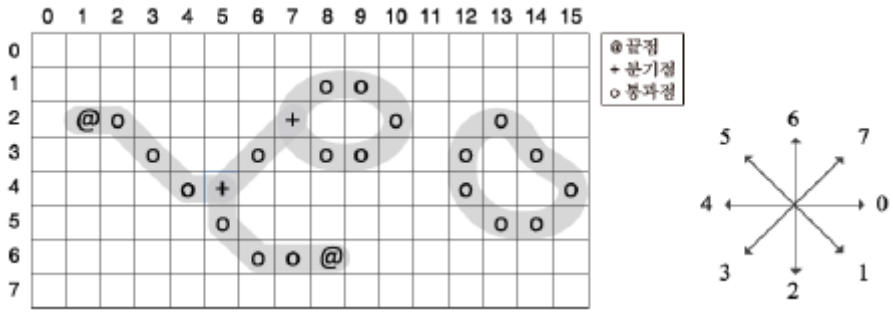
선분 검출

- 에지 맵 -> 에지 토막 -> 선분

에지 연결과 선분 근사



(a) 에지 영상 (b) 체인 코드 표시 기준

에지 토막	에지 열	체인 코드
1	(2,1)(2,2)(3,3)(4,4)(4,5)	(2,1)0110
2	(4,5)(5,5)(6,6)(6,7)(6,8)	(4,5)2100
3	(4,5)(3,6)(2,7)	(4,5)77
4	(2,7)(1,8)(1,9)(2,10)(3,9)(3,8)	(2,7)701345
5	(2,13)(3,14)(4,15)(5,14)(5,13)(4,12)(3,12)	(2,13)113567

그림 3-22 에지 토막의 에지 열과 체인 코드 표현

- 체인 코드 : 시작하는점 ~ 방향숫자
- 선분 근사
  - 두 끝점을 잇는 직선으로부터 가장 먼 점 까지의 거리 h가 임계값 이내가 될 때까지 선분 분할을 **재귀적으로 반복**

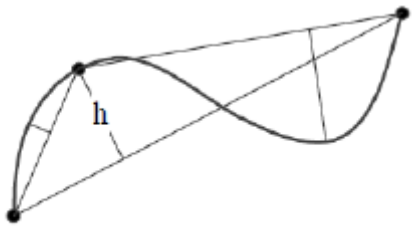


그림 3-27 선분 근사화 알고리즘

  - 차원을 줄이겠다는 의도

허프 변환(Hough Transform)

- 에지 연결 과정 없이 선분 검출 (전역 연산을 이용한 지각 군집화)
- 영상 공간 y-x를 기울기 절편 공간 b-a로 매핑

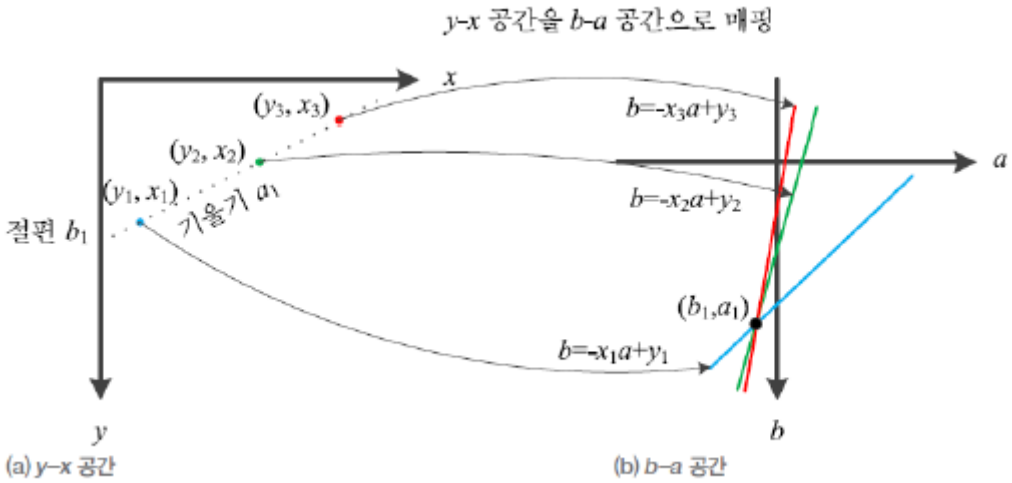


그림 3-28 허프 변환의 원리

- 차원을 줄이면서도 특성을 유지한다

- 수직선의 기울기가 ∞문제
  - 극 좌표계 사용하여 해결

$$y\cos\theta + x\sin\theta = \rho \tag{3.16}$$

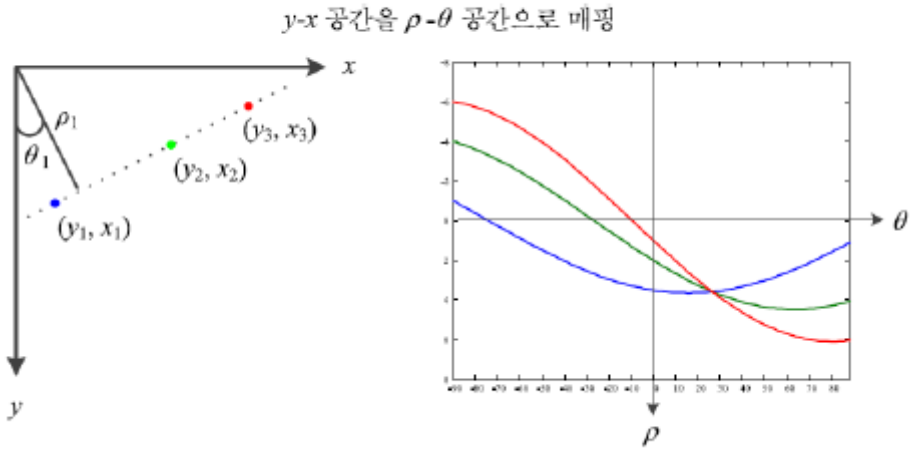


그림 3-29 ρ–θ 공간에서 허프 변환

  - p에 따라서 θ값이 어떻게 변화하는가

- 밀집된 곳 찾기
  - 양자화된 누적 배열 이용하여 해결

알고리즘 3-7 직선 검출을 위한 허프 변환

입력 : 에지 영상 $e(j,i)$, $0 \le j \le M-1$, $0 \le i \le N-1$, 임계값 T // 에지는 1, 비에지는 0인 이진 영상

출력 : (ρ_k, θ_k), $1 \le k \le n$(n개의 직선)

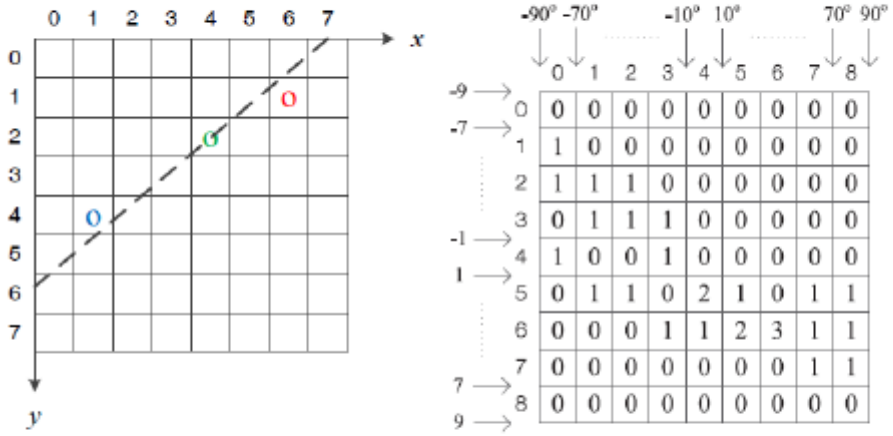
1. 2차원 누적 배열 A를 0으로 초기화한다.
2. for(에지 영상 e에 있는 에지 화소 (y_i, x_i) 각각에 대해)
3. $y_i\cos\theta + x_i\sin\theta = \rho$가 지나는 A의 모든 칸을 1만큼 증가시킨다.
4. A에서 T를 넘는 지역 최대점 (ρ_k, θ_k)를 모두 찾아 직선으로 취한다.

- 원 검출
  - 3차원 누적 배열 사용

$$(y-b)^2 + (x-a)^2 = r^2$$

예제 3-3 허프 변환

[그림 3-30]은 [그림 3-29]를 이산 공간에 다시 그린 것이다. 왼쪽 그림에서 세 점은 $(y_1, x_1)=(4,1)$, $(y_2, x_2)=(2,4)$, $(y_3, x_3)=(1,6)$이다. $(y_1, x_1)=(3.5,1)$이면 세 점이 정확히 일직선 상에 있지만, 디지털 영상의 특성상 약간의 위치 오차가 발생했다고 간주하자.



	0	1	2	3	4	5	6	7	8
0	0	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0
2	1	1	1	0	0	0	0	0	0
3	0	1	1	1	0	0	0	0	0
4	1	0	0	1	0	0	0	0	0
5	0	1	1	0	2	1	0	1	1
6	0	0	0	1	1	2	3	1	1
7	0	0	0	0	0	0	0	1	1
8	0	0	0	0	0	0	0	0	0

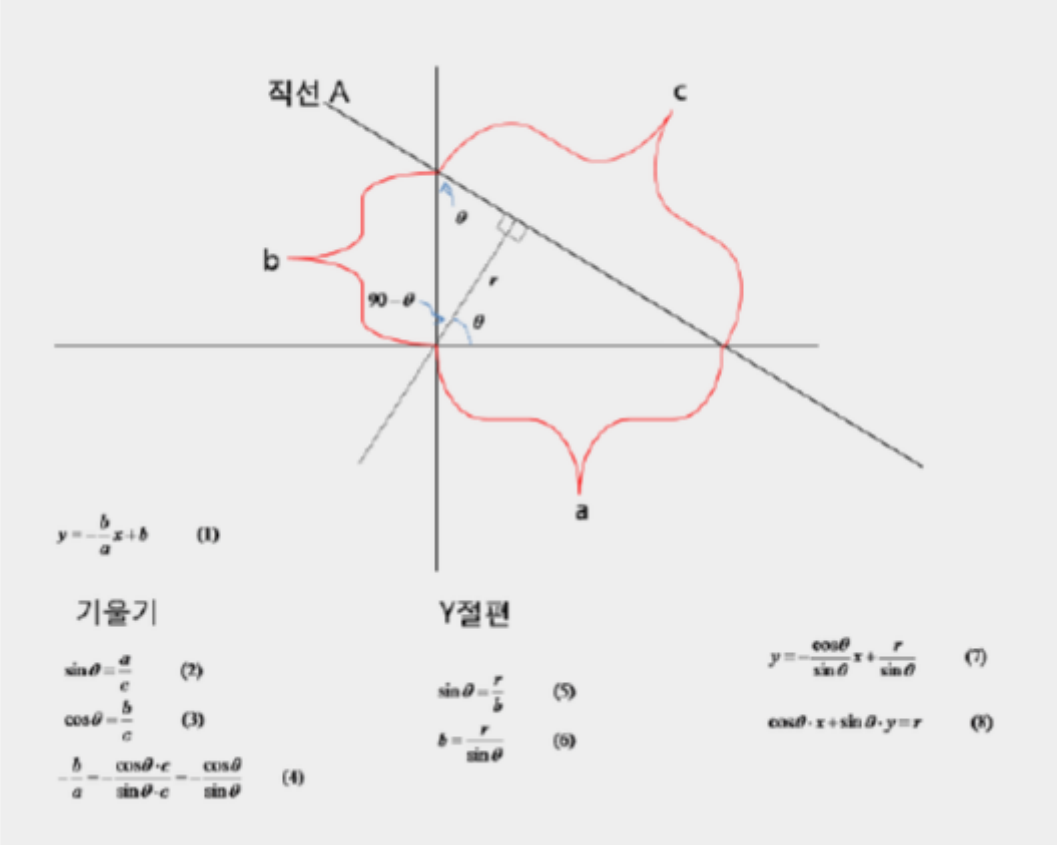
그림 3-30 이산 공간에서 허프 변환

θ축은 20° 간격으로 양자화하여 총 아홉 개의 구간을 가지도록 하였다. ρ축은 범위 $[-9, 9]$를 2 크기의 구간으로 나누어 총 아홉 개의 구간을 가지도록 양자화하였다. 따라서 누적 배열 A는 9×9이다. [알고리즘 3-7]에 따라 A를 0으로 초기화한 후, 2~3행을 수행하여 세 점의 자취를 누적시키면 오른쪽 그림과 같은 배열이 된다. 이 배열에서 지역 최대점은 3을 갖는 (6,6)으로, $(\rho, \theta)=(4, 40°)$에 해당한다. $y \cos 40° + x \sin 40° = 4$라는 직선을 검출한 셈이다. 왼쪽 그림에 있는 점선이 검출한 직선이다.

- An early type of voting scheme

Voting Schemes

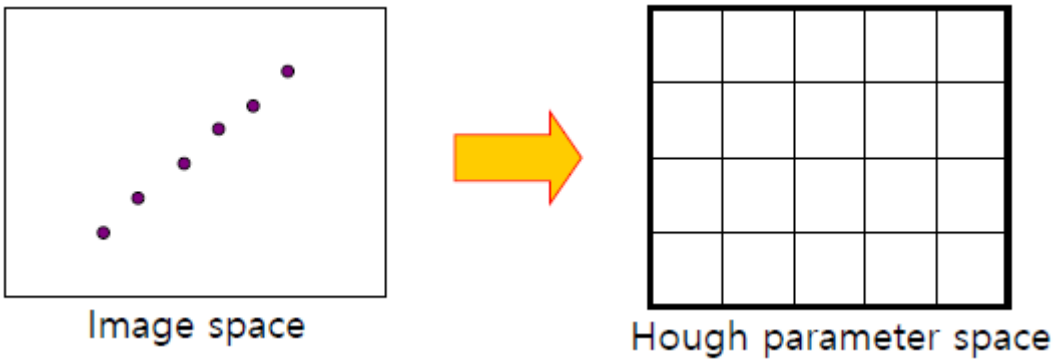
- Let each feature vote for all the models that are compatible with it
- Hopefully the noise features will not vote consistently for any single model
- Missing data doesn't matter as long as there are enough features remaining to agree on a good model



- 주어진 에지 영역이 어떤 특징을 가지고 있는가(몇 개의 점들만 가지고 직선방정식을 그린다)
  - θ와 p의 극좌표계 이용

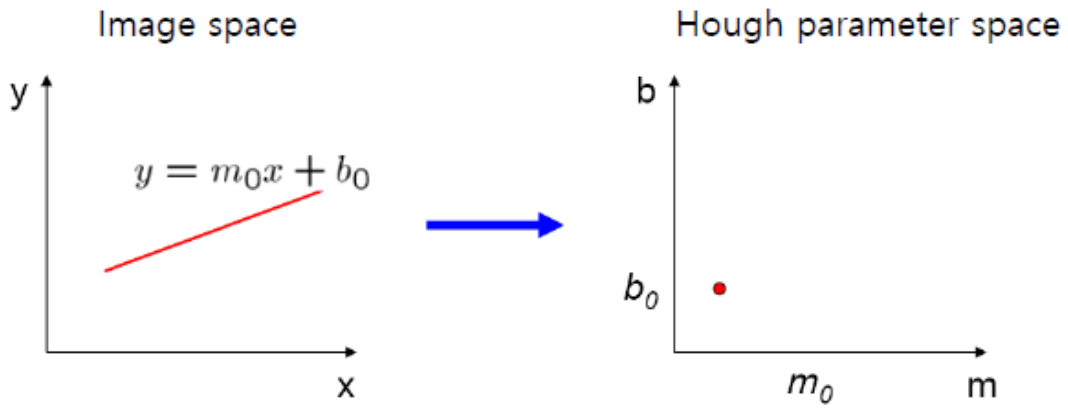
General Outline

- Discretize parameter space into bins
- For each feature point in the image, put a vote in every bin in the parameter space that could have generated this point
- Find bins that have the most votes

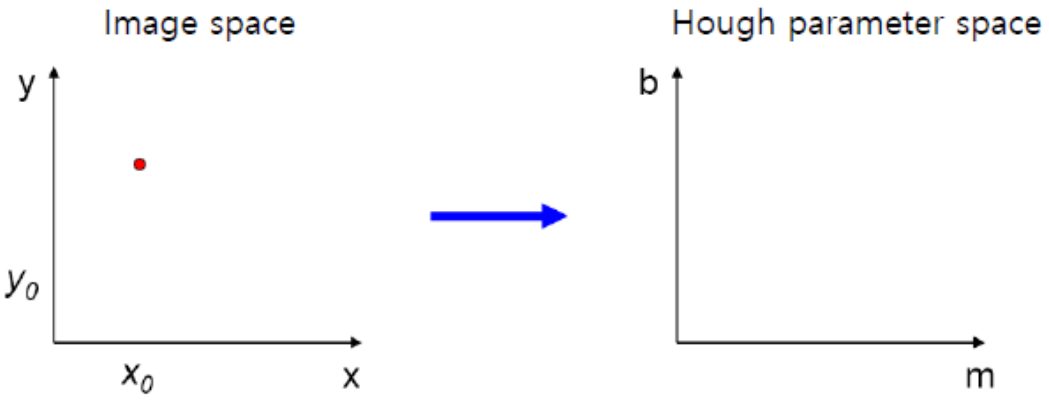


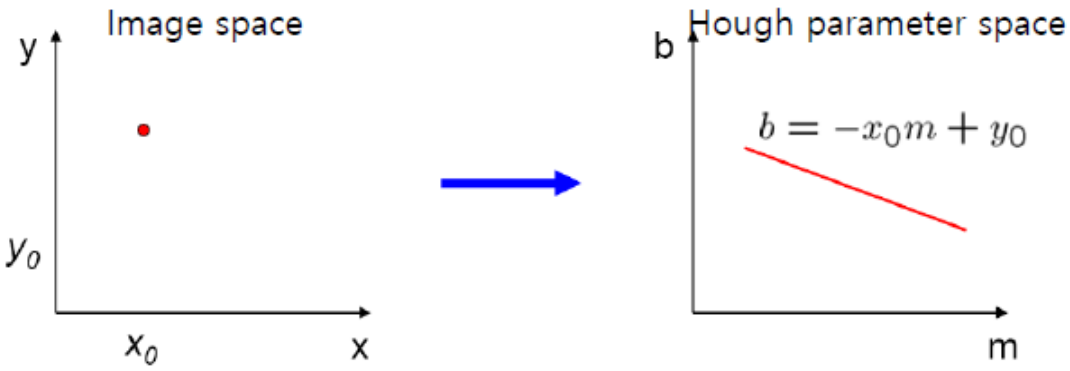
Parameter Space Representation

- A line in the image corresponds to **a point** in Hough Space

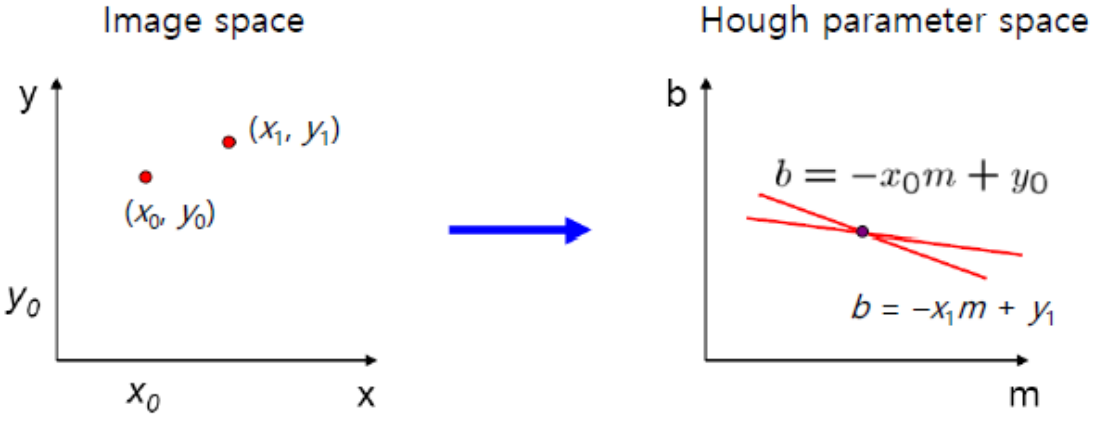


- What does a point (x_0, y_0) in the image space map to in the Hough space



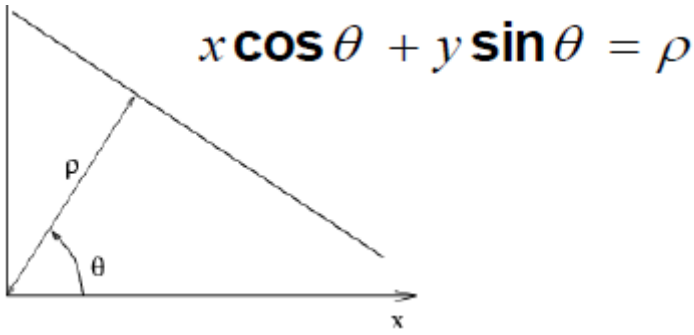


  - the solutions of $b = -x_0m + y_0$
  - this is **a line** in hough space
- Where is the line that contains both (x_0, y_0) and (x_1, y_1)



  - the intersection of $b = -x_0m + y_0$ and $b = -x_1m + y_1$
- Problems with the (m,b) space
  - Unbounded parameter domains

- Vertical lines require infinite m
- Alternative : polar representation



- each point will add a sinusoid in the (θ, p) parameter space

Algorithm Outline

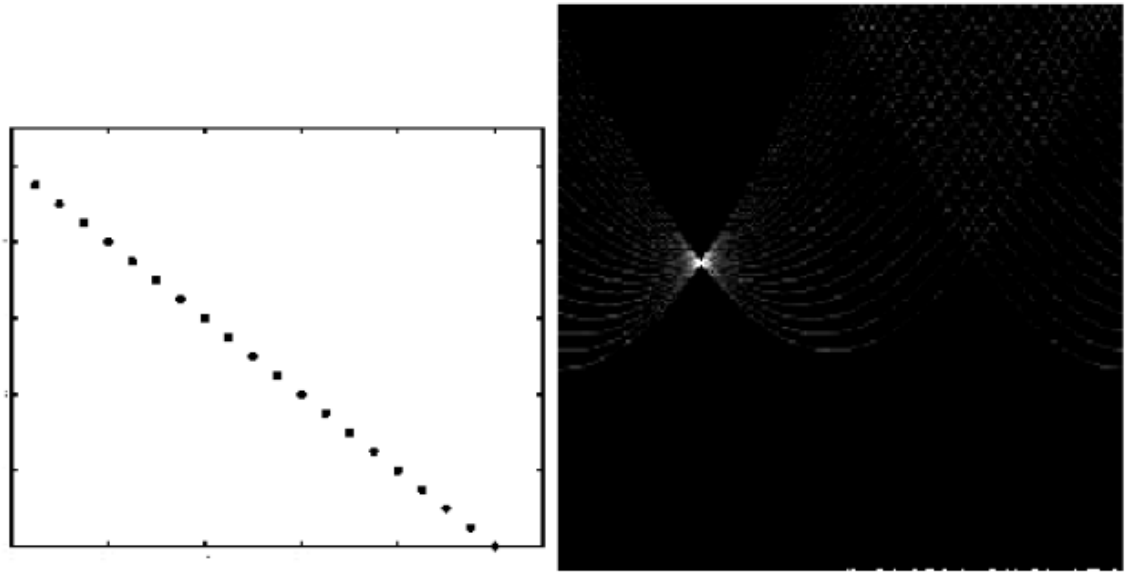
- Initialize accumulator H to all zeros
- For each edge point (x,y) in the image

```
For θ = 0 to 180
    ρ = x cos θ + y sin θ
    H(θ, ρ) = H(θ, ρ) + 1
 end
end
```

- Find the values of (θ, p) where H (θ, p) is a local maximum
  - the detected line in the image is given by

    $p = xcos\theta + ysin\theta$

Basic Illustration

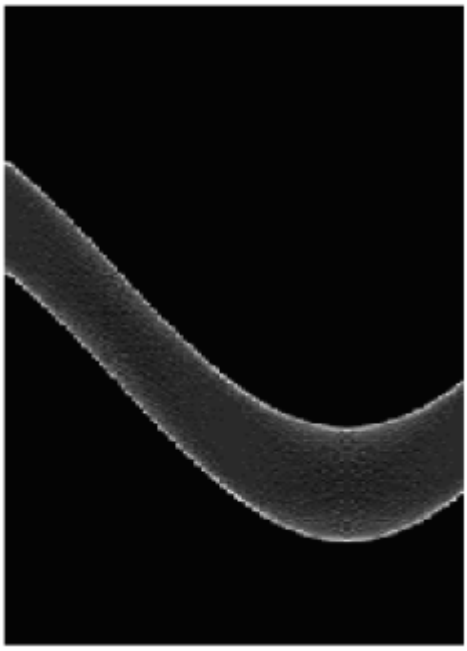


features votes

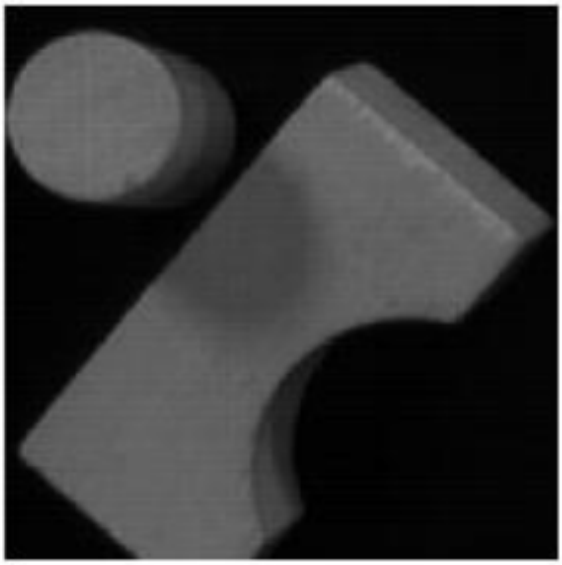
Other shapes

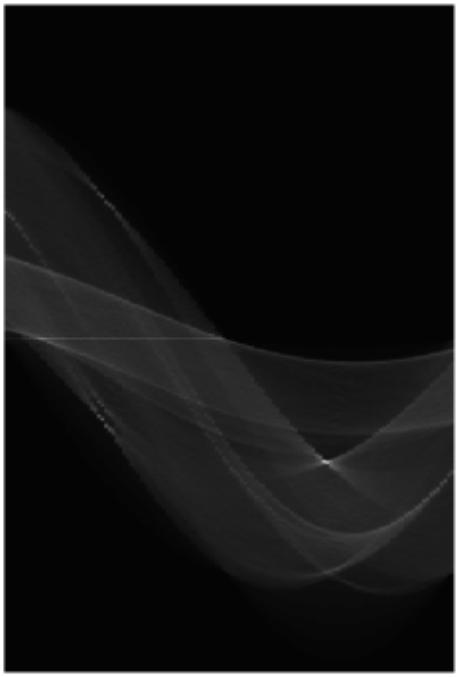
Square

Circle

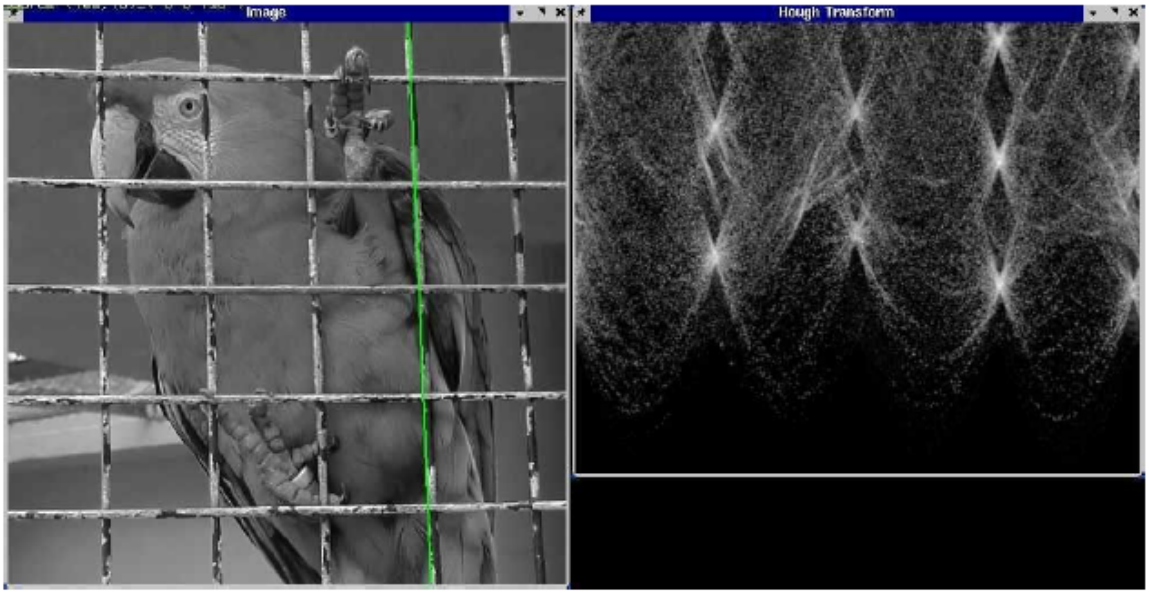


Sevaral Lines





A more complicated image



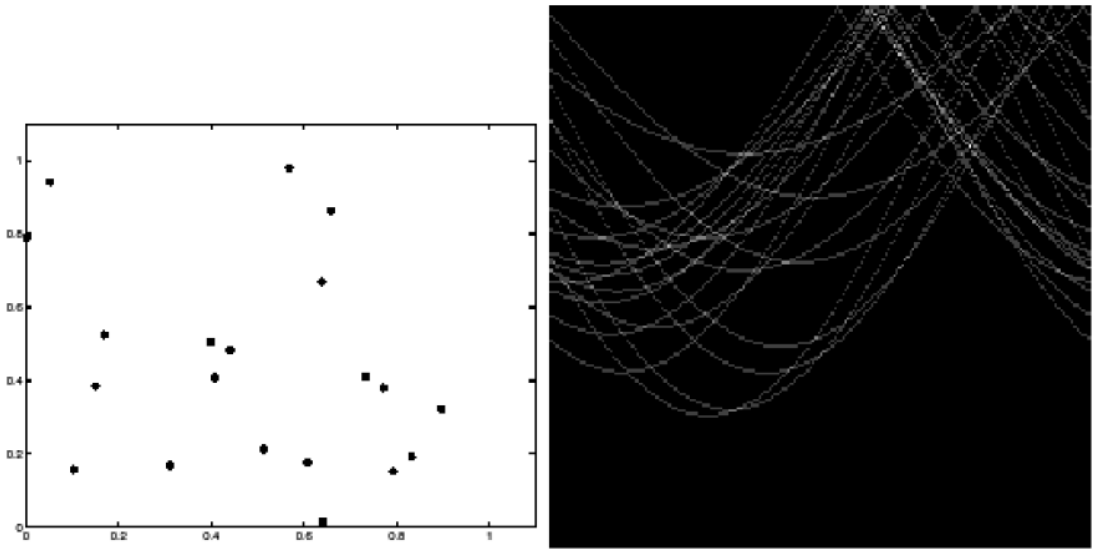
- 많이 겹치는 곳 : 세타가 똑같은 친구들이 겹쳐있는 것
- 허프 트랜스폼을 잘 쓰면 다양한 잡음을 제거할 수 있다

Effect of Noise

- 멀리 본다 : 노이즈 증가
- 가까이 본다 : 정교한 제어 필요

Random Points

- Uniform noise can lead to spurious peaks in the array



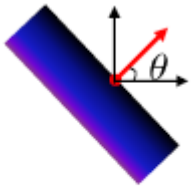
- As the level of noise increases, the maximum number of votes increases too

Dealing with noise

- Choose a grid / discretization
    - Too coarse : large votes obtained when too many different lines correspond to a single bucket
    - Too fine : miss lines because some points that are not exactly collinear cast votes for different buckets
- Increment neighboring bins (smoothing in accumulator array)

- Try to get rid of irrelevant features
    - Take only edge points with significant gradient magnitude

Incorporating image gradients

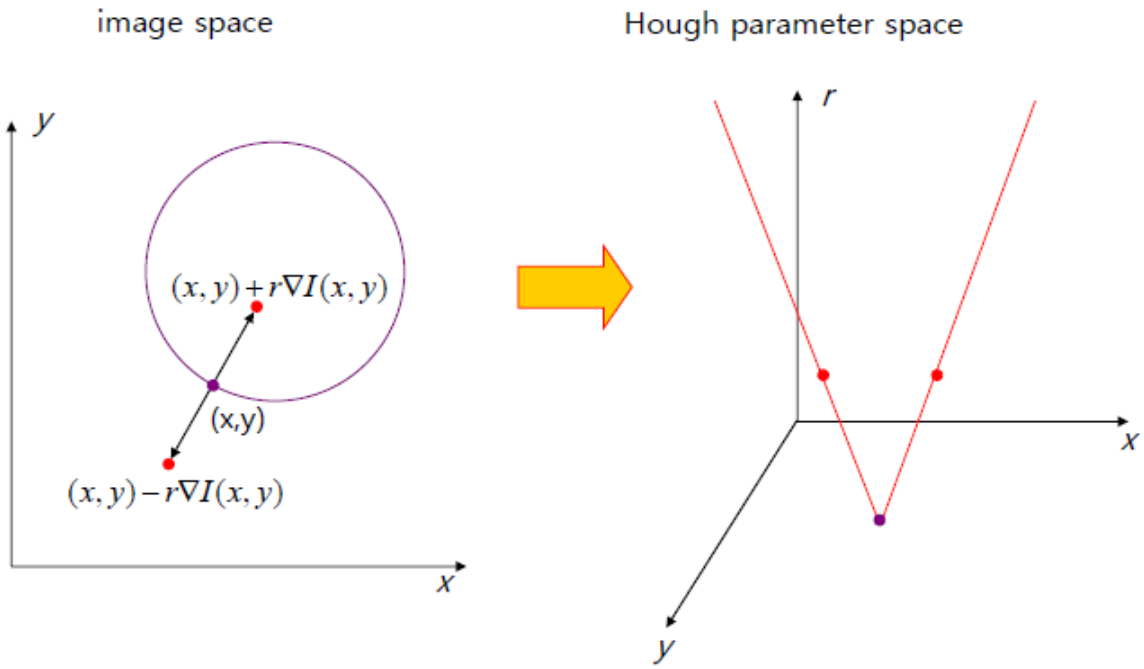


$$\nabla f = \left[\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}\right]$$

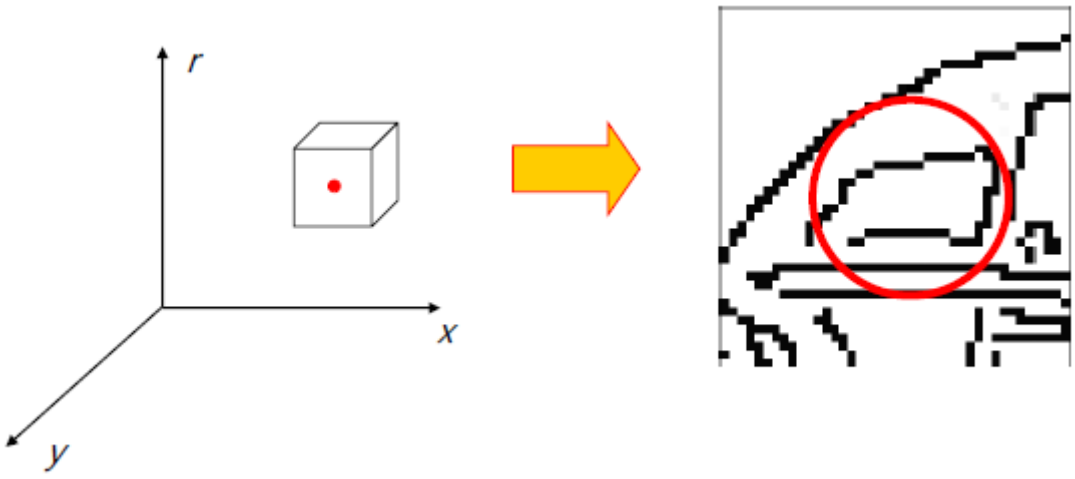
$$\theta = \tan^{-1}\left(\frac{\partial f}{\partial y} / \frac{\partial f}{\partial x}\right)$$

For each edge point (x,y)
θ = gradient orientation at (x,y)
ρ = x cos θ + y sin θ
H(θ, ρ) = H(θ, ρ) + 1
end

Hough tranform for circles

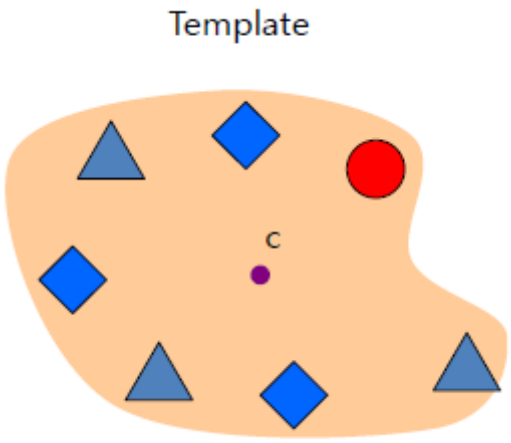


- How many dimensions will the parameter space have?
- Given an oriented edge point, what are all possible bins that it can vote for?
- 반드시 반지름 r 값의 범위가 주어져야 함
- Conceptually equivalent procedure : for each (x,y,r), draw the corresponding circle in the image and compute its "support"

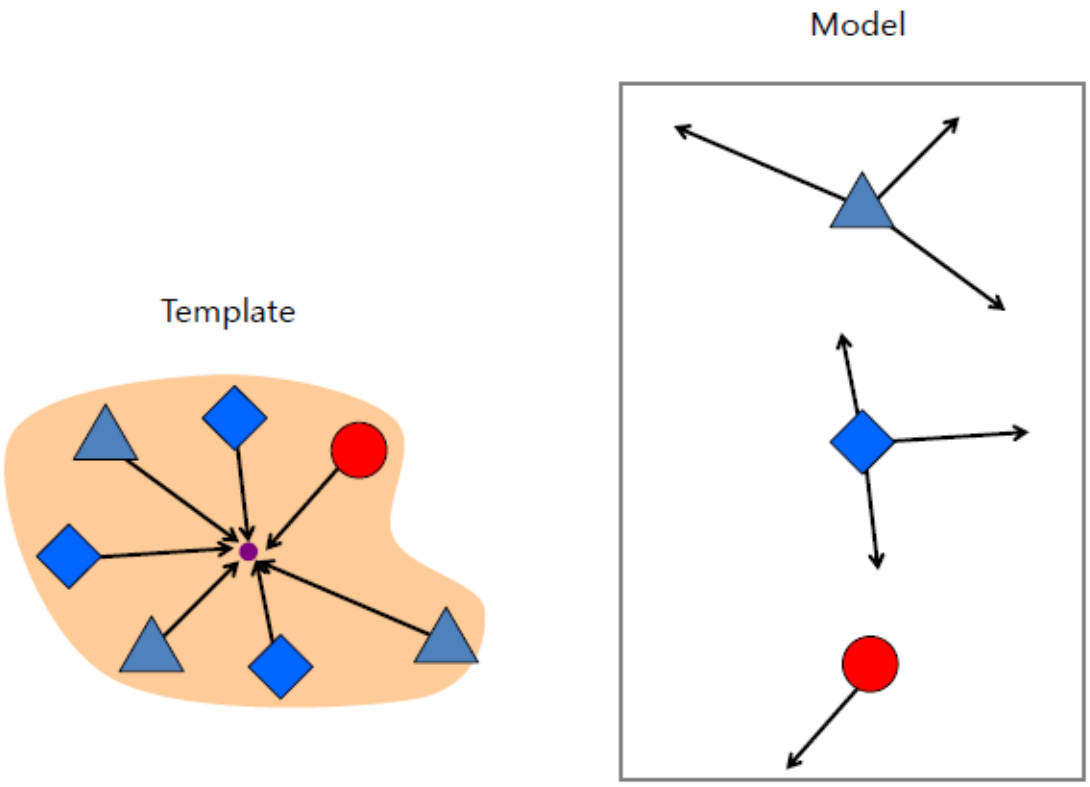


Generalized Hough transform

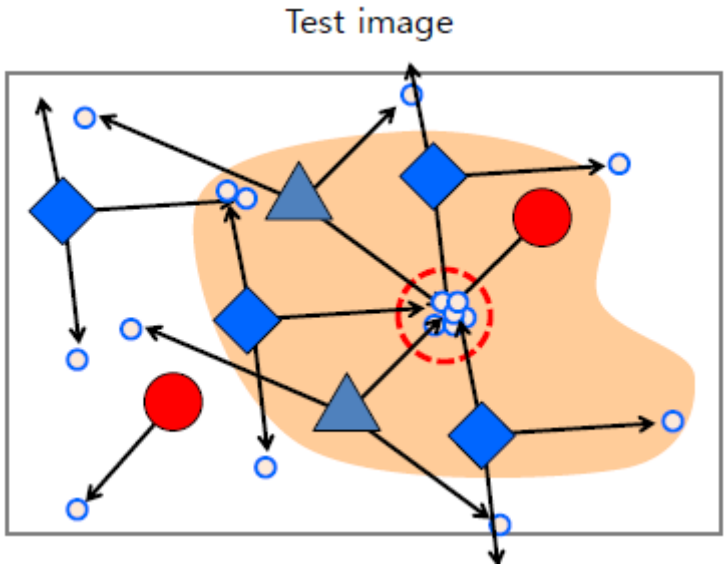
- We want to find a template defined by its reference point(center) and sveral distict types of landmark points in stable spatial configuration



- Template representation : for each type of landmark point, store all possible displacement vectors towards the center

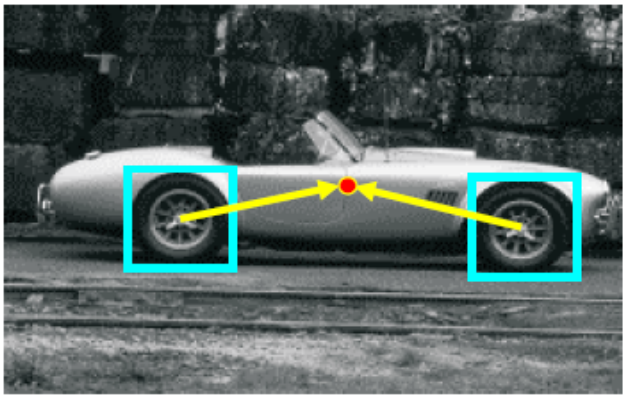


- Detecting the template : for each feature in a new image, look up that feature type in the model and vote for the possible center locations associated with that type in the model



Application in recognition

- Index displacements by 'visual codeword'



training image

visual codeword with displacement vectors

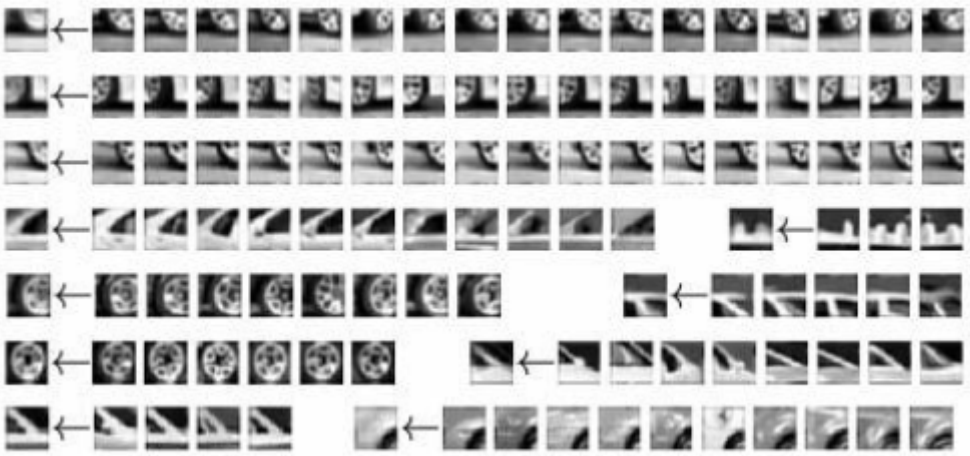
- 바퀴라는 특징을 가지는 모델

test image

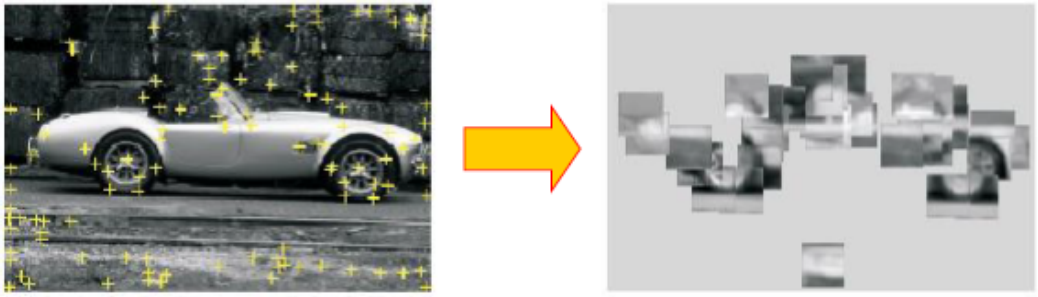
- 두 개 이상의 바퀴가 있는 곳에 자동차가 있을 것이라는 것을 추측가능

Implicit shape models : Training

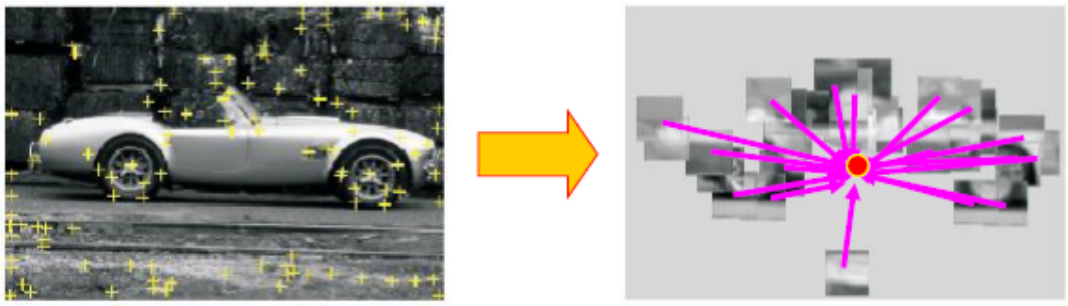
- Build codebook of patches around extracted interest points using clustering



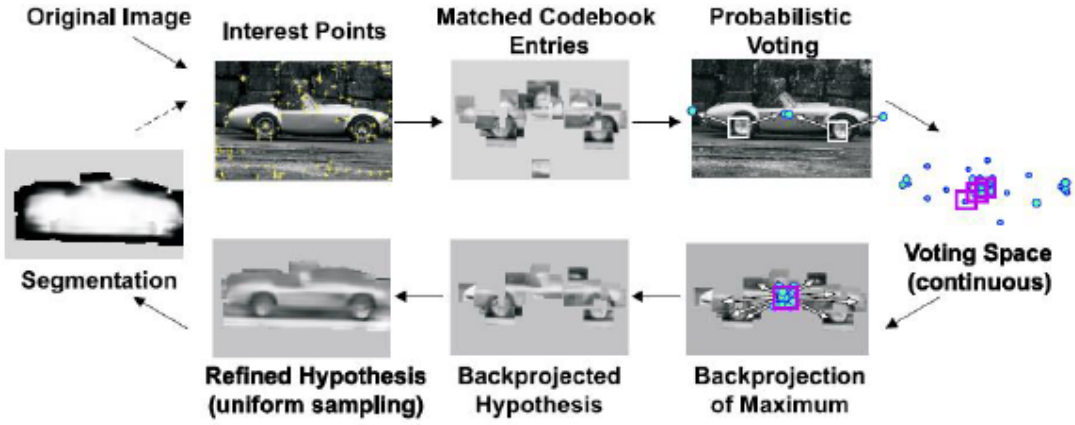
- Map the patch around each interest point to closet codebook entry



- For each codebook entry, store all positions it was found, relative to object center



- Extract weighted segmentation mask based on stored masks for the codebook occurrences



  - 자동차의 중심을 찾아냄

Implicit Shape Models : Details

- **Supervised** Training
  - need reference location and segmentation mask for each training car
- Voting space is continuous, not discrete
  - Clustering algorithm needed to find maxima
- How about dealing with scale changes?
  - Option 1 : Search a range of scales, as in Hough transform for circles
  - Option 2 : Use interest points with characteristic scale
- Verification stage is very important
  - Once we have a location hypothesis, we can overlay a more detailed template over the image and compare pixel-by-pixel, transfer segmentation masks, etc

Hough Transform : Discussion

- Pros
  - Can deal with non-locality and occlusion
  - Can detect multiple instances of a model
  - Some **robustness to noise** : noise points unlikely to contribute consistently to any single bin
- Cons
  - Complexity of search time increases exponentially with the number of model parameters
  - Non-target shapes can produce spurious peaks in parameter space
  - It's har to pick a good grid size

RANSAC

- 1981년 Fischler & Bolles가 제안 [Fischler81]
- 인라이어를 찾아 어떤 모델을 적합시키는 기법
- 난수 생성하여 인라이어 군집을 찾기 때문에 임의성을 지님
- 원리
  - 선분 검출에 적용
  - 모델은 직선의 방정식 $y = ax + b$

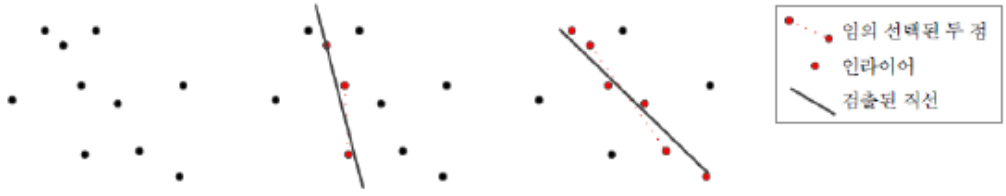
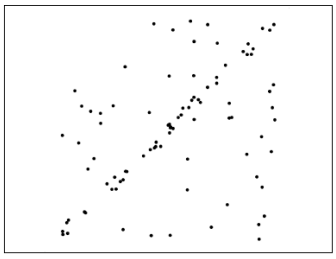
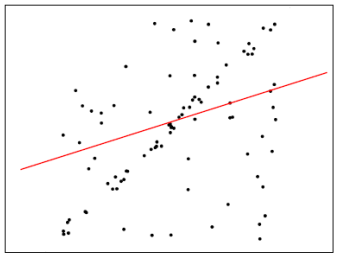
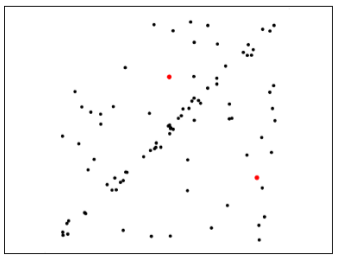


그림 3-31 RANSAC의 원리

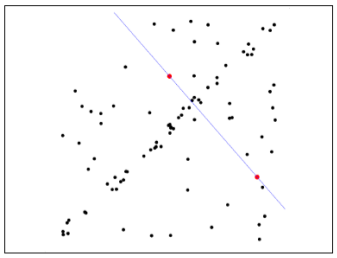
- 매칭 쌍 집합 $X = \{(a_1, b_1), (a_2, b_2), \ldots, (a_n, b_n)\}$을 처리할 수 있게 확장



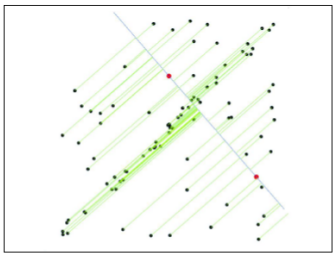




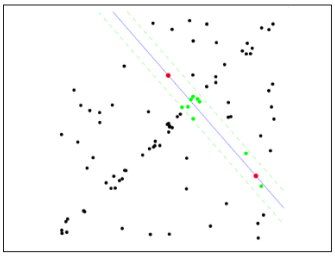
1. Randomly select minimal subset of points



1. Randomly select minimal subset of points
2. Hypothesize a model

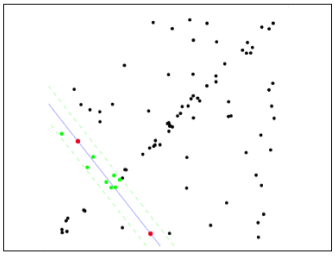


1. Randomly select minimal subset of points
2. Hypothesize a model
3. Compute error function

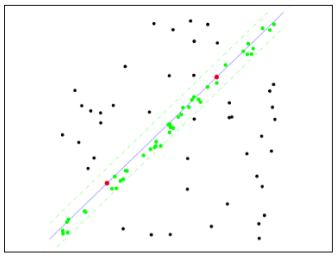


1. Randomly select minimal subset of points
2. Hypothesize a model
3. Compute error function
4. Select points consistent with model

Uncontaminated sample



1. Randomly select minimal subset of points
2. Hypothesize a model
3. Compute error function
4. Select points consistent with model
5. Repeat *hypothesize-and-verify* loop



1. Randomly select minimal subset of points
2. Hypothesize a model
3. Compute error function
4. Select points consistent with model
5. Repeat *hypothesize-and-verify* loop

알고리즘 7-9 기하 변환을 추정하기 위한 RANSAC

입력 : $X=\{(\mathbf{a}_i,\mathbf{b}_i), i=1,2,\cdots,n\}$ // 매칭 쌍 집합
반복 횟수 k, 인라이어 판단 t, 인라이어 집합의 크기 d, 적합 오차 e

출력 : 기하 변환 행렬 $\mathbf{T}$

```
1   Q = Ø;
2   for(j=1 to k) {
3     X에서 세 개 대응점 쌍을 임의로 선택한다.
4     이들 세 쌍을 입력으로 식 (7.14)를 풀어 T_j를 추정한다.
5     이들 세 쌍으로 집합 inlier를 초기화한다.
6     for(이 세 쌍을 제외한 X의 요소 p 각각에 대해) {
7       if(p가 허용 오차 t 이내로 T_j에 적합) p를 inlier에 넣는다.
8     }
9     if(|inlier|≥d)  // 집합 inlier가 d개 이상의 샘플을 가지면
10      inlier에 있는 모든 샘플을 가지고 새로운 T_j를 계산한다.
11    if(T_j의 적합 오류<e ) T_j를 집합 Q에 넣는다.
12  }
13  Q에 있는 변환 행렬 중 가장 좋은 것을 T로 취한다.
```