# Sensors, Localization and Control

D.A. Forsyth, UIUC

## Sensors, Localization and Control

- -1: Intro - who I am, UIUC, what we're going to do
- 0: Big tools of vision
  - geometry; discrimination ; regression
  - Simple visual control
    - Google streetview; various learning paradigms;
    - why imitation learning is hard; Dagger
- I: Sensor review
  - Cameras; LIDAR; timed cameras; GPS
- II: Localization
  - simple 2D, 3D registration by matching; features for matching
  - matching in paired cameras; F, E matrices; odometry

# Vision group at Illinois

## David Forsyth

- Marr prize, 1993; 2 ex students with Marr prizes; IEEE Tech. Achievement, Fellow; ACM Fellow; EIC IEEE TPAMI

## Derek Hoiem

- best paper, CVPR 2006; ACM Doctoral Dissertation honorable mention; Sloan Fellow;PAMI-TC Young Researcher

## Lana Lazebnik

- Microsoft Faculty Fellow; Sloan Fellow; Koenderink Prize (2016)

## Alex Schwing

- Visual learning, segmentation and GAN models

## Saurabh Gupta

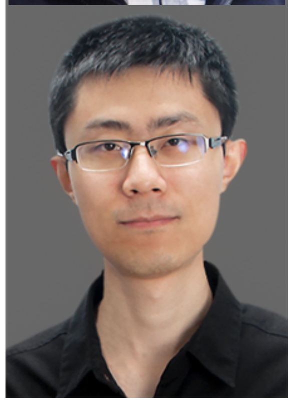
- Linking visual sensing to motion

## Liangyan Gui

- Understanding human movement

## Shenlong Wang

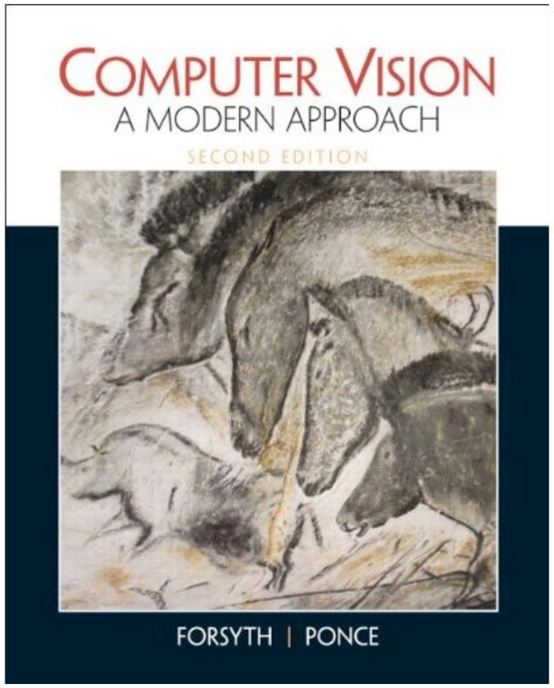
- Simulation and sensing for autonomous vehicles



## Yuxiong Wang

- Learning to detect and classify with very little data

# Vision group



Well-known ex-students:
Lana Lazebnik (UIUC)
Tamara Berg (UNC)
Pinar Duygulu (Hacettepe U.)
Ian Endres
Ali Farhadi (UW)
Varsha Hedau
Nazli Ikizler (Hacettepe U.)
Brett Jones
Kevin Karsch
Zicheng Liao
Deva Ramanan (CMU)
Raj Sodhi
Gang Wang (now Alibaba)
Amin Sadeghi
Zicheng Liao (Zhejiang U.)

The New Computer Vision

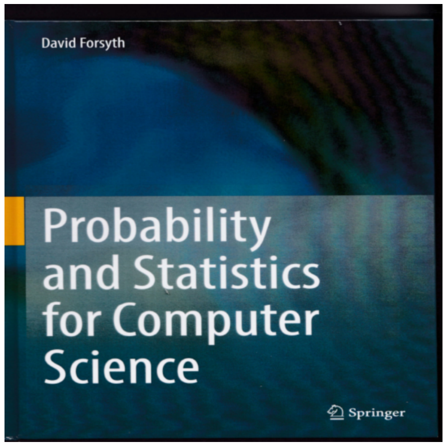
D.A. Forsyth

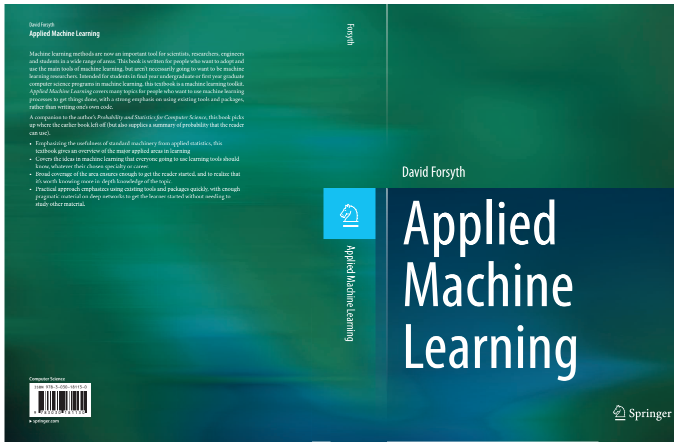
Likely about 2024

Cover design opportunity!

Startups:

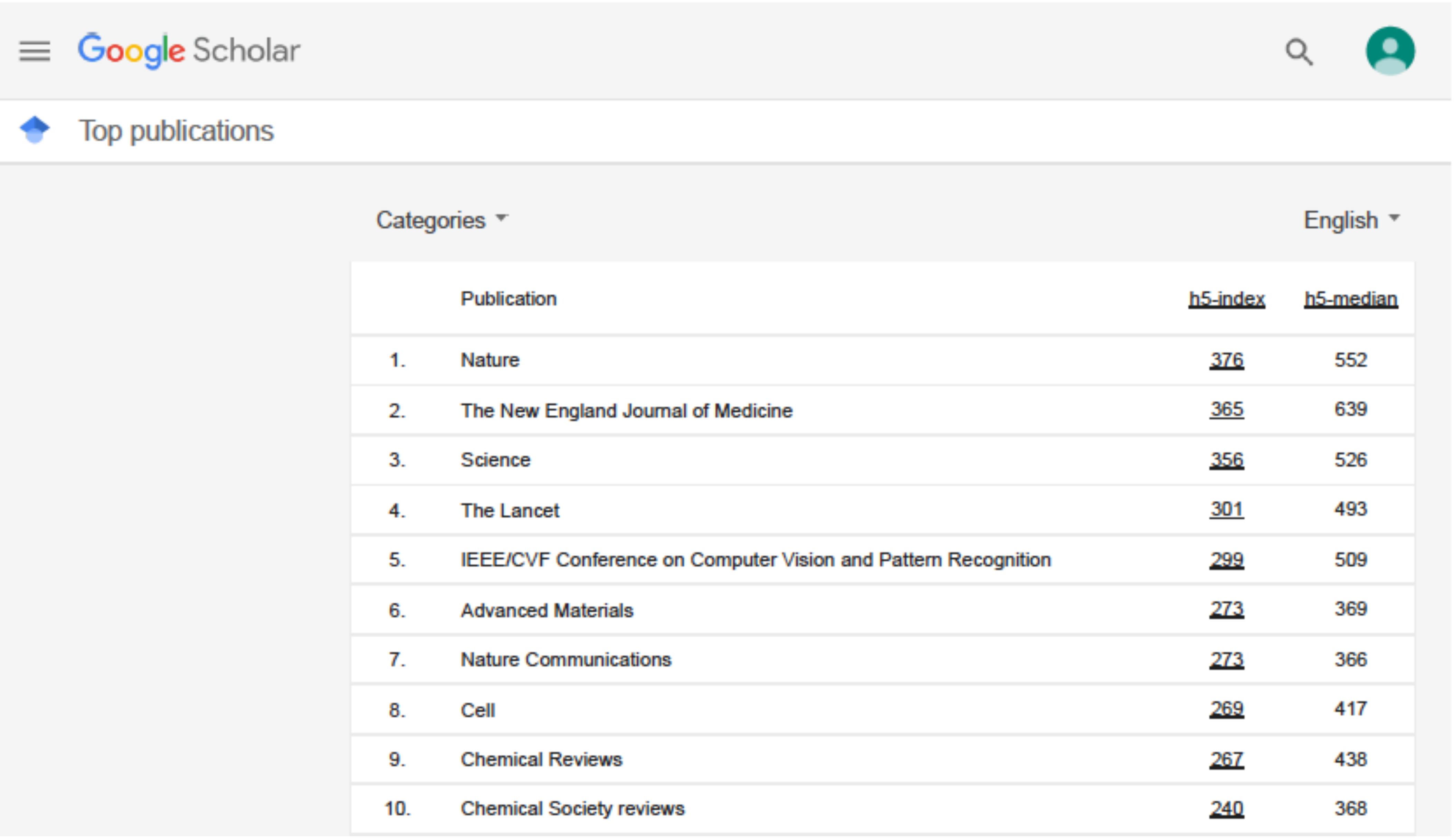
Lightform
Revery.ai
Reconstruct
Depix





- UIC has autonomous vehicles class
    - WITH ACTUAL VEHICLE!

Class project: brake for pedestrian



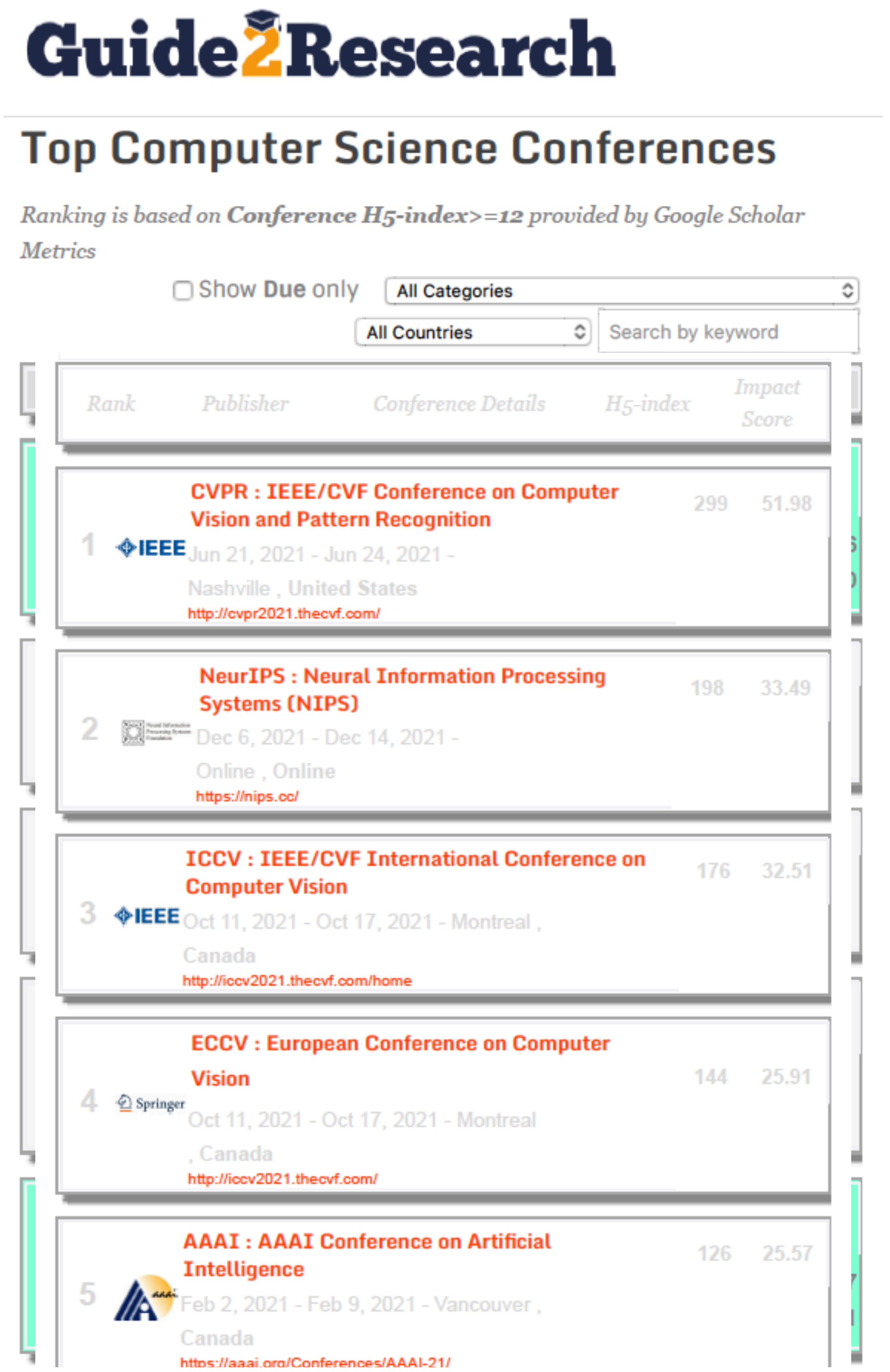
Top publications

| | Publication | h5-index | h5-median |
|---|---|---|---|
| 1. | Nature | 376 | 552 |
| 2. | The New England Journal of Medicine | 365 | 639 |
| 3. | Science | 356 | 526 |
| 4. | The Lancet | 301 | 493 |
| 5. | IEEE/CVF Conference on Computer Vision and Pattern Recognition | 299 | 509 |
| 6. | Advanced Materials | 273 | 369 |
| 7. | Nature Communications | 273 | 366 |
| 8. | Cell | 269 | 417 |
| 9. | Chemical Reviews | 267 | 438 |
| 10. | Chemical Society reviews | 240 | 368 |

Vision

Vision

Vision



Guide2Research

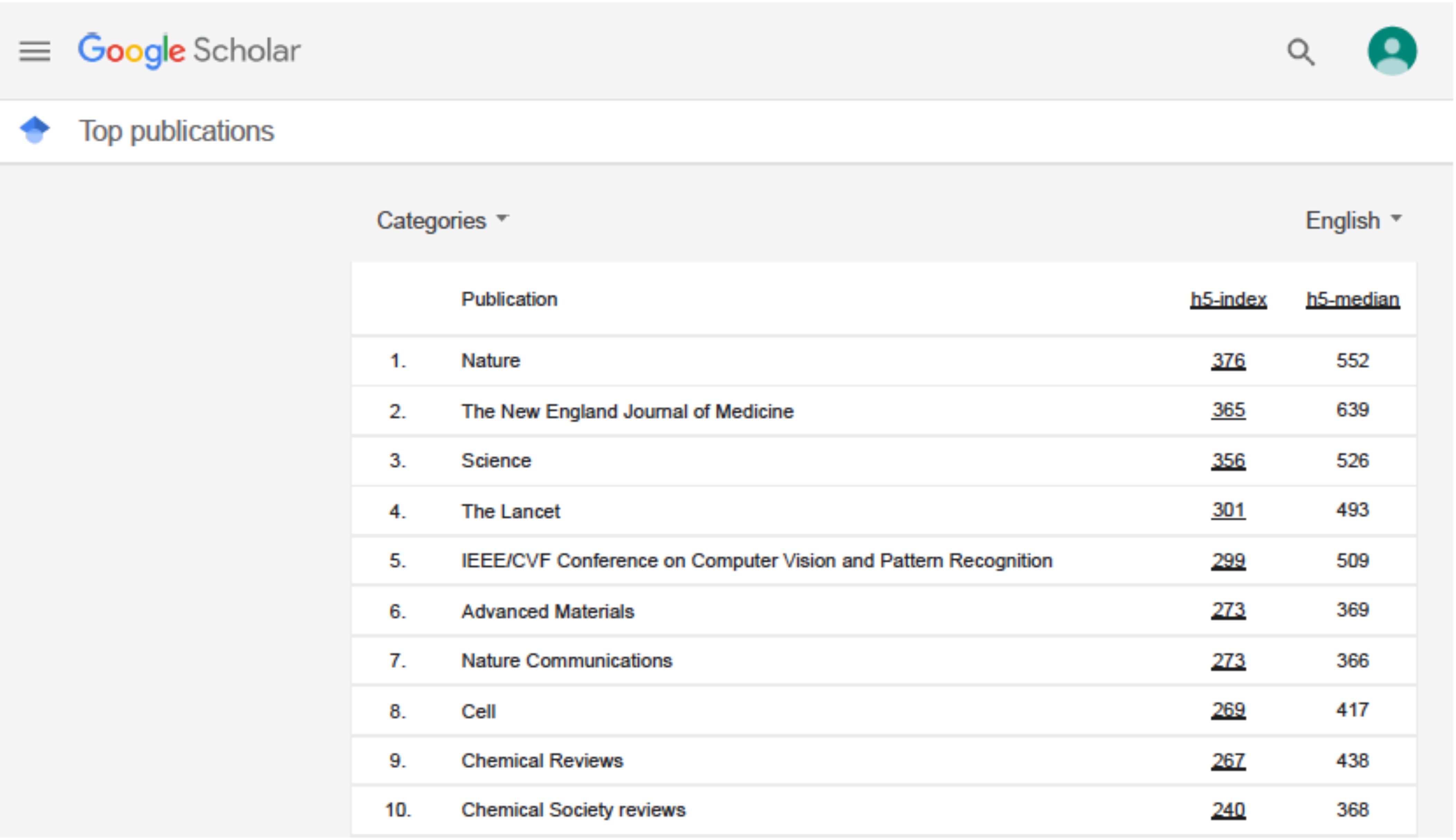
## Top Computer Science Conferences

*Ranking is based on **Conference H5-index>=12** provided by Google Scholar Metrics*

☐ Show **Due** only | All Categories

All Countries | Search by keyword

| *Rank* | *Publisher* | *Conference Details* | *H5-index* | *Impact Score* |
|---|---|---|---|---|
| 1 | IEEE | CVPR : IEEE/CVF Conference on Computer Vision and Pattern Recognition<br>Jun 21, 2021 - Jun 24, 2021 - Nashville , United States<br>http://cvpr2021.thecvf.com/ | 299 | 51.98 |
| 2 | Neural Information Processing Systems Foundation | NeurIPS : Neural Information Processing Systems (NIPS)<br>Dec 6, 2021 - Dec 14, 2021 - Online , Online<br>https://nips.cc/ | 198 | 33.49 |
| 3 | IEEE | ICCV : IEEE/CVF International Conference on Computer Vision<br>Oct 11, 2021 - Oct 17, 2021 - Montreal , Canada<br>http://iccv2021.thecvf.com/home | 176 | 32.51 |
| 4 | Springer | ECCV : European Conference on Computer Vision<br>Oct 11, 2021 - Oct 17, 2021 - Montreal , Canada<br>http://iccv2021.thecvf.com/ | 144 | 25.91 |
| 5 | aaai | AAAI : AAAI Conference on Artificial Intelligence<br>Feb 2, 2021 - Feb 9, 2021 - Vancouver , Canada<br>https://aaai.org/Conferences/AAAI-21/ | 126 | 25.57 |

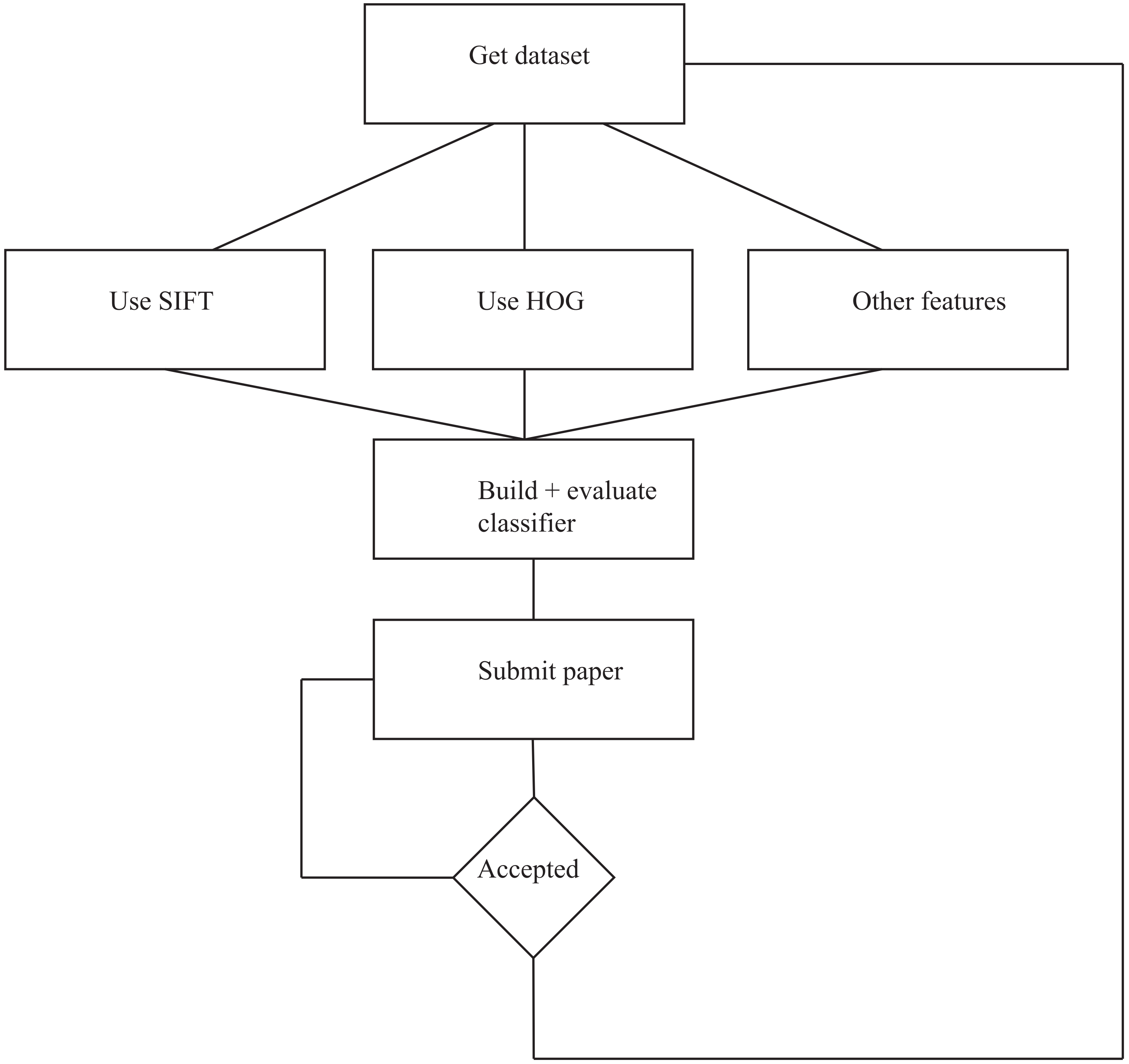


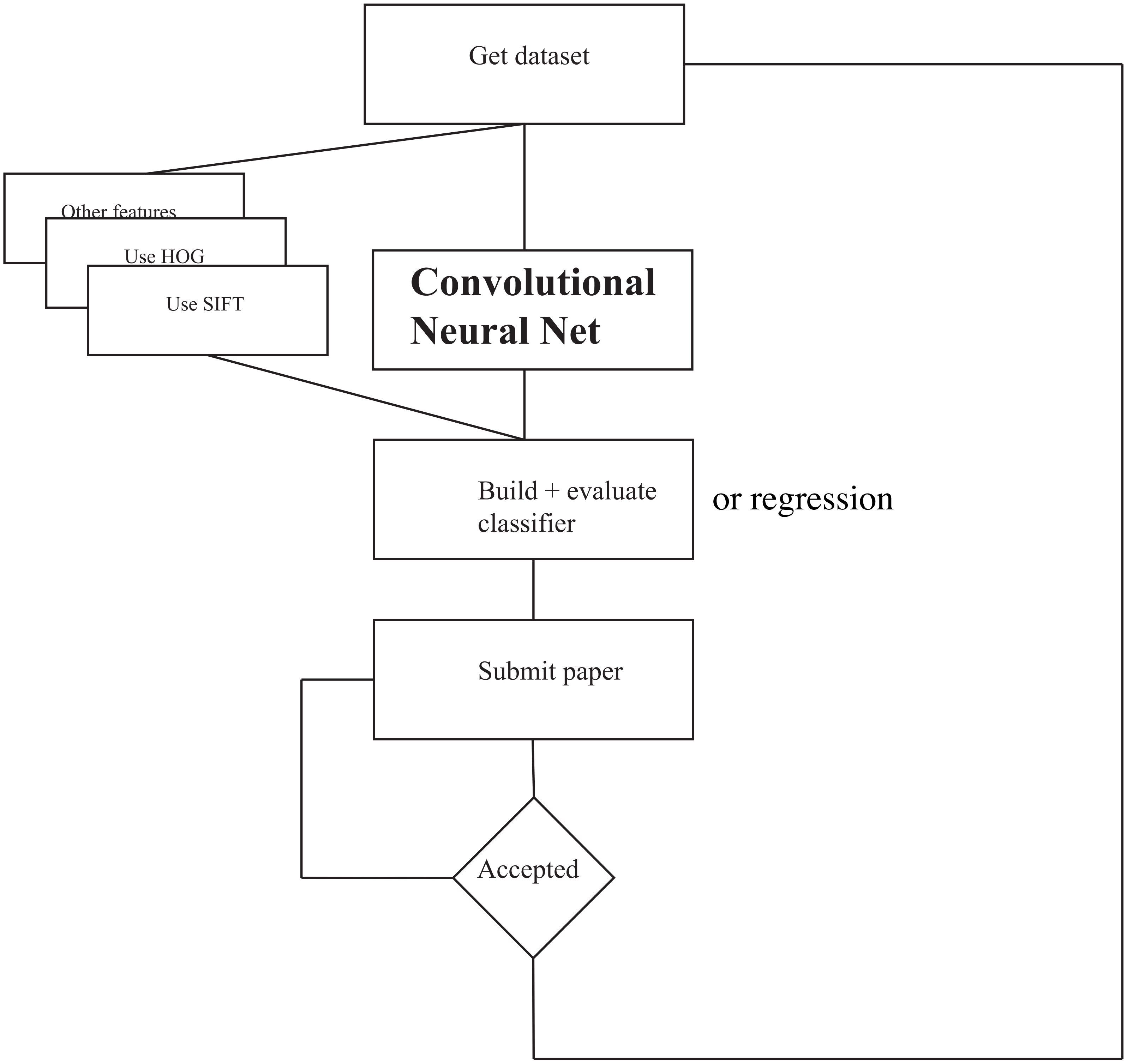
Google Scholar

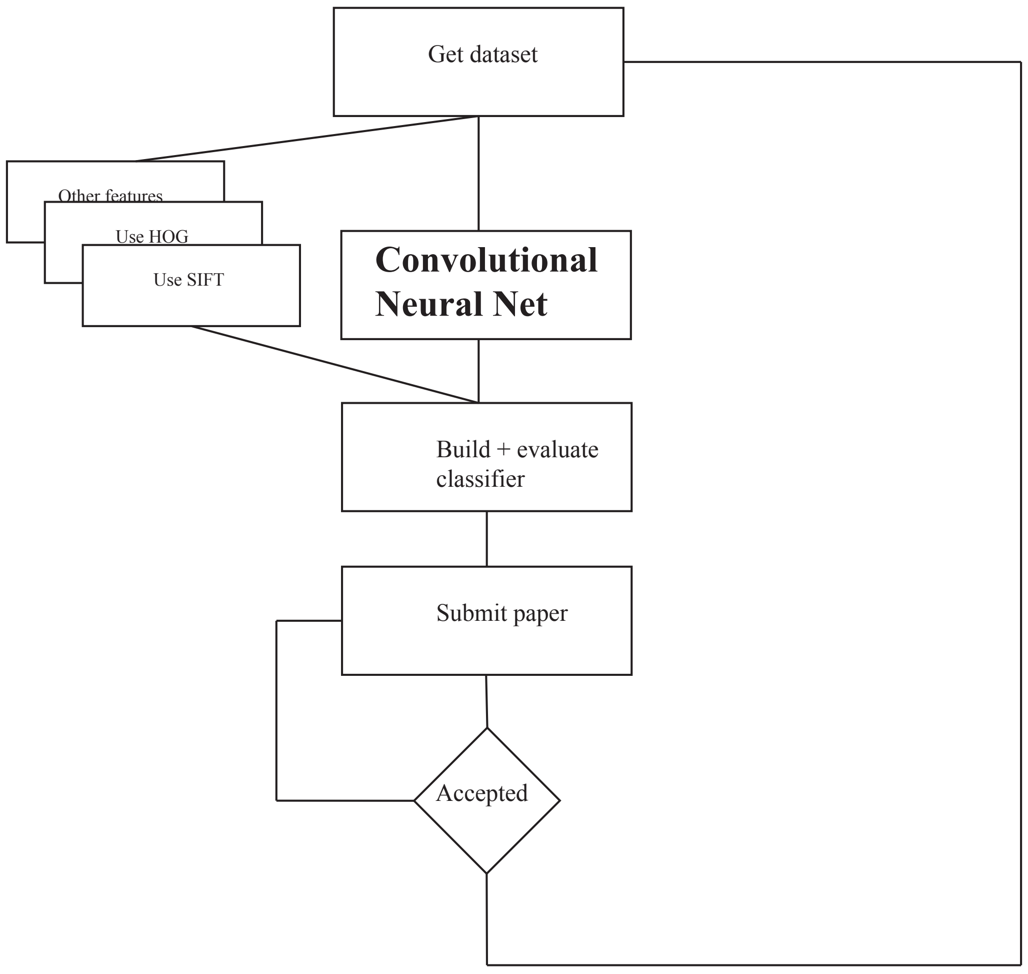
Top publications

Categories English

| | Publication | h5-index | h5-median |
|---|---|---|---|
| 1. | Nature | 376 | 552 |
| 2. | The New England Journal of Medicine | 365 | 639 |
| 3. | Science | 356 | 526 |
| 4. | The Lancet | 301 | 493 |
| 5. | IEEE/CVF Conference on Computer Vision and Pattern Recognition | 299 | 509 |
| 6. | Advanced Materials | 273 | 369 |
| 7. | Nature Communications | 273 | 366 |
| 8. | Cell | 269 | 417 |
| 9. | Chemical Reviews | 267 | 438 |
| 10. | Chemical Society reviews | 240 | 368 |


Get dataset
Use SIFT
Use HOG
Other features
Build + evaluate classifier
Submit paper
Accepted


Get dataset
Other features
Use HOG
Use SIFT
Convolutional Neural Net
Build + evaluate classifier
or regression
Submit paper
Accepted



- But
  - This is increasingly expensive
  - It is surprisingly fragile
  - It is routine

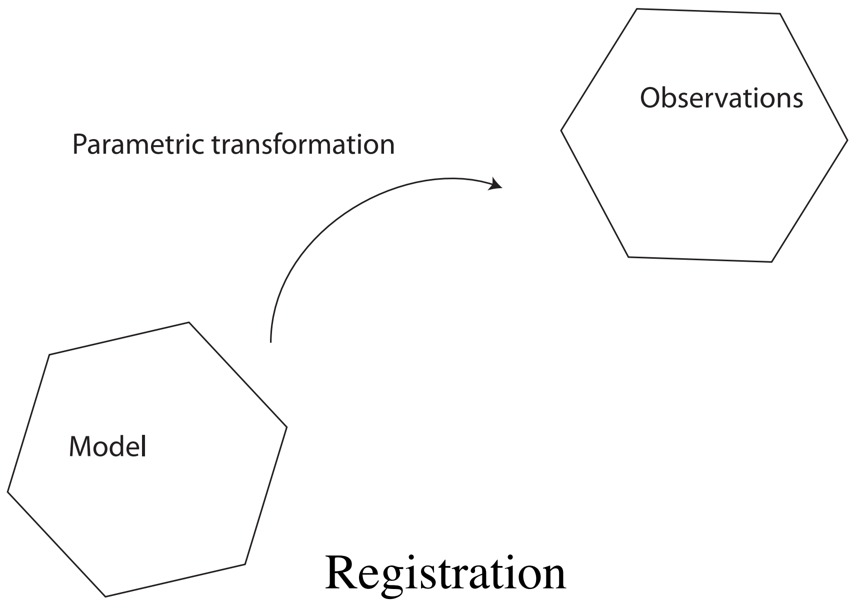
## Computer Vision Today

- Vision in general is flourishing
  - Greatest strengths
    - using data to construct effective, highly polished features
    - increasingly focused on discrimination and regression for huge datasets
    - a money game
    - vast flood of "important" papers
- What should academic vision do?
  - Join a crew
    - works for some; boring; narrowing
  - Contrarianism
    - do stuff industry can't or won't do
    - low and no data learning

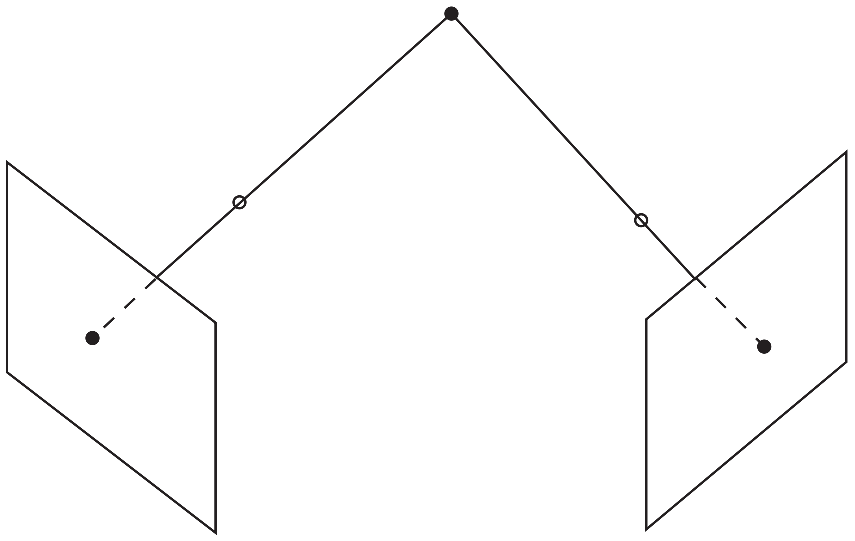
# Sensors, Localization and Control

- -1: Intro - who I am, UIUC, what we’re going to do
- 0: Big tools of vision
  - geometry; discrimination ; regression
  - Simple visual control
    - Google streetview; various learning paradigms;
    - why imitation learning is hard; Dagger
- I: Sensor review
  - Cameras; LIDAR; timed cameras; GPS
- II: Localization
  - simple 2D, 3D registration by matching; features for matching
  - matching in paired cameras; F, E matrices; odometry

# Big ideas in vision: Geometry

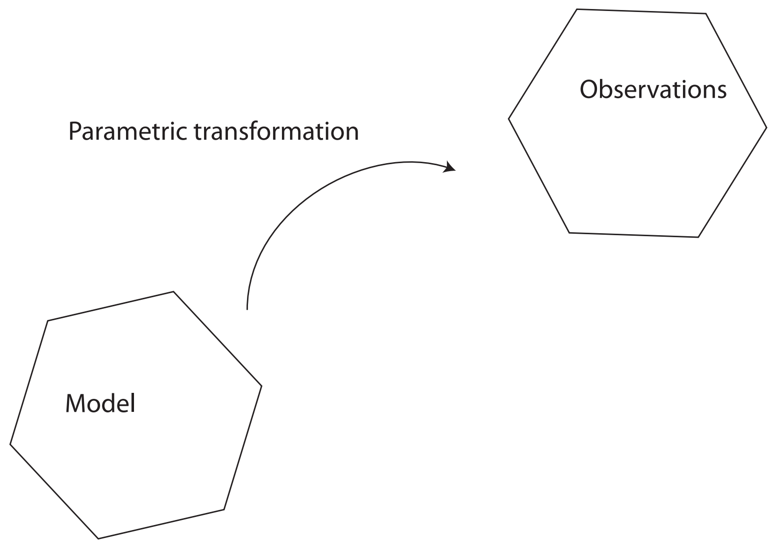


Registration



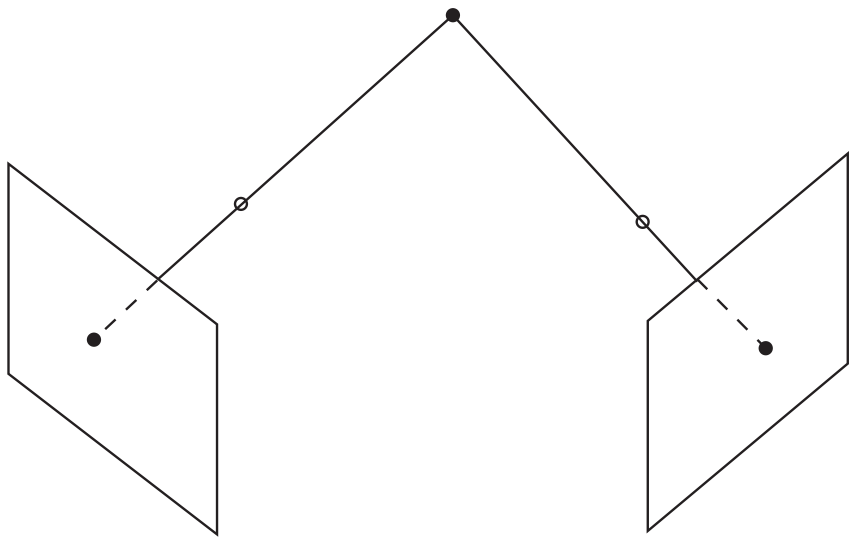
Multiple view geometry

# Registration



- Extremely general recipe
  - 2D to 2D; 3D to 3D; 3D to 2D; etc
  - Representations
    - as points
    - as primitives
    - etc
- Fantastically useful

# Multiple view geometry



- Q: What can we impute from
  - 2 or more views
- A: Lots
  - Motion of the camera
  - How things are moving in the image
  - 3D structure

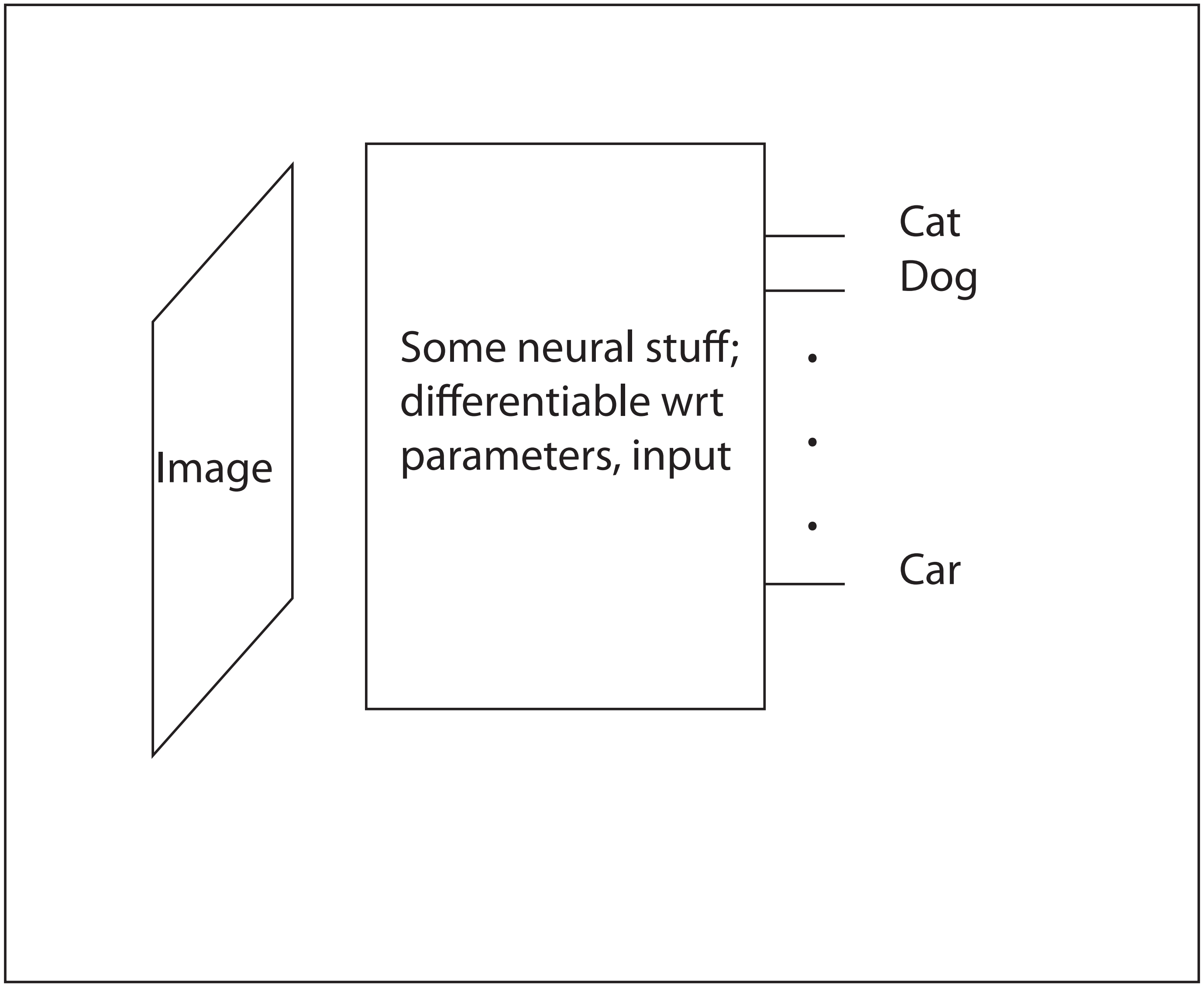
# Sensors, Localization and Control

- -1:  Intro - who I am, UIUC, what we’re going to do
- 0: Big tools of vision
  - geometry ; discrimination ; regression
  - Simple visual control
    - Google streetview; various learning paradigms;
    - why imitation learning is hard; Dagger
- I: Sensor review
  - Cameras; LIDAR; timed cameras; GPS
- II: Localization
  - simple 2D, 3D registration by matching; features for matching
  - matching in paired cameras; F, E matrices; odometry

## Big ideas in vision: Discrimination

- Use some procedure to attach a label to
  - an image; some images; video; range data; lidar data; etc, etc
- “Label” can be very loosely interpreted
- “Procedure” could be
  - learned
  - hand-tuned
  - determined by physics; the problem; etc

# Image classification



## Key ideas

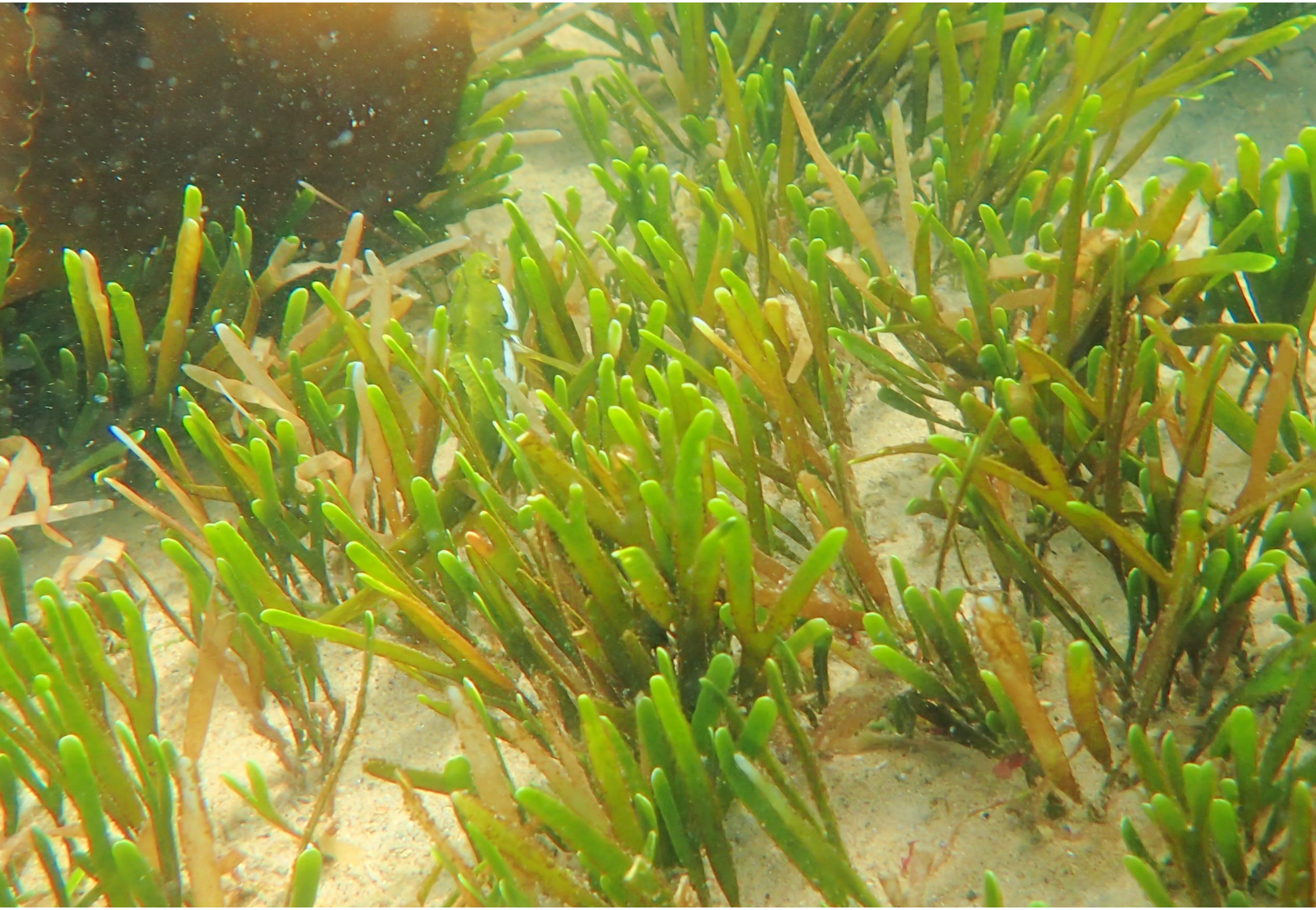
- Goal:
  - Adjust classifier so that it accurately classifies *UNSEEN* data
- Procedure:
  - Adjust so that it
    - classifies training data well
    - generalizes
      - regularization term, either explicit or implicit
- Evaluation:
  - Use held out data to check accuracy on *UNSEEN* data

## Classification or detection

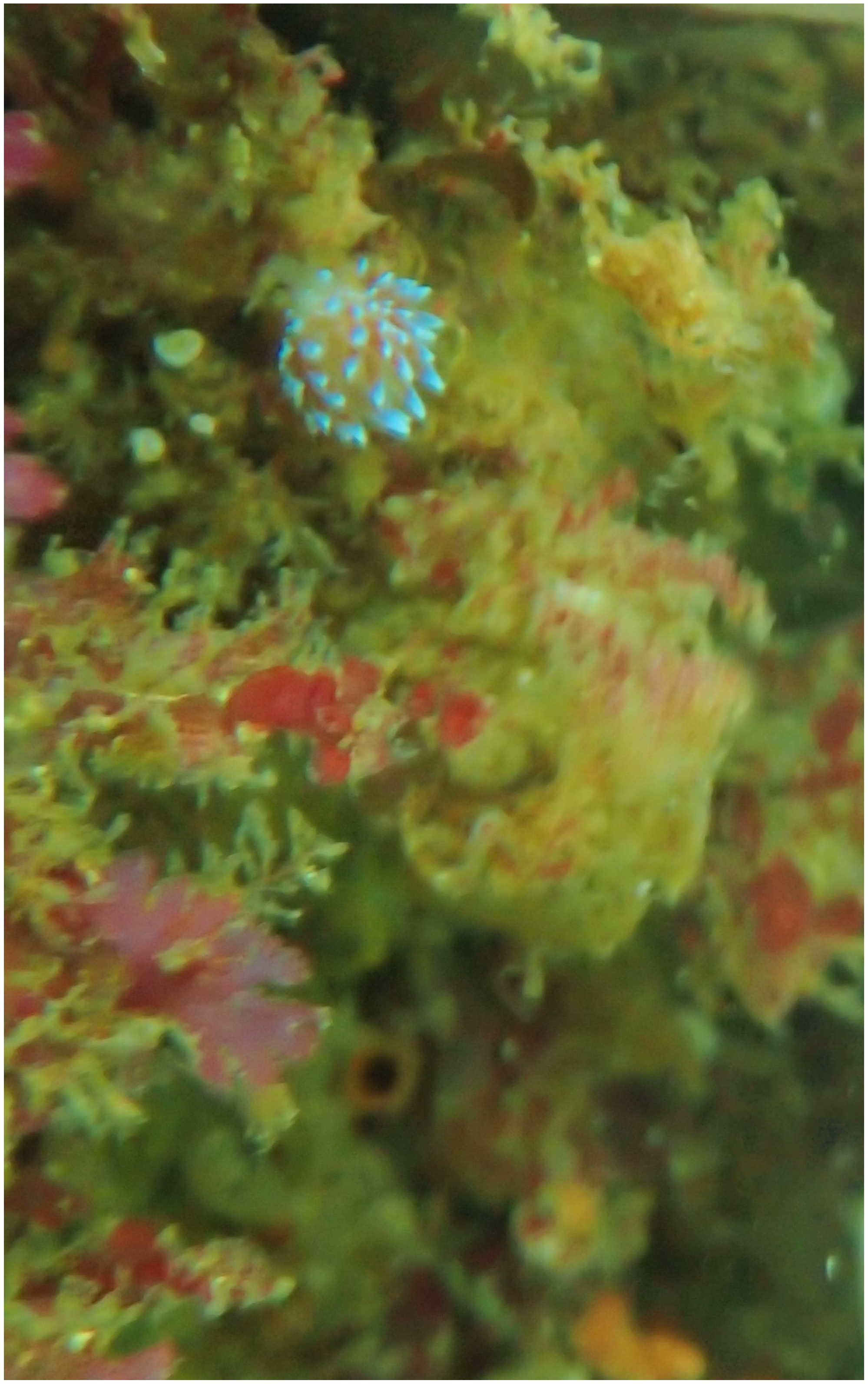
- Classification:
  - there is an X in this image
    - what
- Detection:
  - there is an X HERE in this image
    - what AND where

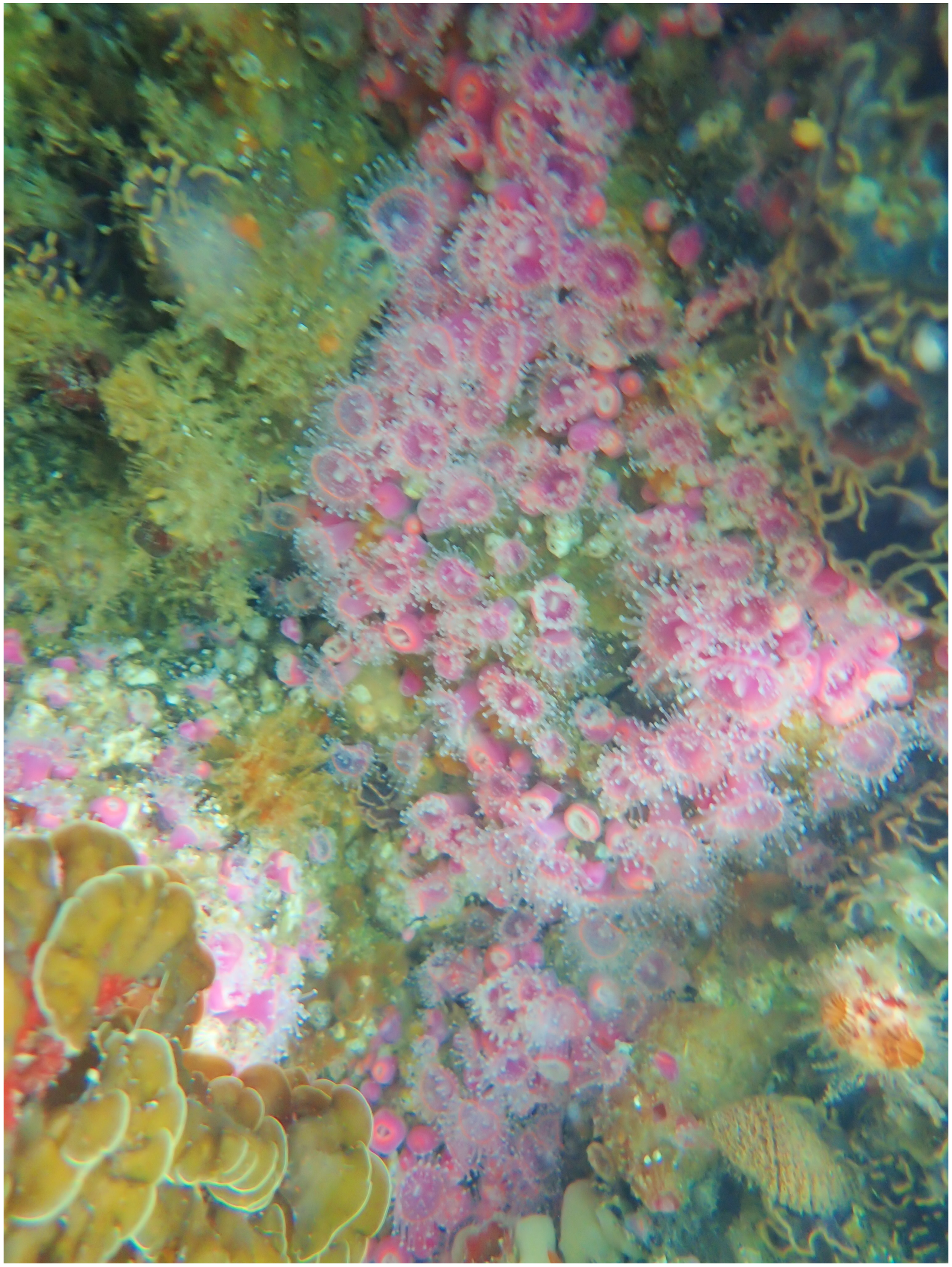
- Key issues
  - how to specify where
  - relationship between what and where
    - efficiency, etc
  - evaluation
    - surprisingly fiddly

## It can be hard to find things



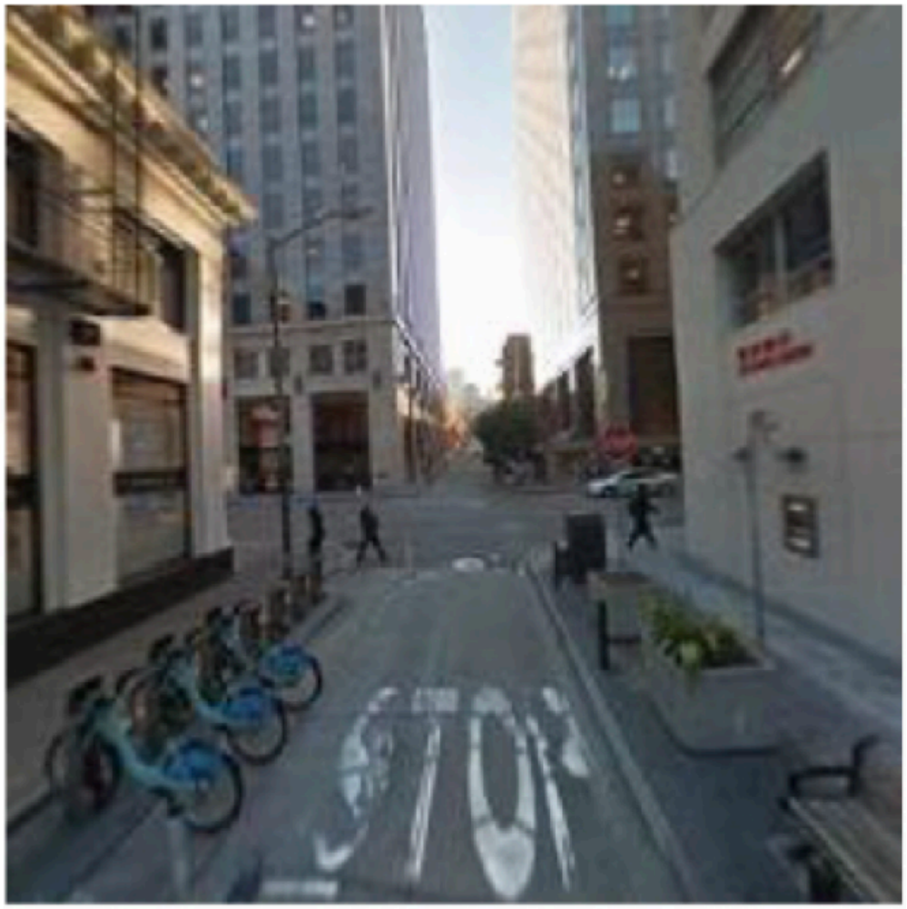
## You may not know the right label





# Exploiting registration and classification

- Use a classifier to tell:
  - how far to the next intersection?
  - what is it like?
  - is there a bike lane?
  - etc.



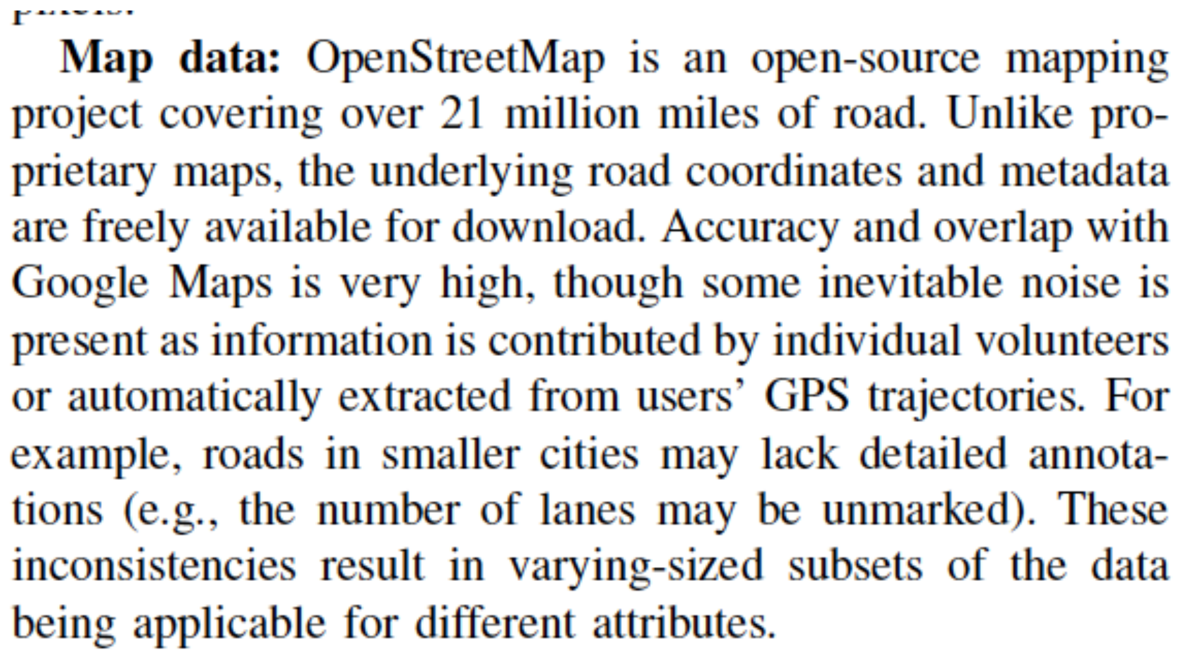
Pred = 18.5 m

## Road layout maps

- Potential cues
  - streetview
  - openmaps
  -

## Partially supervised cues

- Open Street Maps (OSM)



**Map data:** OpenStreetMap is an open-source mapping project covering over 21 million miles of road. Unlike proprietary maps, the underlying road coordinates and metadata are freely available for download. Accuracy and overlap with Google Maps is very high, though some inevitable noise is present as information is contributed by individual volunteers or automatically extracted from users' GPS trajectories. For example, roads in smaller cities may lack detailed annotations (e.g., the number of lanes may be unmarked). These inconsistencies result in varying-sized subsets of the data being applicable for different attributes.

Seff+Xiao

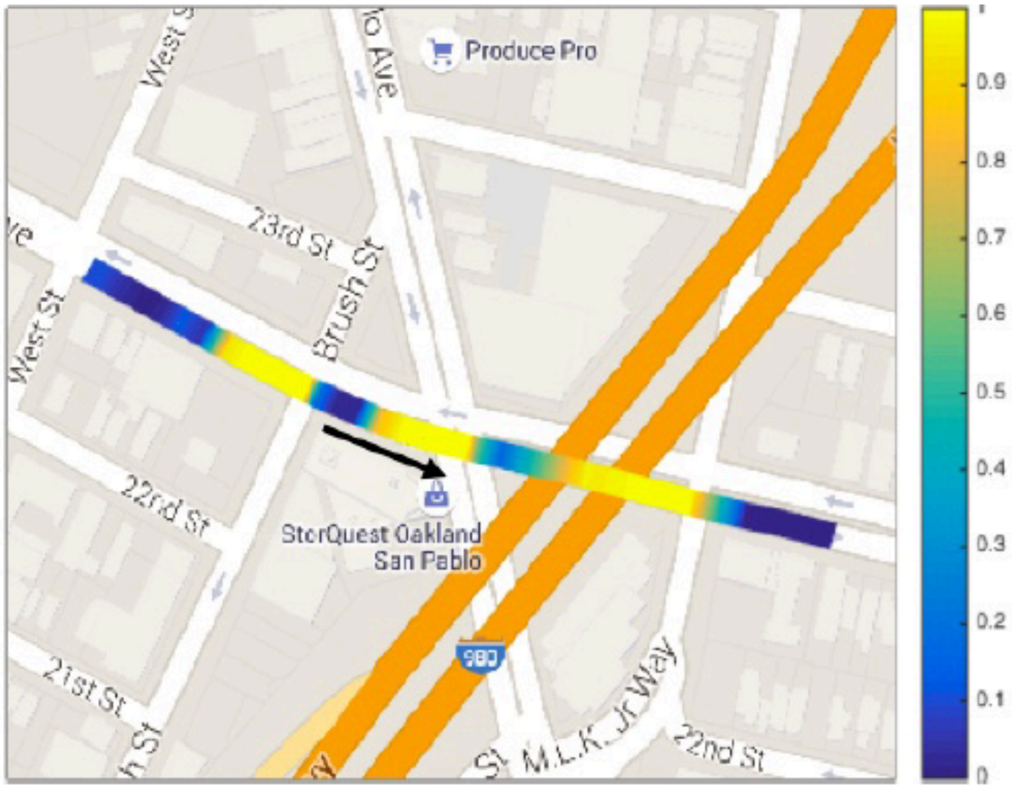


Fig. 3. Intersection detection heatmap. Images are cropped from test set GSV panoramas in the direction of travel indicated by the black arrow. The probabilities of "approaching" an intersection output by the trained ConvNet are overlaid on the road. (The images are from the ground level road, not the bridge.)

Seff+Xiao

## Partially supervised cues

- Google street view

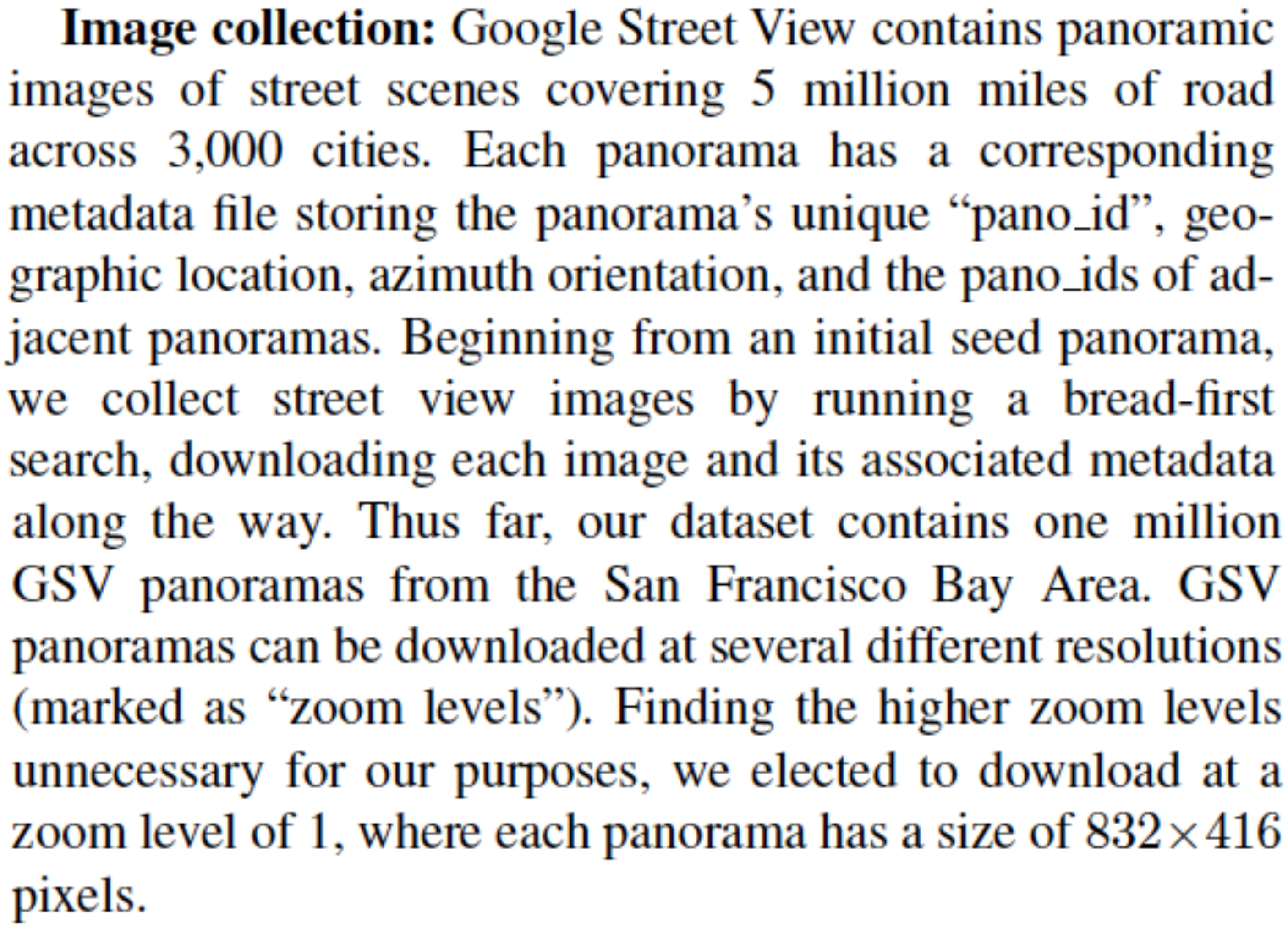


**Image collection:** Google Street View contains panoramic images of street scenes covering 5 million miles of road across 3,000 cities. Each panorama has a corresponding metadata file storing the panorama's unique "pano_id", geographic location, azimuth orientation, and the pano_ids of adjacent panoramas. Beginning from an initial seed panorama, we collect street view images by running a bread-first search, downloading each image and its associated metadata along the way. Thus far, our dataset contains one million GSV panoramas from the San Francisco Bay Area. GSV panoramas can be downloaded at several different resolutions (marked as "zoom levels"). Finding the higher zoom levels unnecessary for our purposes, we elected to download at a zoom level of 1, where each panorama has a size of 832×416 pixels.

Seff+Xiao

## Labelling - I

- Match panoramas to roads
  - panorama center location, orientation is known
  - (essentially) project to plane
  - thresholded nearest neighbor to road center polyline
    - thresholding removes panoramas inside buildings, etc.
  - some noise
    - under bridges, etc.
- Annotations
  - Intersections
  - Drivable heading
  - Heading angle
  - Bike lane
  - Speed limit, wrong way, etc.

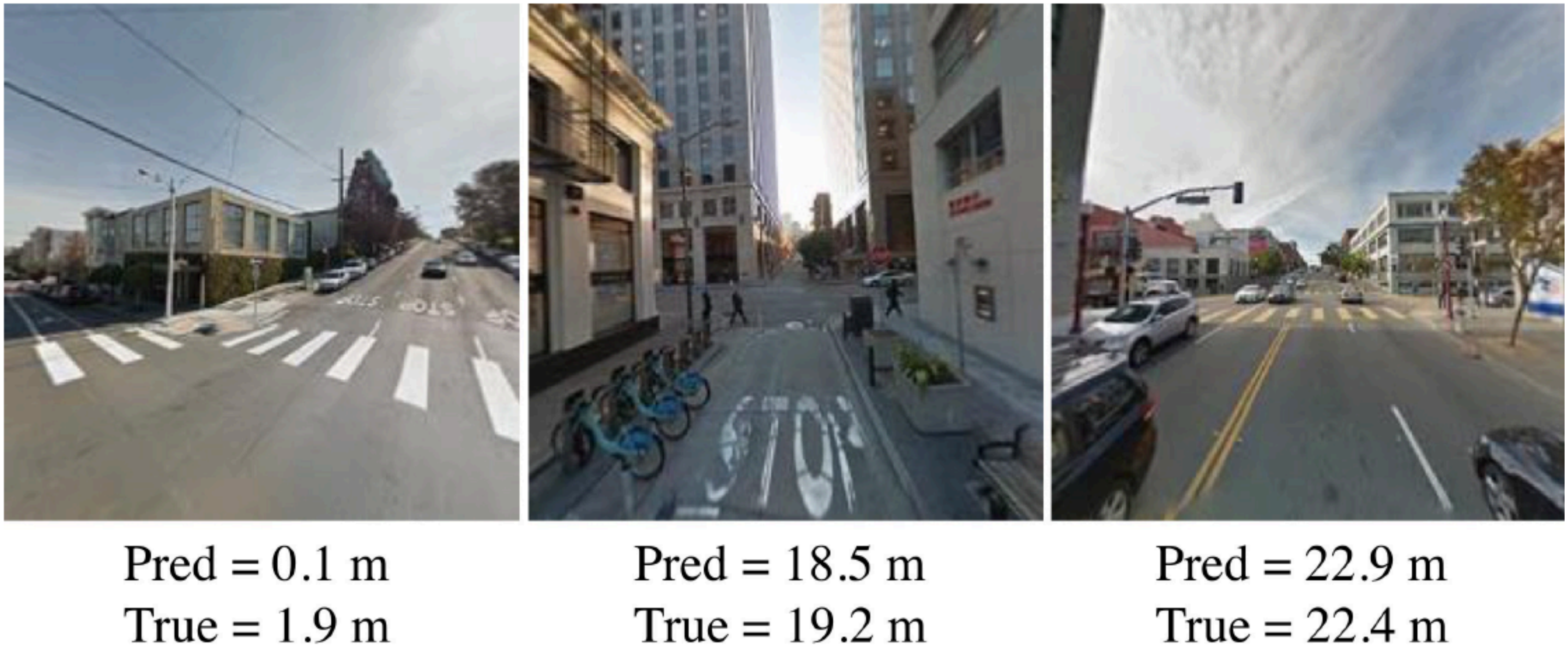


Fig. 4. Distance to intersection estimation. For images within 30 m of true intersections, our model is trained to estimate the distance from the host car to the center of the intersection across a variety of road types.

Seff+Xiao

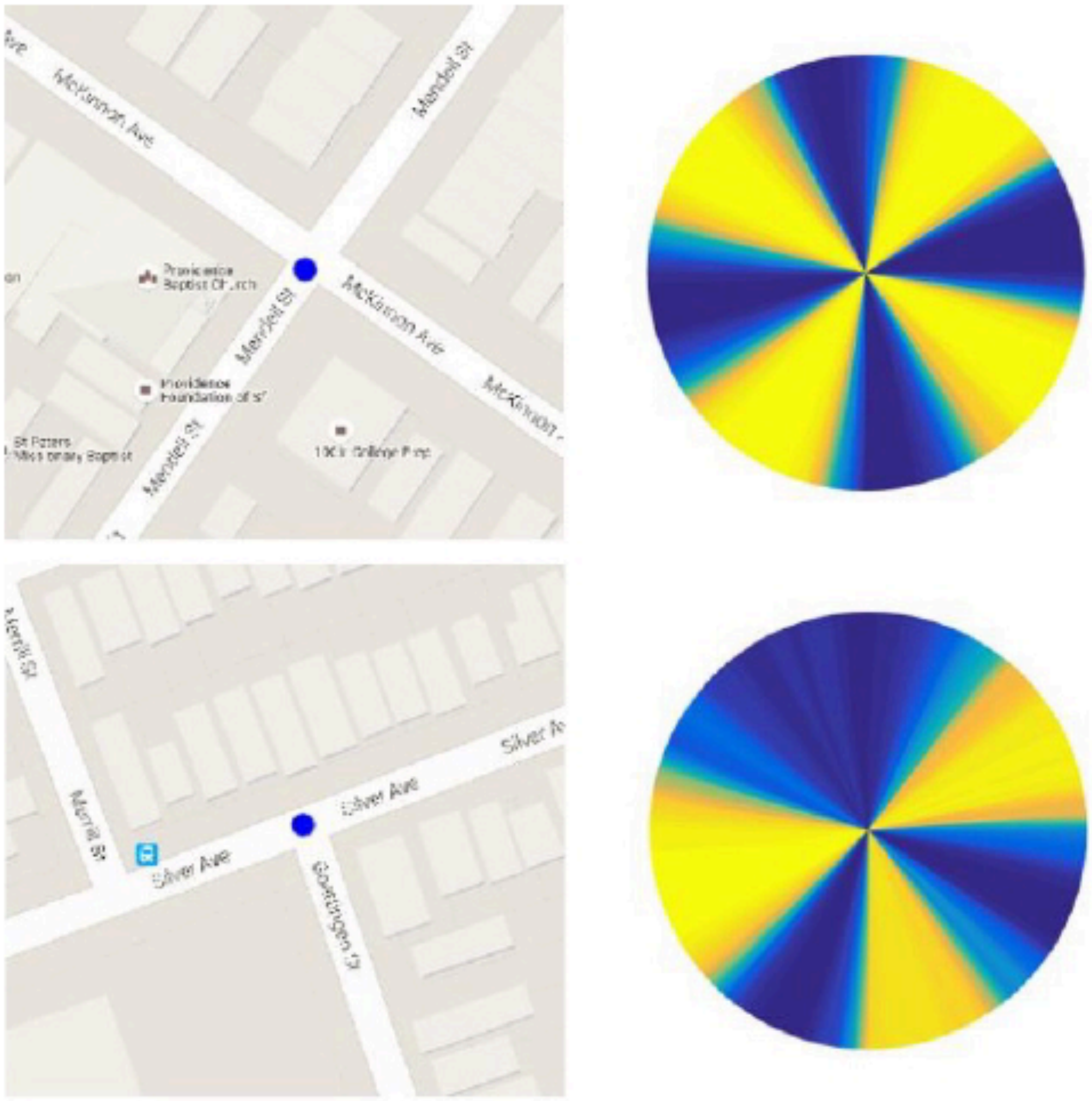


Fig. 5. Intersection topology is one of several attributes our model learns to infer from an input GSV panorama. The blue circles on the Google Maps extracts to the left show the locations of the input panoramas. The pie charts display the probabilities output by the trained ConvNet of each heading angle being on a driveable path (see Figure 3 for colormap legend).

Seff+Xiao

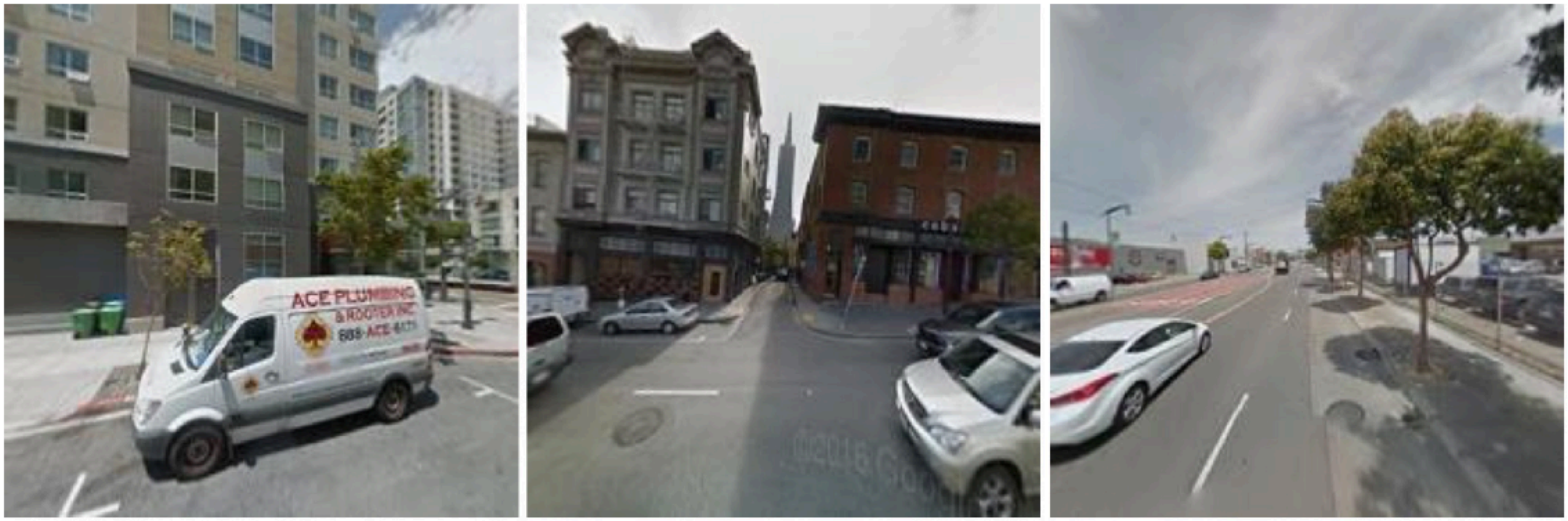
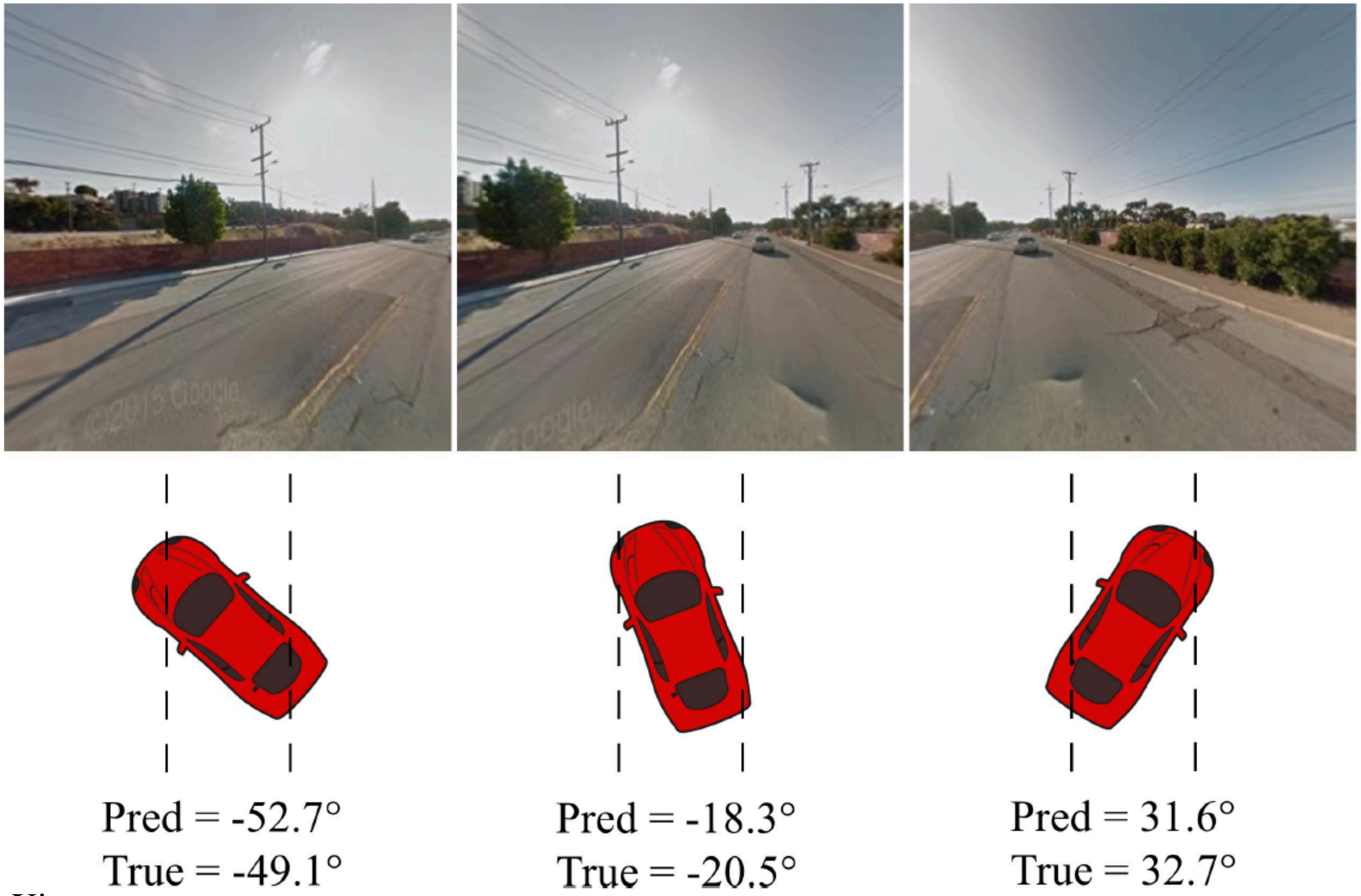


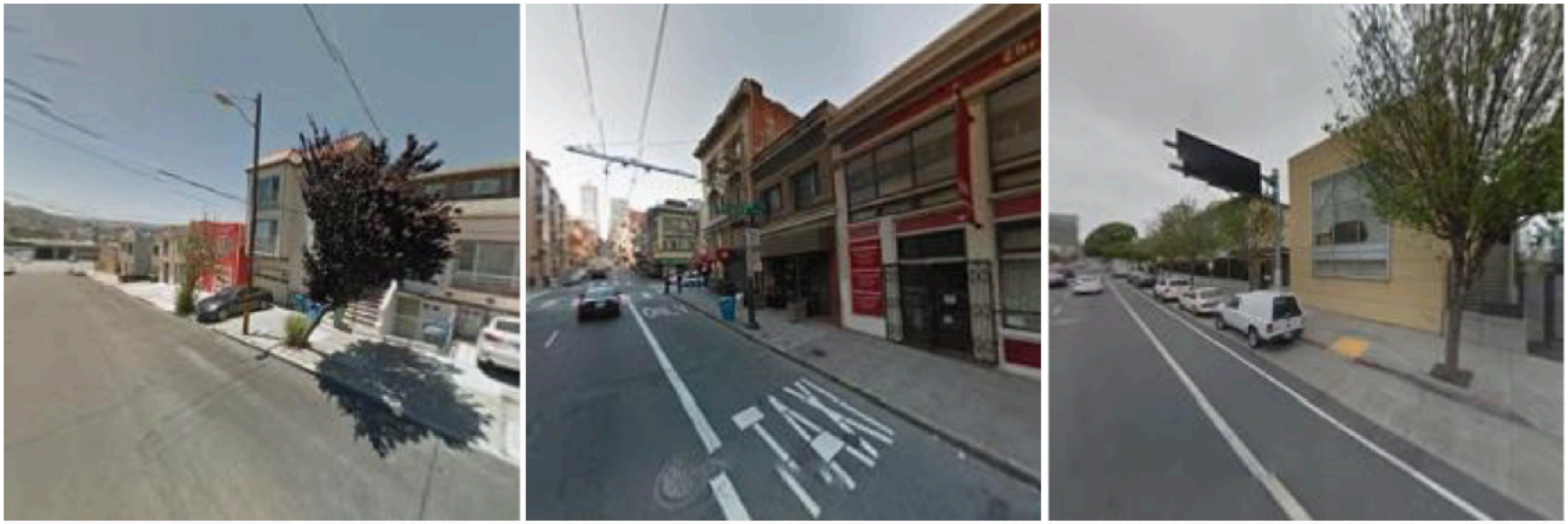
Fig. 6. Driveable headings. A ConvNet is trained to distinguish between non-drivable headings (left) and drivable headings aligned with the road (right). The ConvNet weakly classifies the middle example as drivable because the host car's heading is facing the alleyway between the buildings.

Seff+Xiao



Seff+Xiao

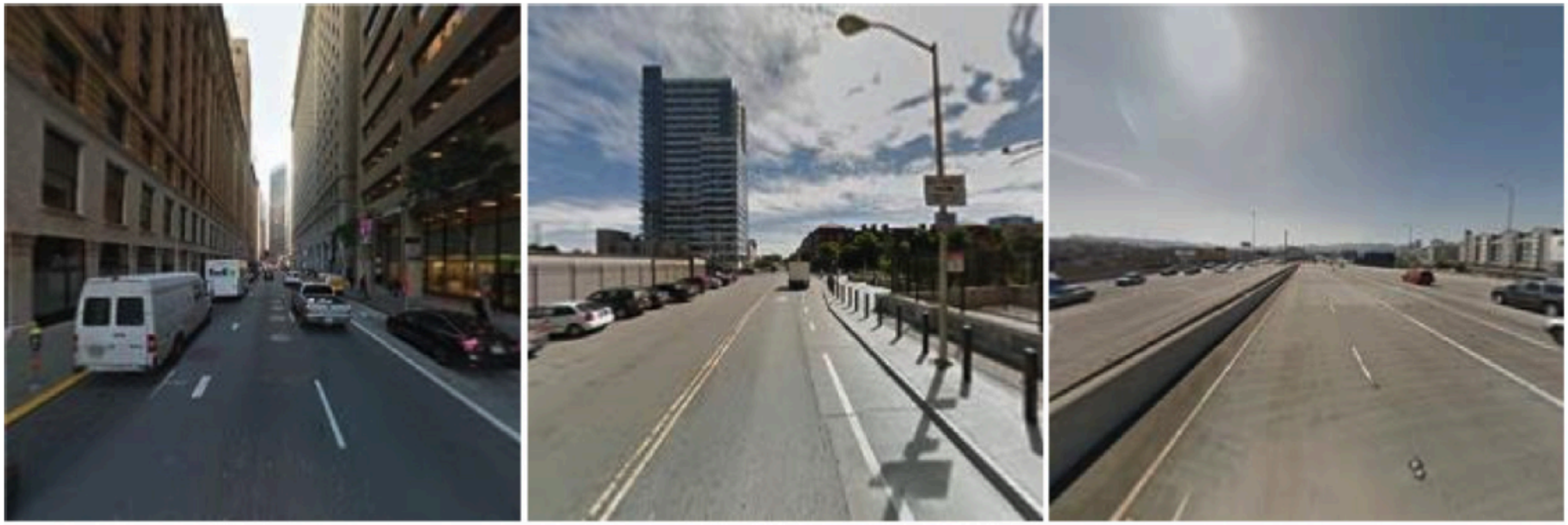
Fig. 7. Heading angle regression. The network learns to predict the relative angle between the street and host vehicle heading given a single image cropped from a GSV panorama. Below each GSV image, the graphic visualizes the ground truth heading angle.



p(bike lane) = 0.043 p(bike lane) = 0.604 p(bike lane) = 0.988

Fig. 8. The ConvNet learns to detect bike lanes adjacent to the vehicle. The GSV images are arranged from left to right in increasing order of probability output by the ConvNet of a bike lane being present (ground truth labels from left to right are negative, negative, positive). The middle example contains a taxi lane, resulting in a weak false positive.

Seff+Xiao



Pred = 26.1 mph
True = 30 mph

Pred = 30.0 mph
True = 50 mph

Pred = 54.3 mph
True = 50 mph

Fig. 9. Speed limit regression. The network learns to predict speed limits given a GSV image of road scene. The model significantly underestimates the speed limit in the middle example as this type of two-way road with a single lane in each direction would generally not have a speed limit as high as 50 mph.

Seff+Xiao

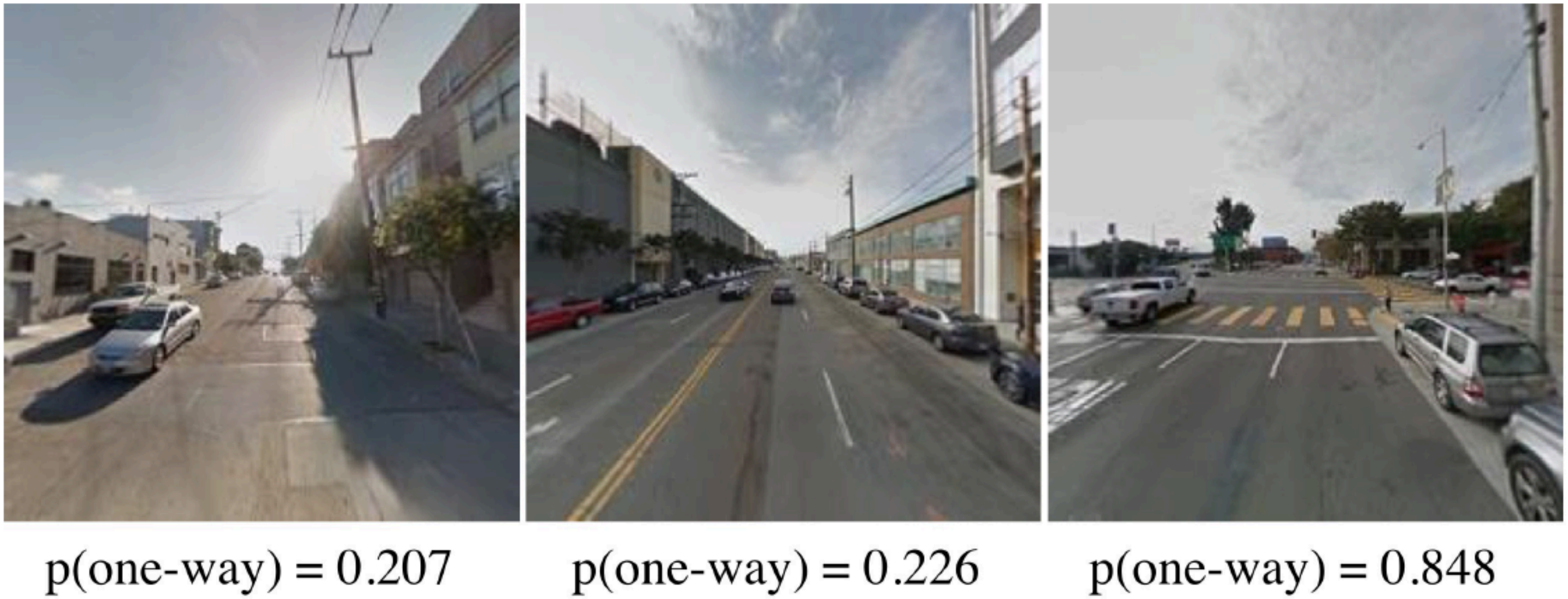


Fig. 10. One-way vs. two-way road classification. The probability output by the ConvNet of each GSV scene being on a one-way road is shown. From left to right the ground truth labels are two-way, two-way, and one-way. The image on the left is correctly classified as two-way despite the absence of the signature double yellow lines.

Seff+Xiao

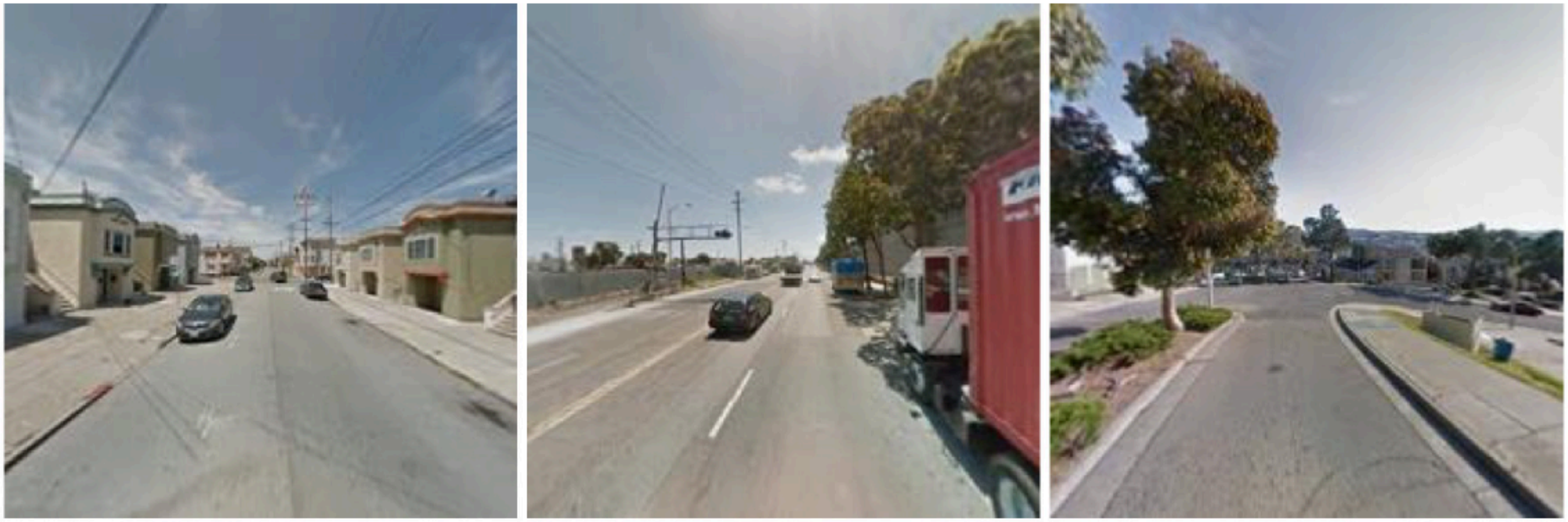


Fig. 11. Wrong way detection. The probability output by the ConvNet of each GSV image facing the wrong way on the road is displayed. From left to right the ground truth labels are wrong way, right way, and right way. For two-way roads with no lane markings (left), this is an especially difficult problem as it amounts to estimating the horizontal position of the host car. The problem can also be quite ill-defined if there are no context clues as is the case with the rightmost image.

Seff+Xiao

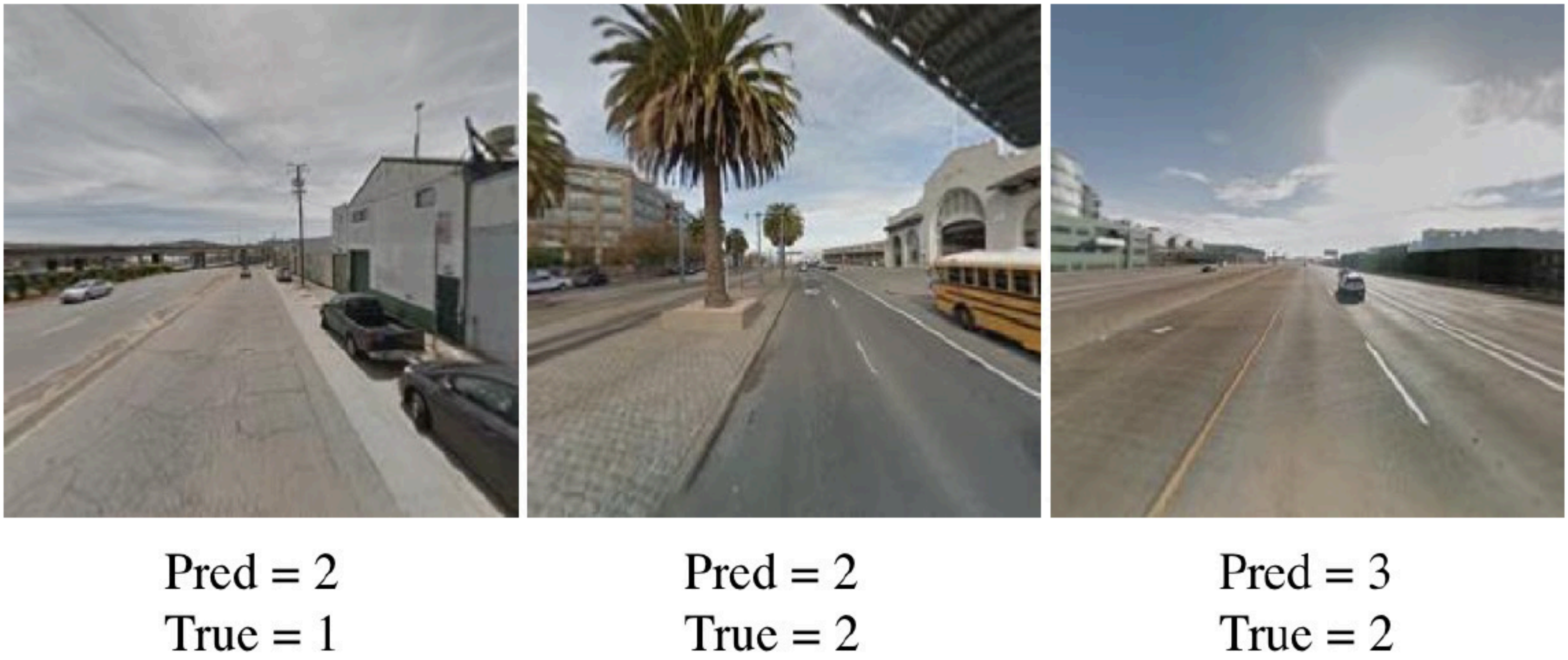


Fig. 12. Number of lanes estimation. The predicted and true number of lanes for three roads are displayed along with the corresponding GSV images. For streets without clearly visible lane markings (left), this is especially challenging. Although the ground truth for the rightmost image is two lanes, there is a third lane that merges just ahead.

Seff+Xiao

## At this point

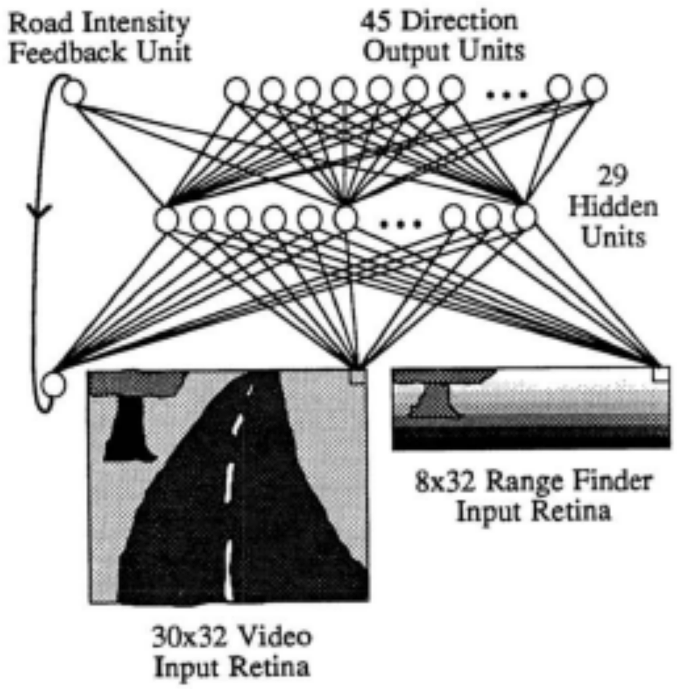
- I can tell from an image whether
  - I’m pointing in the right direction
  - going the right way
  - facing an intersection
  - available turns, etc.
  - what and where street signs are
  - …
- Can I build a reliable controller?

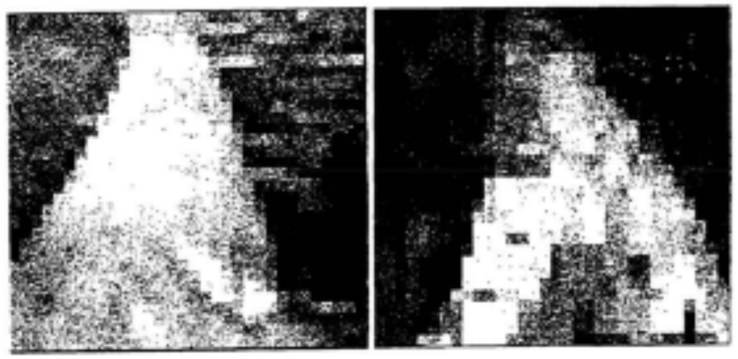
## BIG GOOD QUESTIONS

- Mashup of openmaps and street view
  - it could predict drivable directions, steering directions, lanes, signs, etc.
- Q:  WHY IS THIS NOT DRIVING AROUND NOW?
  - A: (pretty obviously) because it doesn’t work
- Q: WHY NOT?
  - A: interesting

# Imitation learning

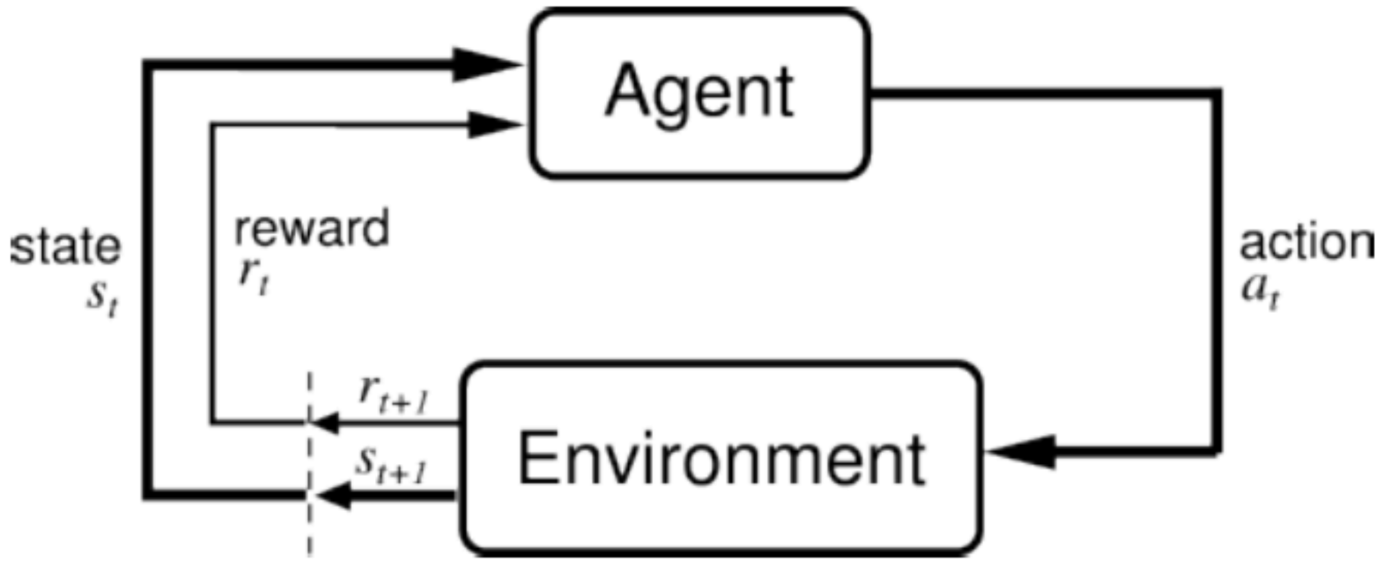
- Approaches
  - Imitation learning:
    - Train a policy that does “the same thing” as an expert


Road Intensity Feedback Unit
45 Direction Output Units
29 Hidden Units
8x32 Range Finder Input Retina
30x32 Video Input Retina



*simulated road images should help eliminate these difficulties and facilitate better performance*. “ *ALVINN: An autonomous Land vehicle in a neural Network, Pomerleau 1989*

## Markov Decision Process

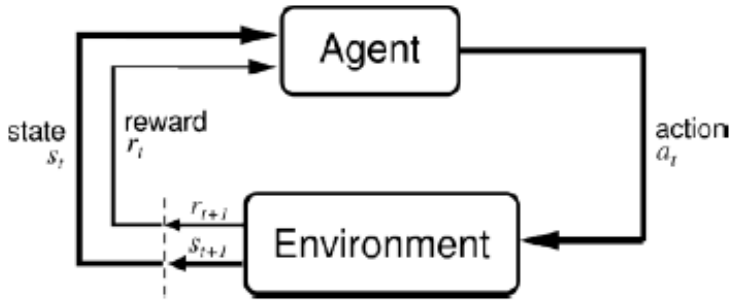


Assumption: agent gets to observe the state

[Drawing from Sutton and Barto, Reinforcement Learning: An Introduction, 1998]

Abbeel slides

## Markov Decision Process (S, A, T, R, H)



Given

- S: set of states
- A: set of actions
- T: S x A x S x {0,1,…,H} → [0,1], $T_t(s,a,s') = P(s_{t+1} = s' \mid s_t = s, a_t = a)$
- R: S x A x S x {0, 1, …, H} → $\Re$ $R_t(s,a,s')$ = reward for $(s_{t+1} = s', s_t = s, a_t = a)$
- H: horizon over which the agent will act

Goal:

- Find $\pi$ : S x {0, 1, …, H} → A that maximizes expected sum of rewards, i.e.,

$$\pi^* = \arg\max_{\pi} \mathrm{E}[\sum_{t=0}^{H} R_t(S_t, A_t, S_{t+1}) | \pi]$$

$\uparrow$

This is usually discounted by gamma

Abbeel slides

# Questions

- Known environment, rewards
  - Assume
    - we know T(s, a, s'), R(s, a, s')
  - What should our policy be?
    - do math

Solving MDPs

- Unknown environment, rewards

Reinforcement learning

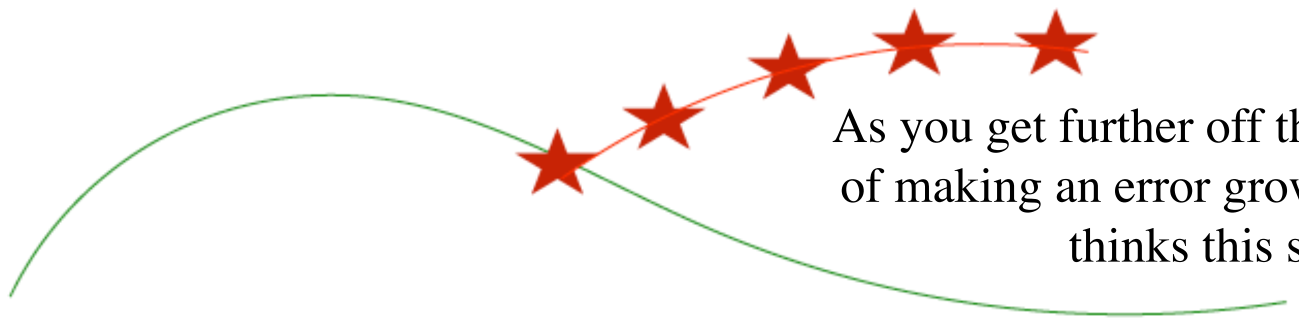
  - What should our policy be?
    - act and adjust policy to improve rewards
- Unknown environment, rewards, but access to expert
  - What should our policy be?
    - (a1) do what the expert does

Imitation learning

    - (a2) figure out the experts reward function, and maximize that

Inverse reinforcement learning

## Compounding Errors



As you get further off the path, the probability of making an error grows, cause the classifier thinks this state is rare

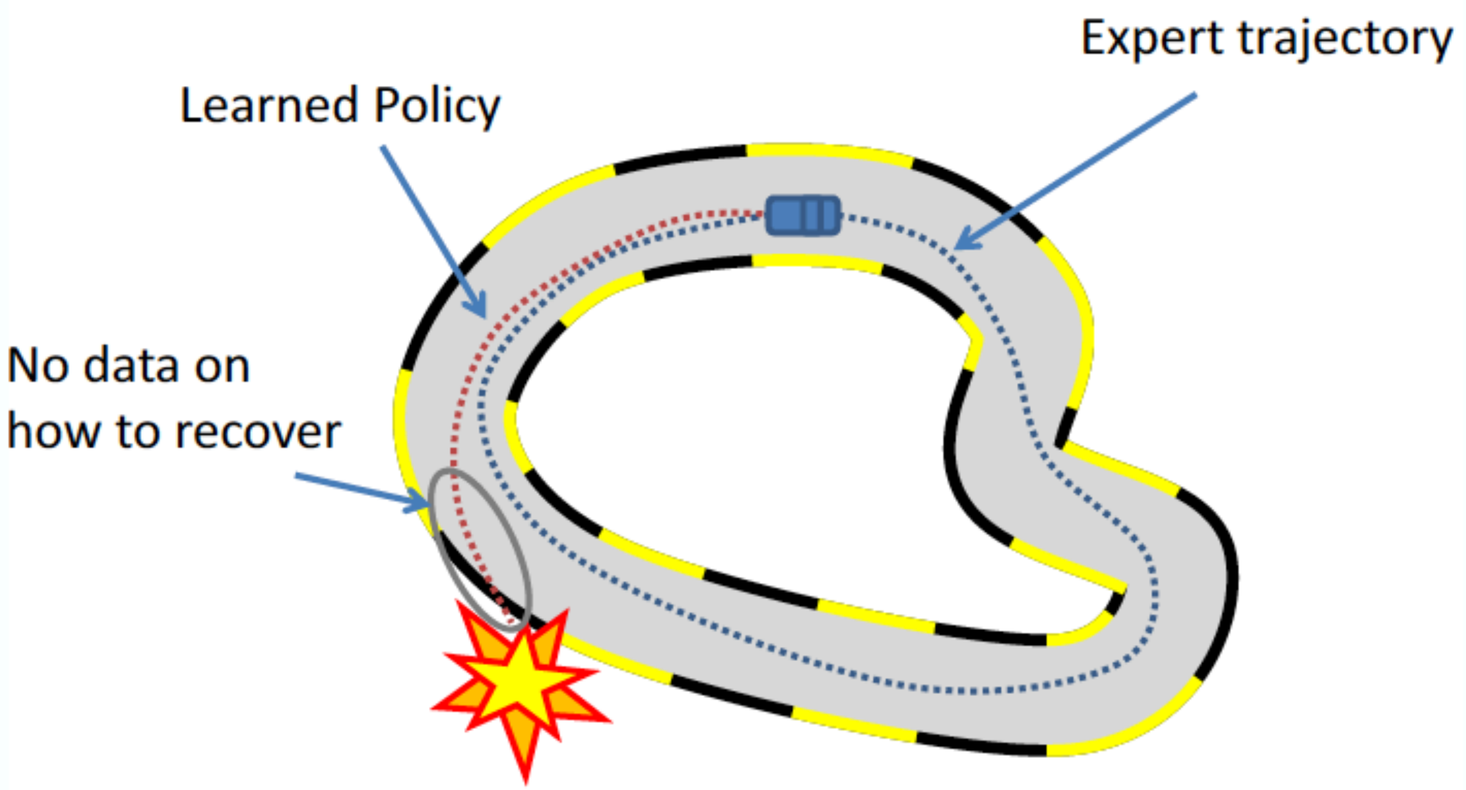
error at time t with probability ε

E[Total errors] $\lesssim \varepsilon(T + (T-1) + (T-2) + \ldots + 1) \propto \varepsilon T^2$

Fragkiadaki, ND A Reduction of Imitation Learning and Structured Prediction to No-Regret Online Learning, Ross et al. 2011

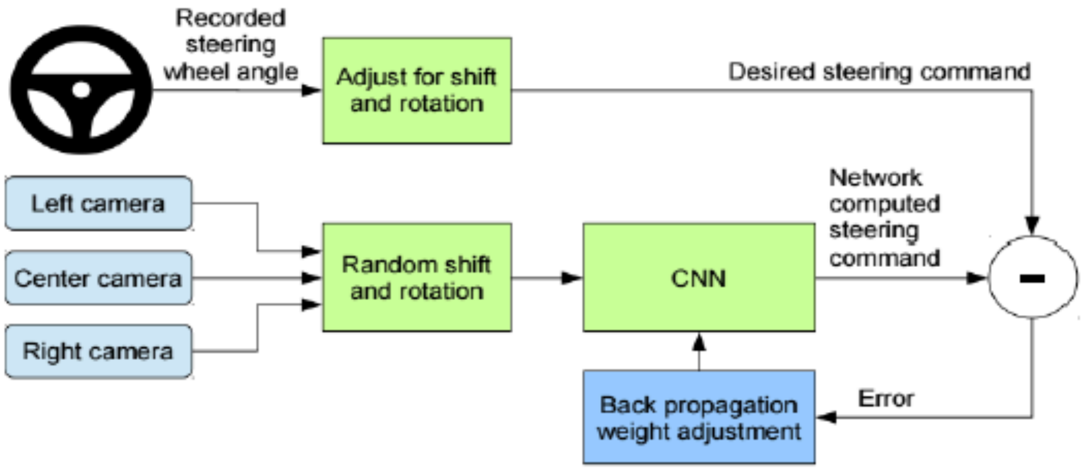
## Data Distribution Mismatch!

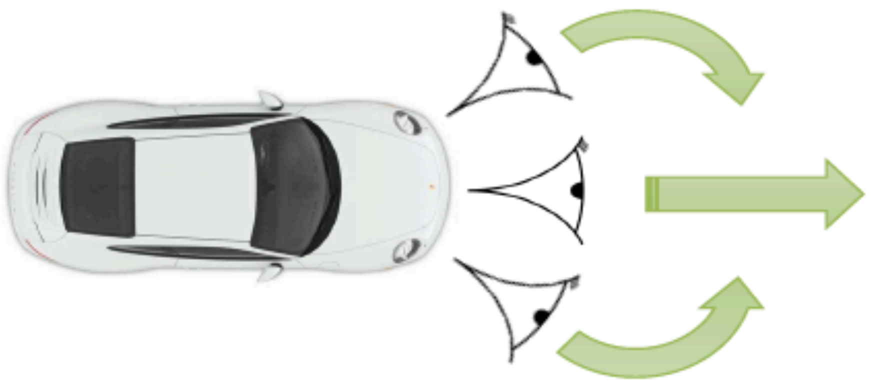
$$p_{\pi^*}(o_t) \neq p_{\pi_\theta}(o_t)$$



Fragkiadaki, ND

## Demonstration Augmentation: NVIDIA 2016


Recorded steering wheel angle
Adjust for shift and rotation
Desired steering command
Left camera
Center camera
Right camera
Random shift and rotation
CNN
Network computed steering command
-
Back propagation weight adjustment
Error



Additional, left and right cameras with automatic grant-truth labels to recover from mistakes

"*DAVE-2 was inspired by the pioneering work of Pomerleau [6] who in 1989 built the Autonomous Land Vehicle in a Neural Network (ALVINN) system*. Training with data from only the human driver is not sufficient. The network must learn how to recover from mistakes. …",

End to End Learning for Self-Driving Cars , Bojarski et al. 2016

# DAGGER (in simulation)

Dataset AGGregation: bring learner's and expert's trajectory distributions closer by labelling additional data points resulting from applying the current policy

1. train $\pi_\theta(u_t|o_t)$ from human data $\mathcal{D}_{\pi^*} = \{o_1, u_1, ..., o_N, u_N\}$
2. run $\pi_\theta(u_t|o_t)$ to get dataset $\mathcal{D}_\pi = \{o_1, ..., o_M\}$
3. Ask human to label $\mathcal{D}_\pi$ with actions $u_t$
4. Aggregate: $\mathcal{D}_{\pi^*} \leftarrow \mathcal{D}_{\pi^*} \cup \mathcal{D}_\pi$
5. GOTO step 1.

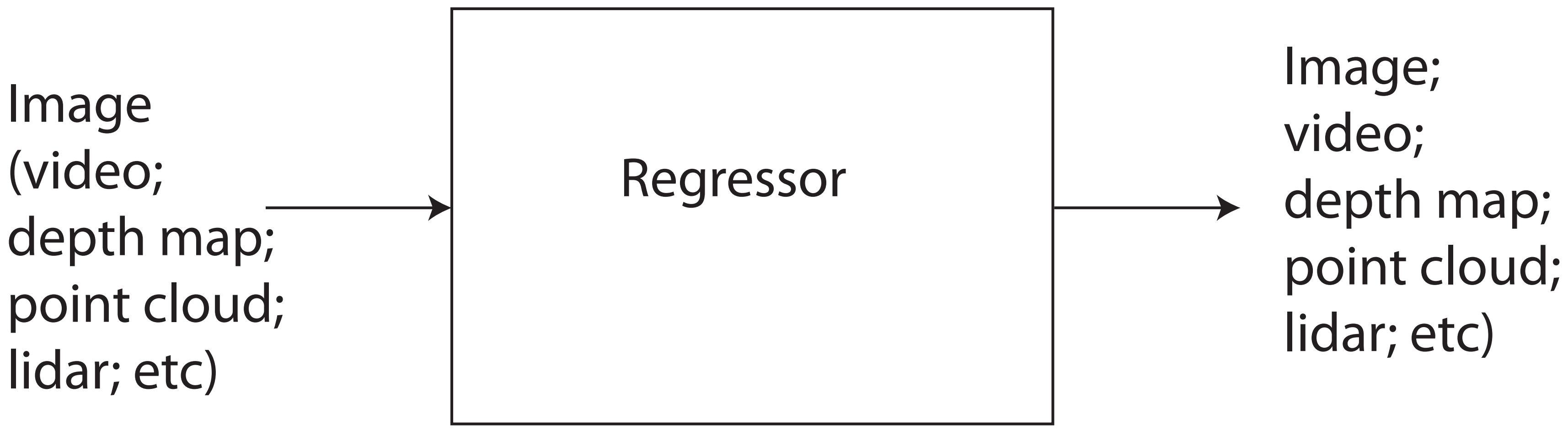
Problems:

- execute an unsafe/partially trained policy
- repeatedly query the expert

# Sensors, Localization and Control

- -1: Intro - who I am, UIUC, what we're going to do
- 0: Big tools of vision
  - geometry ; discrimination ; regression
  - Simple visual control
    - Google streetview; various learning paradigms;
    - why imitation learning is hard; Dagger
- I: Sensor review
  - Cameras; LIDAR; timed cameras; GPS
- II: Localization
  - simple 2D, 3D registration by matching; features for matching
  - matching in paired cameras; F, E matrices; odometry

## Big ideas in vision: Regression



## Why regression?

- It’s useful
  - Depth map from a single image
  - Ground map from aerial image
  - Modified/improved image from image
  - etc.

## Key ideas

- Goal:
  - Adjust regressor so that it accurately predicts for *UNSEEN* input
- Procedure:
  - Adjust so that it
    - predicts for well for training data
    - generalizes
      - regularization term, either explicit or implicit
- Evaluation:
  - Use held out data to check accuracy on *UNSEEN* data

## Key questions for this part

- Generally: produce representations that improve control
  - by interpreting sensor output
- Today:
  - Where am I?
    - registration and building simple maps
  - How have I moved?
    - visual odometry
    - (SLAM mentioned, but omitted as requiring filtering)

# Sensors, Localization and Control

- -1: Intro - who I am, UIUC, what we're going to do
- 0: Big tools of vision
  - geometry ; discrimination ; regression
- I: Sensor review
  - Cameras; LIDAR; timed cameras; GPS
- II: Localization
  - simple 2D, 3D registration by matching; features for matching
  - matching in paired cameras; F matrix; odometry; SLAM

## Sensors

# Sensors, Localization and Control

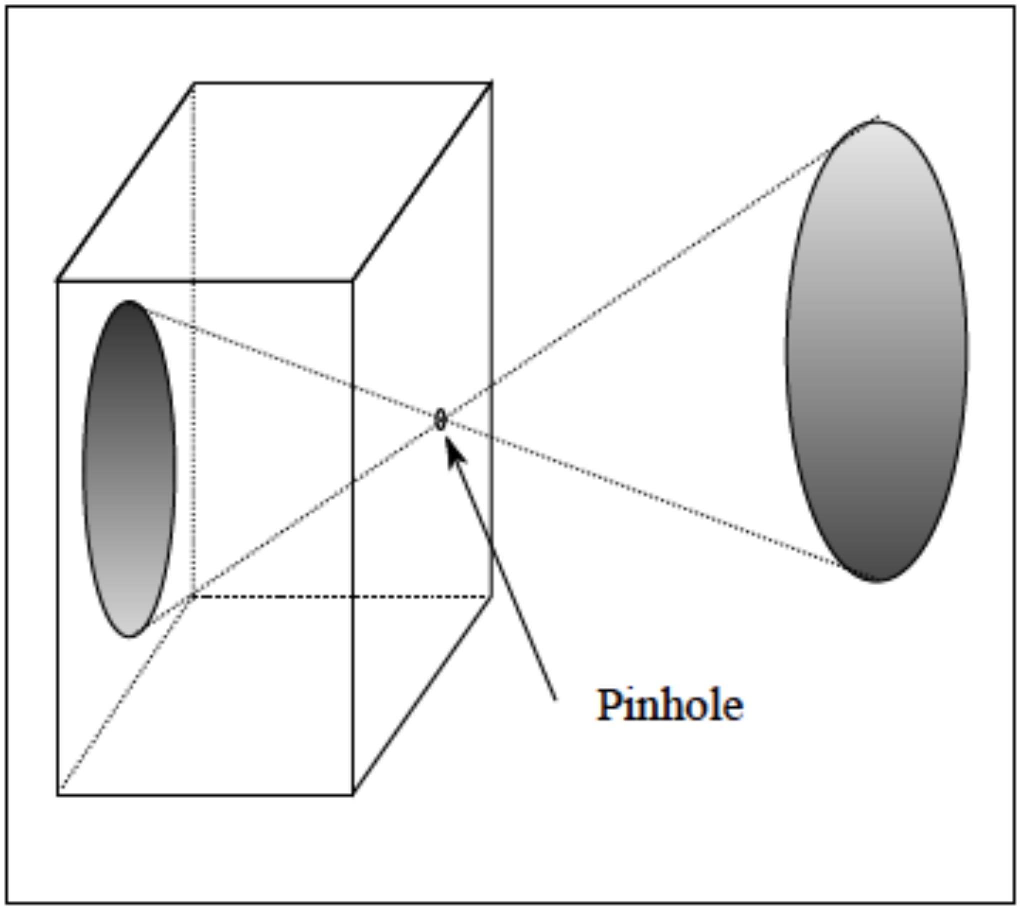
- -1: Intro - who I am, UIUC, what we're going to do
- 0: Big tools of vision
  - geometry; discrimination ; regression
  - Simple visual control
    - Google streetview; various learning paradigms;
    - why imitation learning is hard; Dagger
- I: Sensor review
  - Cameras; LIDAR; timed cameras; GPS
- II: Localization
  - simple 2D, 3D registration by matching; features for matching
  - matching in paired cameras; F, E matrices; odometry

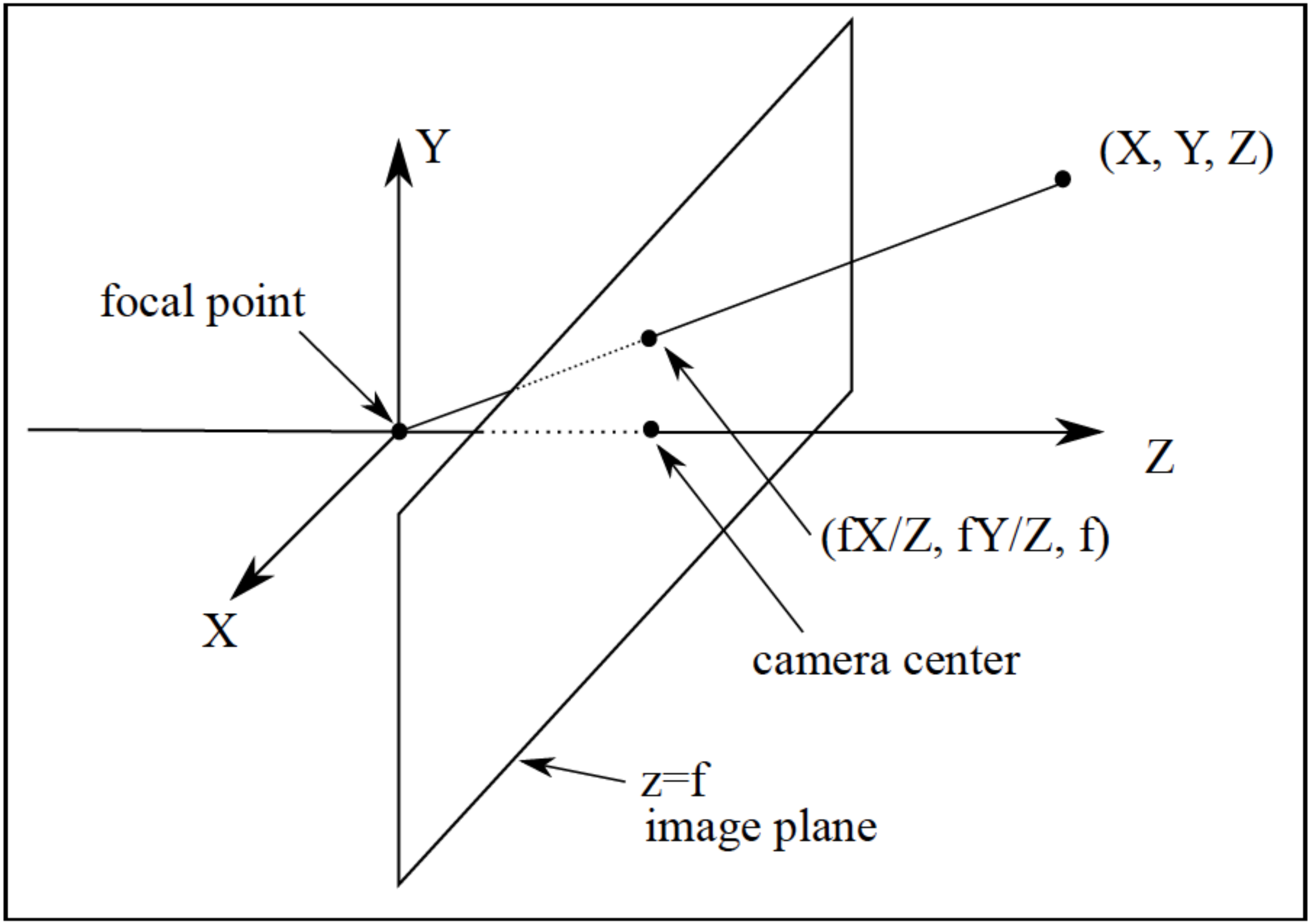
## Camera basics

- Perspective
- Lenses
- Extrinsics and intrinsics
- Calibration

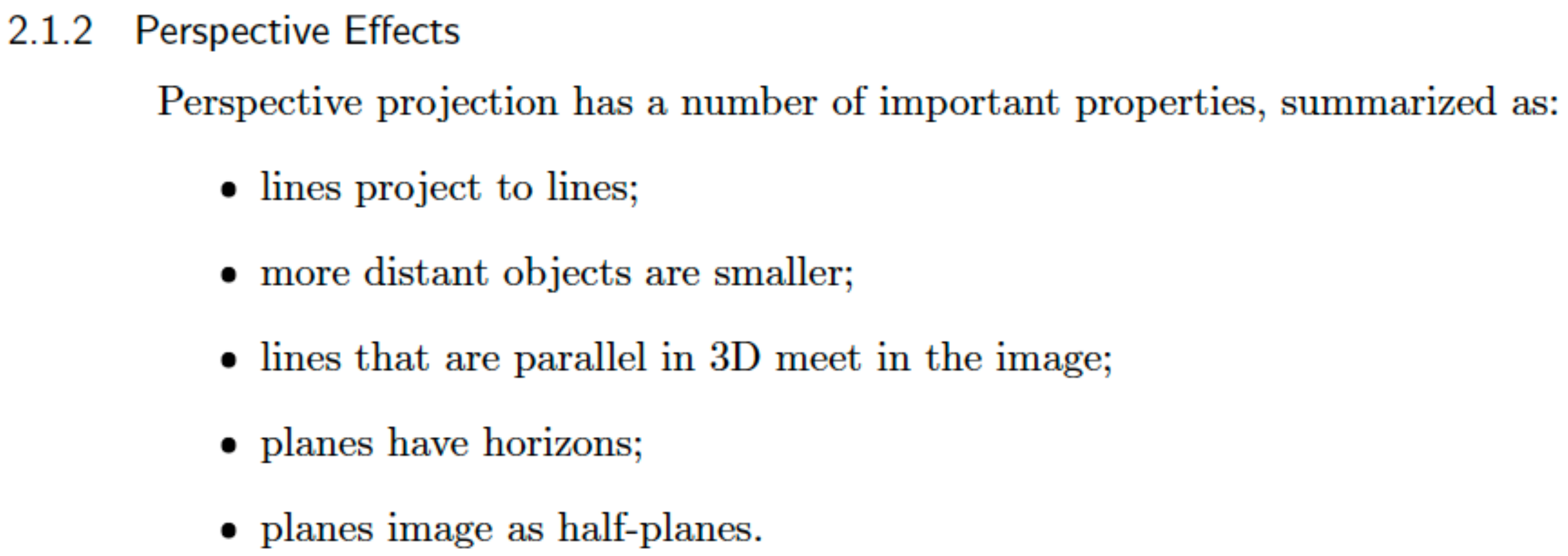
## Pinhole cameras

- Abstract camera model - box with a small hole in it
- Pinhole cameras work in practice




Y
(X, Y, Z)
focal point
Z
X
(fX/Z, fY/Z, f)
camera center
z=f
image plane

# Perspective effects



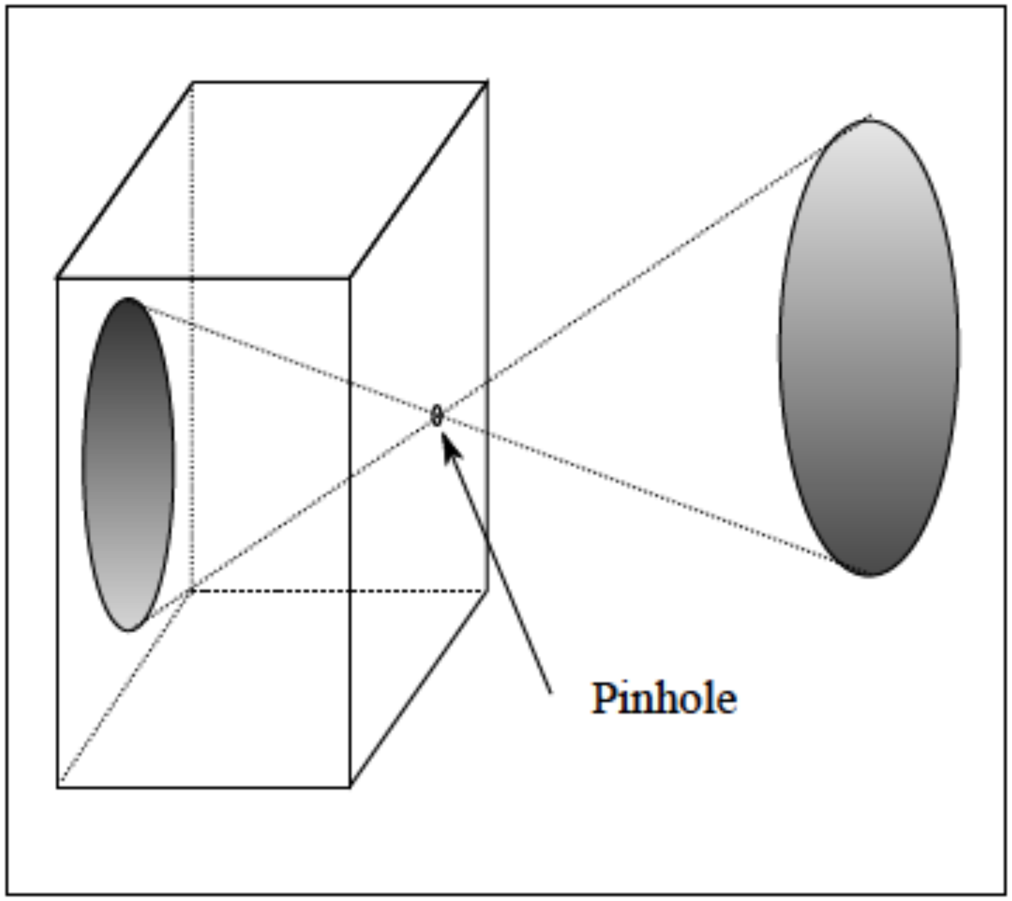
### 2.1.2 Perspective Effects

Perspective projection has a number of important properties, summarized as:

- lines project to lines;
- more distant objects are smaller;
- lines that are parallel in 3D meet in the image;
- planes have horizons;
- planes image as half-planes.

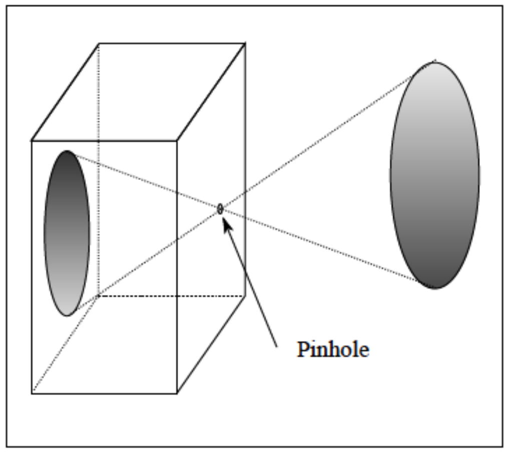
## Lenses

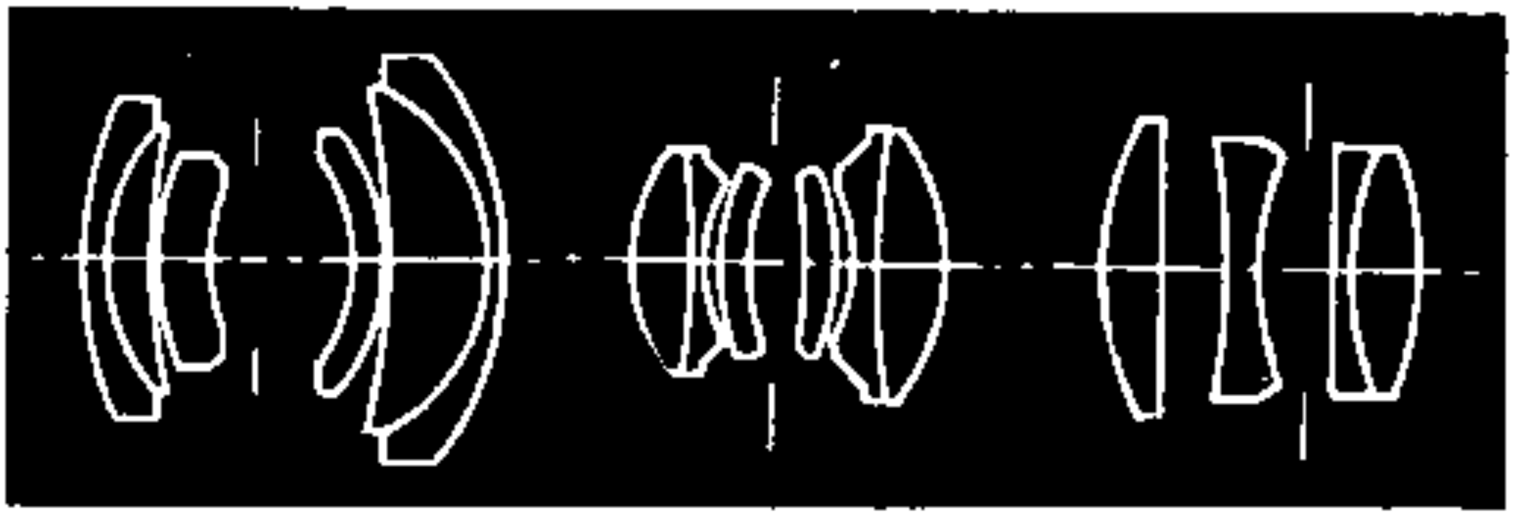
- Pinholes admit very little light
  - and only work because they are small



# Lenses

- Pinholes admit very little light
  - and only work because they are small
- Lens system
  - Collect much more light
  - Ensure camera behaves like a pinhole camera
    - roughly!





## Possibly annoying lens phenomena

- Spherical aberration
  - Lens is not a perfect thin lens, and point is defocused
- Chromatic aberration
  - Light at different wavelengths follows different paths; hence, some wavelengths are defocussed
  - Machines: coat the lens
  - Humans: live with it
- Scattering at the lens surface
  - Light in lens system reflects off surfaces
  - Machines: coat the lens, interior
  - Humans: live with it (scattering phenomena are visible in the human eye)
- Geometric phenomena (Barrel distortion, etc.)

## Geometric distortions in lenses

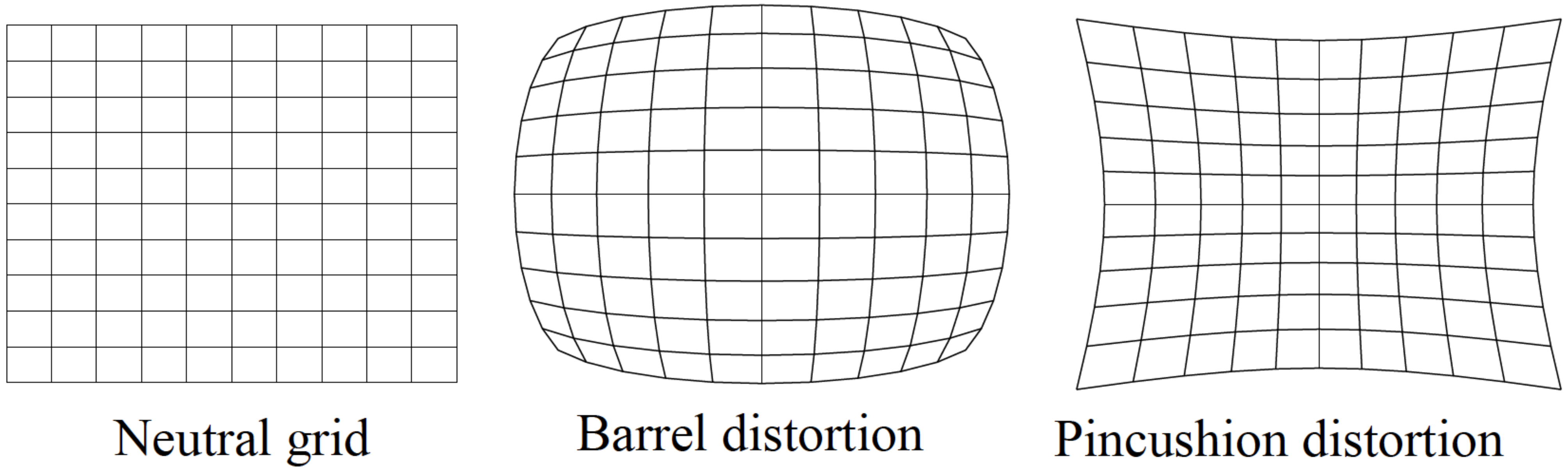


FIGURE 2.6: *On the* **left** *a neutral grid observed in a non-distorting lens (and viewed frontally to prevent any perspective distortion).* **Center** *shows the same grid, viewed in a lens that produces barrel distortion.* **Right**, *the same grid, now viewed in a lens that produces pincushion distortion.*

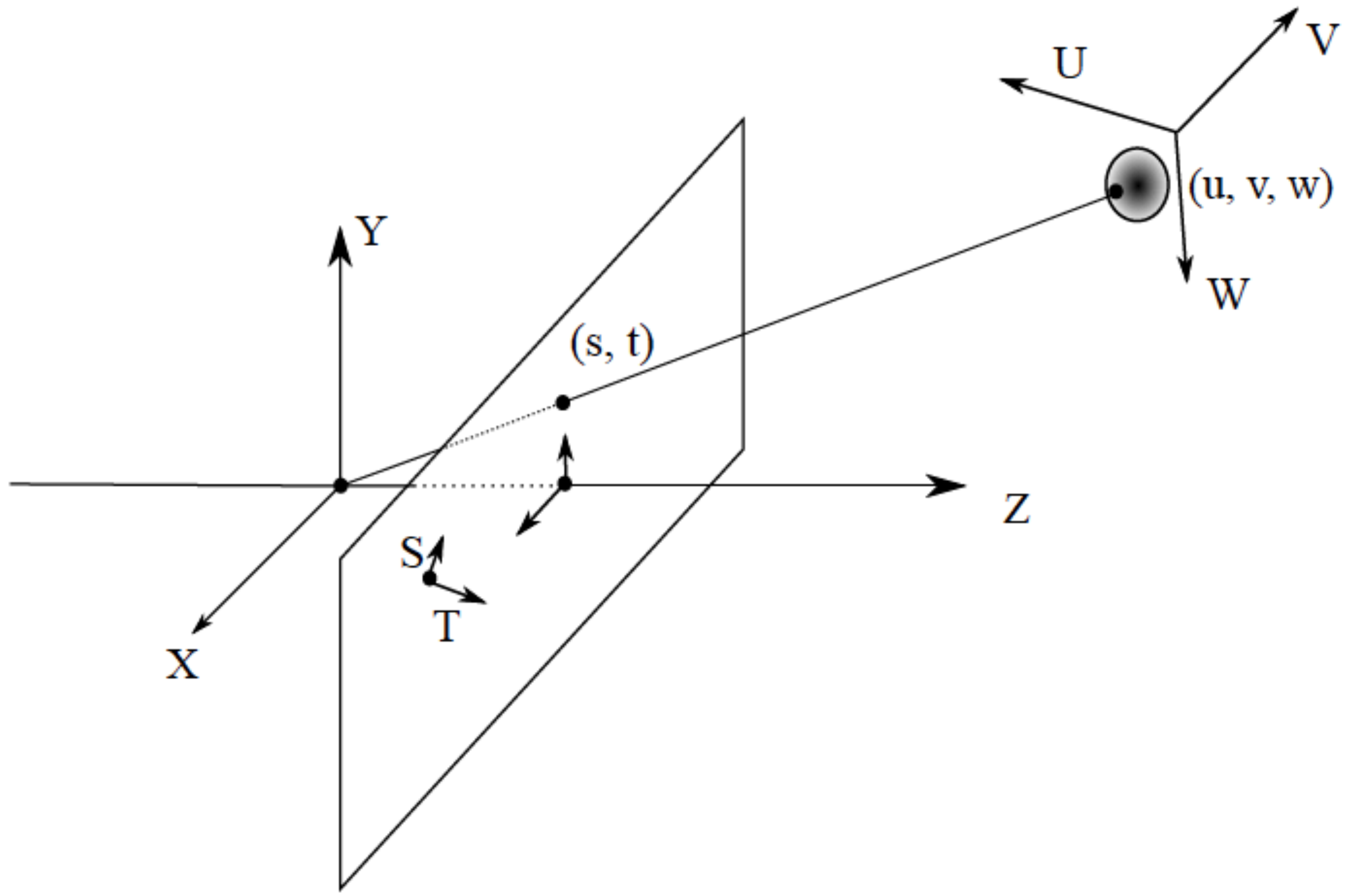


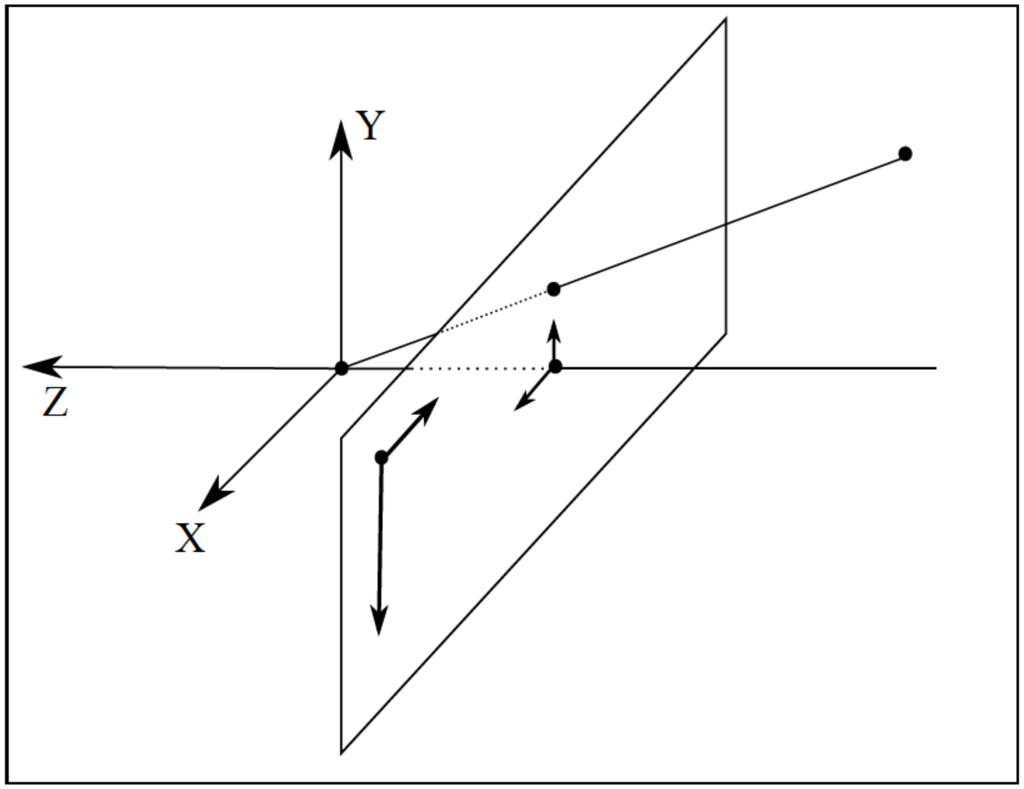
FIGURE 2.7: *A perspective camera (in its own coordinate system, given by $X$, $Y$ and $Z$ axes) views a point in world coordinates (given by $(u, v, w)$ in the $UVW$ coordinate system) and reports the position of points in $ST$ coordinates. We must model the mapping from $(u, v, w)$ to $(s, t)$, which consists of a transformation from the $UVW$ coordinate system to the $XYZ$ coordinate system followed by a perspective projection followed by a transformation to the $ST$ coordinate system.*

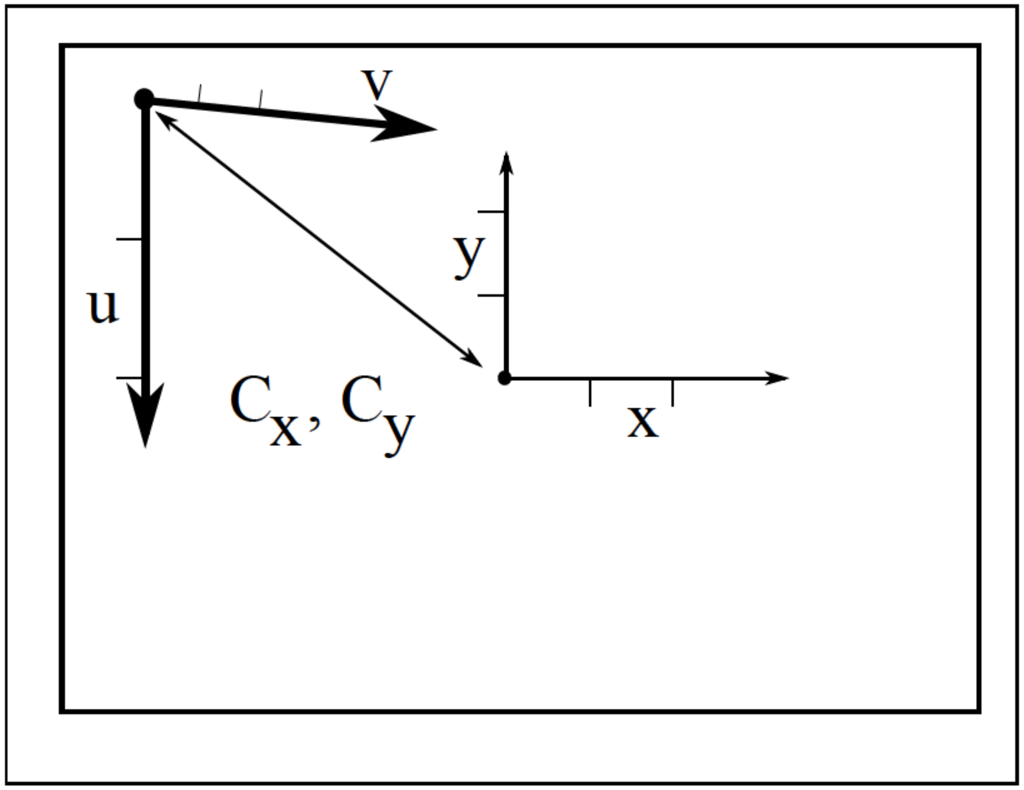
## The camera matrix - II

- Turn previous expression into HC’s
  - HC’s for 3D point are (X,Y,Z,T)
  - HC’s for point in image are (U,V,W)

$$\begin{bmatrix} I_1 \\ I_2 \\ I_3 \end{bmatrix} = \begin{bmatrix} \text{Transformation} \\ \text{mapping image} \\ \text{plane coords to} \\ \text{pixel coords} \end{bmatrix} \mathcal{C}_p \begin{bmatrix} \text{Transformation} \\ \text{mapping world} \\ \text{coords to camera} \\ \text{coords} \end{bmatrix} \begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \end{bmatrix}$$

## Camera intrinsics

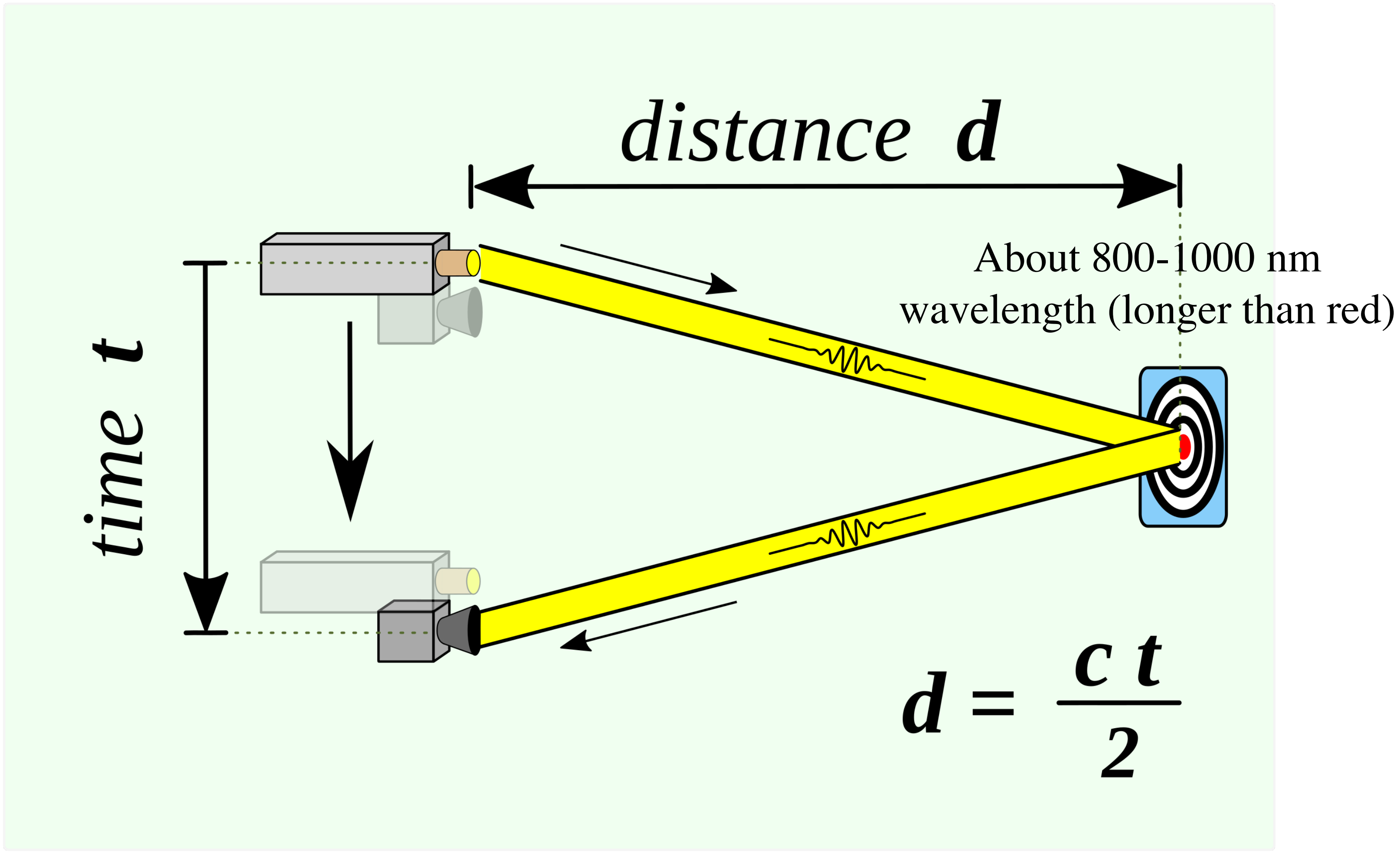




## Camera calibration

- Issues:
  - what are intrinsic parameters of the camera?
  - what is the camera matrix? (intrinsic+extrinsic)
- General strategy:
  - view calibration object; identify image points in image
  - obtain camera matrix by minimizing error
  - obtain intrinsic parameters from camera matrix

# Lidar



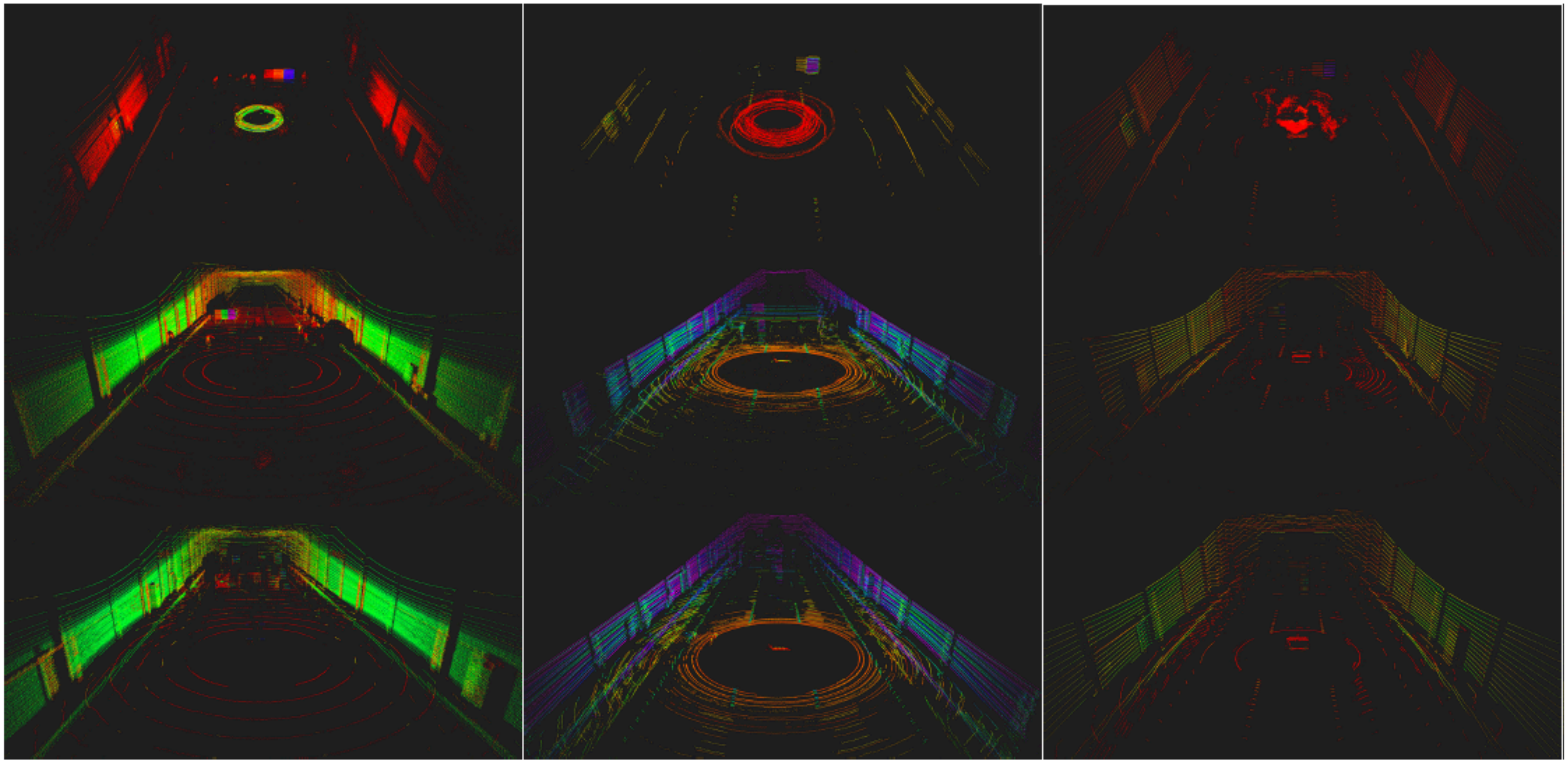
$$\mathbf{d} = \frac{\mathbf{c\,t}}{2}$$

Wikipedia

Fog

Rain

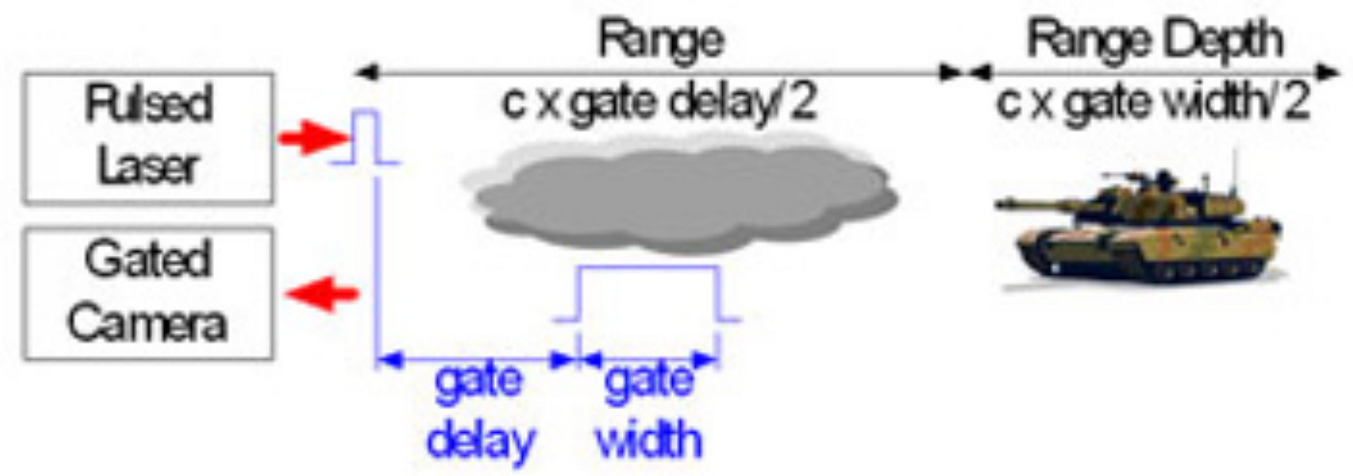
Very
bright
light



(a) VLS-128 (b) HDL-64S2 (c) HDL-32E

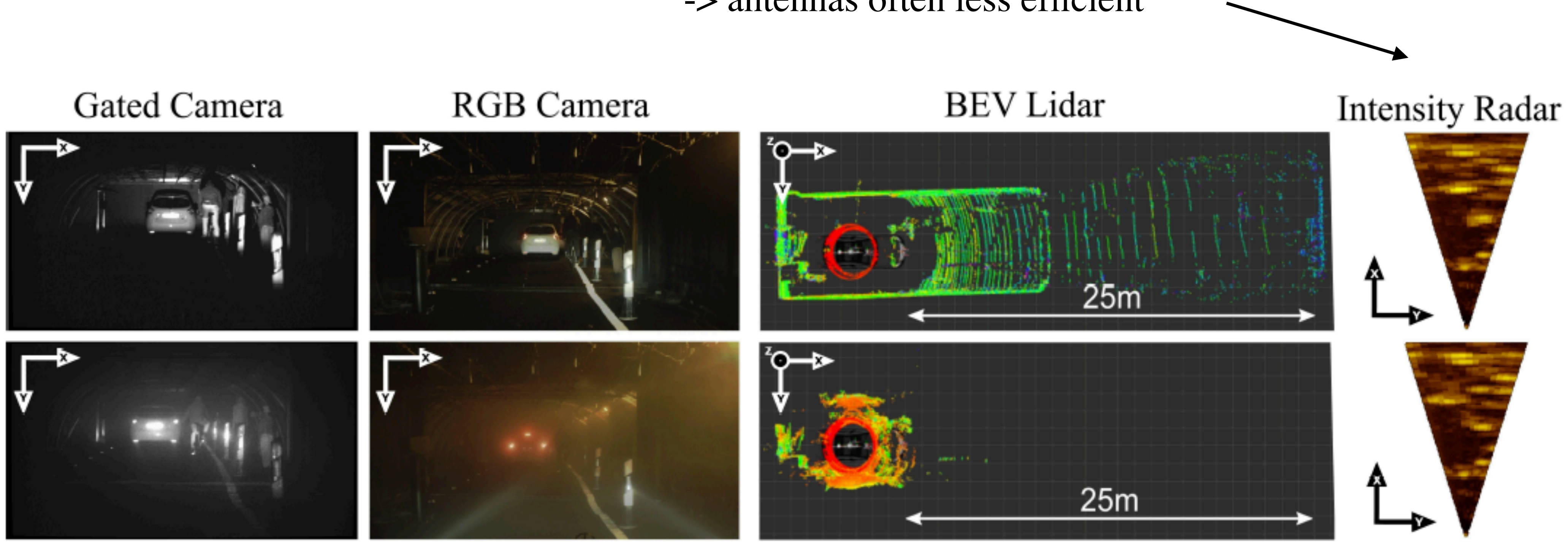
Carballo, 20

## Gated cameras



From sensors unlimited website

Very like lidar, just at much longer wavelengths
-> Bigger antennas for transmit/receive
-> reduced spatial resolution
-> antennas often less efficient



Bijelic et al 20

# Sensors, Localization and Control

- -1: Intro - who I am, UIUC, what we're going to do
- 0: Big tools of vision
  - geometry ; discrimination ; regression
  - Simple visual control
    - Google streetview; various learning paradigms;
    - why imitation learning is hard; Dagger
- I: Sensor review
  - Cameras; LIDAR; timed cameras; GPS
- II: Localization
  - simple 2D, 3D registration by matching; features for matching
  - matching in paired cameras; F matrix; odometry; SLAM

## Point set registration

## Issues

- Where am I?
  - Simplest: register observations and motion to a map
    - correspondence and robustness
- Build a map
  - Register observations to one another
    - global consistency
- Incorporating motion models
  - Registration should benefit from knowledge of motion
    - Filtering

## Simplest case

- Registration with known correspondence
  - No motion model
  - 3D observations of known beacons at known 3D locations
    - beacons y_i; observations x_i
    - (for generality) weights w_i
- Problem:
  - choose rotation R, translation t to minimize

$$C(R, \mathbf{t}) = \sum_i w_i \left(R\mathbf{x}_i + \mathbf{t} - \mathbf{y_i}\right)^T \left(R\mathbf{x}_i + \mathbf{t} - \mathbf{y_i}\right)$$

  - THIS CAN BE DONE IN CLOSED FORM!

## So far

- Given two sets of points
  - with known correspondences
  - weights
- We can find optimal rotation, translation to register
  - easily
  - in closed form

- We now know where we are
  - for (say) x_i 3D measurements, y_i beacons
- Missing:
  - what happens if we *don’t* have correspondences?
  - robustness

## ICP = Iterated closest points

- What if we *don’t* have correspondences?
- Idea:
  - Repeat until convergence:
    - each x corresponds to “closest” y
    - register
- Big simple idea, lots of variants
  - What is “closest”?
  - What if you have lots of points?

## Robustness is a serious problem

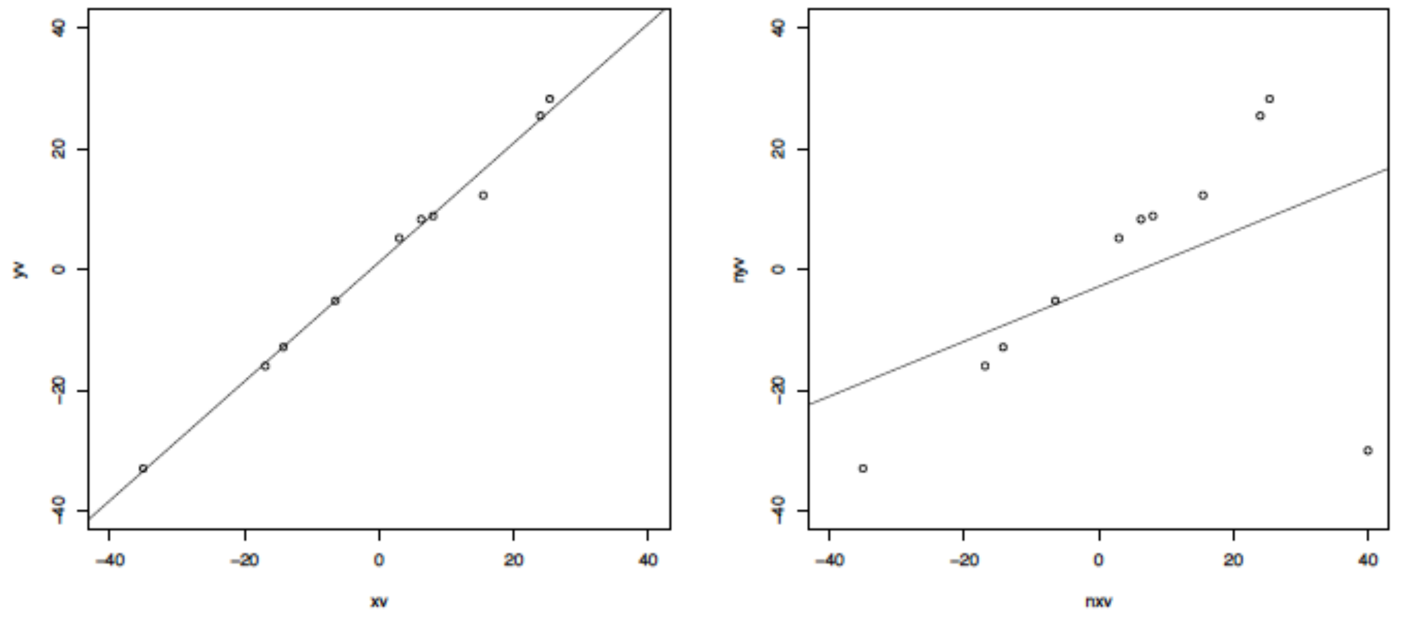


FIGURE 10.6: *On the* **left**, *a synthetic dataset with one independent and one explanatory variable, with the regression line plotted. Notice the line is close to the data points, and its predictions seem likely to be reliable. On the* **right**, *the result of adding a single outlying datapoint to that dataset. The regression line has changed significantly, because the regression line tries to minimize the sum of squared vertical distances between the data points and the line. Because the outlying datapoint is far from the line, the squared vertical distance to this point is enormous. The line has moved to reduce this distance, at the cost of making the other points further from the line.*

## Robustness is a serious problem

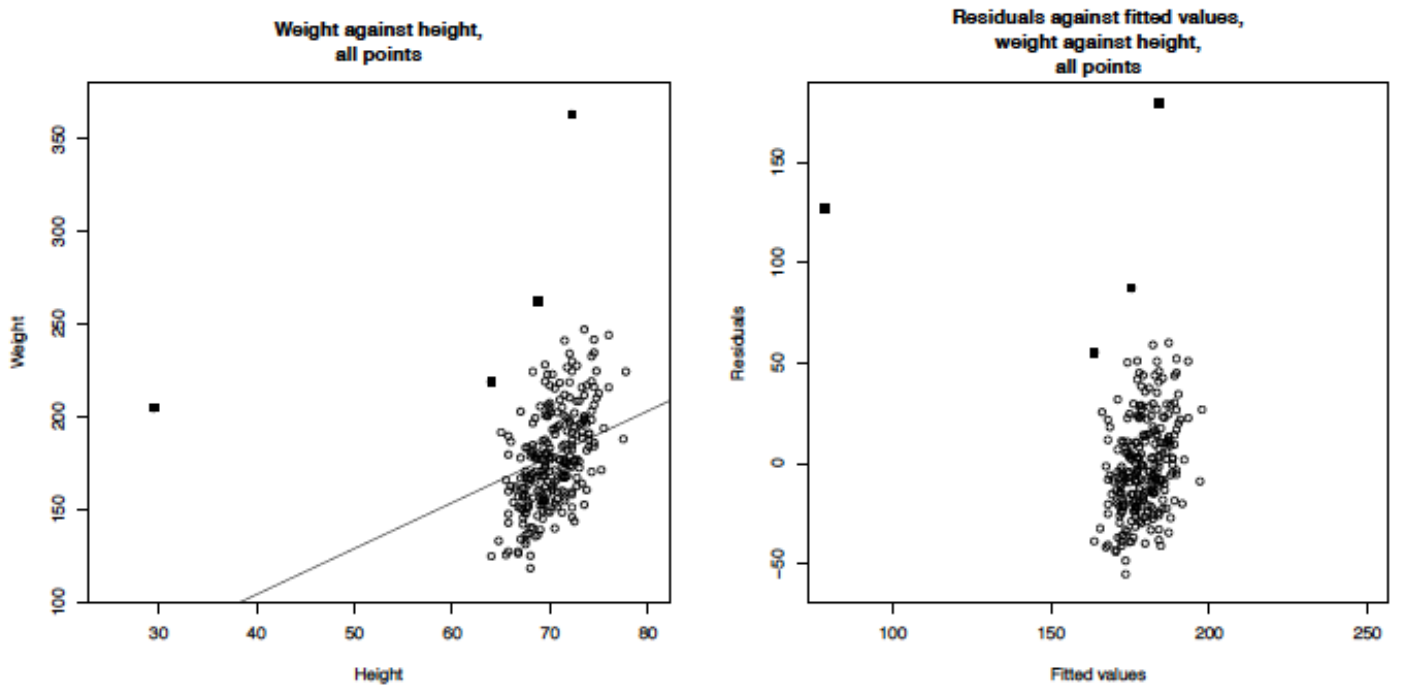


FIGURE 10.7: *On the* **left**, *weight regressed against height for the bodyfat dataset. The line doesn't describe the data particularly well, because it has been strongly affected by a few data points (filled-in markers). On the* **right**, *a scatter plot of the residual against the value predicted by the regression. This doesn't look like noise, which is a sign of trouble.*

## Key issue:

- Squaring a large number produces a huge number
- A few wildly mismatch points can throw off R, t
- Fixes:
  - remove matches with “large” distances
    - actually, quite good
    - but what happens if new such pairs emerge?
  - apply an M-estimator
    - deals with new pairs

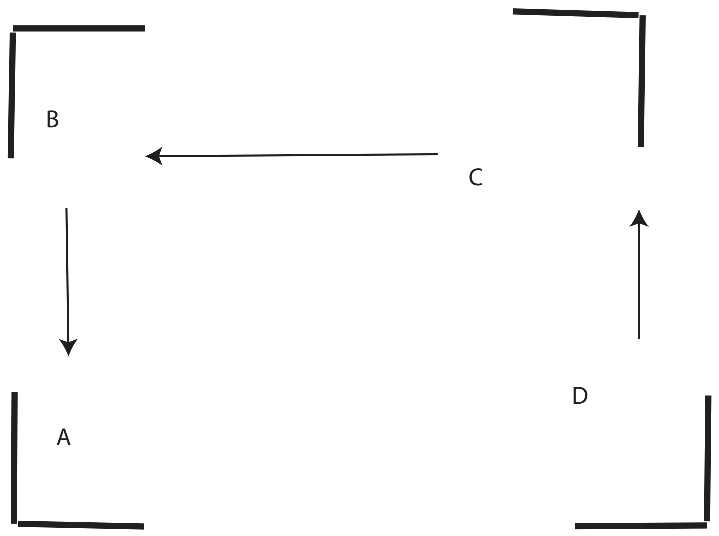
## Localization

- We can now robustly register a point set to a point set
    - This is localization
        - Look at LIDAR
        - register to map
        - I know where I am
- Q:
    - how to make the map?
        - registration + bundle adjustment
    - how to incorporate movement estimates and model uncertainty?
        - important; huge literature (filtering); but out of scope

# Bundle adjustment

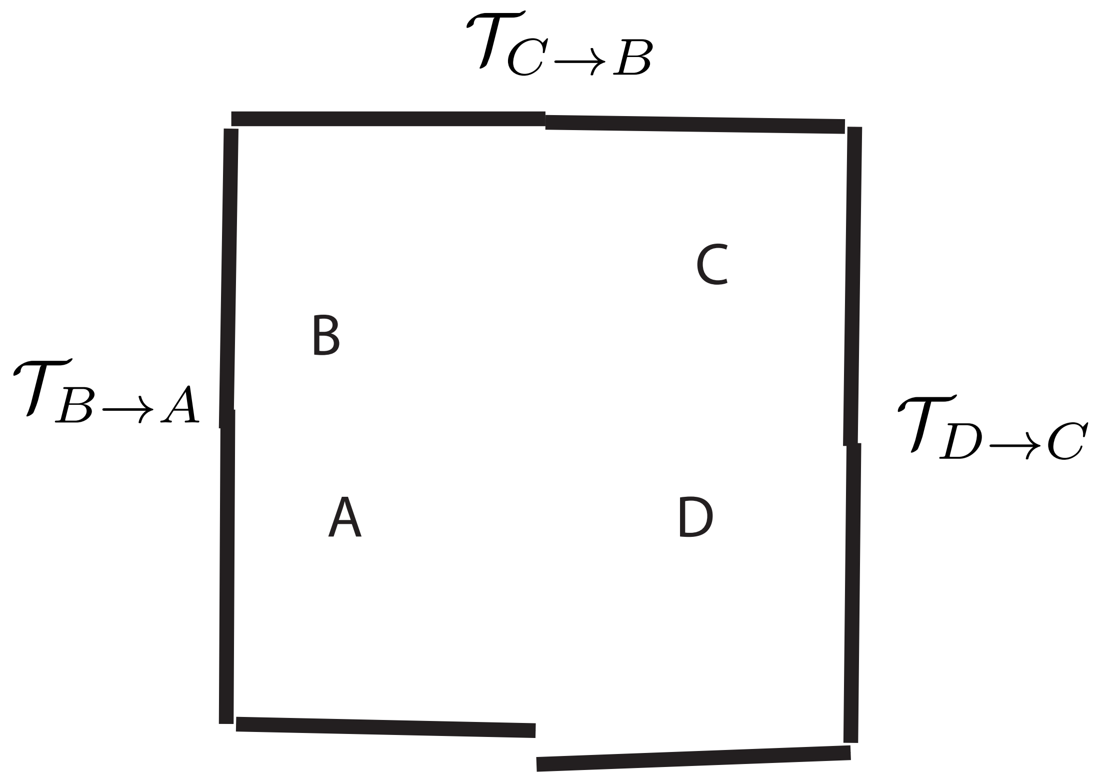
- The problem:
  - Register B to A, C to B, D to C
    - -> then you know D->A, but it isn't very good…

Q: Why not C->A?



## Bundle adjustment

- Loop closure problems
  - Why is this happening?



$$\mathcal{T}_{D\to A} = \mathcal{T}_{B\to A} \circ \mathcal{T}_{C\to B} \circ \mathcal{T}_{D\to C}$$

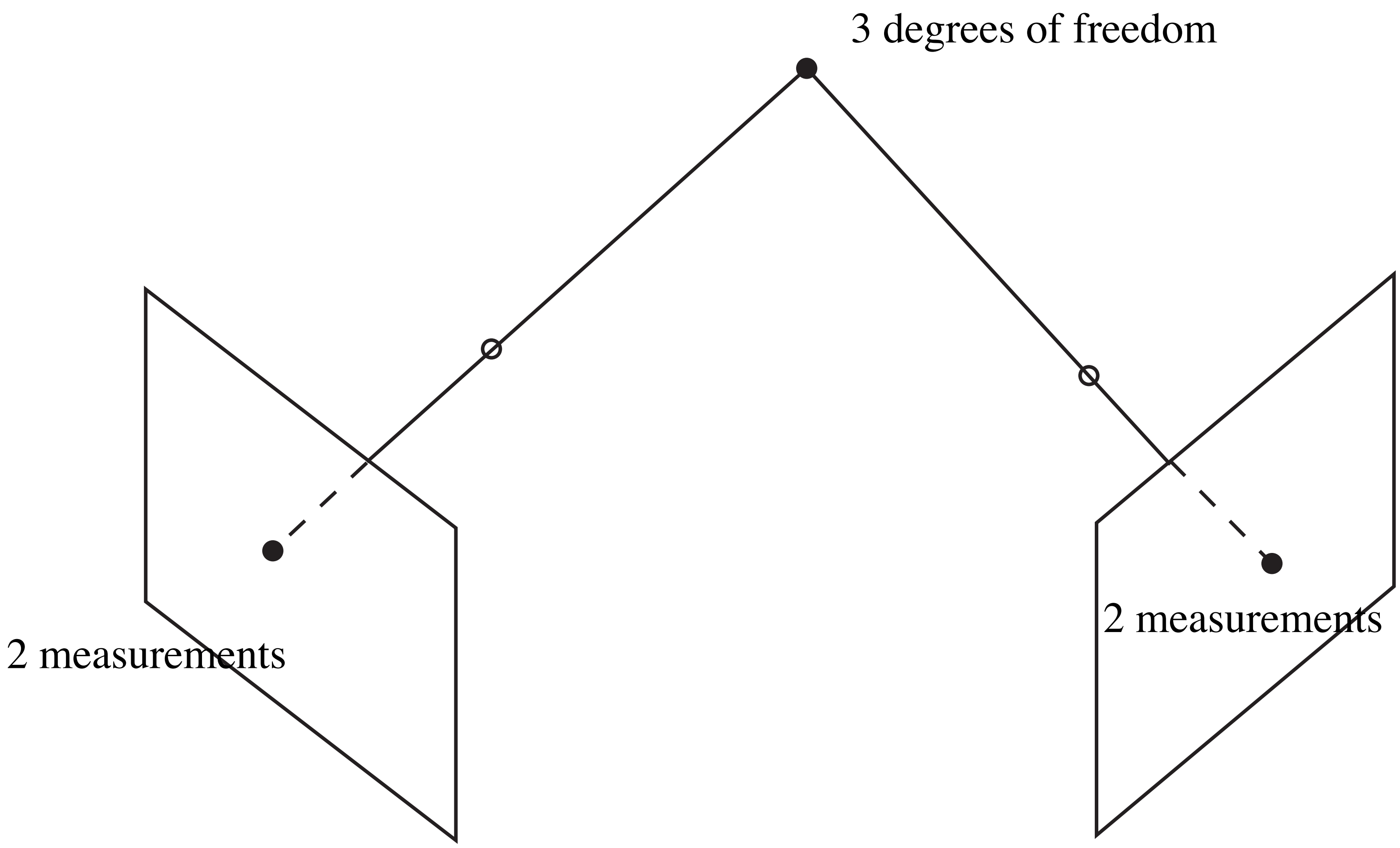
But this isn't the one that minimizes the D, A overlap errors!

## Strategies

- (Pretty much always)
  - Fix one set of points
  - Register in sequence
    - usually defined by (say) time
  - Now fix the resulting estimates
    - these are a start point

## Pairs of cameras

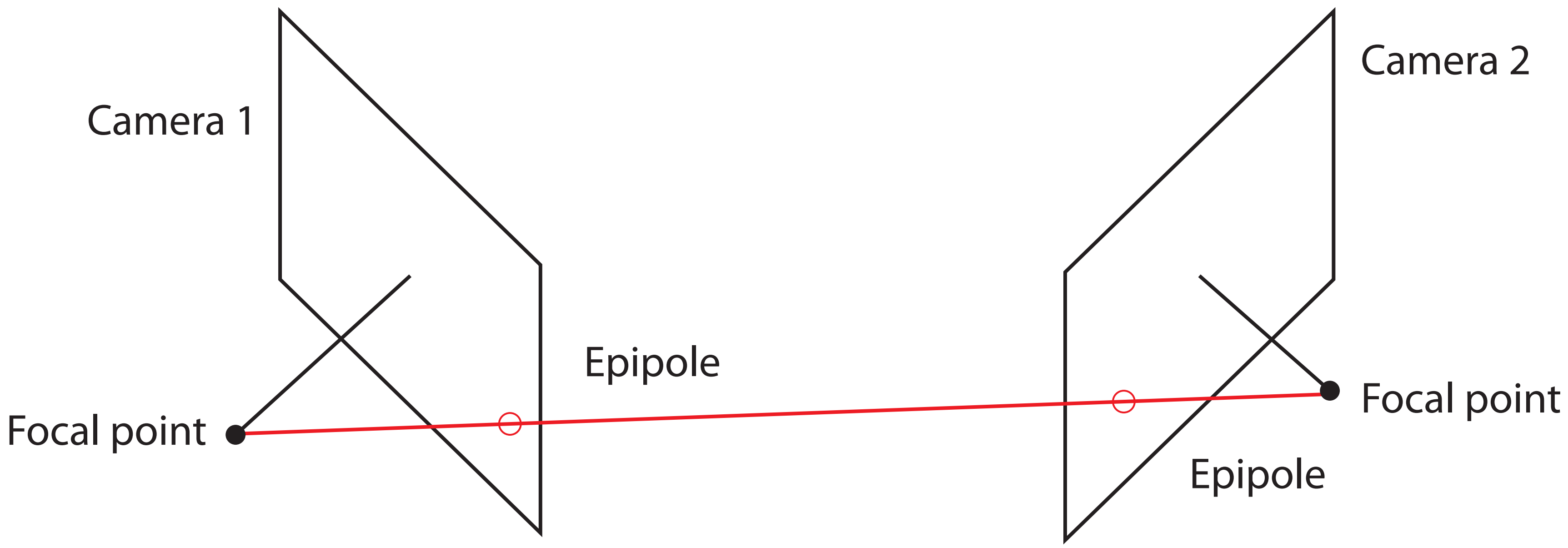
# What happens in two views



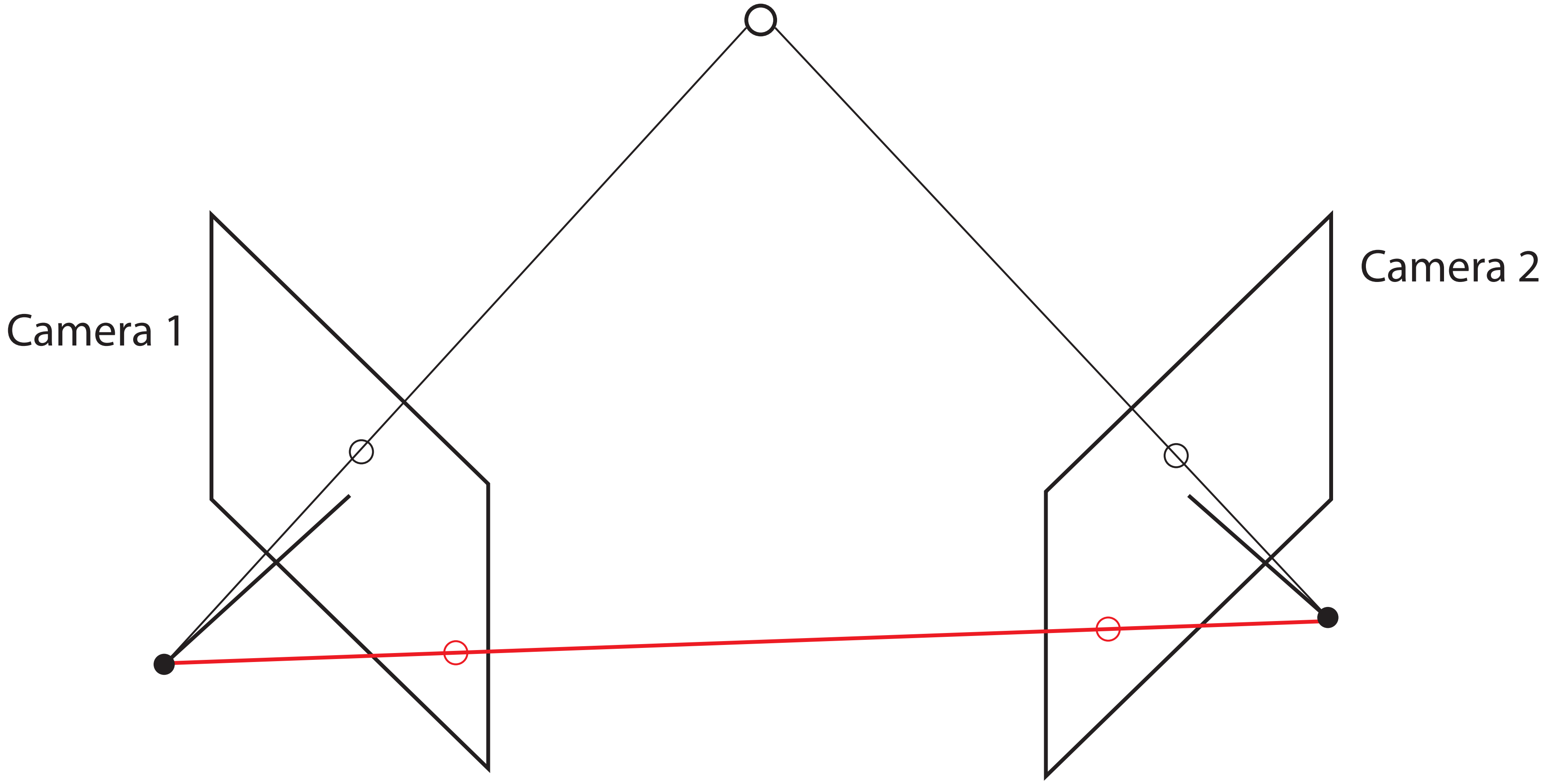
## All of Camera Geometry

- From the picture
  - two views of a point give four measurements of three DOF
  - this means
    - correspondence is constrained
    - if we have enough points and enough pix we can recover
      - points
      - cameras

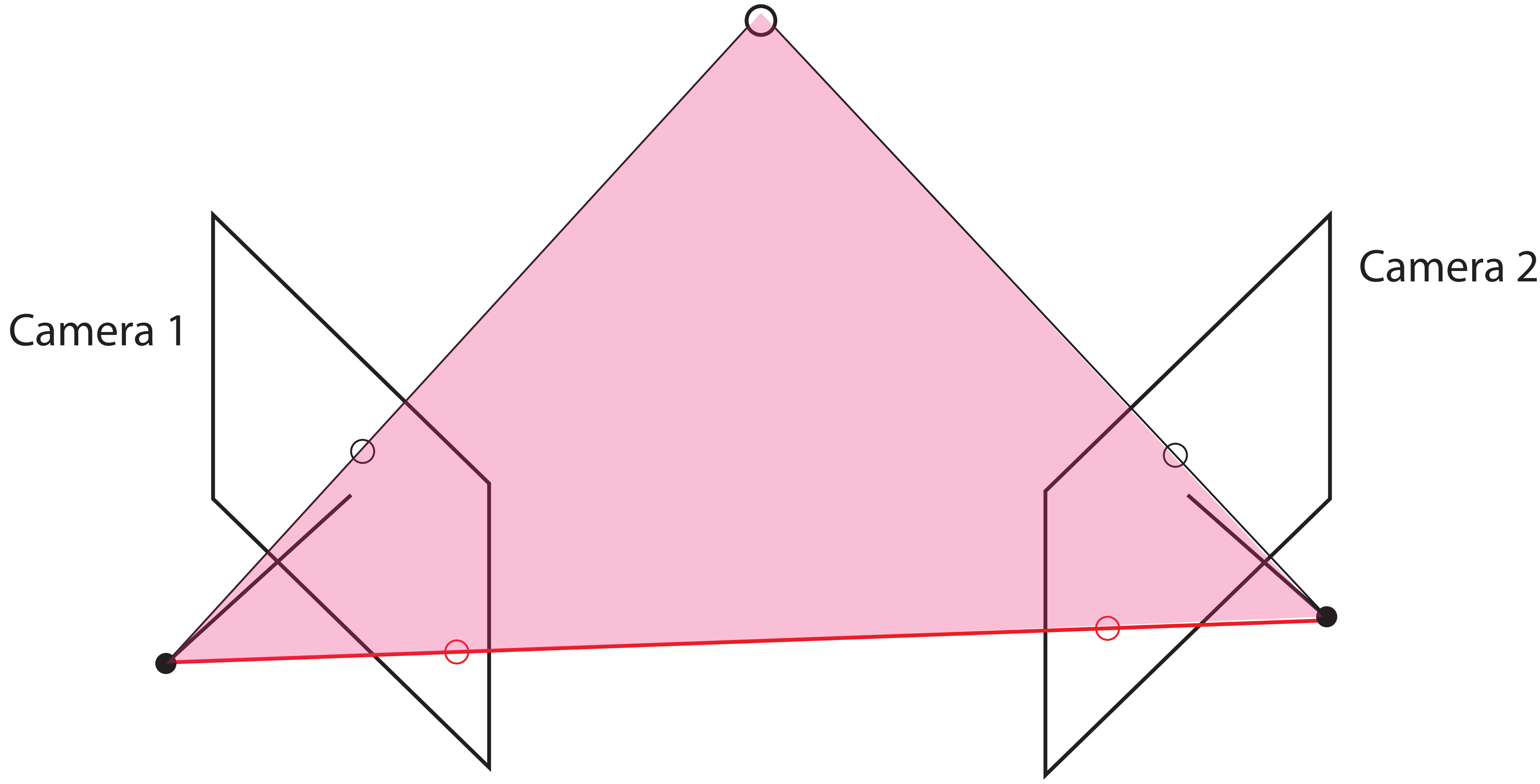
# The Epipoles



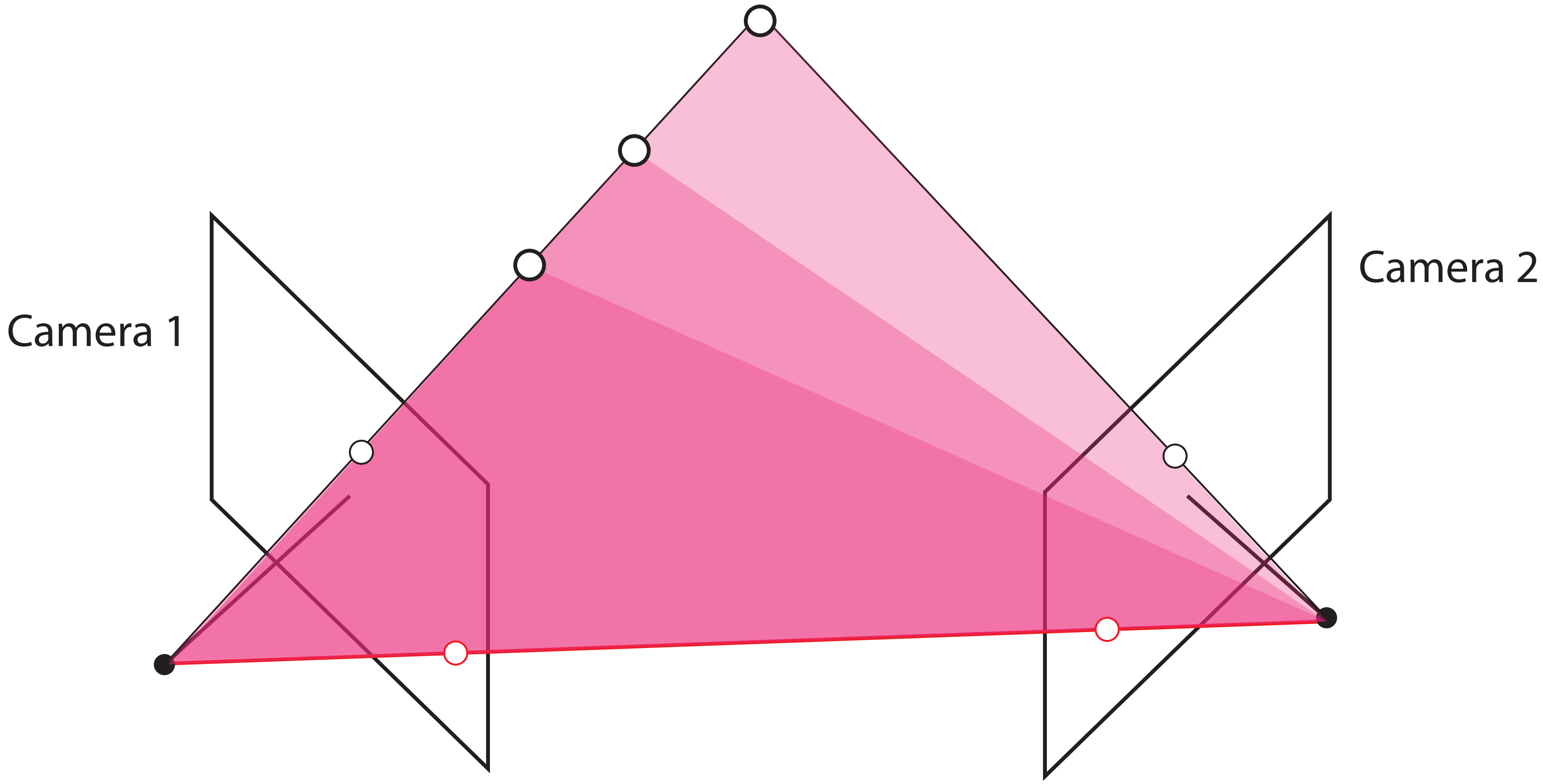
## Constraints on correspondence



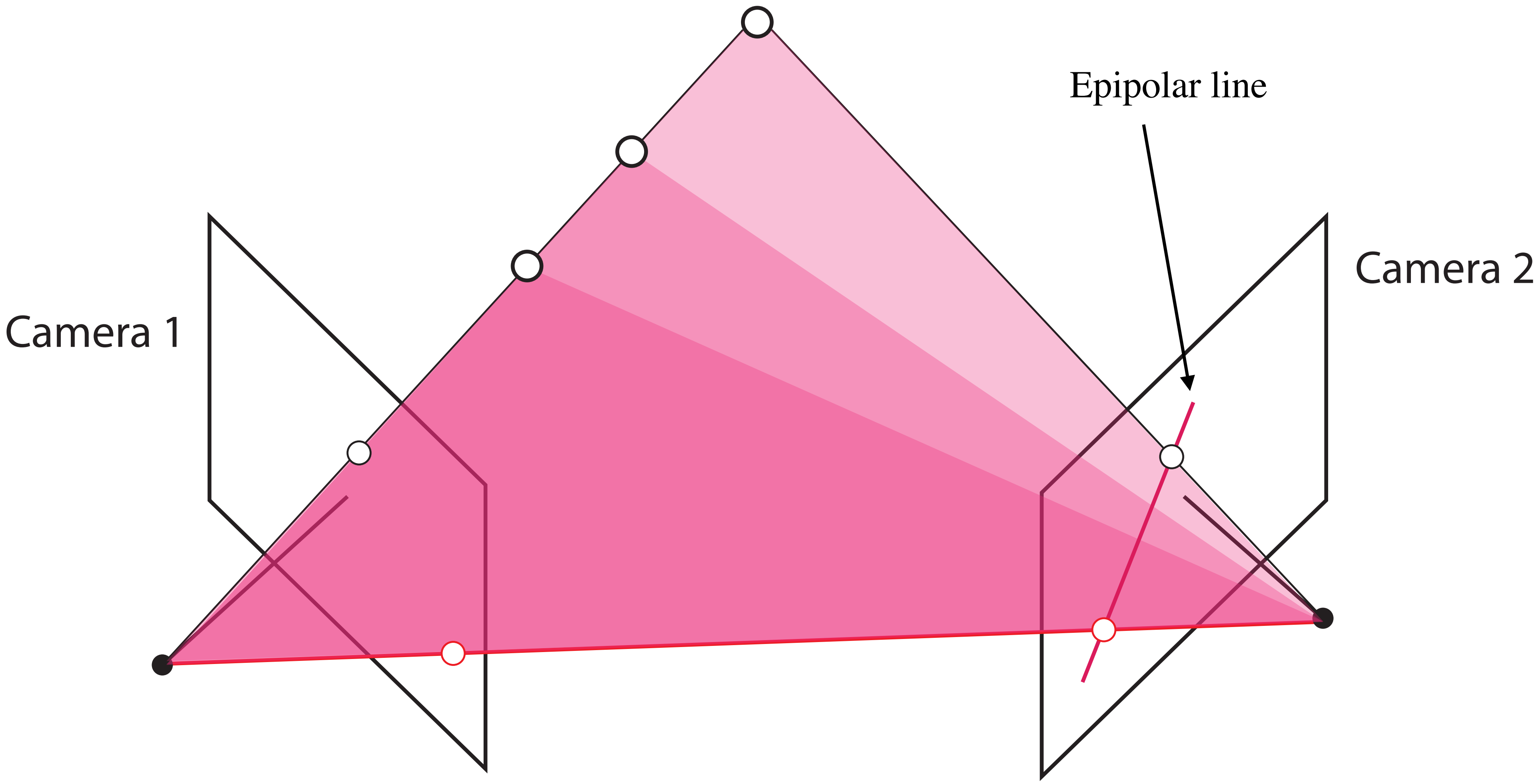
# Constraints on CSP -II



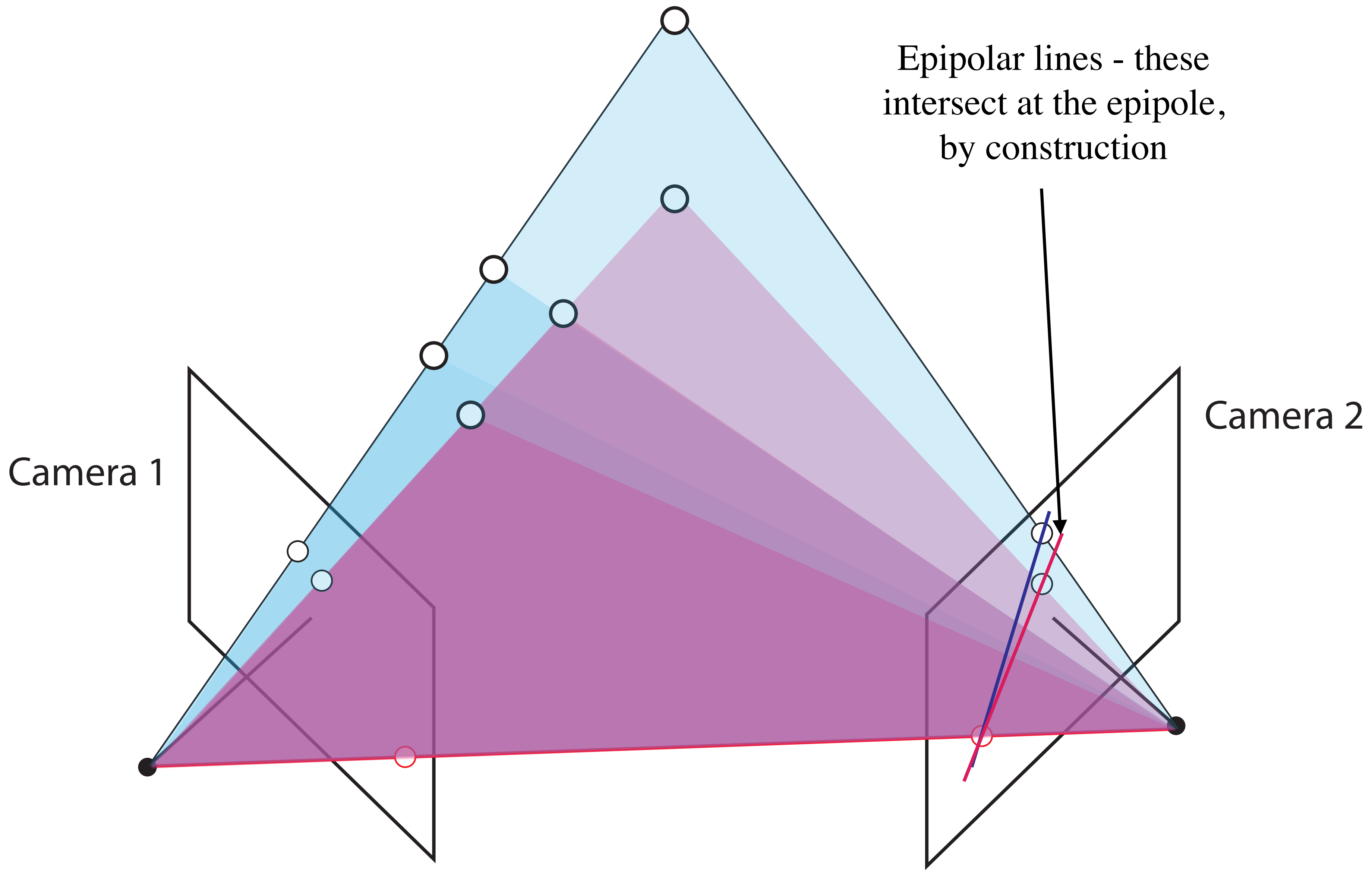
# Constraints on CSP - III



# Constraints on CSP - III

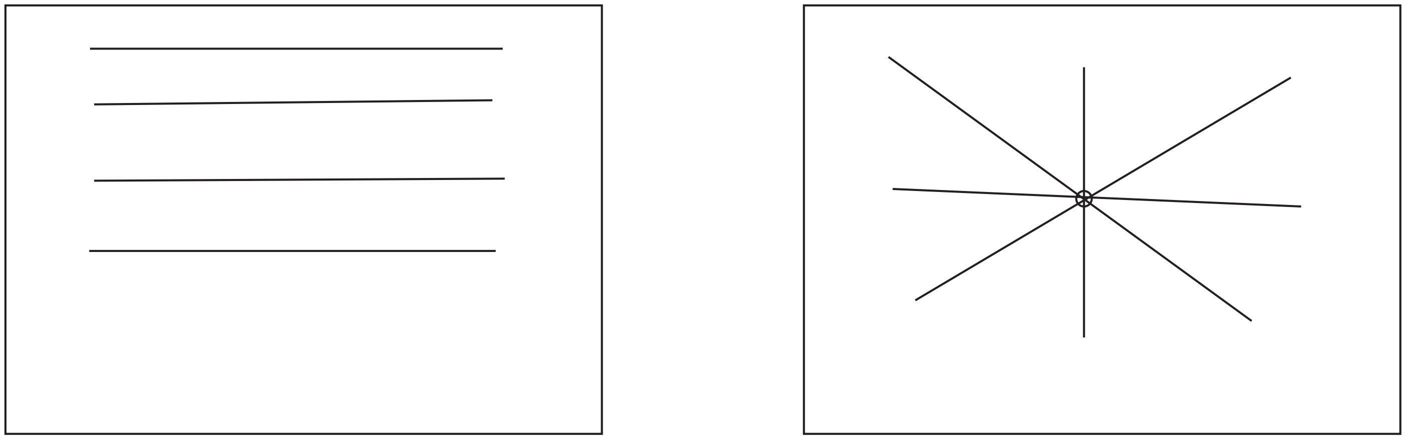


# Constraints on CSP - IV


Epipolar lines - these intersect at the epipole, by construction
Camera 1
Camera 2

## Epipoles (resp. epipolar lines)

- Informative



Epipole and epipolar lines in camera 1 - where is camera 2?

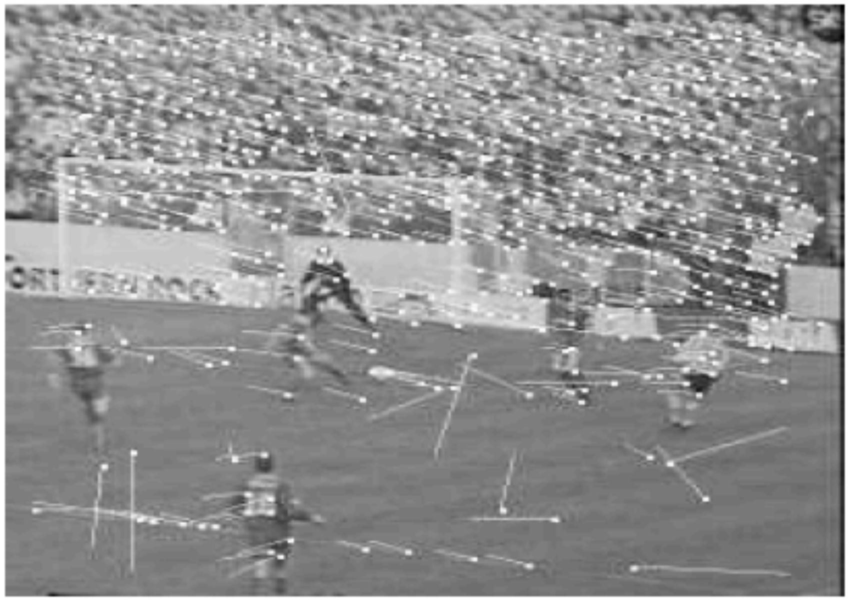
## The Fundamental Matrix

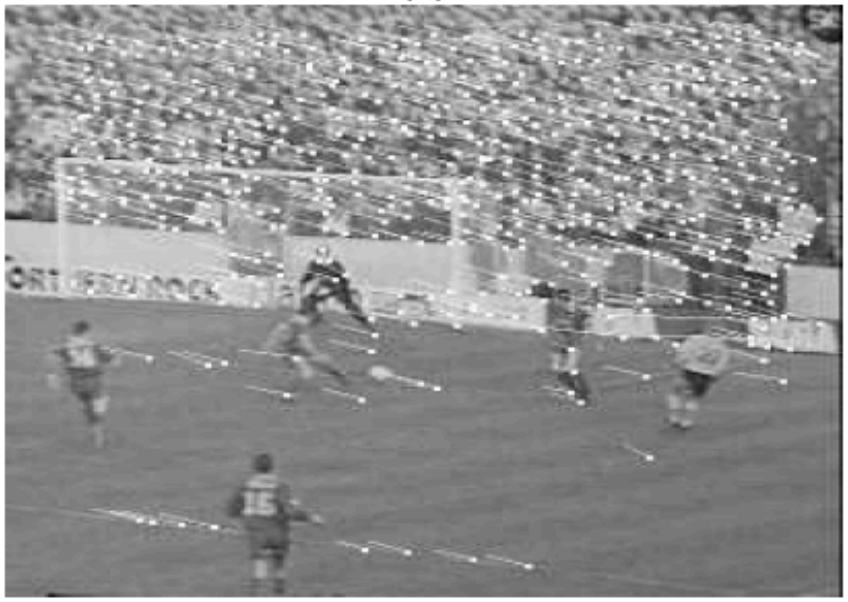
$$\mathbf{p_1}^T \overset{\downarrow}{\mathcal{F}} \mathbf{p_2} = 0$$

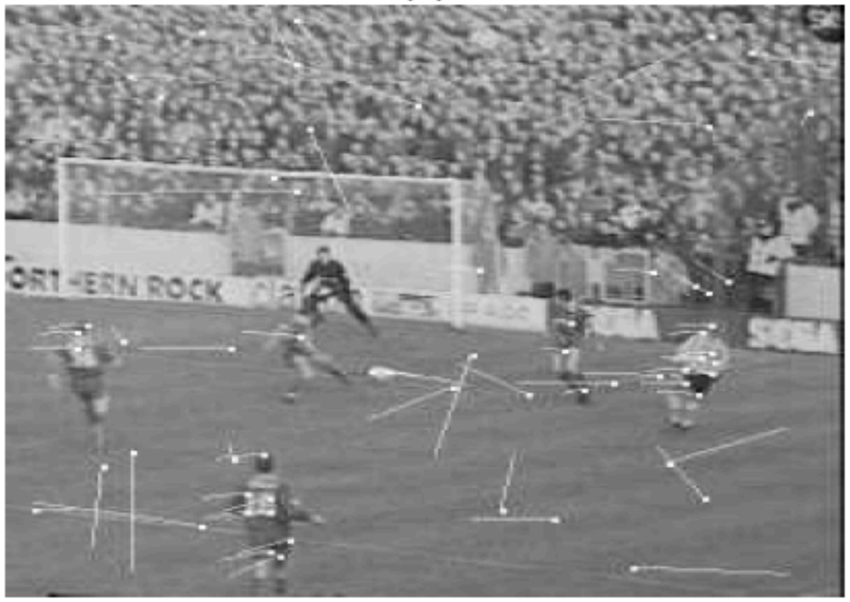
- Easy closed form expression exists
  - in terms of rot, trans between cameras, intrinsics
  - following slides

- Can be fit a pair of images using feature correspondences
  - 8 point algorithm
  - robustness is an important issue
    - understood - standard procedures

(a)


(b)


(c)


(d)

Torr, 96

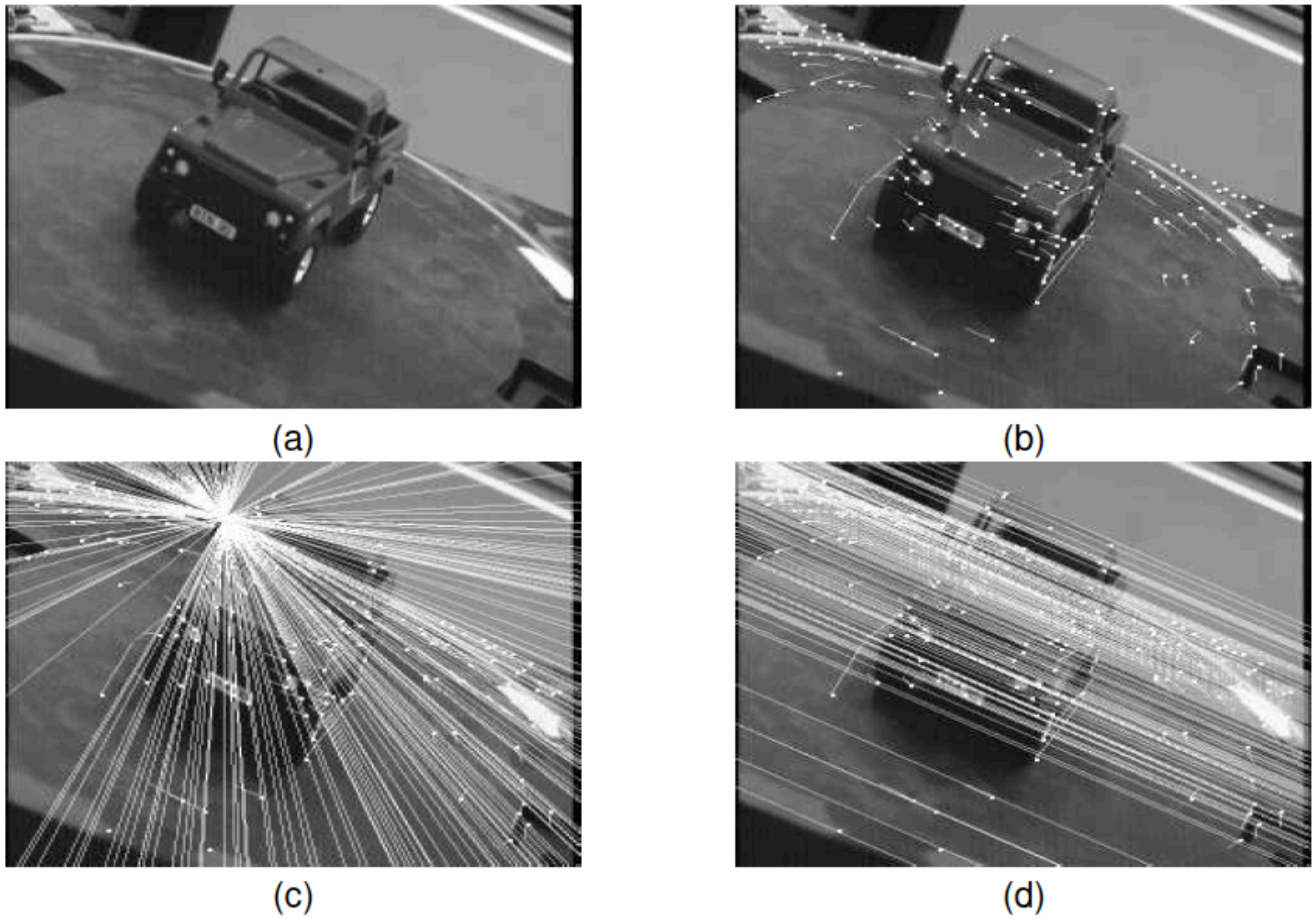


*Fig. 18.* In (a) (b) two consecutive images of a buggy rotating on a turntable. (b) has 167 matches superimposed on the second image. (c) (d) show two epipolar geometries generated by two distinct fundamental matrices, 139 correspondences are consistent with the fundamental matrix in (a), 131 are consistent with the fundamental matrix in (b) yet the two epipolar geometries obviously differ.

Torr, 96

## What to match

# Image neighborhoods

- We want to find patches that are “worth representing”
  - to match from image to image
  - to represent textures
  - to represent objects
- Requirements
  - Covariant to translation, rotation, scale
    - i.e. if the image is translated, rotated, scaled, so are the neighborhoods
    - important to ensure that the representation of the patch is stable
  - Localizable in translation, rotation, scale
    - we can estimate the position, orientation and size of the patch
    - and get the answer about right
- Methods exist for richer sets of requirements

## Learning to detect and describe keypoints

- You should be able to learn all this
  - keypoints are stable under rotation, translation, scale (homographies)
  - descriptions are stable under rotation, translation, scale (homographies)

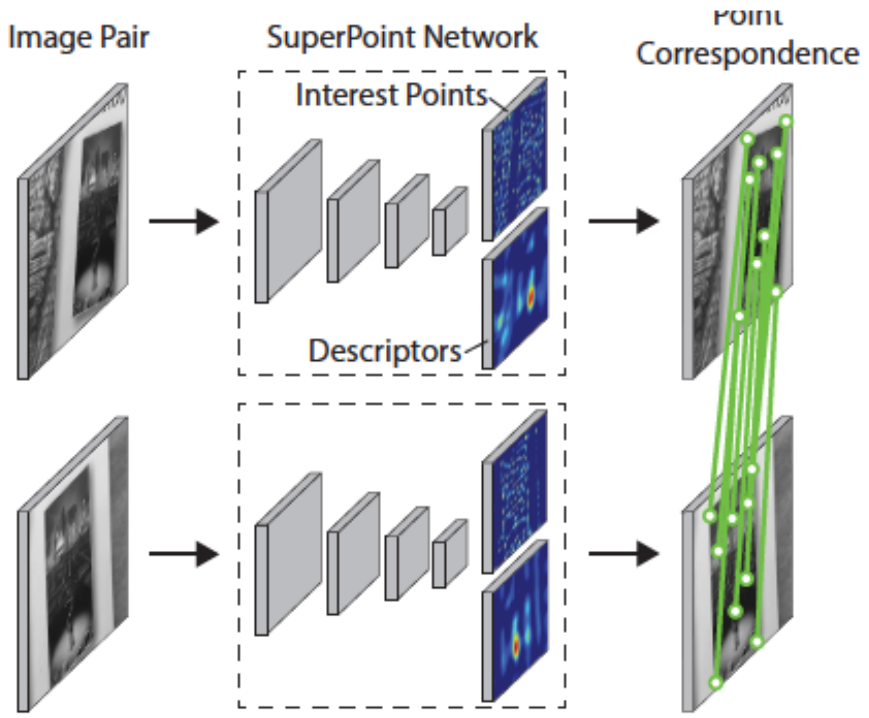


Figure 1. **SuperPoint for Geometric Correspondences.** We present a fully-convolutional neural network that computes SIFT-like 2D interest point locations and descriptors in a single forward pass and runs at 70 FPS on 480 × 640 images with a Titan X GPU.

DeTone et al, 18

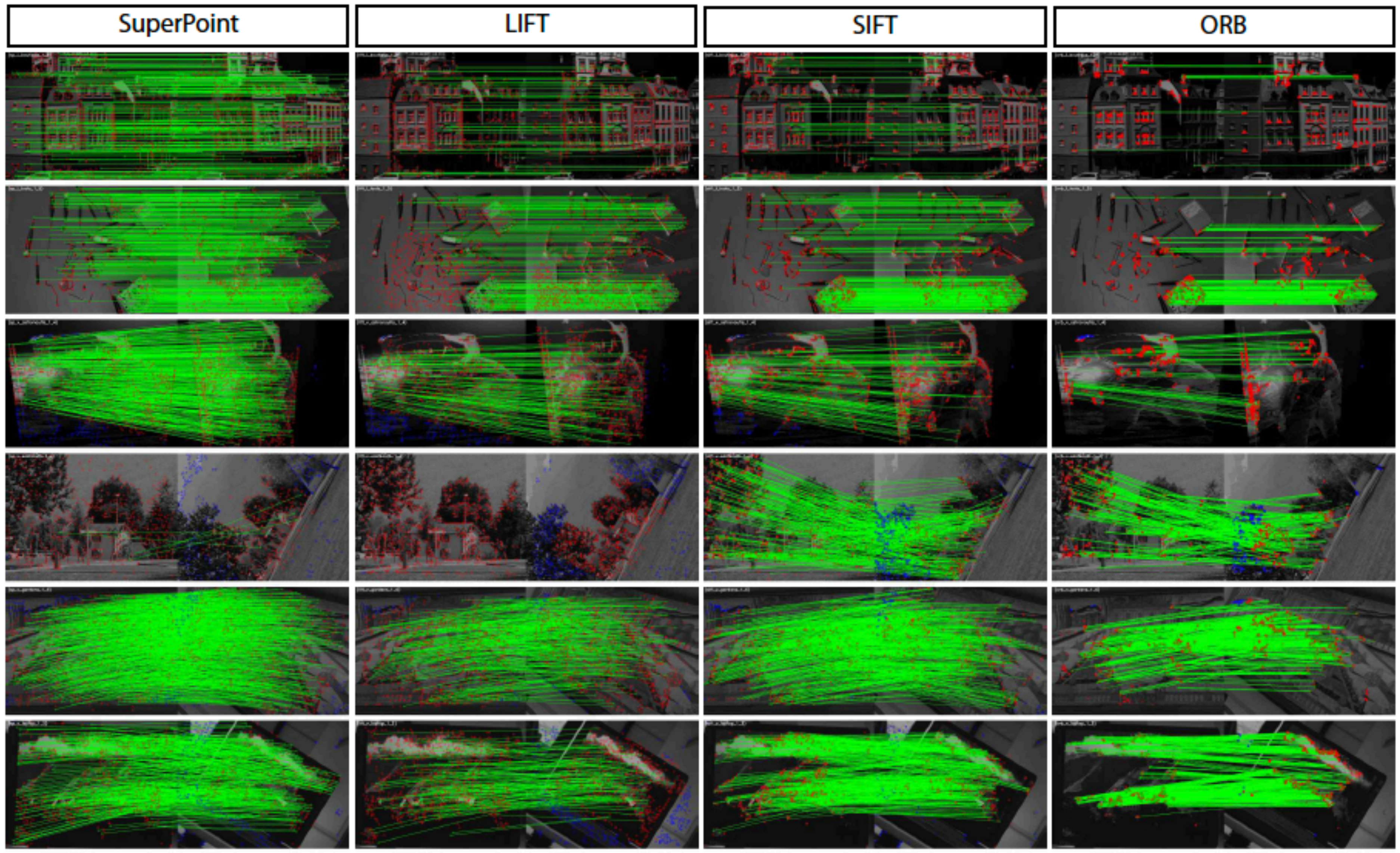


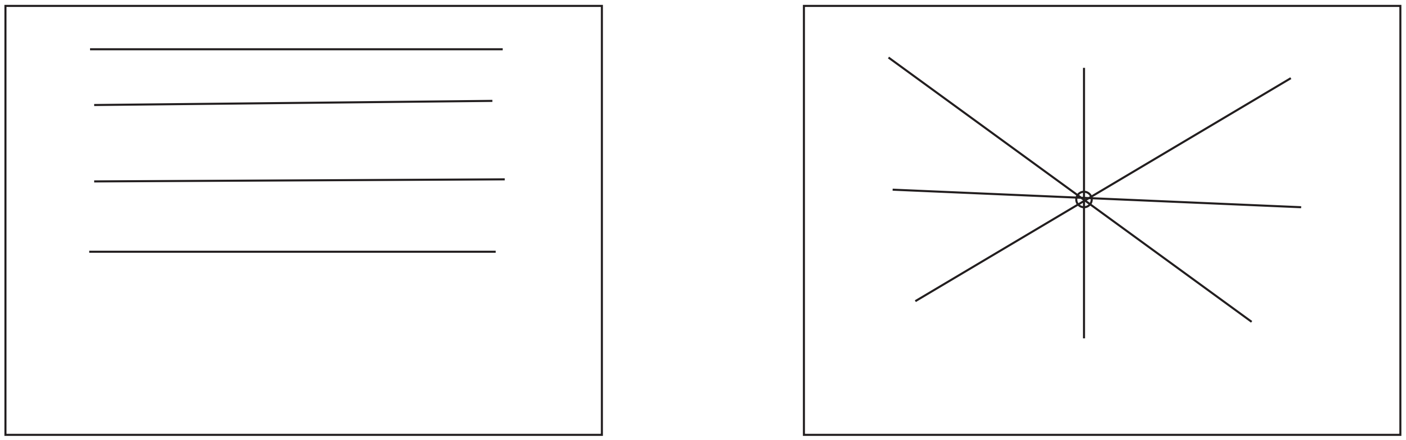
Figure 8. **Qualitative Results on HPatches**. The green lines show correct correspondences. SuperPoint tends to produce more dense and correct matches compared to LIFT, SIFT and ORB. While ORB has the highest average repeatability, the detections cluster together and generally do not result in more matches or more accurate homography estimates (see 4). Row 4: Failure case of SuperPoint and LIFT due to extreme in-plane rotation not seen in the training examples. See Appendix D for additional homography estimation example pairs.

DeTone et al, 18

## Visual Odometry with Single Camera

# Epipoles (resp. epipolar lines)

- Informative



Epipole and epipolar lines in camera 1 - where is camera 2?

# Odometry from two camera geometry

- Idea:
  - use calibrated camera
  - move; track some points
    - in reading slides
  - compute essential matrix (calibrated fundamental matrix) to get
    - rotation
    - translation up to scale
- Options:
  - fix scale later
  - use (say) wheel measurements + Kalman filter to fix
  - use stereo

## RECALL: The Fundamental Matrix

$$\mathbf{p_1}^T \mathcal{F} \mathbf{p_2} = 0$$

- Can be fit a pair of images using feature correspondences
  - 8 point algorithm
  - robustness is an important issue
  - we'll do this

$$\mathcal{F} = k\mathcal{C}_L^{-T}\mathcal{R}\mathcal{S}\mathcal{C}_R^{-1}$$

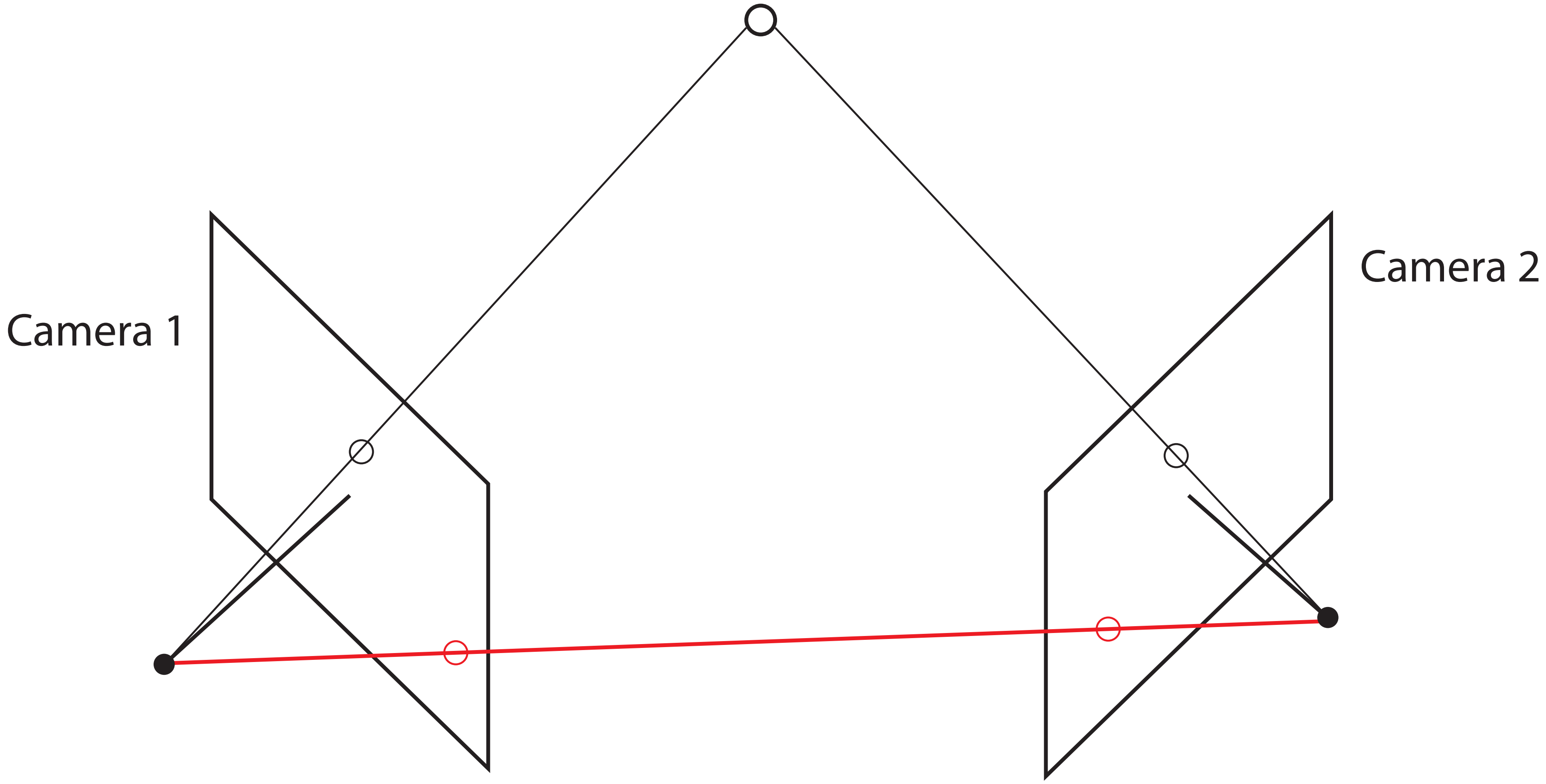
If we know these

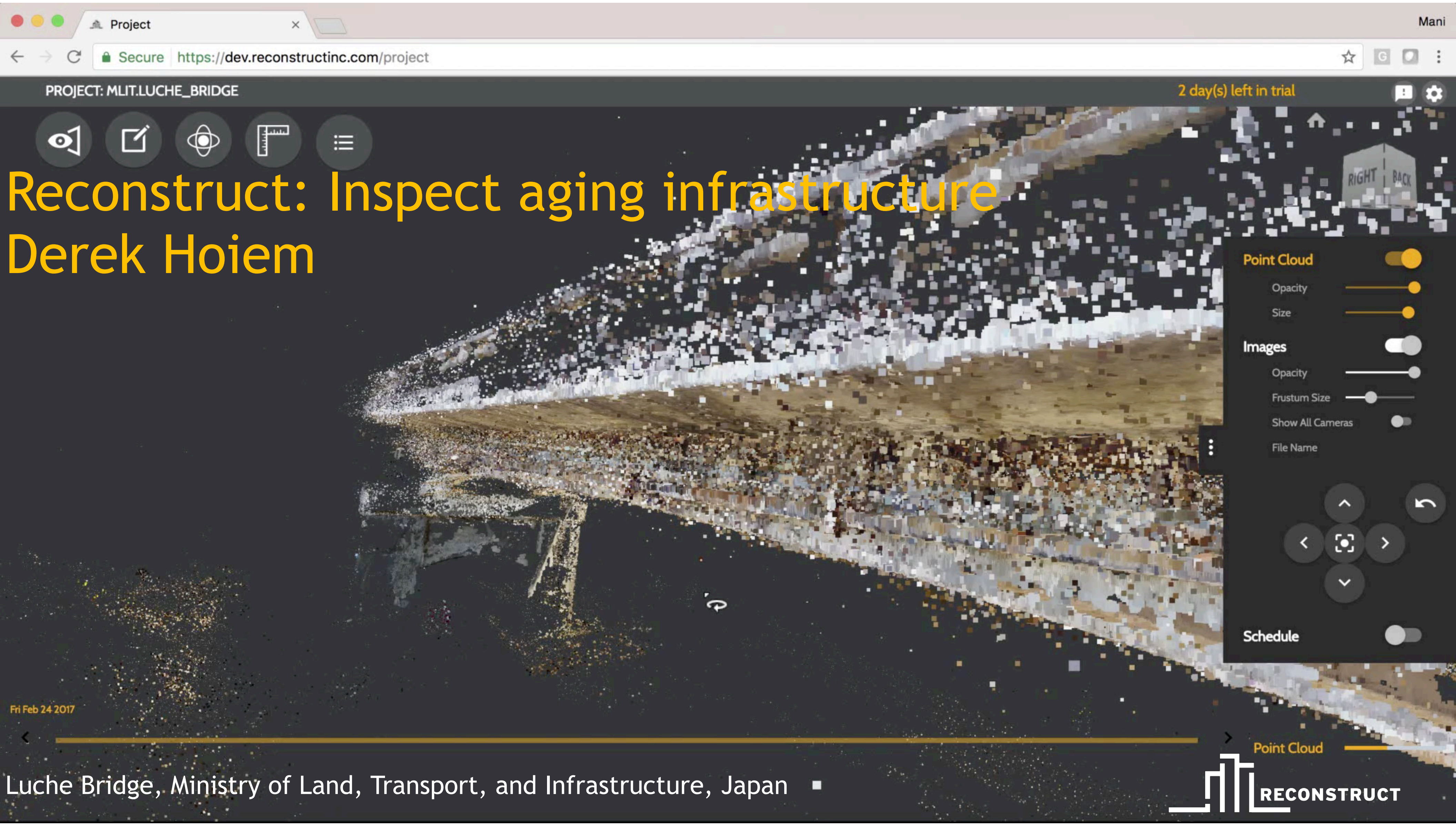
we can recover info about R, T from F

## VO example; Bloesch et al 2015

https://www.youtube.com/watch?v=ZMAISVy-6ao

# Correspondence yields 3D configuration

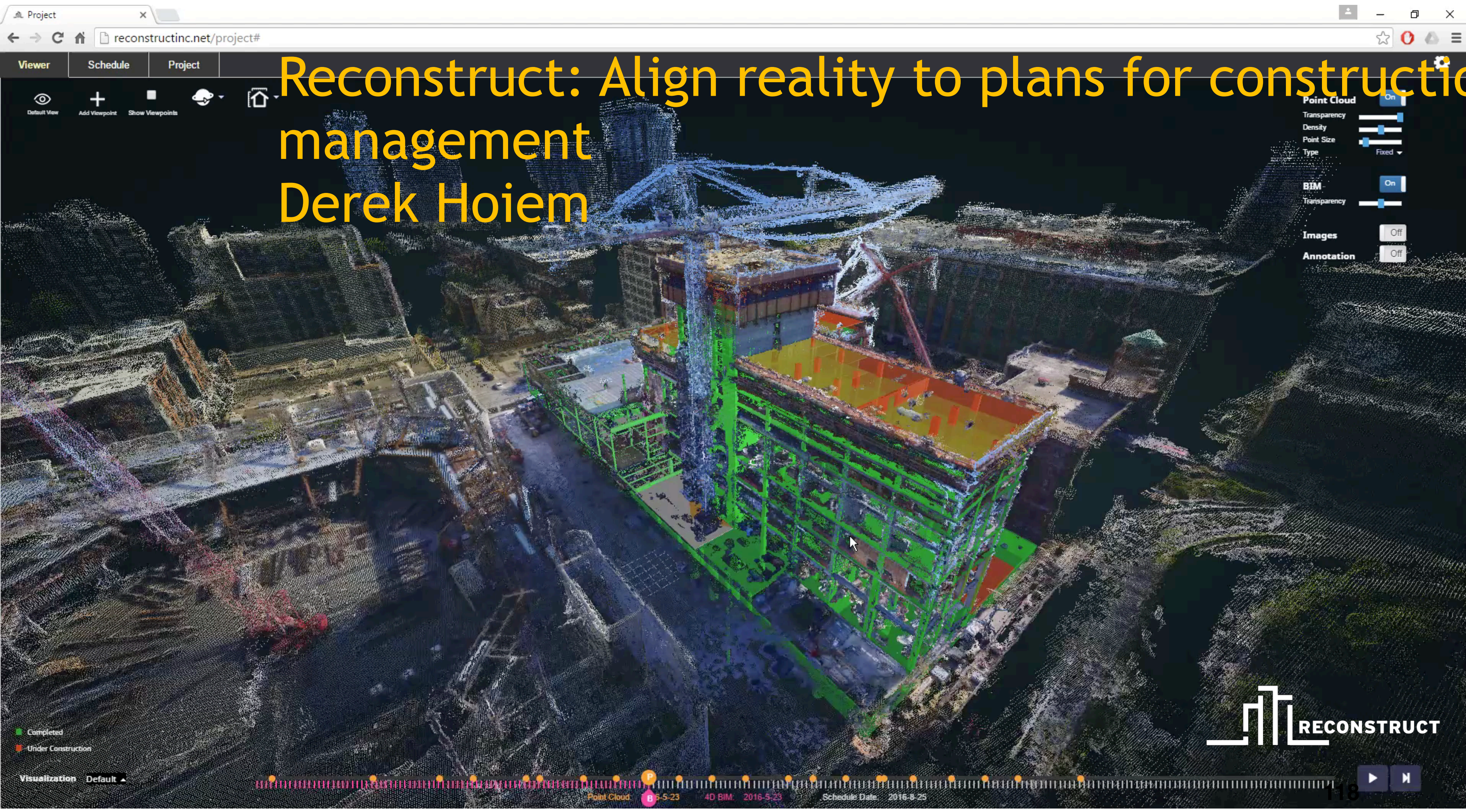




# Reconstruct: Inspect aging infrastructure
Derek Hoiem

Luche Bridge, Ministry of Land, Transport, and Infrastructure, Japan

# Reconstruct: Align reality to plans for constructio management
Derek Hoiem



# Filtering

- We may have more than one cue to our movement
    - visual odometry
    - inertial sensor
    - acceleration constraints
    - control inputs
    - vehicle dynamics
- Filtering
    - Combine these cues to come up with best posterior estimate of
        - camera motion (odometry)
        - features (SLAM)
            - SLAM=simultaneous localization and mapping

## SLAM example from Scaramuzza

https://www.youtube.com/watch?v=XpuRpKHp_Bk