CSCI 4830 / 5722
Computer Vision

University of Colorado **Boulder**

CSCI 4830 / 5722
Computer Vision

Dr. Ioana Fleming

Spring 2019
Lecture 18

Reminders

Submissions:
- Homework 3: Sat 3/2 at 11 pm

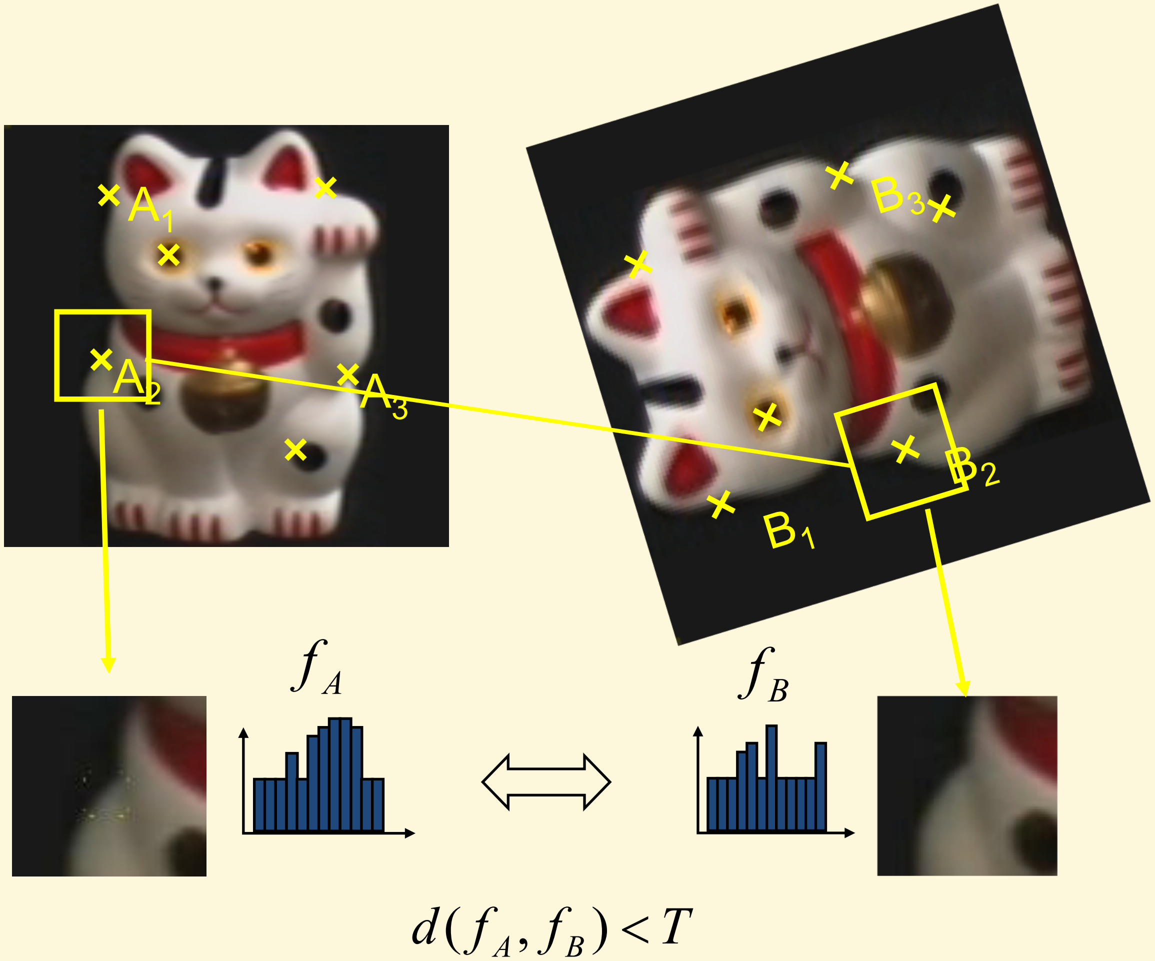
Readings:
- Szeliski:
  - chapter 4.1 (Feature detection – Points and patches)
  - chapter 11 (Stereo correspondence)
- P&F:
  - chapter 5 (Local features – corners, SIFT features)
  - chapter 7 (Stereopsis)

Today

- Feature matching
- Clustering with Hough
- Stereo vision - intro

K. Grauman, B. Leibe

Overview of Keypoint Matching



1. Find a set of distinctive key-points
2. Define a region around each keypoint
3. Extract and normalize the region content
4. Compute a local descriptor from the normalized region
5. Match local descriptors

Feature matching

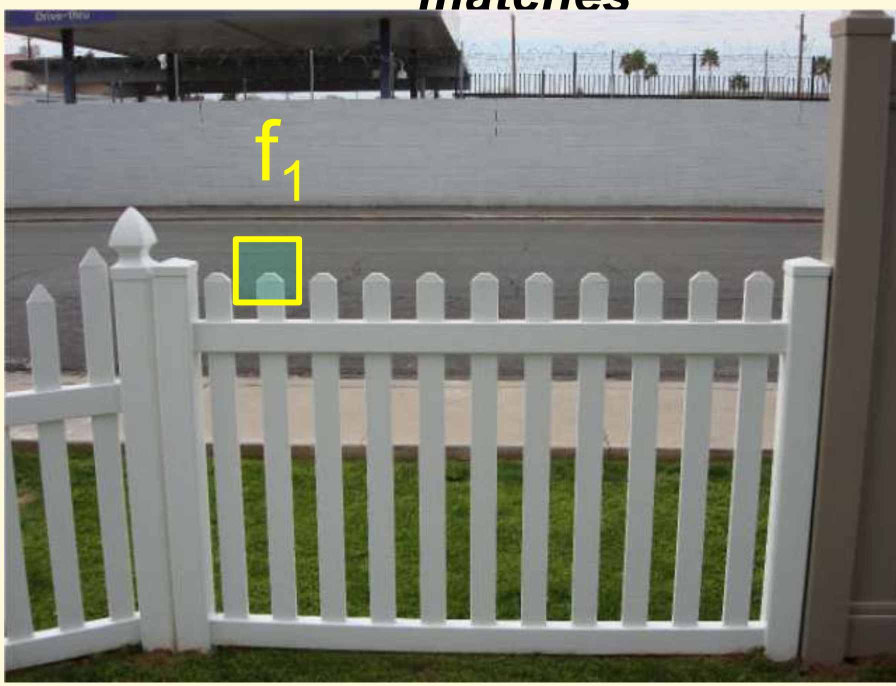
Given a feature in I_1, how to find the best match in I_2?

1. Define distance function that compares two descriptors
2. Test all the features in I_2, find the one with min distance

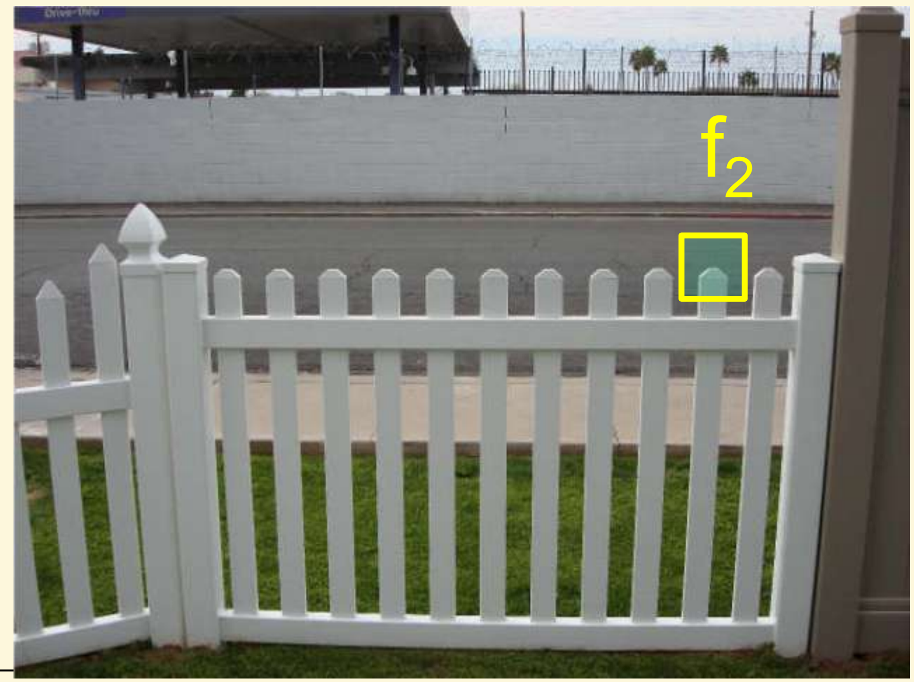
Feature distance

How to define the difference between two features f_1, f_2?

- Simple approach is SSD(f_1, f_2)
  - ***sum of square differences between entries of the two descriptors***
  - ***<u>can give good scores to very ambiguous (bad)</u> matches***



I_1

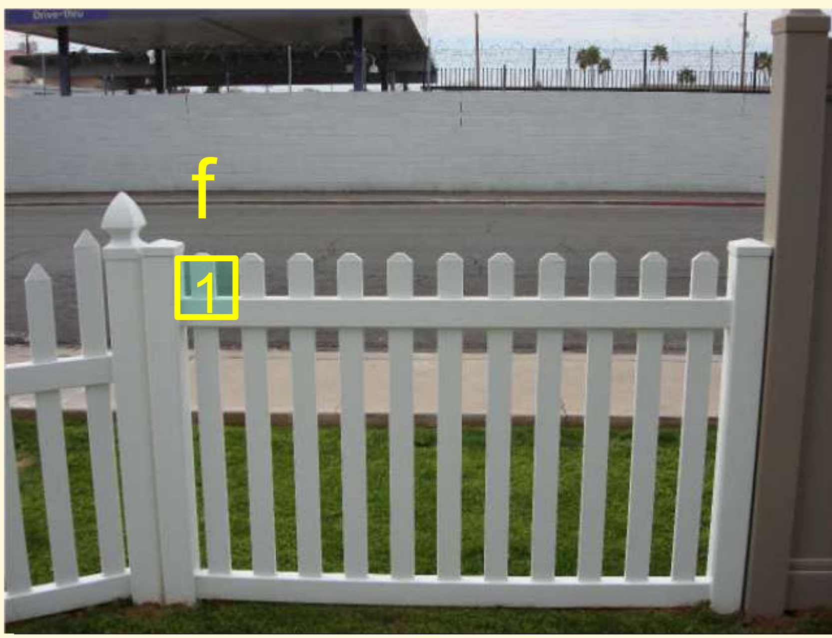


I_2

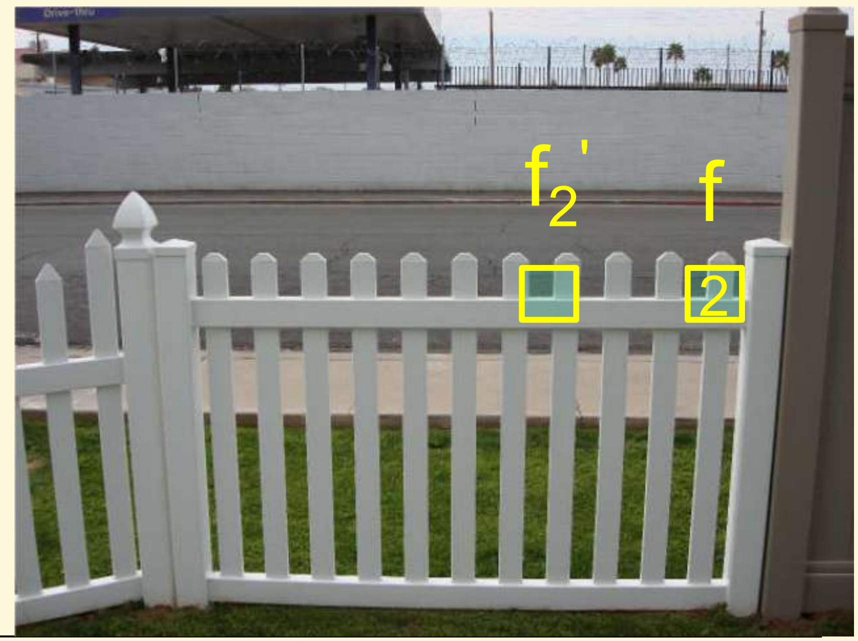
Feature distance

How to define the difference between two features f_1, f_2?

– Better approach: **ratio distance** = $SSD(f_1, f_2) / SSD(f_1, f_2')$
  - ***f_2 is best SSD match to f_1 in I_2***
  - ***f_2' is 2nd best SSD match to f_1 in I_2***
  - ***gives small values for ambiguous matches***

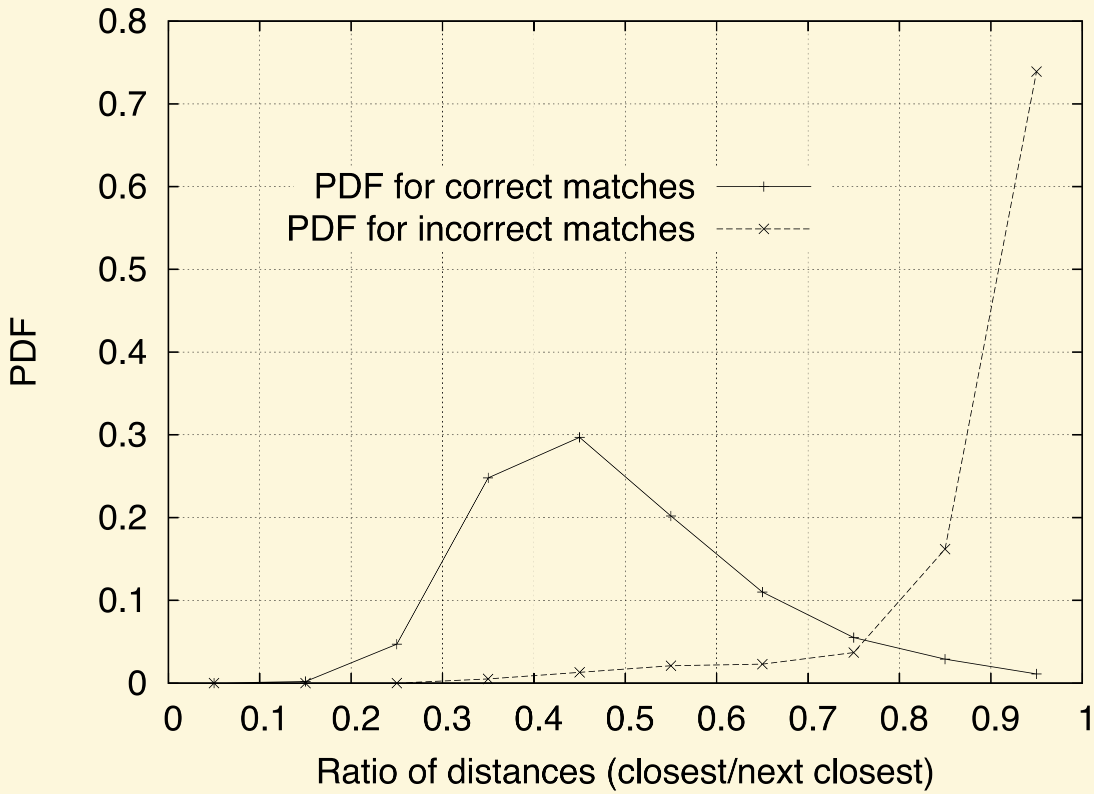


I_1



I_2

PDF = probability density function



Matches for which the nearest neighbor was a correct match have a PDF that is centered at a much lower ratio than that for incorrect matches. For our object recognition implementation, we reject all matches in which the distance ratio is greater than 0.8, which eliminates 90% of the false matches while discarding less than 5% of the correct matches.

Feature matching example

 51 matches

Confusion Matrix

	True matches	True non-matches		
Predicted matches	TP = 18	FP = 4	P' = 22	PPV = 0.82
Predicted non-matches	FN = 2	TN = 76	N' = 78	
	P = 20	N = 80	Total = 100	
	TPR = 0.90	FPR = 0.05		ACC = 0.94

Table 4.1 The number of matches correctly and incorrectly estimated by a feature matching algorithm, showing the number of true positives (TP), false positives (FP), false negatives (FN) and true negatives (TN). The columns sum up to the actual number of positives (P) and negatives (N), while the rows sum up to the predicted number of positives (P') and negatives (N'). The formulas for the true positive rate (TPR), the false positive rate (FPR), the positive predictive value (PPV), and the accuracy (ACC) are given in the text.

- true positive rate (TPR),

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{TP+FN}} = \frac{\mathrm{TP}}{\mathrm{P}};$$

- false positive rate (FPR),

$$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{FP+TN}} = \frac{\mathrm{FP}}{\mathrm{N}};$$

- positive predictive value (PPV),

$$\mathrm{PPV} = \frac{\mathrm{TP}}{\mathrm{TP+FP}} = \frac{\mathrm{TP}}{\mathrm{P'}};$$

- accuracy (ACC),

$$\mathrm{ACC} = \frac{\mathrm{TP+TN}}{\mathrm{P+N}}.$$

Hough Transform Clustering

- Distance ratio test - discards many of the false matches arising from background clutter.
- But, we still have matches that belong to different objects.

We want to cluster those features that belong to the same object and reject the matches that are left out in the clustering process.

Hough Transform Clustering

Goal: identify clusters of features with a consistent interpretation

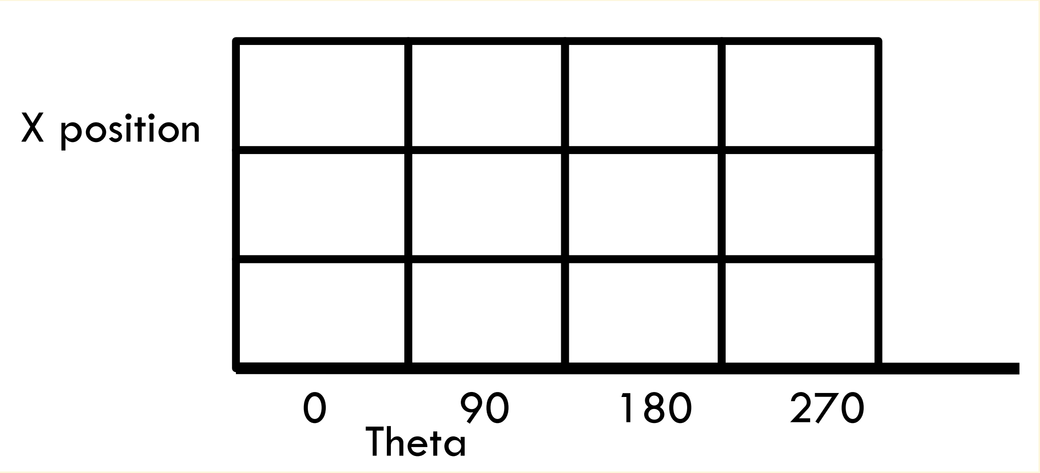
- Use each feature to vote for all object poses that are consistent with the feature.
- An entry in a hash table is created predicting the model location, orientation, and scale from the match hypothesis.
- The hash table is searched to identify all clusters of at least 3 entries in a bin, and the bins are sorted into decreasing order of size.

Hough Transform Clustering

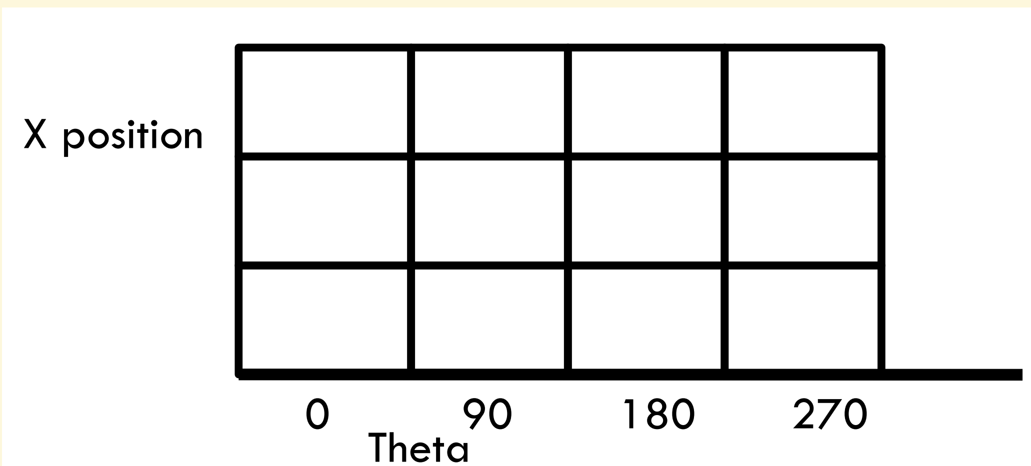
- Create 4D Hough Transform (HT) Space for each reference image
  1. Orientation bin: 30° increments
  2. Scale bin: factors of 2
  3. X location bin: 0.25*ref_image width
  4. Y location bin: 0.25*ref_image height
- If key point "votes" for reference image, tally its vote in 4D HT Space.
  - This gives estimate of location and pose
  - Keep list of which key points vote for a bin

Hough Transform - simplified

- If we take each feature and align the database image at that feature we can vote for the x position of the center of the object and the theta of the object based on all the poses that align



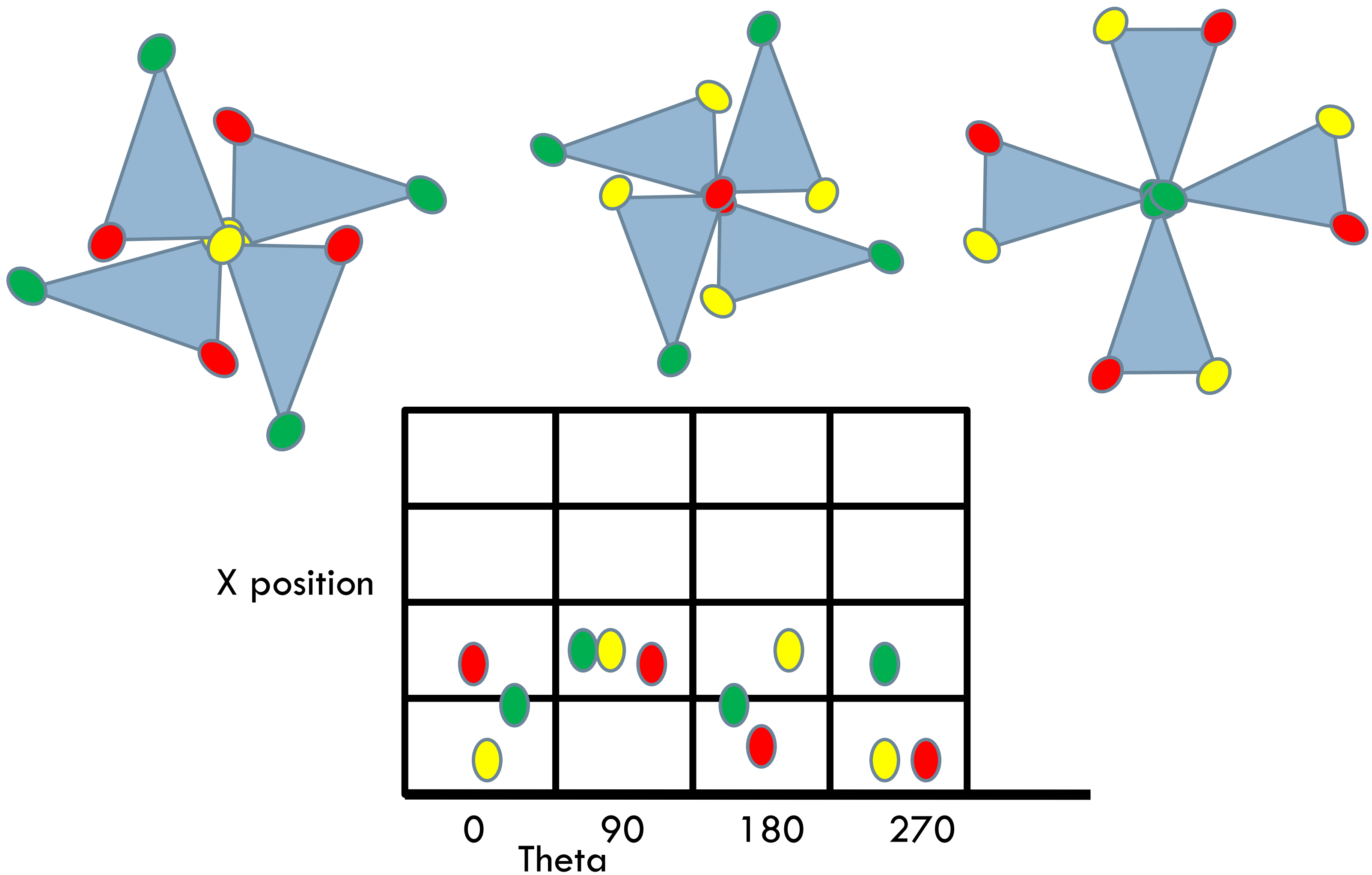
Hough Transform - simplified



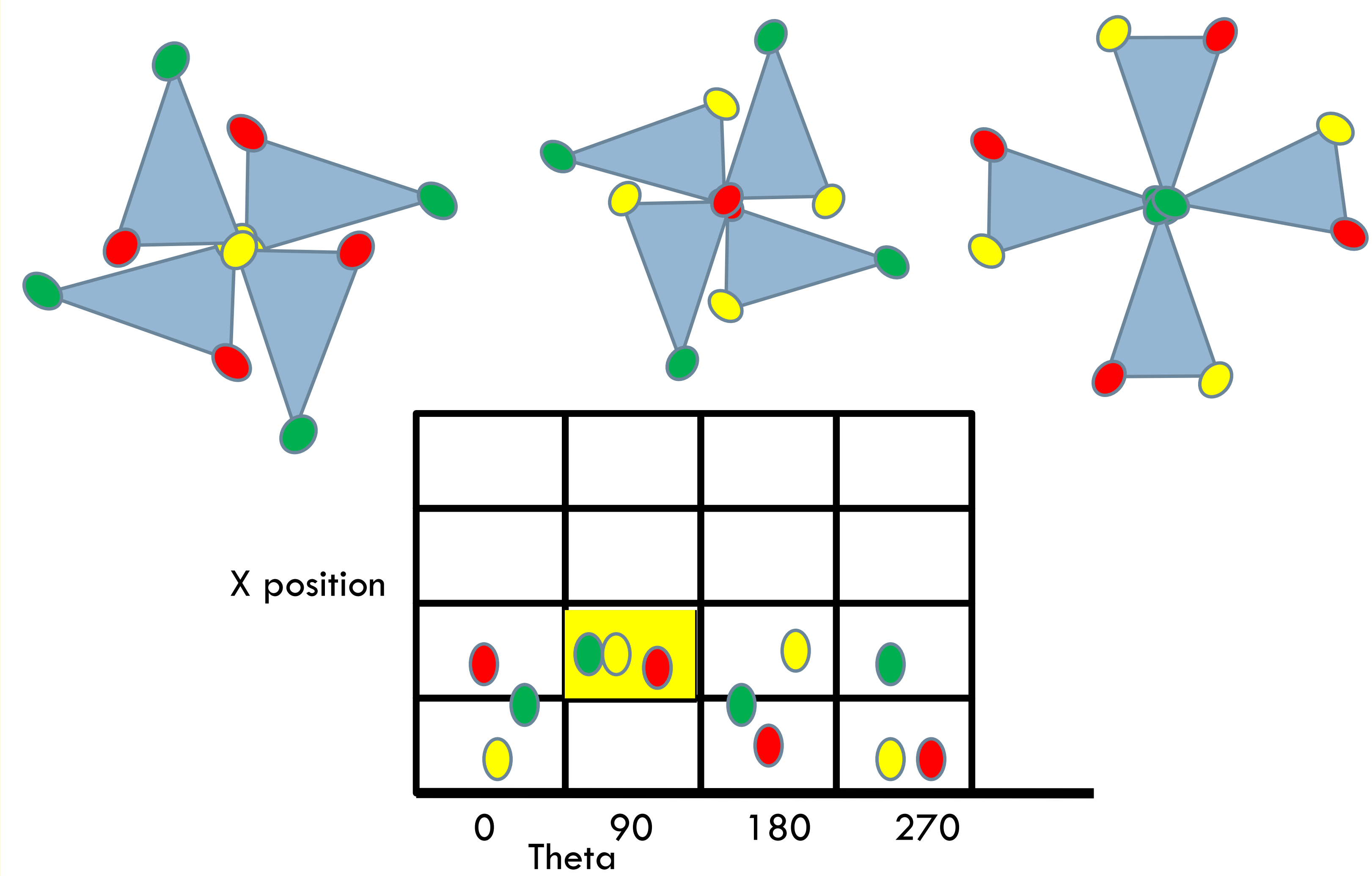
Assume we have 4 x locations
And only 4 possible rotations (thetas)

Then the Hough space can look like the
Diagram to the left

Hough Transform - simplified



Hough Transform - simplified



Verification

- Identify bins with largest votes (must have at least 3).
- Using list of keypoints which voted for a cell, compute affine transformation parameters (m, t)
  - Use corresponding coordinates of reference model (x,y) and target image (u,v).

$$\begin{bmatrix} u \\ v \end{bmatrix} = \begin{bmatrix} m_1 & m_2 \\ m_3 & m_4 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} t_x \\ t_y \end{bmatrix}$$

Verification

- If more than three points, solve in least-squares sense

$$\begin{bmatrix} x & y & 0 & 0 & 1 & 0 \\ 0 & 0 & x & y & 0 & 1 \\ & & \cdots & & & \\ & & \cdots & & & \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \\ t_x \\ t_y \end{bmatrix} = \begin{bmatrix} u \\ v \\ \vdots \end{bmatrix}$$

$$\mathbf{Ax = b}$$

$$\mathbf{x} = [\mathbf{A^T A}]^{-1} \mathbf{A^T b}$$

Verification: Remove Outliers

- After applying affine transformation to key points, determine difference between calculated location and actual target image location
- Throw out if:
  - Orientation different by > 15°
  - Scale off by sqrt(2)
  - X,Y location off by 0.2*model size
- Repeat least-squares solution until no points are thrown out

Object recognition (David Lowe - SIFT)

Hough Transform Clustering