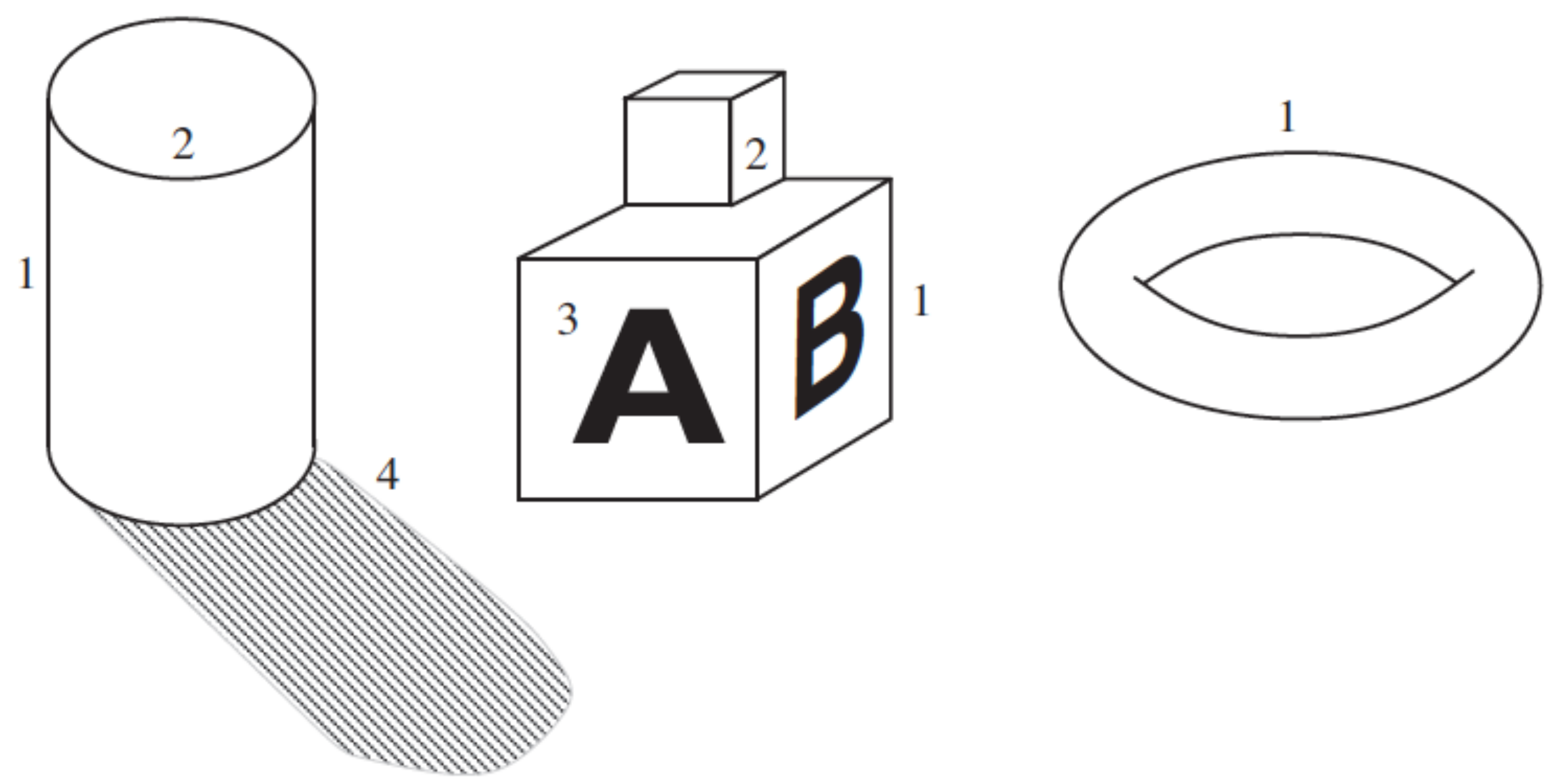


**Figure 24.6** Different kinds of edges: (1) depth discontinuities; (2) surface orientation discontinuities; (3) reflectance discontinuities; (4) illumination discontinuities (shadows).

[R&N]

# Finding Edges

That gives us a hint!
Compute the derivative of brightness with respect to position.

Brightness:

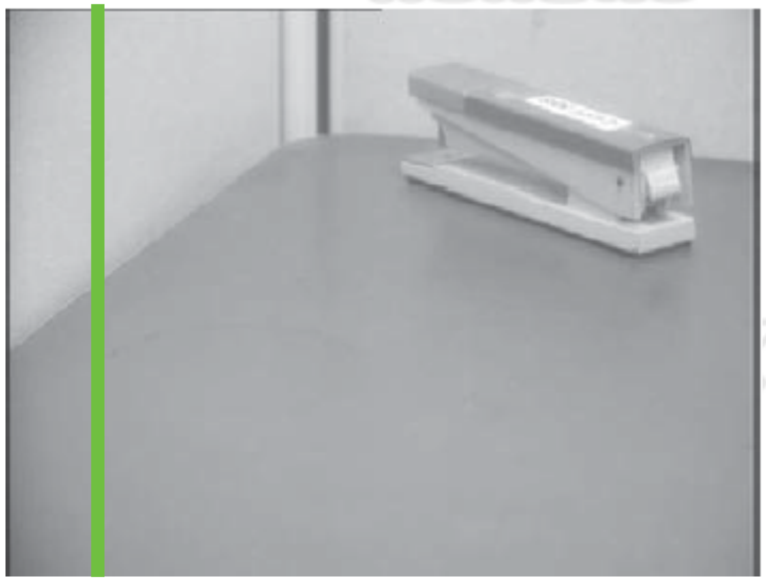
- Average RGB pixel values:

$$BIm(x, y) = (Im(x, y).r + Im(x,y).g + Im(x, y).b)/3$$

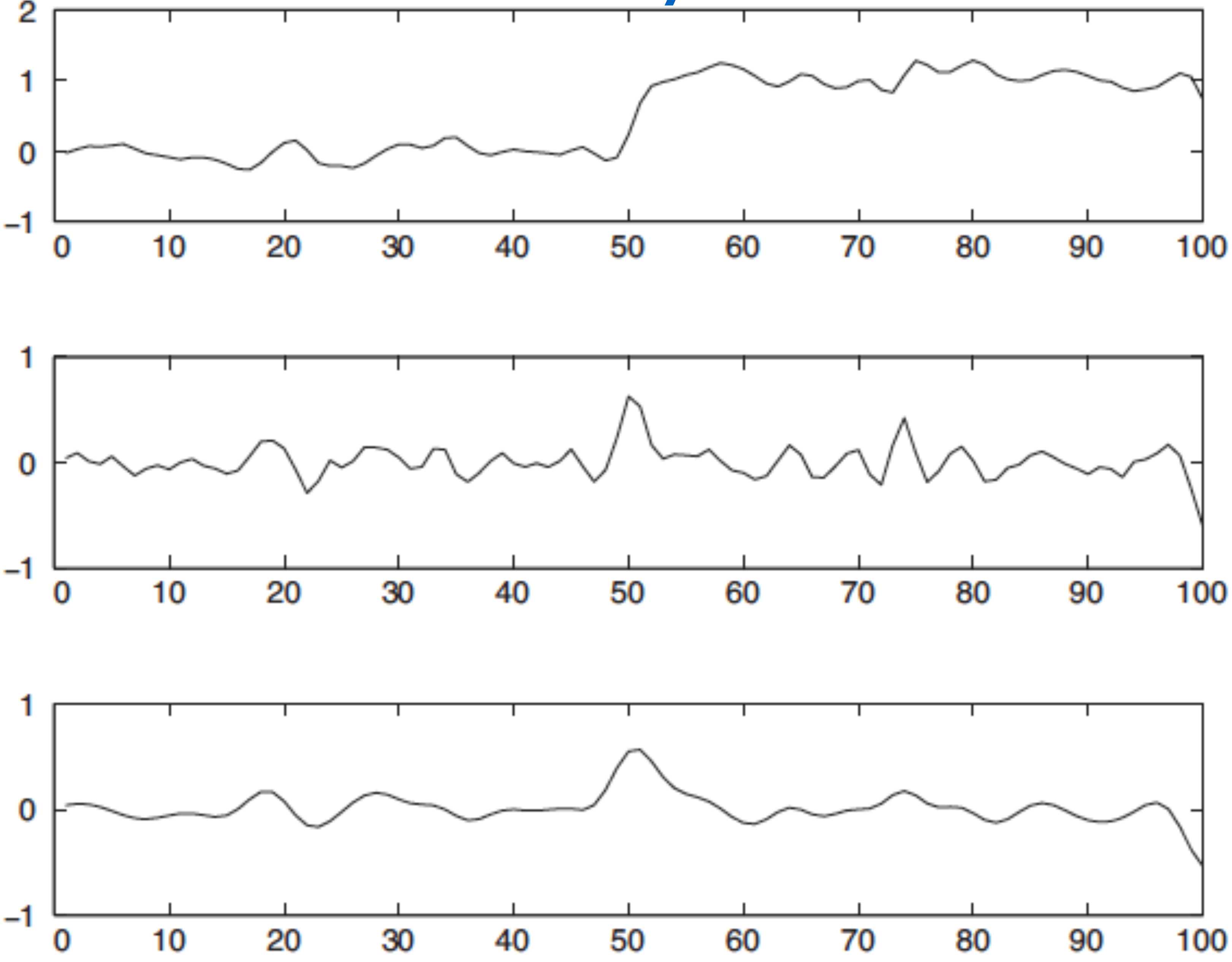
Derivative:

- Take a vertical slice of the image $H_i = BIm(i, :)$
- Compute brightness difference between $H_i(x)$ and $H_i(x+1)$

# Finding Edges



intensity



derivative

smoothed derivative

[R&N]

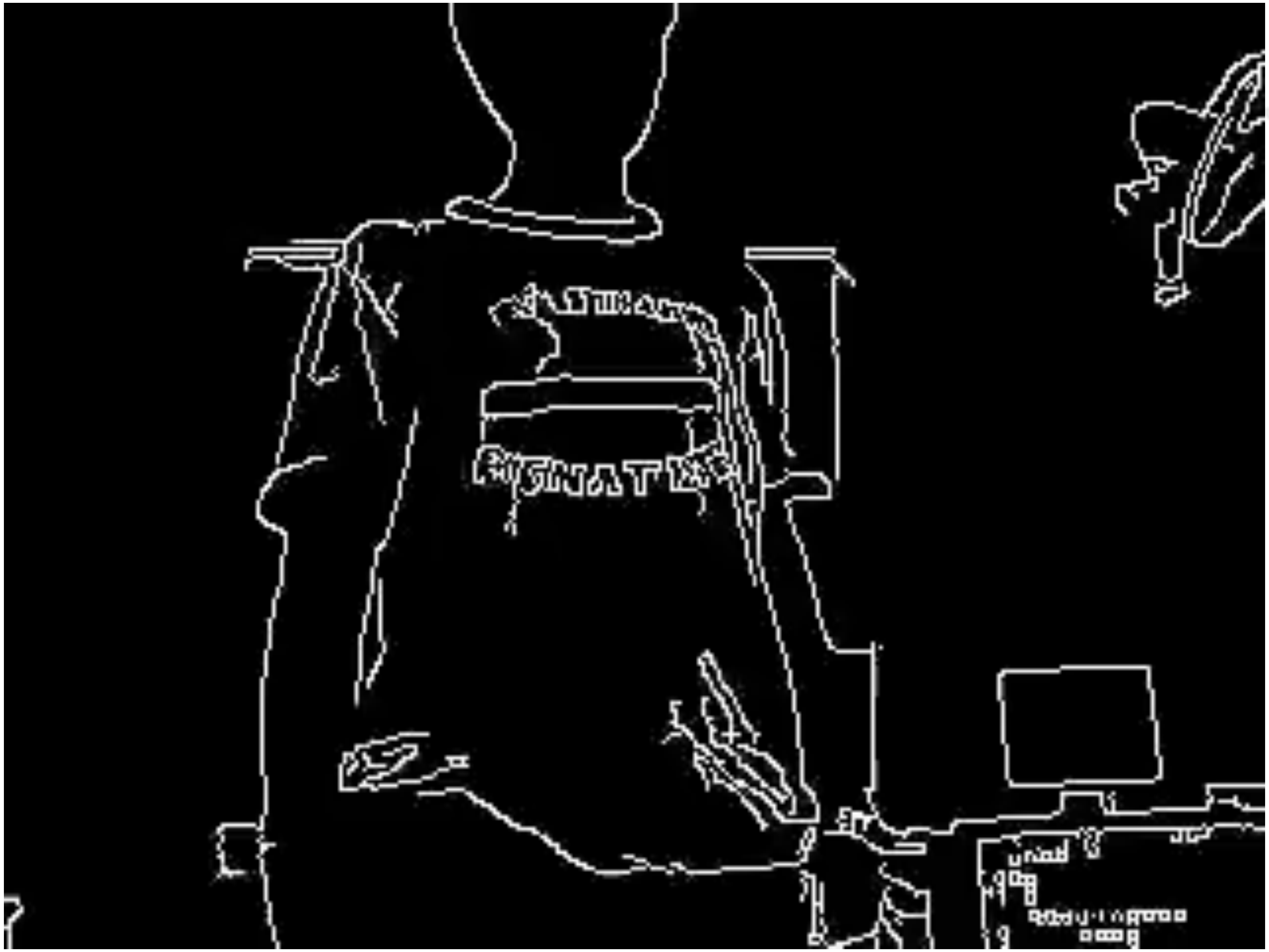
# Canny Edge Detector

Classic and very accurate edge detector.

Five steps:

- Gaussian filter to smooth image (reduce noise)
- Find intensity gradients (horizontal, vertical, diagonal)
- Non-maximum suppression
- Threshold to get edges
- Edge tracking: keep only "connected" edges.

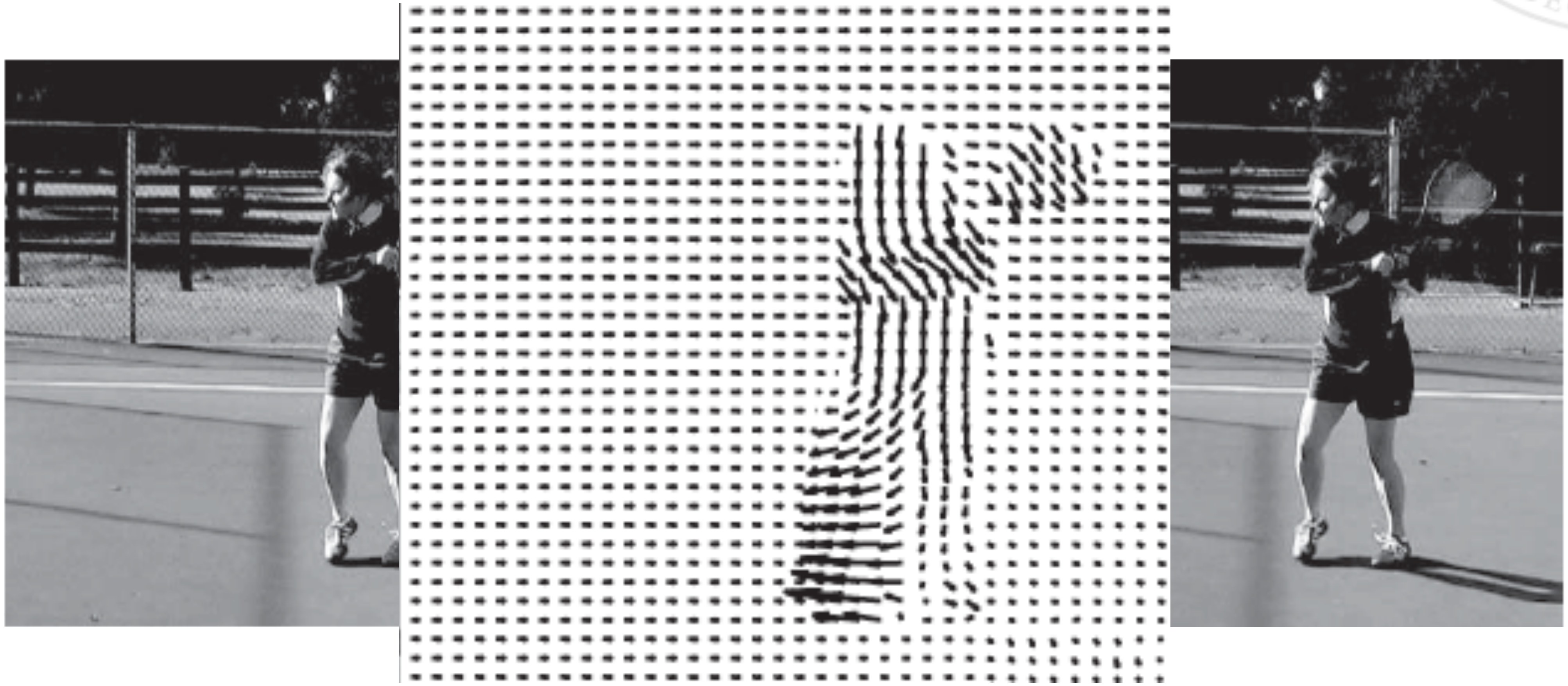
# Canny Edge Detection



[via Michael Jacob Matthew, Youtube]

# Optical Flow

Useful for understanding movement



[R&N]

# Optical Flow

Formally!

Given two images $I_1$ and $I_2$

- Produce optical flow field F
  - $F(x, y) = (dx, dy)$
  - where pixel $I_1[x,y]$ moves to $I_2[x + dx, y + dy]$

This boils down to finding ***correspondences***.

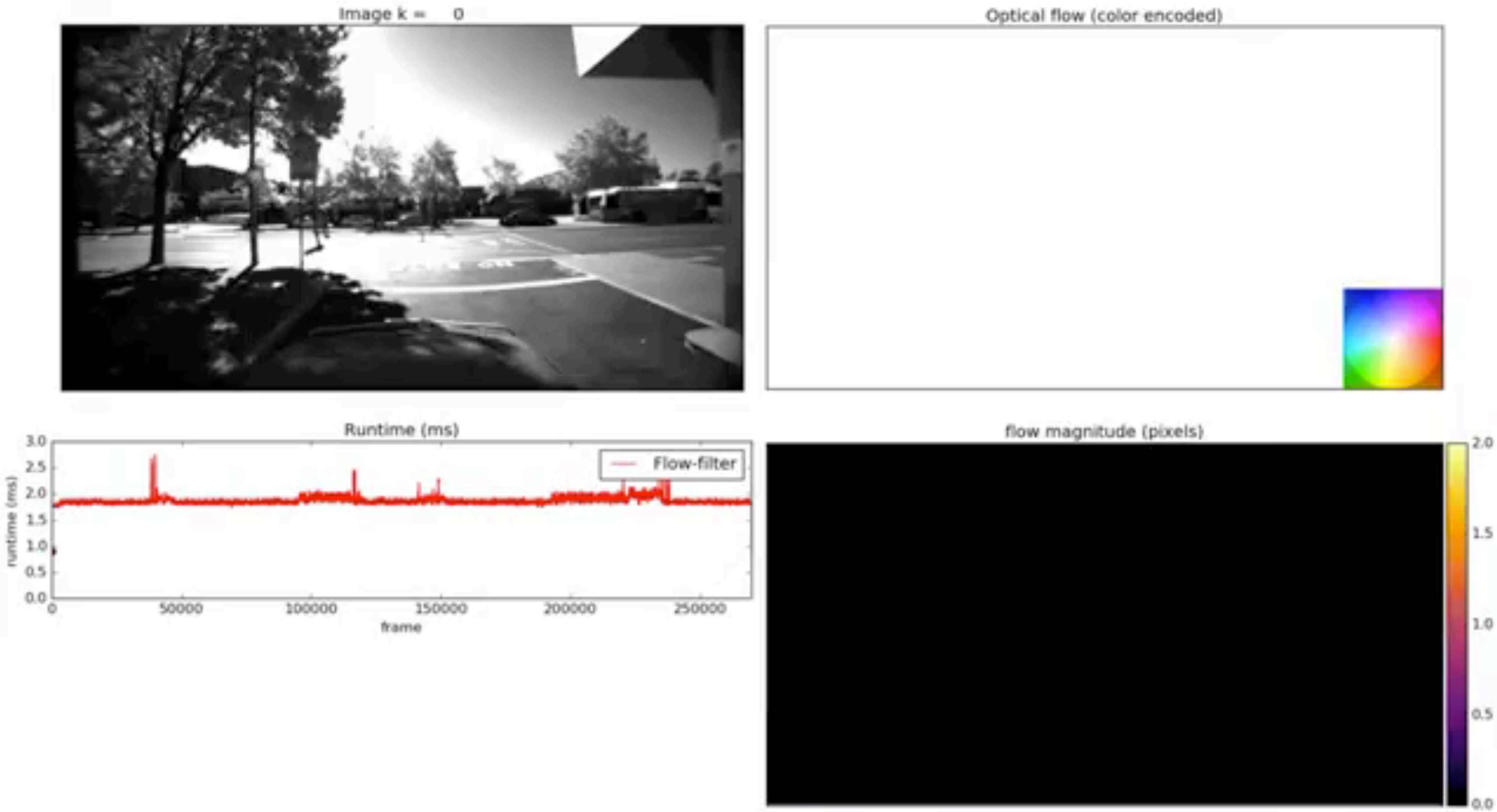
One approach

- Find correspondences that minimize "patch" error
- Regularize for smaller movements

# Optical Flow

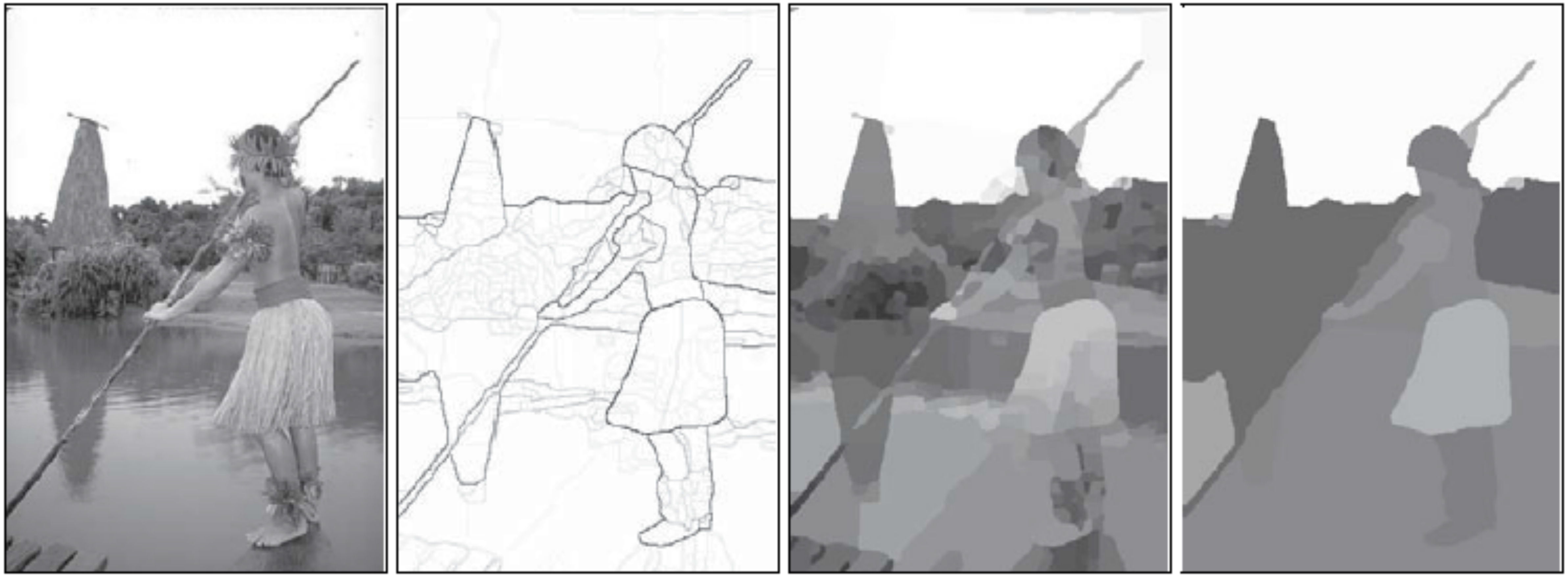
[via Matthieu Garrigues, YouTube]

# Optical Flow



[via Juan Adarve, YouTube]

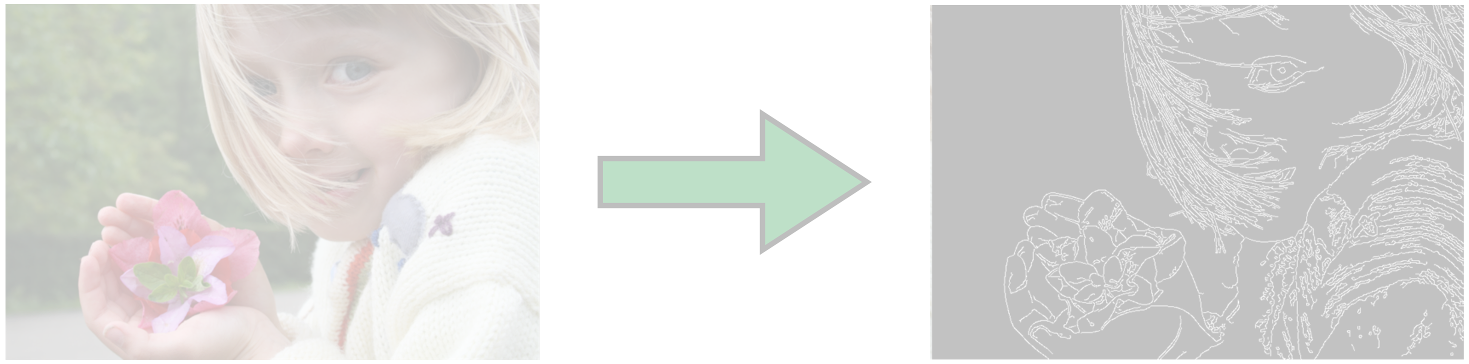
# Image Segmentation



[R&N]

# Computer Vision

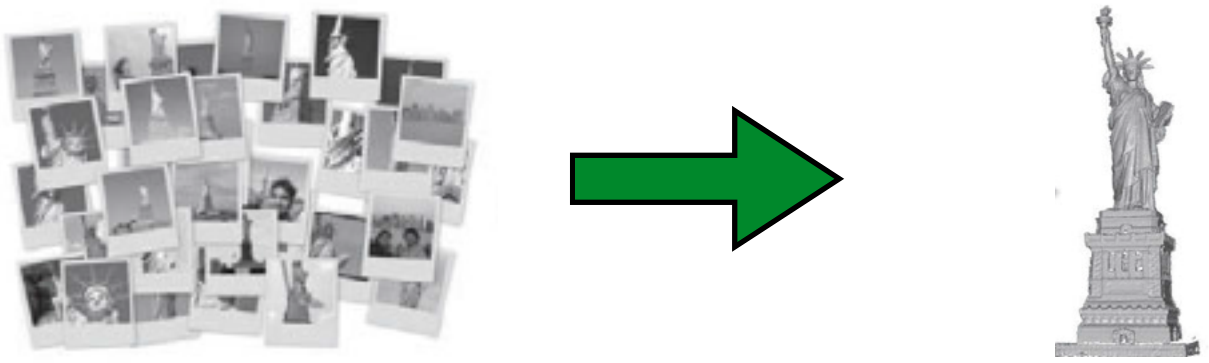
## Image preprocessing



## Recognition

flower

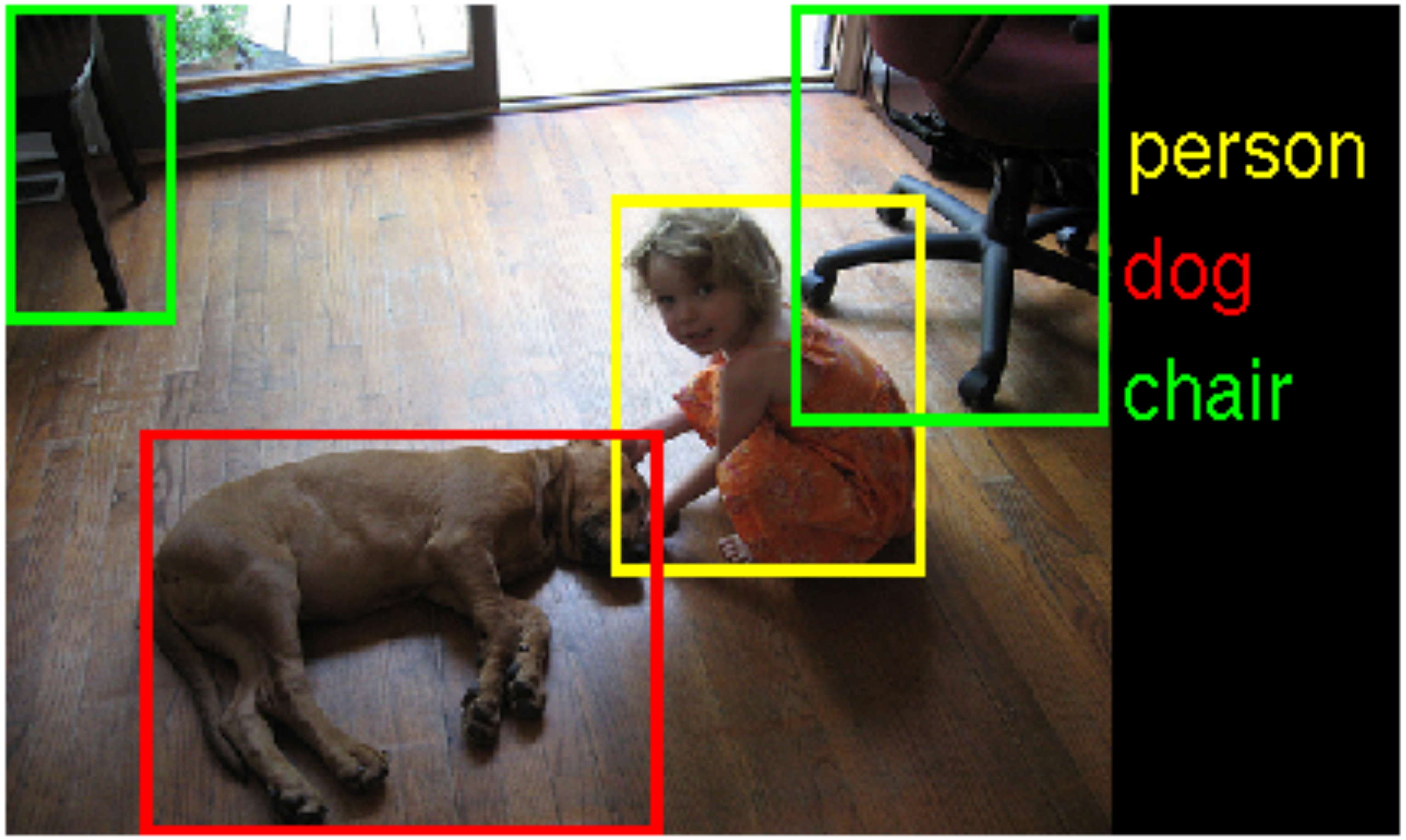
## Reconstruction



[R&N]

# Recognition

→ **flower**



[ImageNet]

# Recognition

Given:

- Object classes $O_1, \ldots, O_n$
- An image size $I$
- A collection of labeled data points $\{I_i, O_i\}_n$

Find:

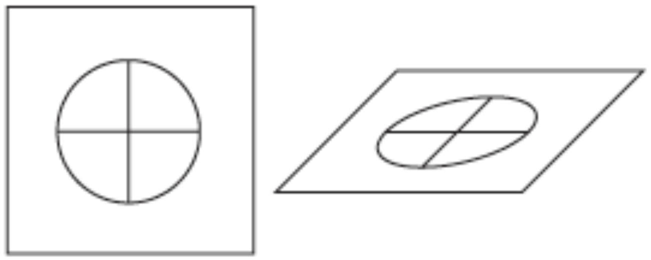
- $f : I \rightarrow O_i$

Minimizing expected error.

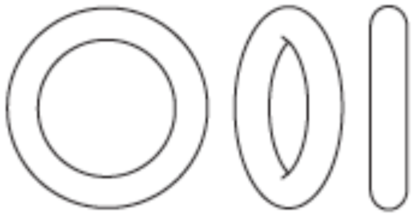
**Classification**

# Recognition

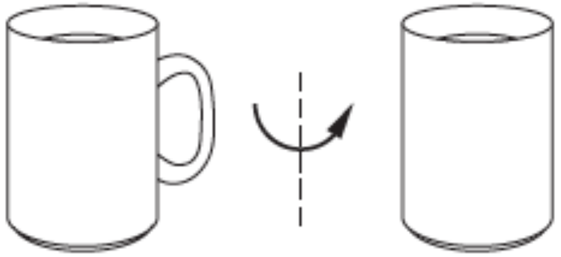
## Why is this hard?



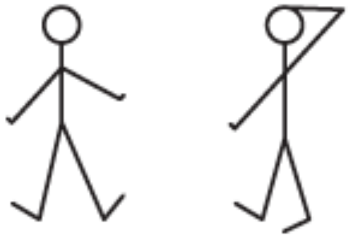
Foreshortening



Aspect



Occlusion



Deformation

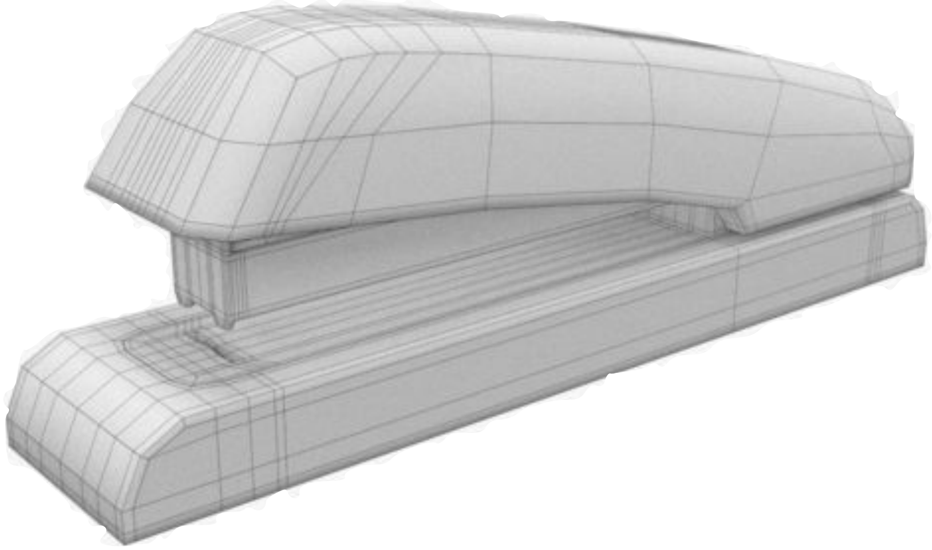
St Bernard's Lily (*Anthericum liliago*)
Bermuda Buttercup (*Oxalis pes-caprae*)
Oleander (*Nerium oleander*)
Lantana (*Lantana camara*)
Scarlet Pimpernel (*Anagallis arvensis*)
Verbascum (Verbascum sinuatum)
Common Mallow (Malva sylvestris)
Spanish Oyster (Scolymus hispanicum)
Stork's bill (*Erodium malacoides*)
Bindweed (*Convolvulus arvensis*)
Blue Gem (*Hebes x franciscana*)
Calla Lily (*Zantedeschia aethiopica*)

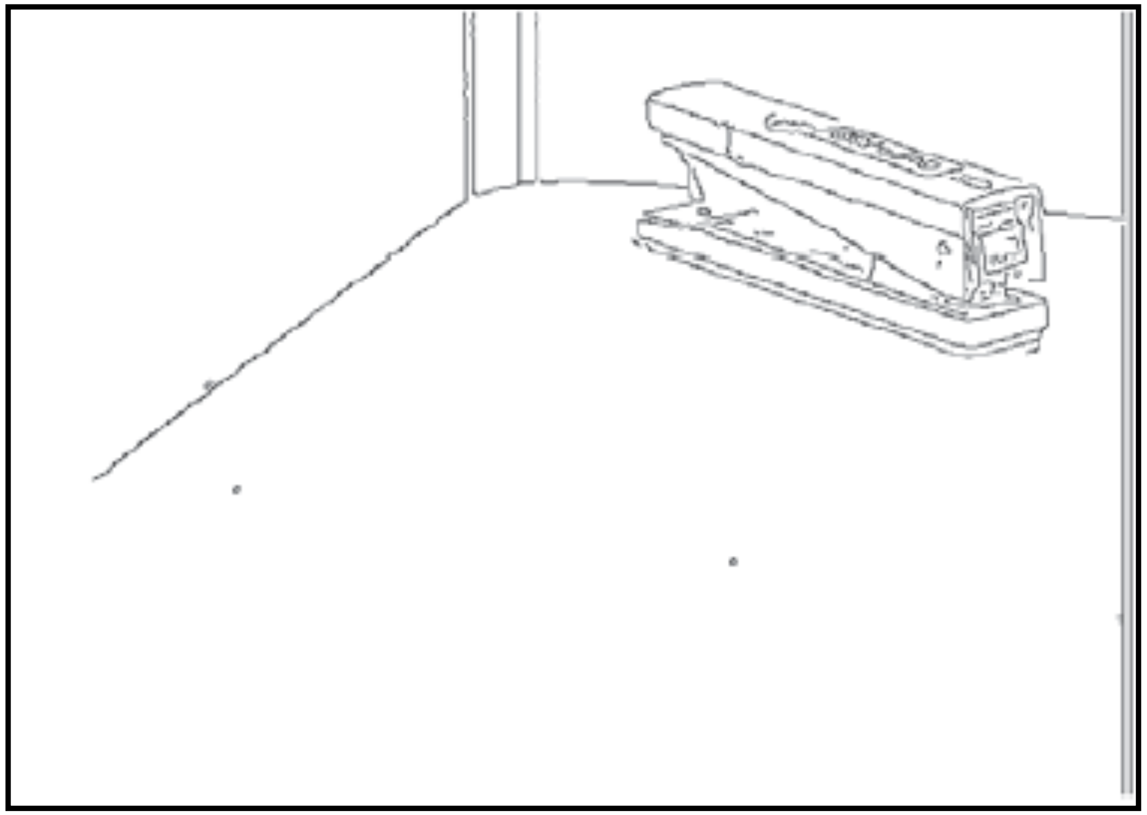
# Recognition

Two main ways of going about this:

- Use a geometric object model
- Use machine learning

First: use an object model to *match* an object in a scene.





# Recognition by ML

Just do ML:

- Get lots of labeled data
- Learn a classifier

Primary challenge:

- Objects of the same class look different
- The same object looks different from different orientations

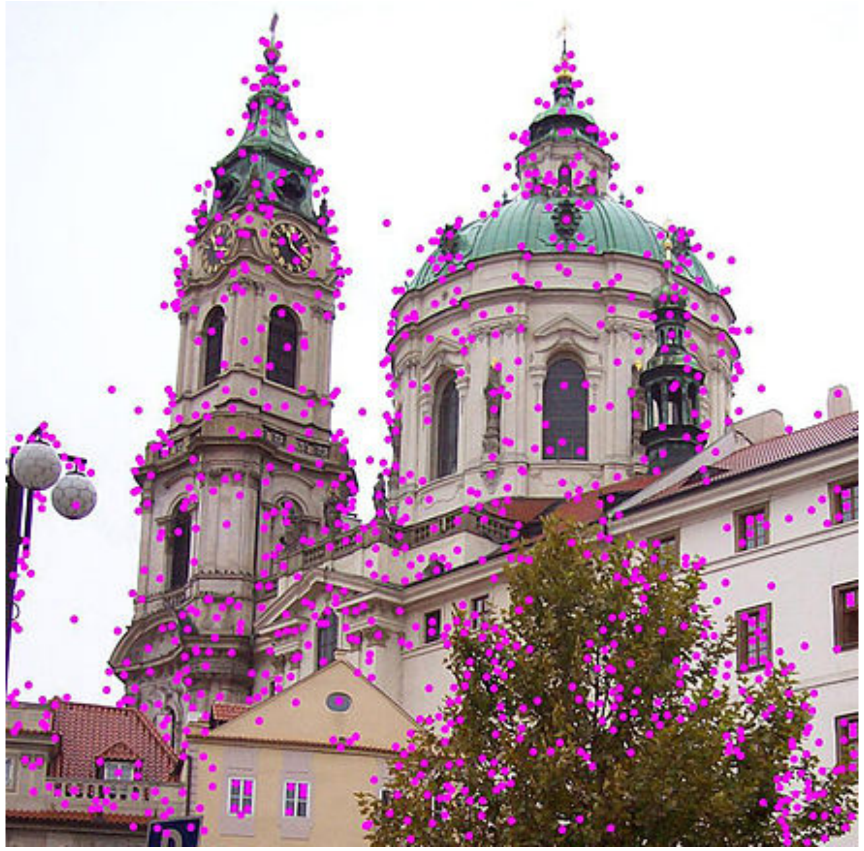
# Recognition by ML

**Solution:**

- Compute features from the image
- Features should be invariant to scale, translation, etc.
  - This is a form of *special knowledge* about images.
- Use these as input to classifier instead of image

**SIFT features**

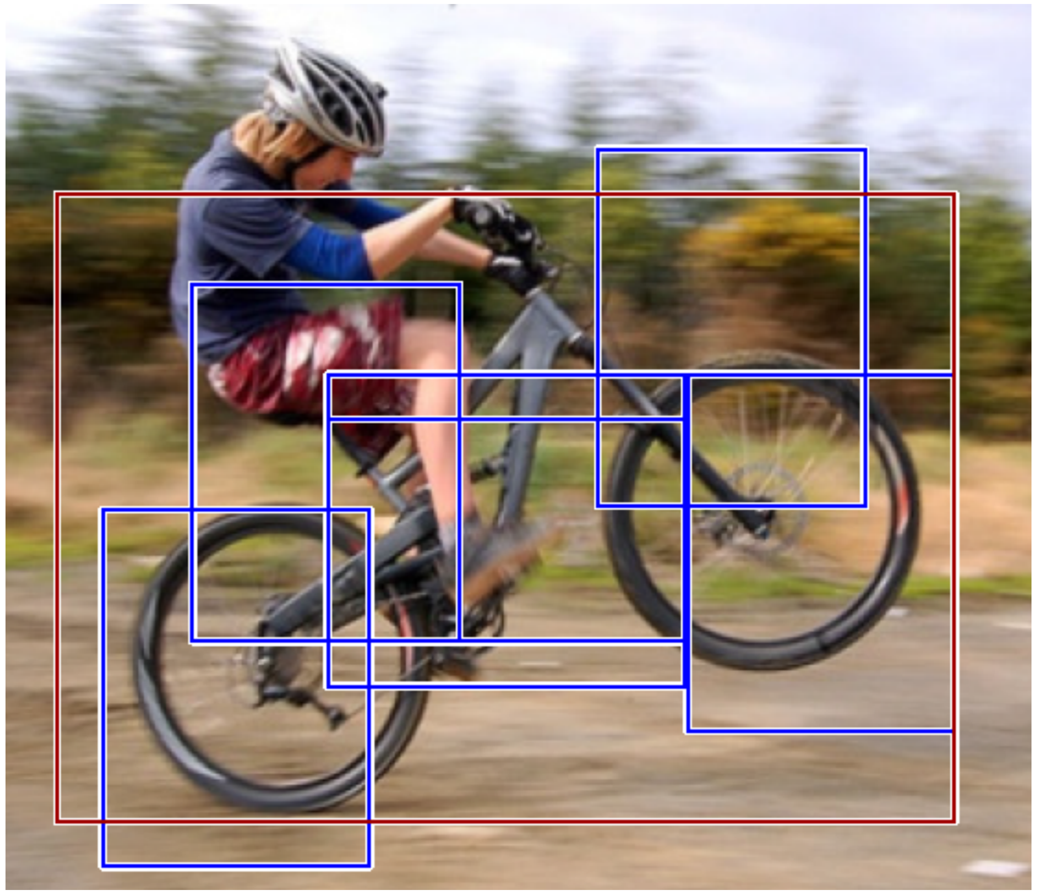
- Scale-invariant feature transform
- Most widely used
- Many applications in industry

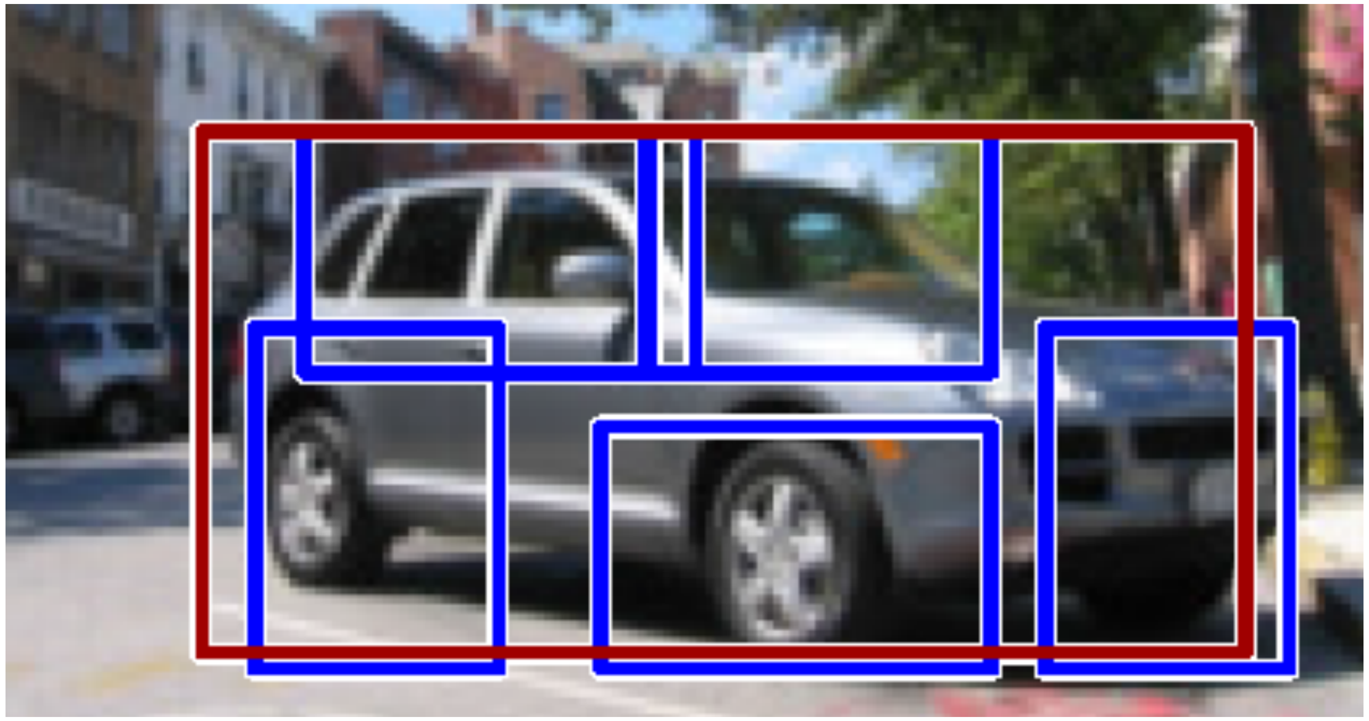


# Recognition by Parts

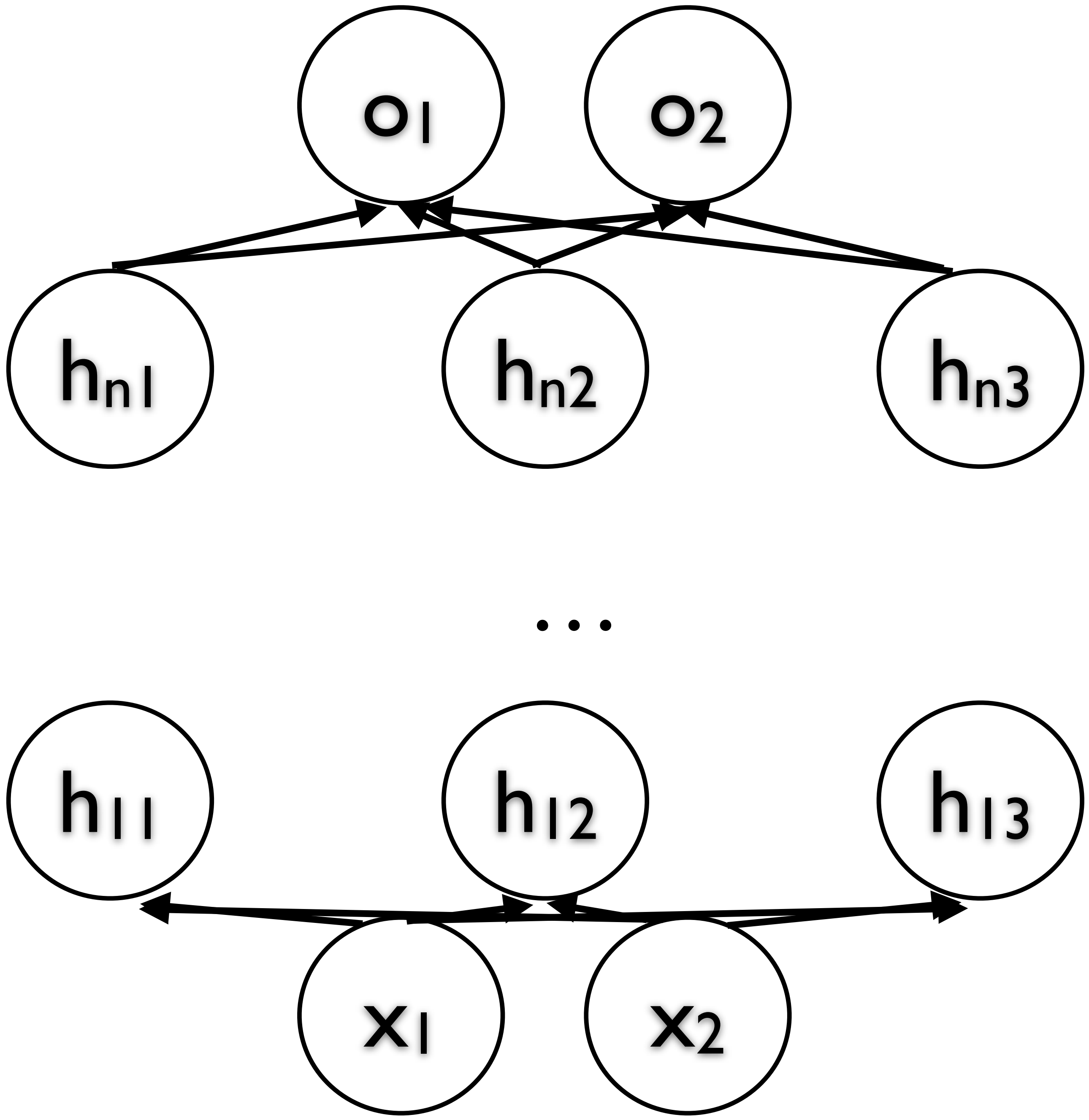
**Combine ML and object-models**

- Objects are made up of "parts"
- Parts have specific relationships to each other
- Match parts by ML, objects by templates or ML
- Best performing: *deformable parts*





# Deep Nets for Object Recognition

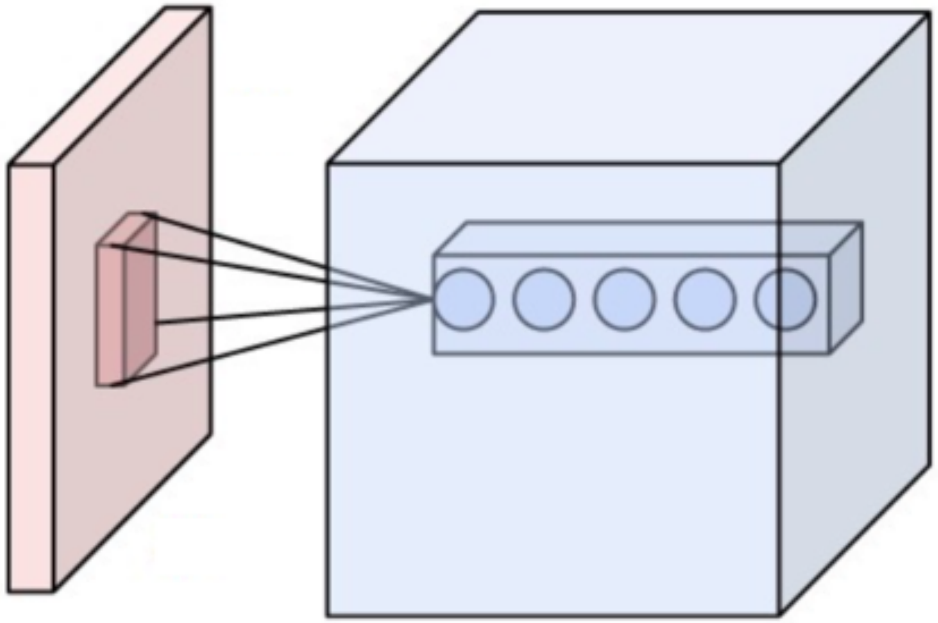


**flower**

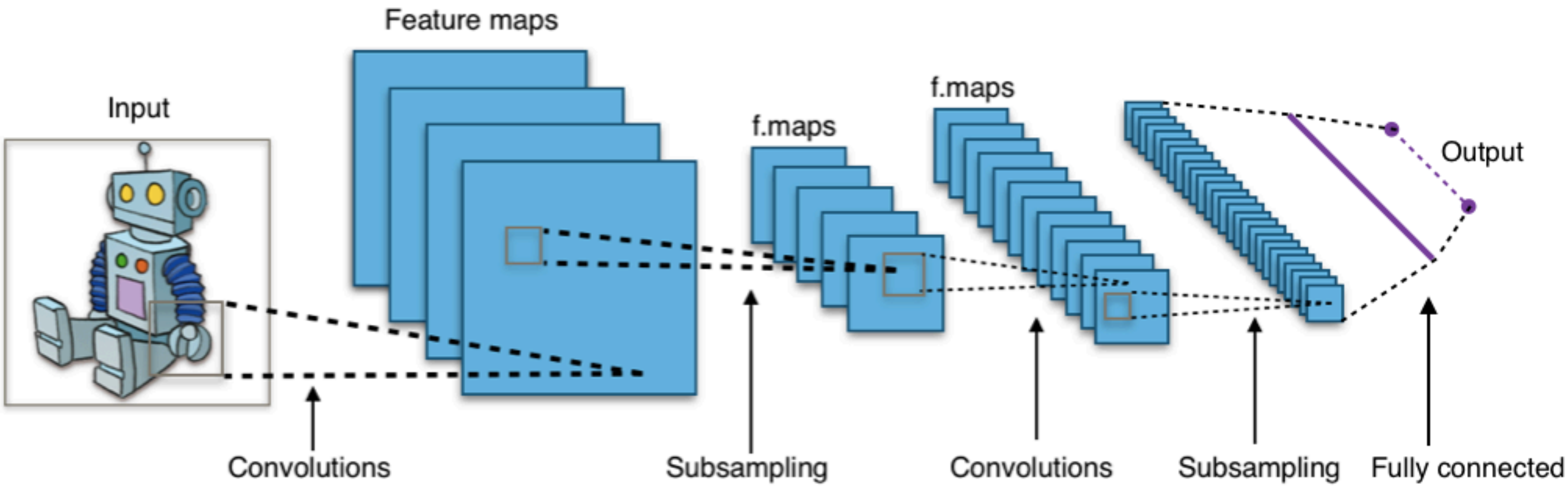
# Convolutional Deep Nets

Key idea:

- The first few layers of processing in a deep net construct features automatically.
- Those features should be *location invariant.*
- Create a layer of neurons with *spatially local input.*
- *Constrain their weights to be the same*.



# Convolutional Deep Nets



(wiki)

# Convolutional Deep Nets

All the usual tricks apply:

- Training vs. test set
- Pretraining
- Can generate synthetic data!
- Must design network architecture
- But no need to think hard about features
- Very powerful hypothesis class
- Lots of data available!

0.23% error rate

# Computer Vision

## Image preprocessing

## Recognition

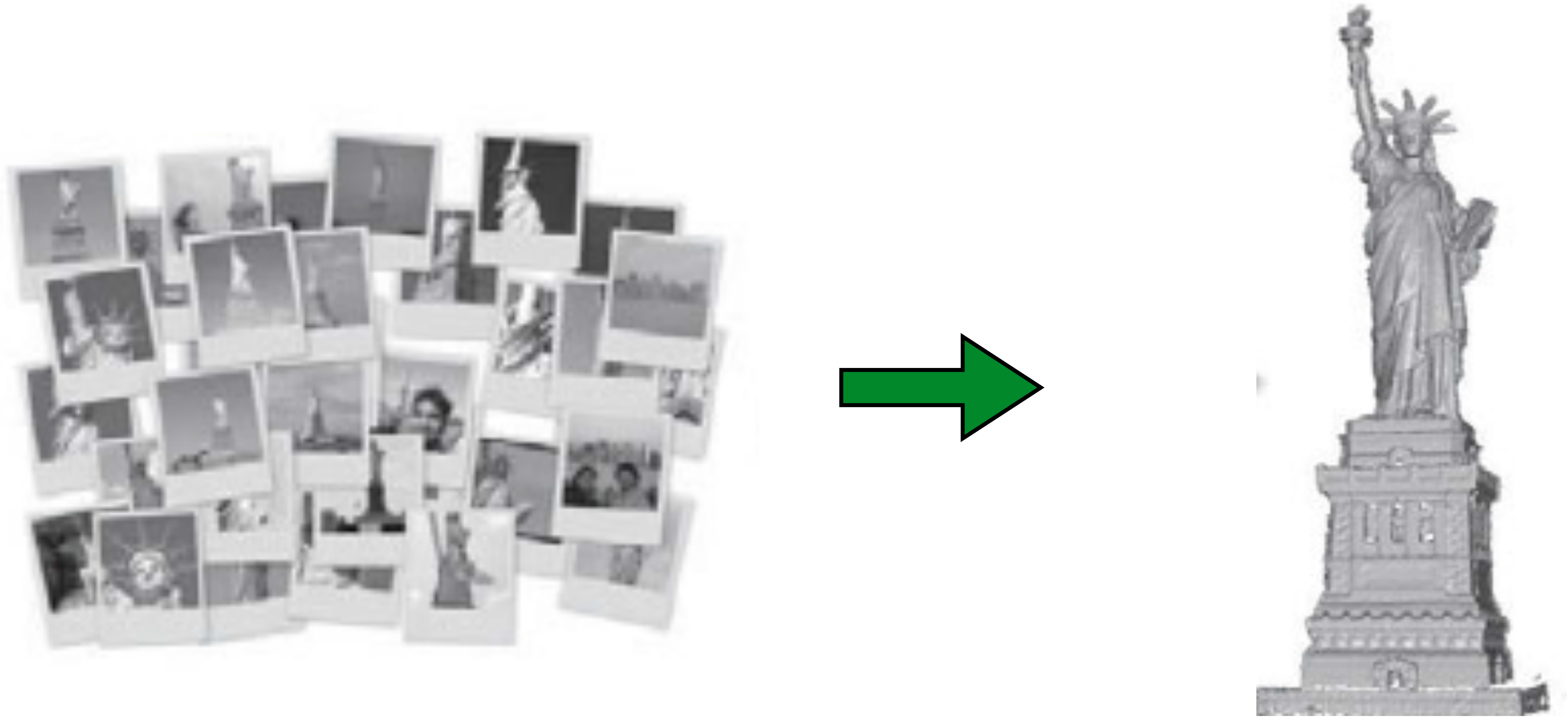
flower

## Reconstruction

[R&N]

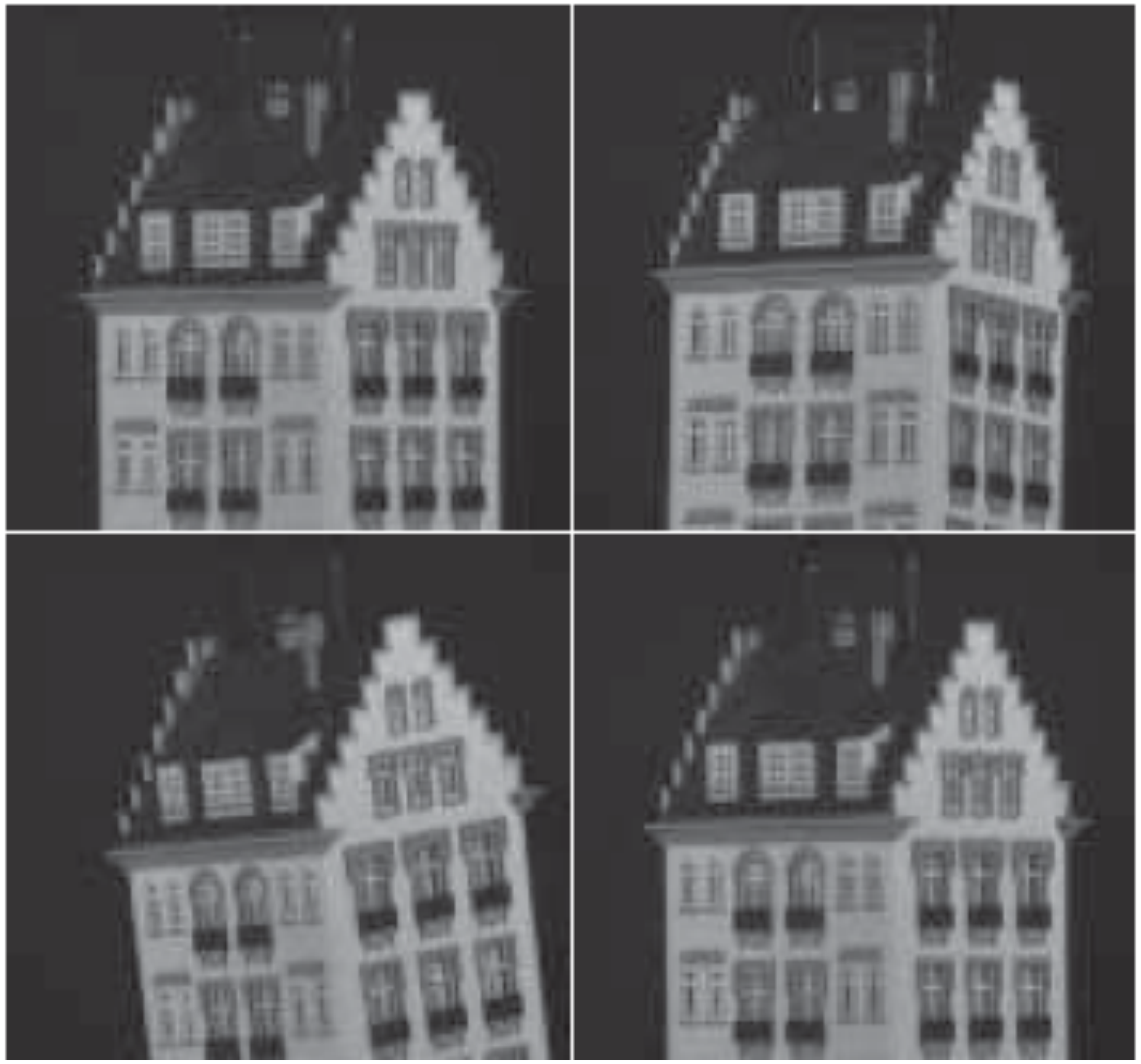
# Reconstruction

Recover 3D information and structure from collection of images.



[R&N]

# Reconstruction

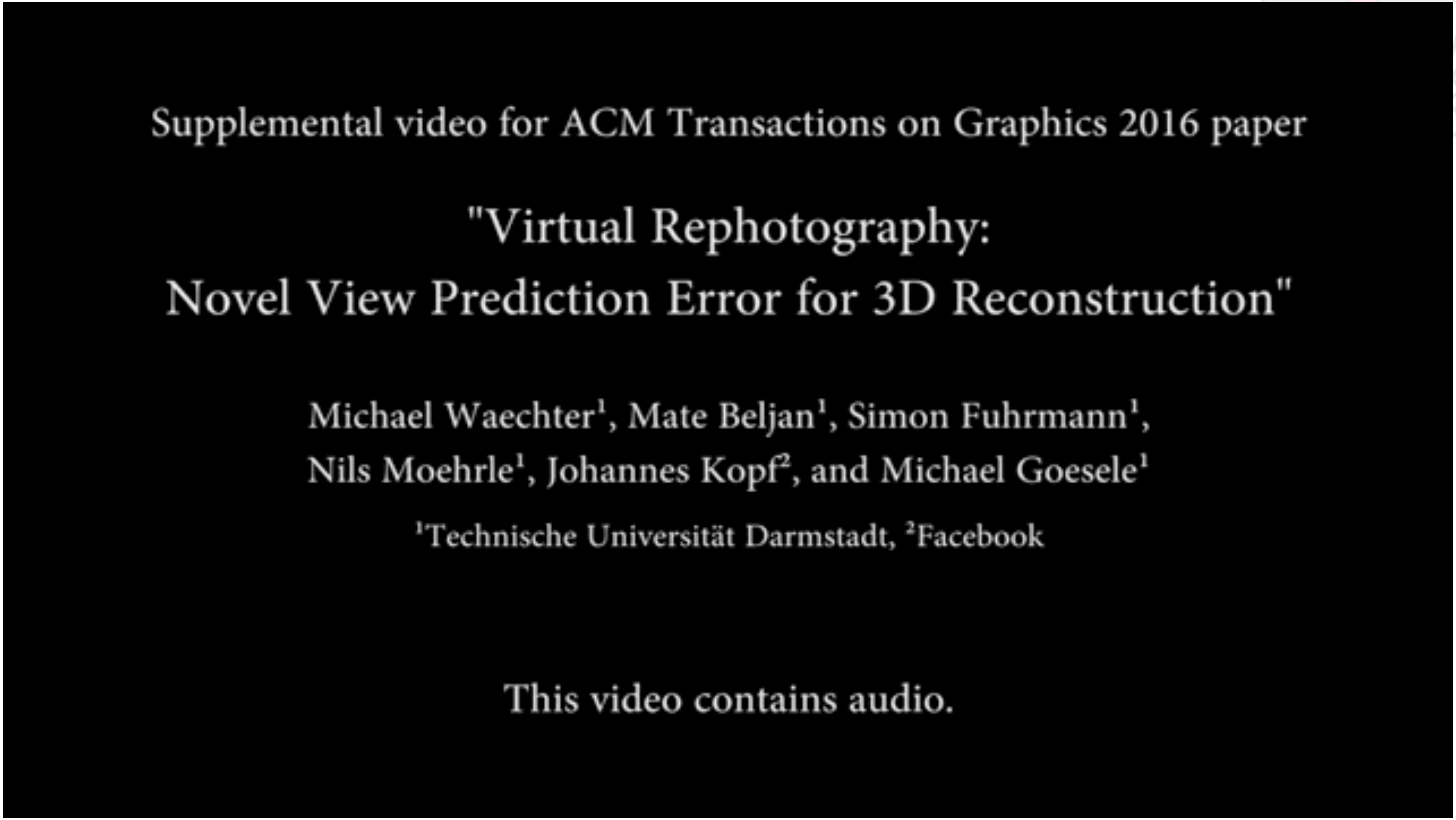


[Tomasi, R&N]

# Reconstruction

Real-time Monocular Scene Reconstruction
in a Public Environment
(Home Improvement Store)

# Reconstruction



Supplemental video for ACM Transactions on Graphics 2016 paper

"Virtual Rephotography:
Novel View Prediction Error for 3D Reconstruction"

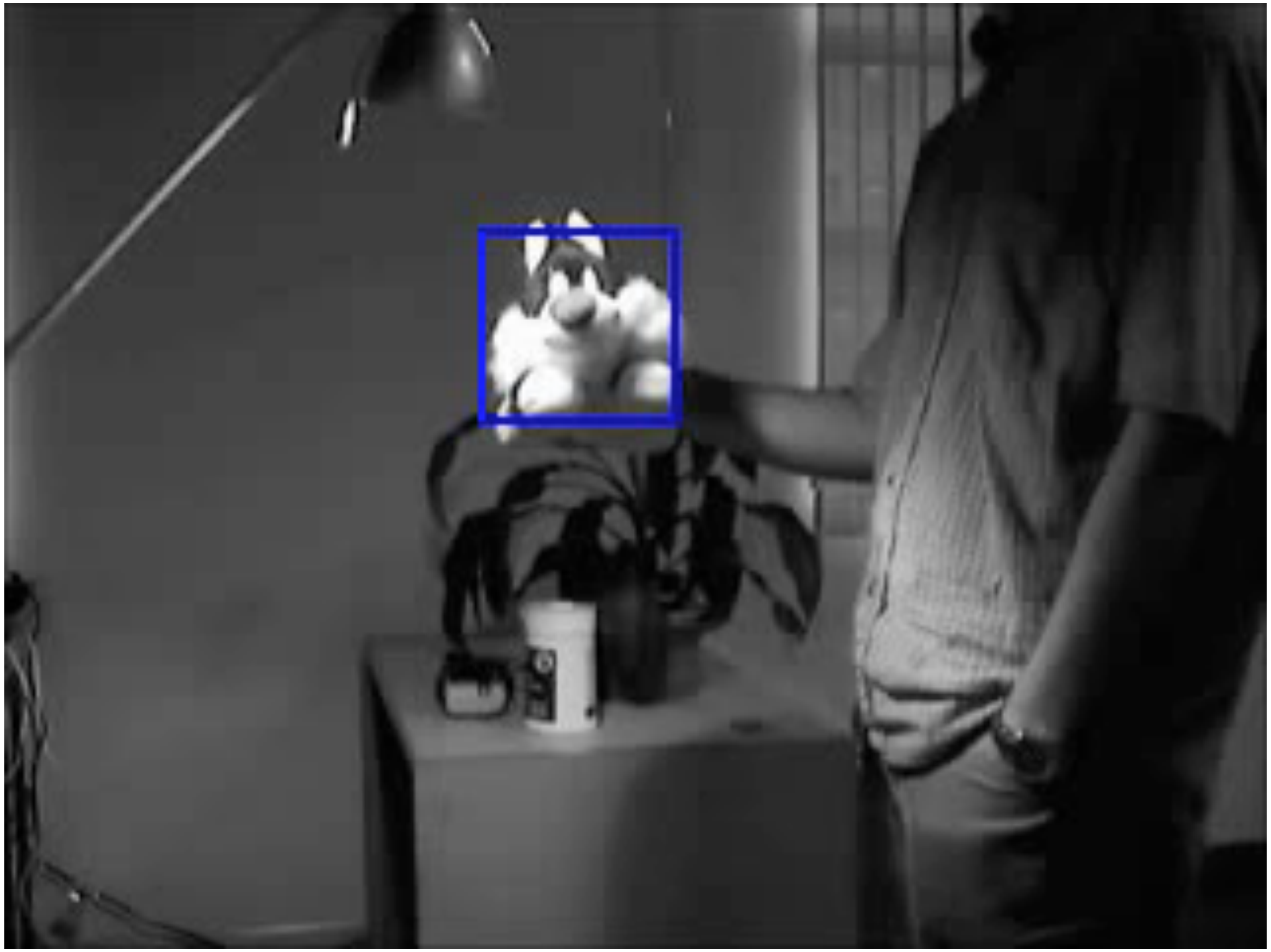
Michael Waechter[1], Mate Beljan[1], Simon Fuhrmann[1],
Nils Moehrle[1], Johannes Kopf[2], and Michael Goesele[1]

[1]Technische Universität Darmstadt, [2]Facebook

This video contains audio.

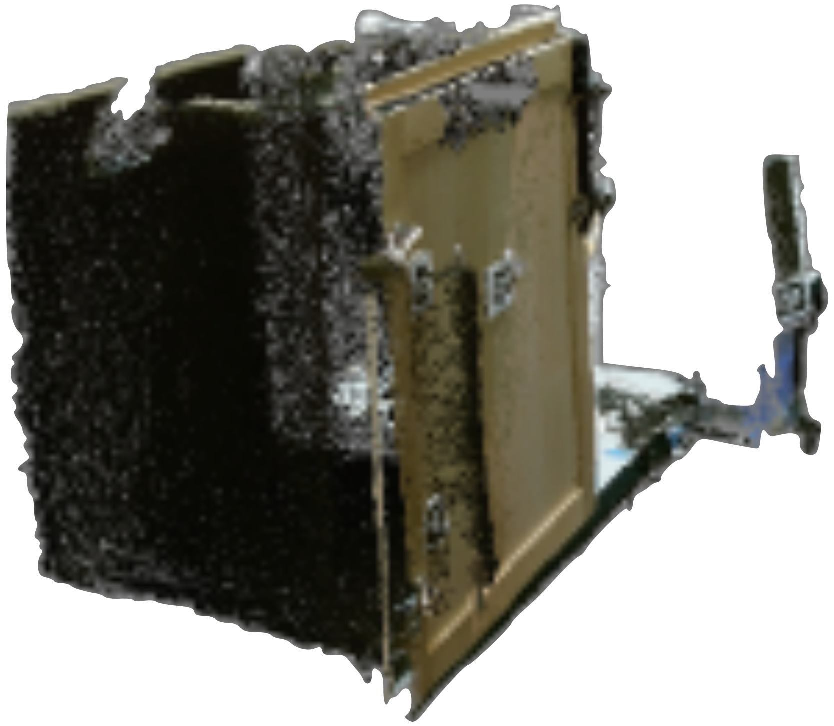
# Reconstruction

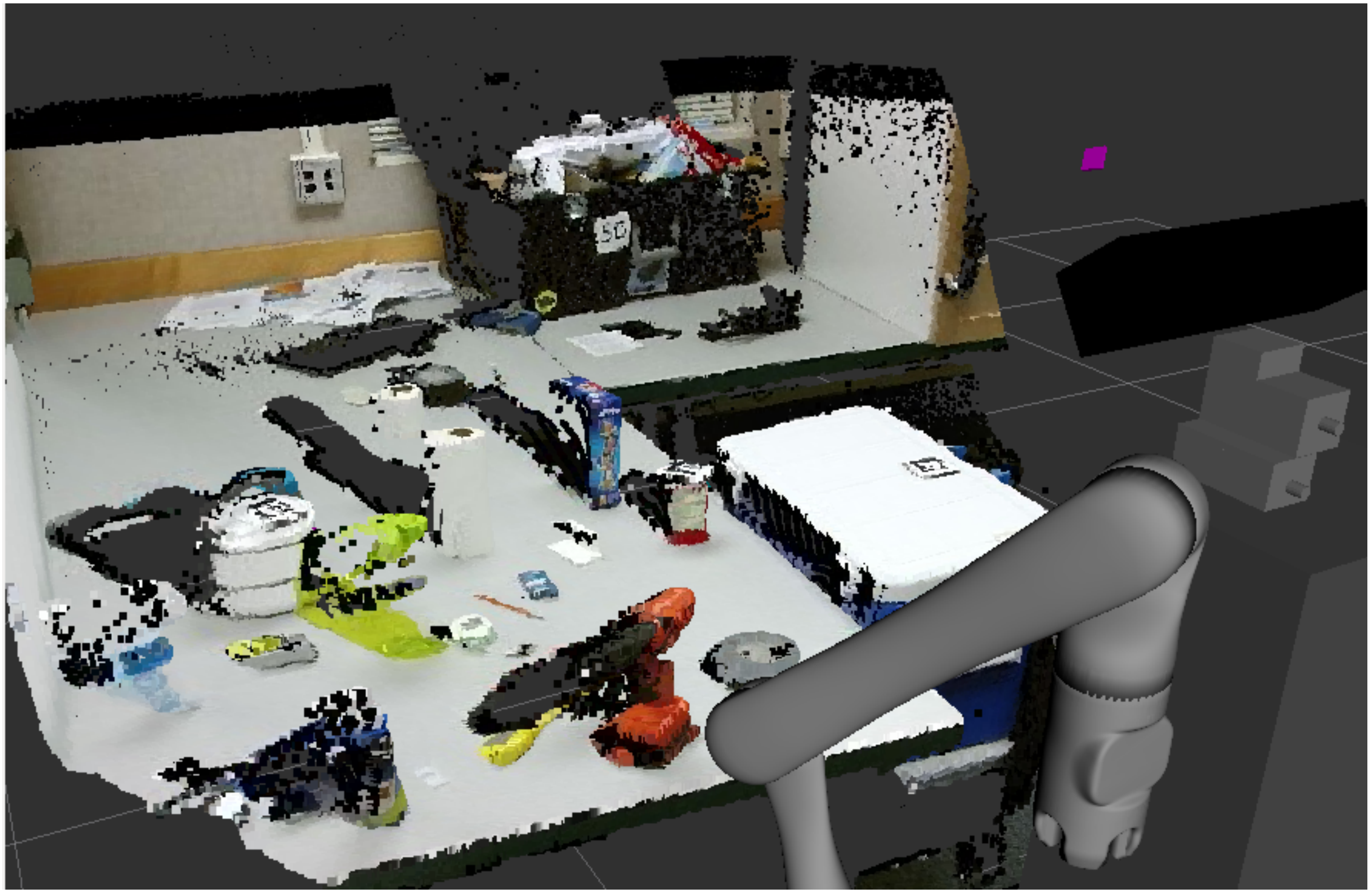
# Tracking



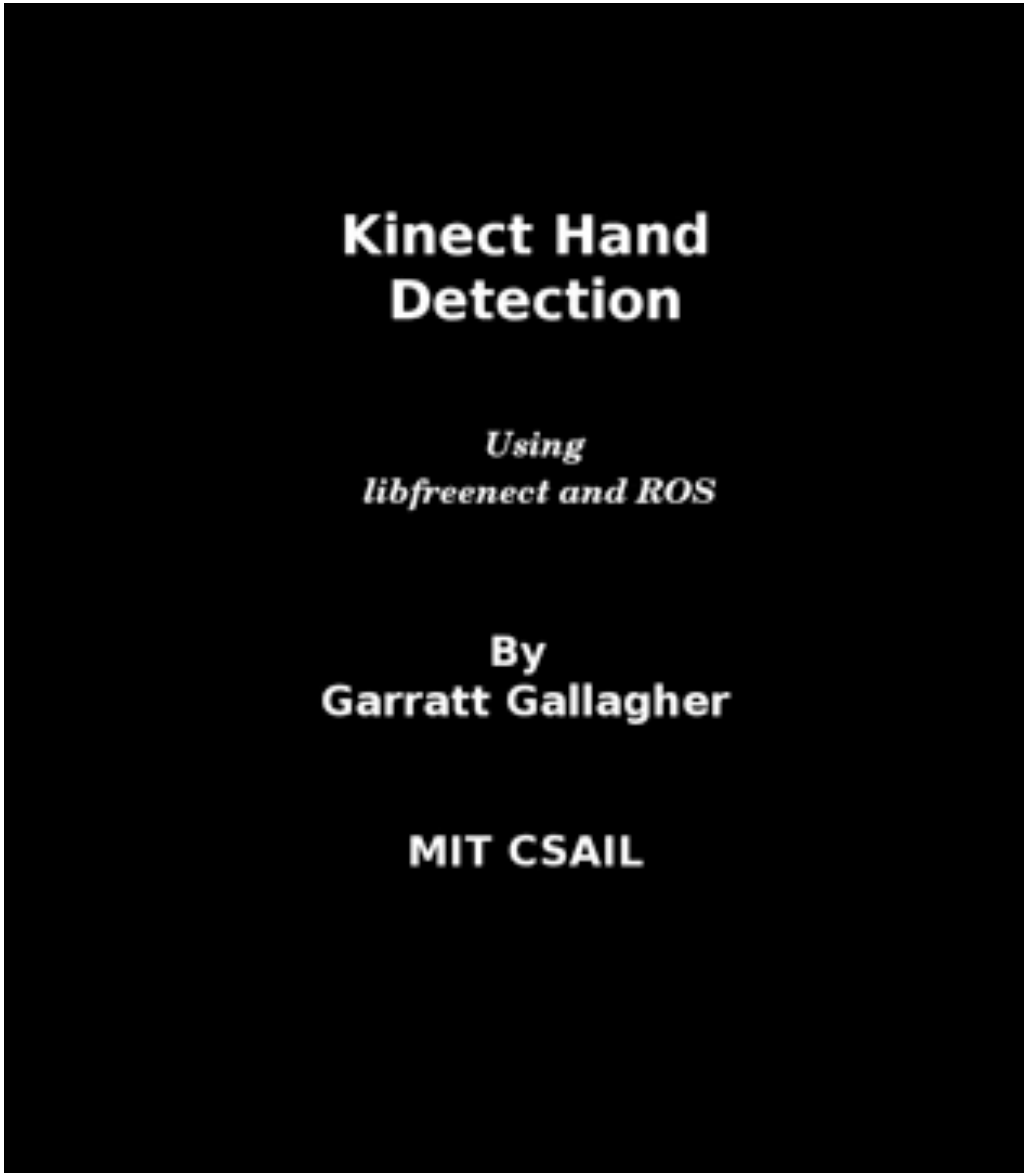
[Sevilla]

# Depth Sensors





# Depth Sensors



**Kinect Hand Detection**

*Using libfreenect and ROS*

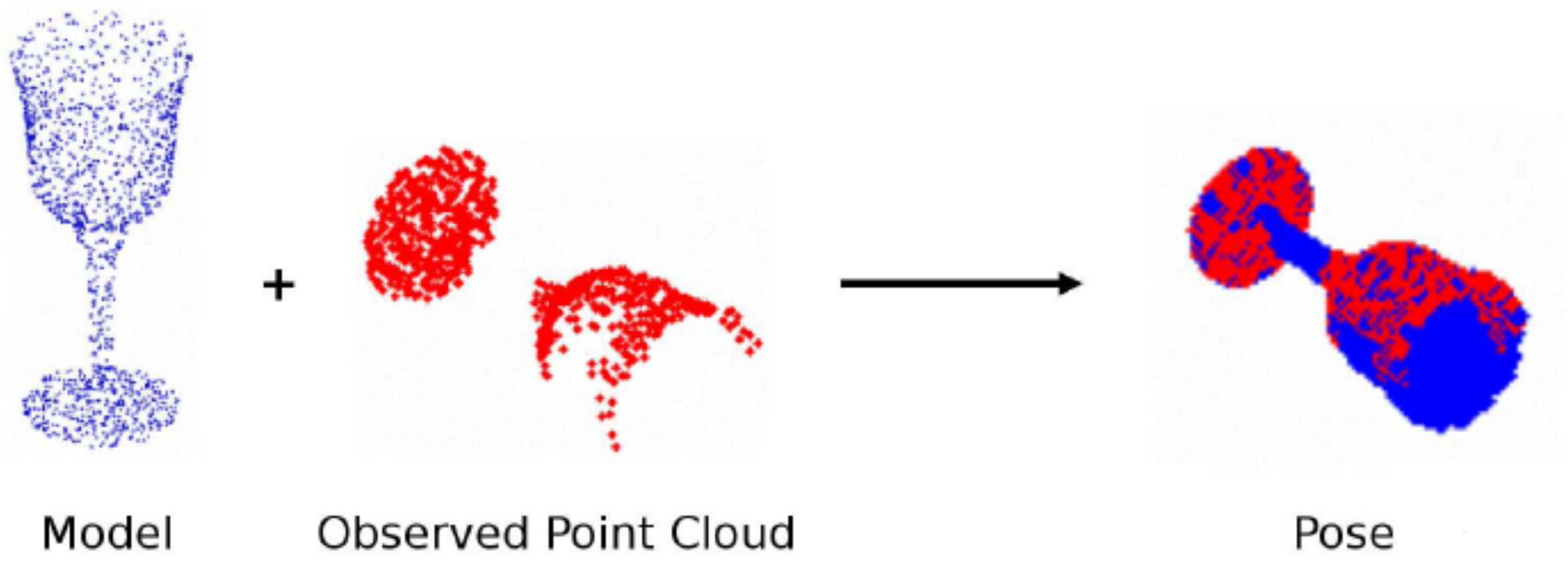
**By Garratt Gallagher**

**MIT CSAIL**

# 3D Perception

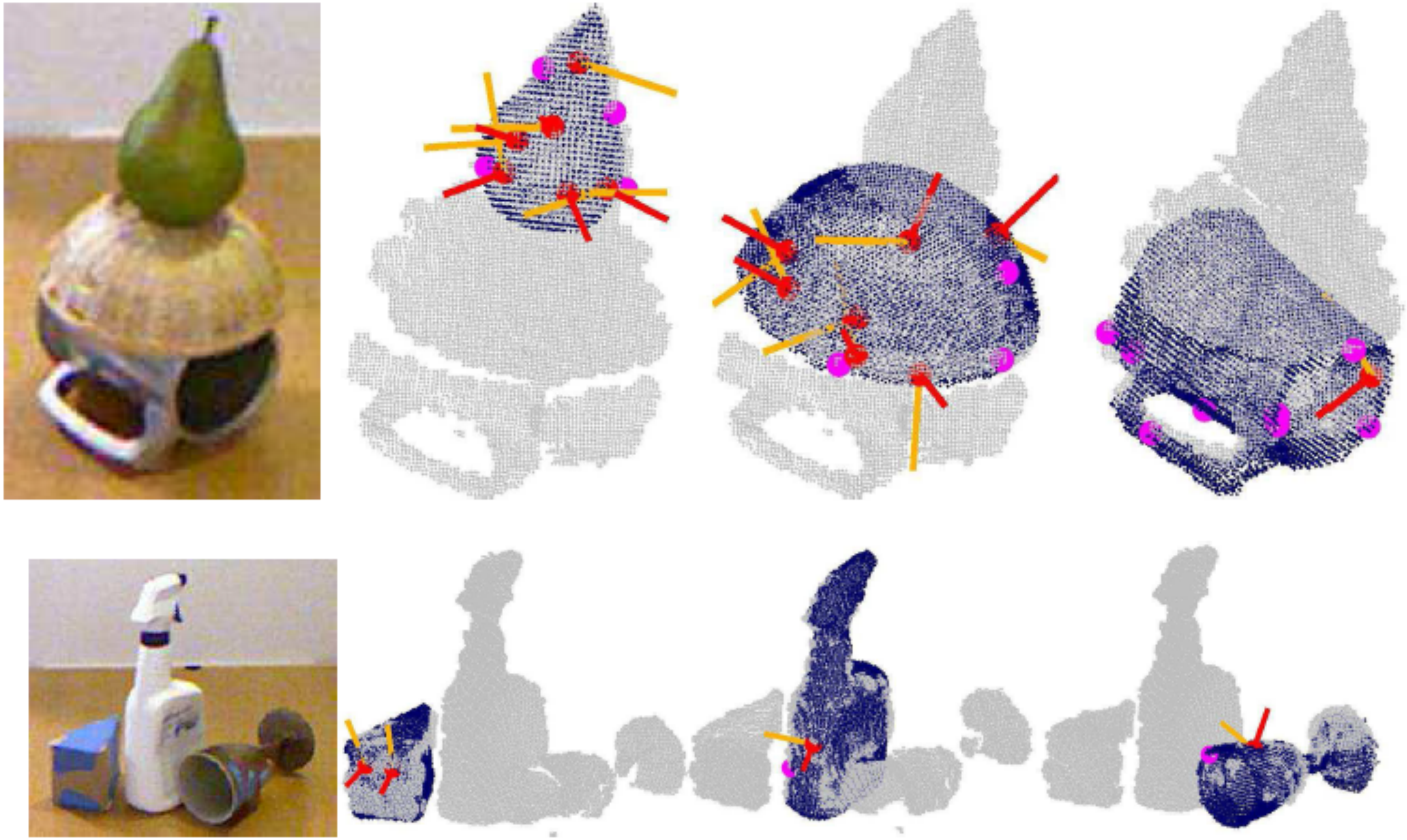
Typically given 3D model of ***specific*** object:

- Identify it from a partial view.
- Pose estimate.
- Complete.



[Glover]

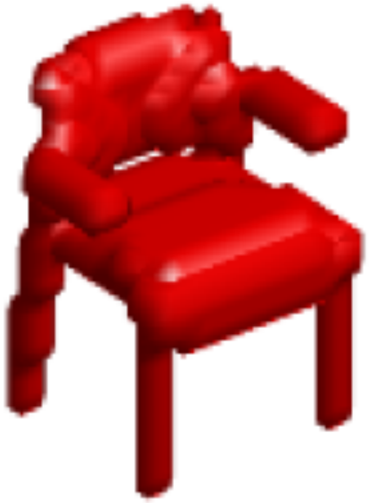
# 3D Perception in Clutter



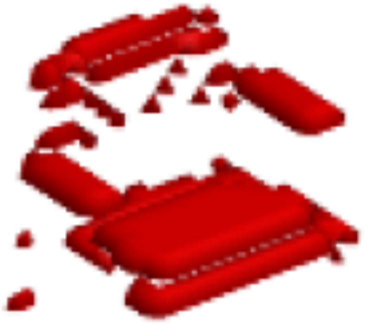
[Glover]

# 3D Perception

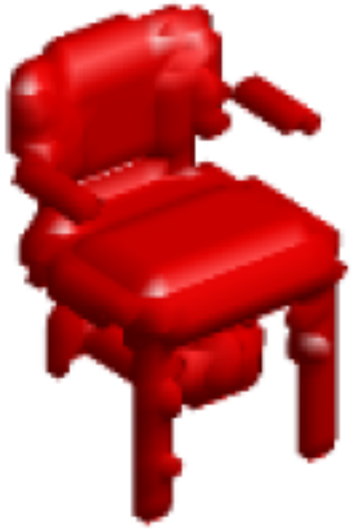
Sensing a ***novel*** chair



true model

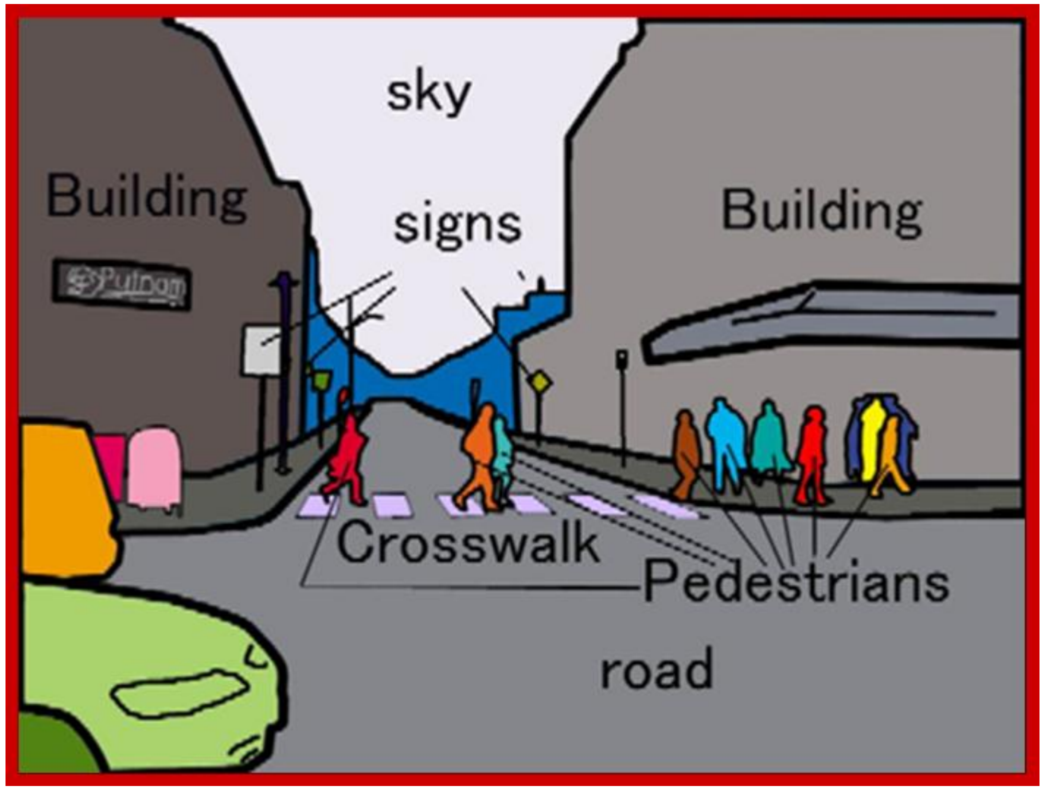


observation



reconstruction

# Autonomous Cars



[Wolf]