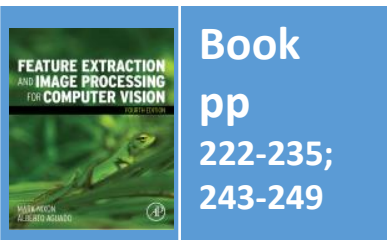


Lecture 8 Finding Shapes

COMP3204 & COMP6223 Computer Vision

How can we group points to find shapes?



Department of
Electronics and
Computer Science

UNIVERSITY OF
Southampton
School of Electronics
and Computer Science

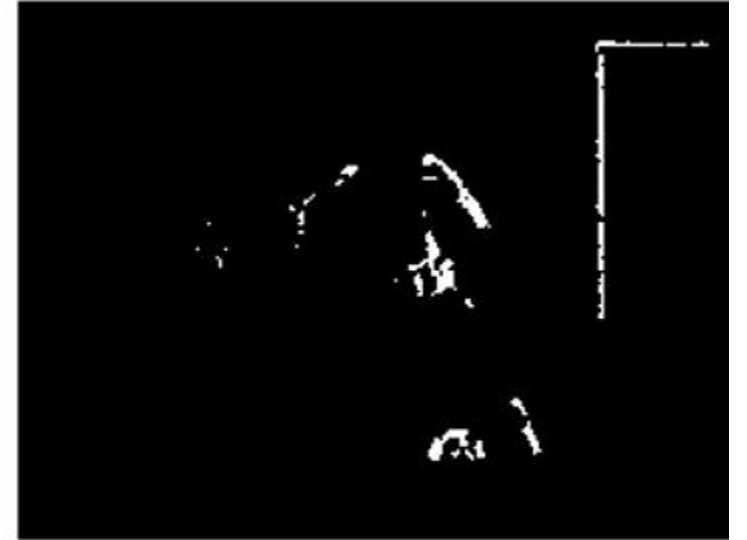
Feature extraction by thresholding



(a) image



(b) low threshold



(c) high threshold

- Conclusion: we need **shape**!



Template Matching

- Intuitively **simple**
- **Correlation** and convolution
- Implementation via **Fourier**
- Relationship with matched filter, viz: **optimality**



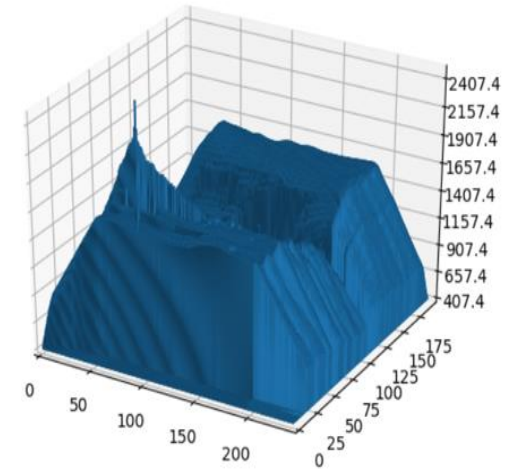
image



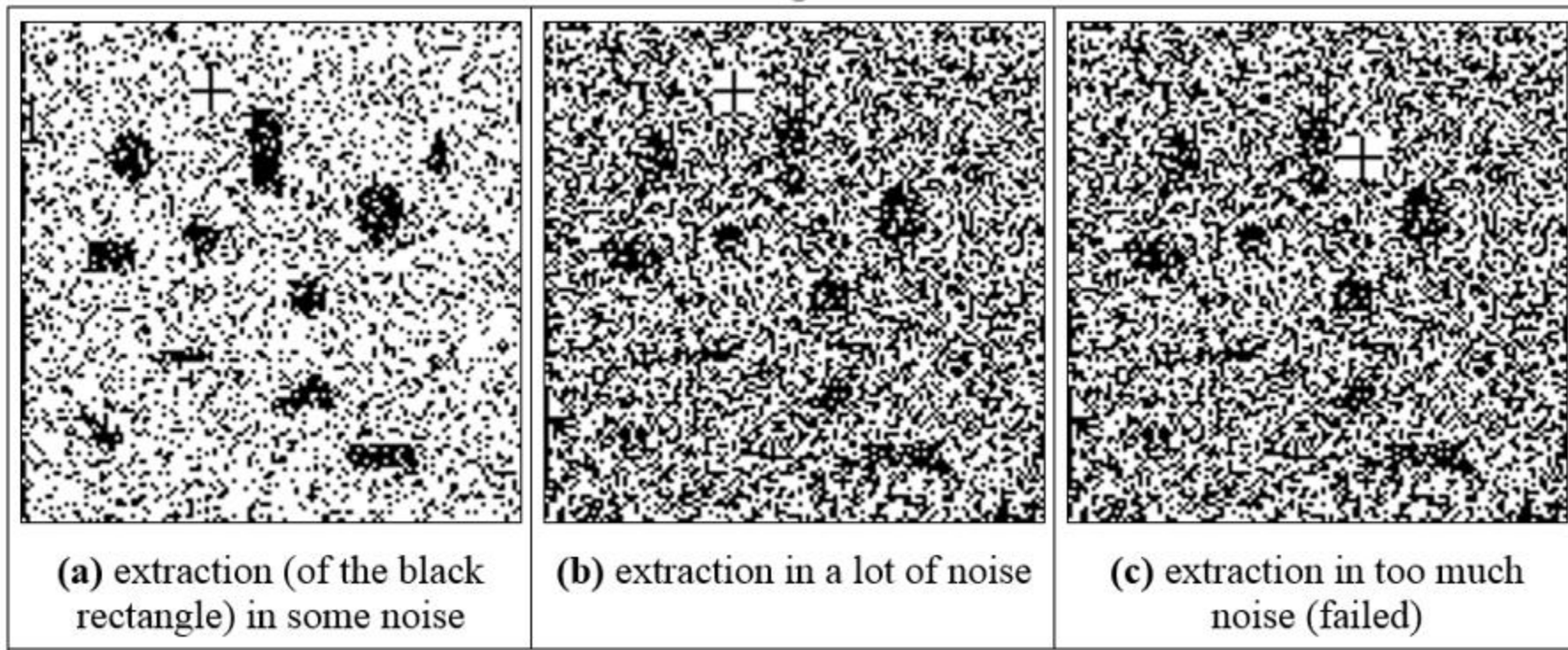
template



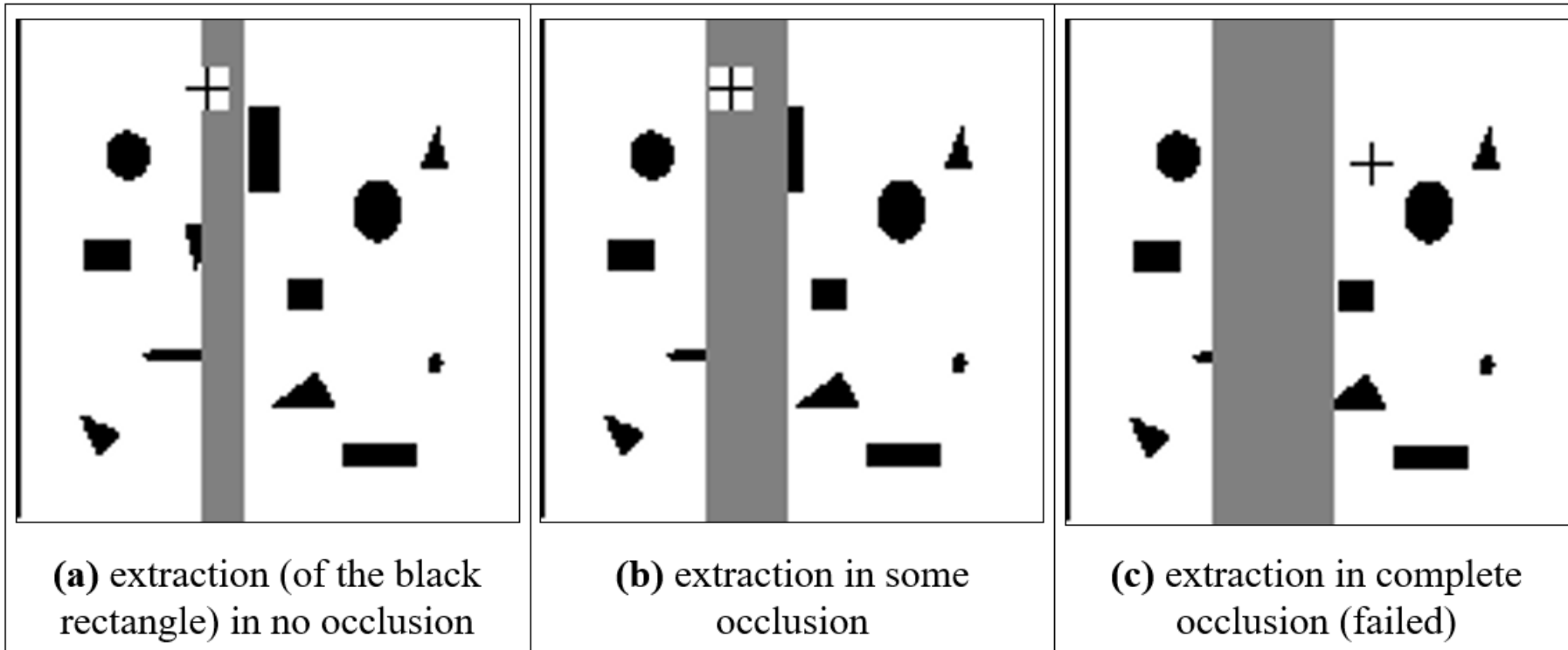
accumulator space



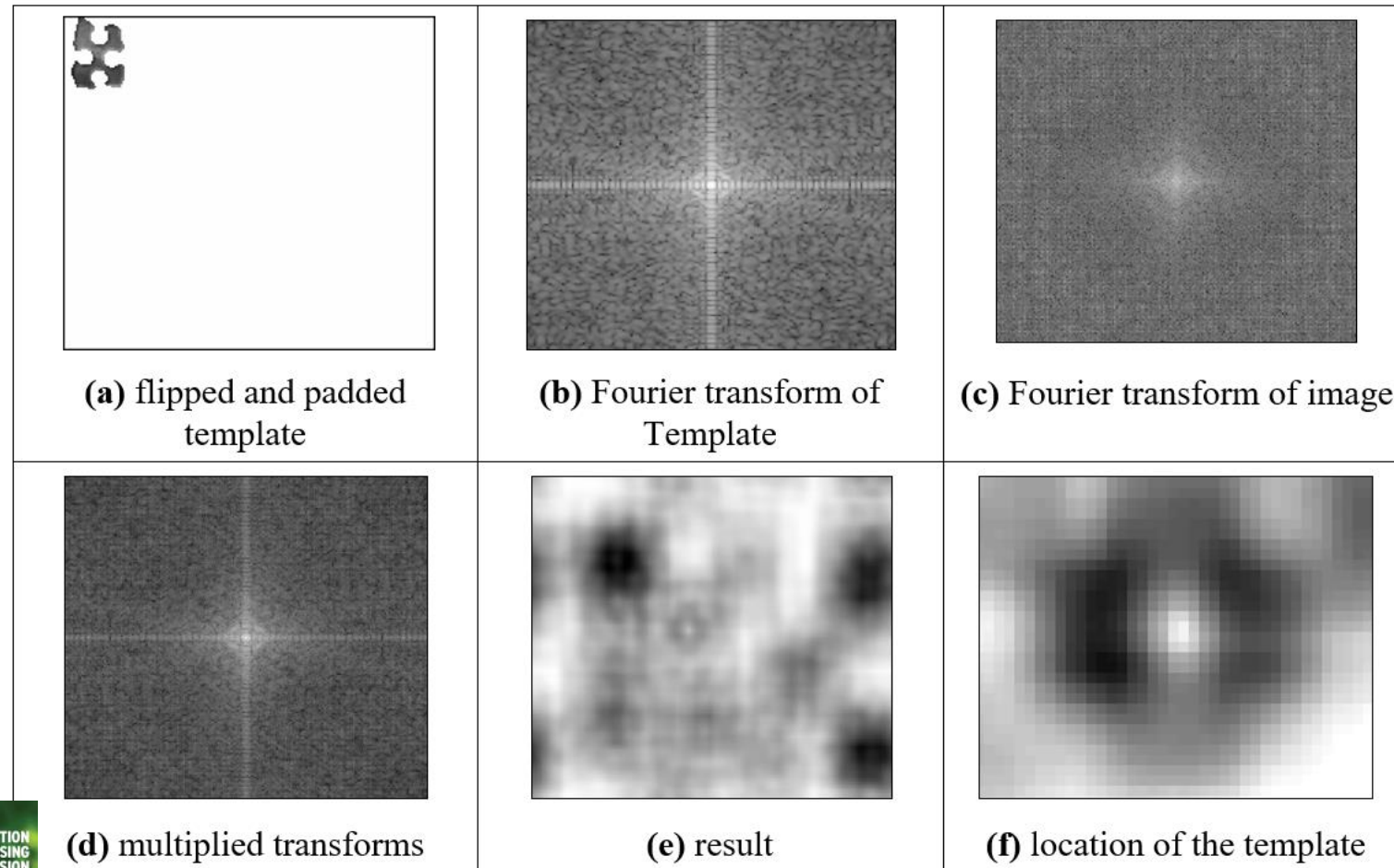
Template matching in noisy images



Template matching in occluded images



Encore, Monsieur Fourier!



$$\mathbf{P} \otimes \mathbf{T} = F^{-1} \left(F(\mathbf{P}) \times (F(\mathbf{T}))^c \right) \\ = \sum_{i \in \mathbf{P}} \sum_{j \in \mathbf{P}} \mathbf{P}_{i,j} \mathbf{T}_{i+n,j+m}$$

No sliding of
templates
here;

cost is Fourier
Transform plus
multiplication

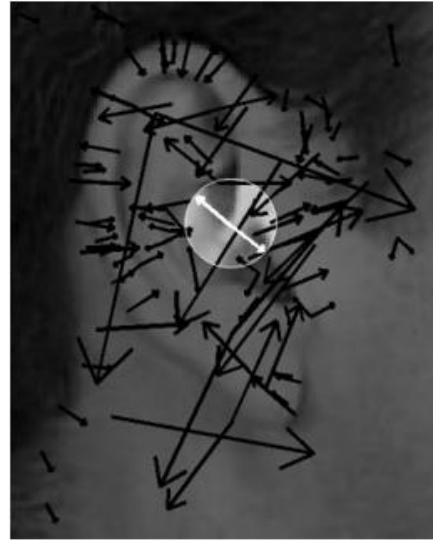
Applying template matching



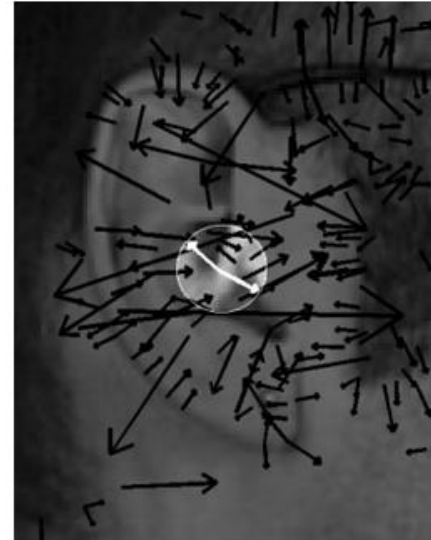
Applying SIFT in ear biometrics



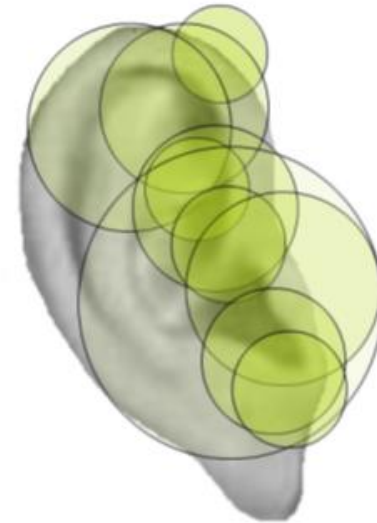
(a) detected SIFT points



(b) one feature



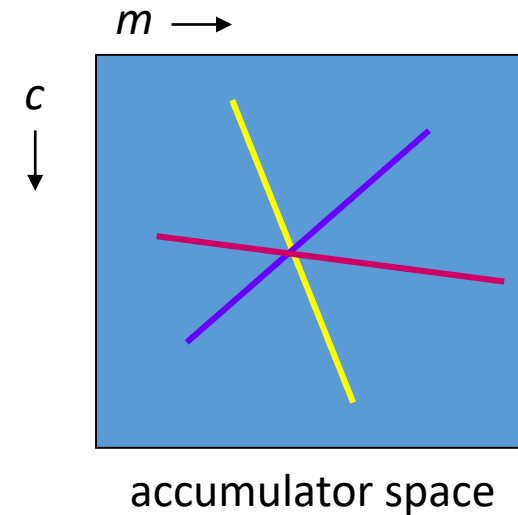
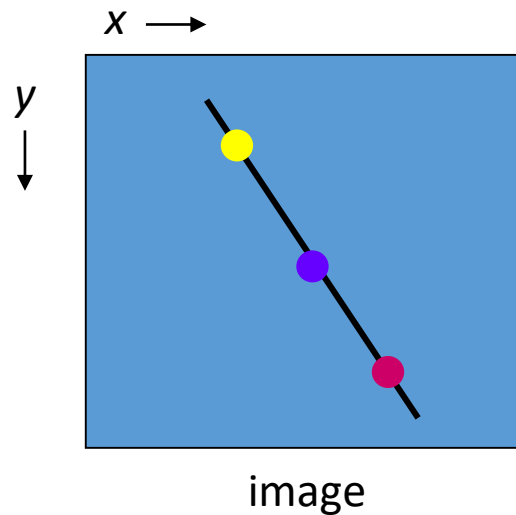
(c) same feature as (b) in a different ear



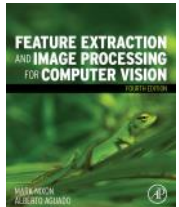
(d) regions of influence

Hough Transform

- **Performance** equivalent to template matching, but **faster**
- A line is points x,y gradient m intercept c $y = m \times x + c$
- **and** is points m,c gradient $-x$ intercept y $c = -x \times m + y$



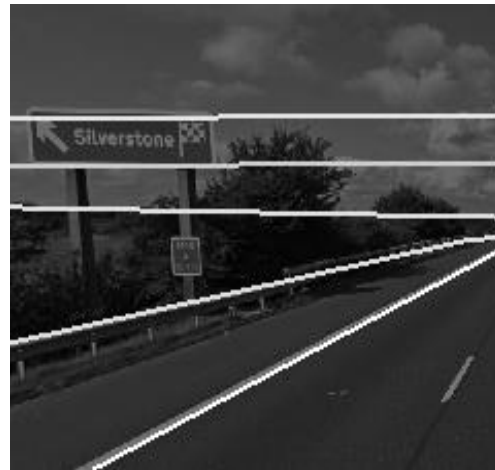
- In maths it's the **principle of duality**



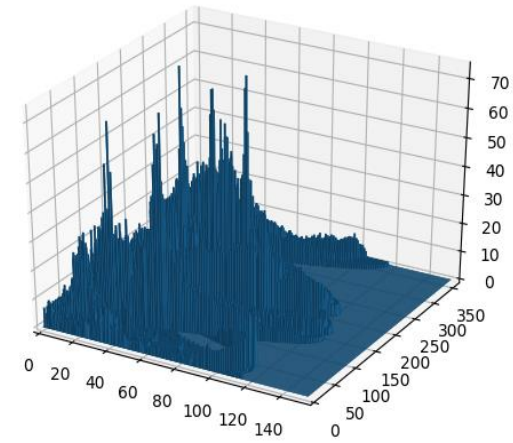
Applying the Hough transform for lines



image



detected lines



accumulator space



Hough Transform for Lines ... problems

- m, c tend to infinity
- Change the parameterisation
- Use foot of normal $\rho = x \cos \theta + y \sin \theta$
- Gives polar HT for lines

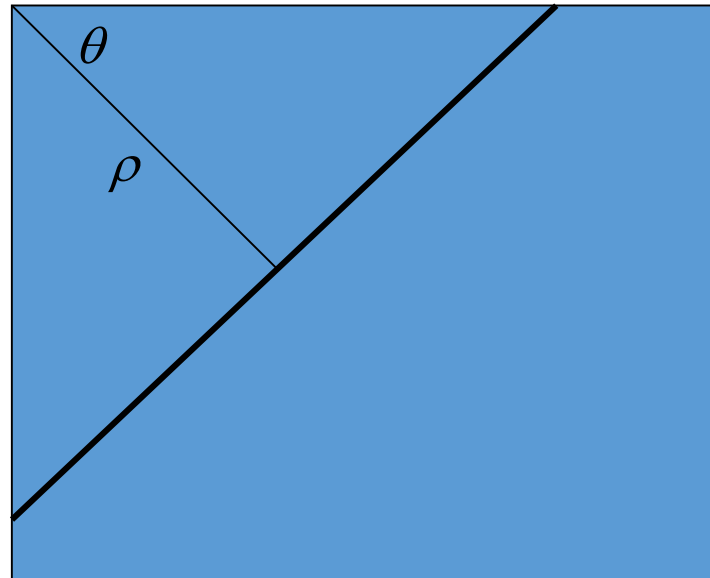
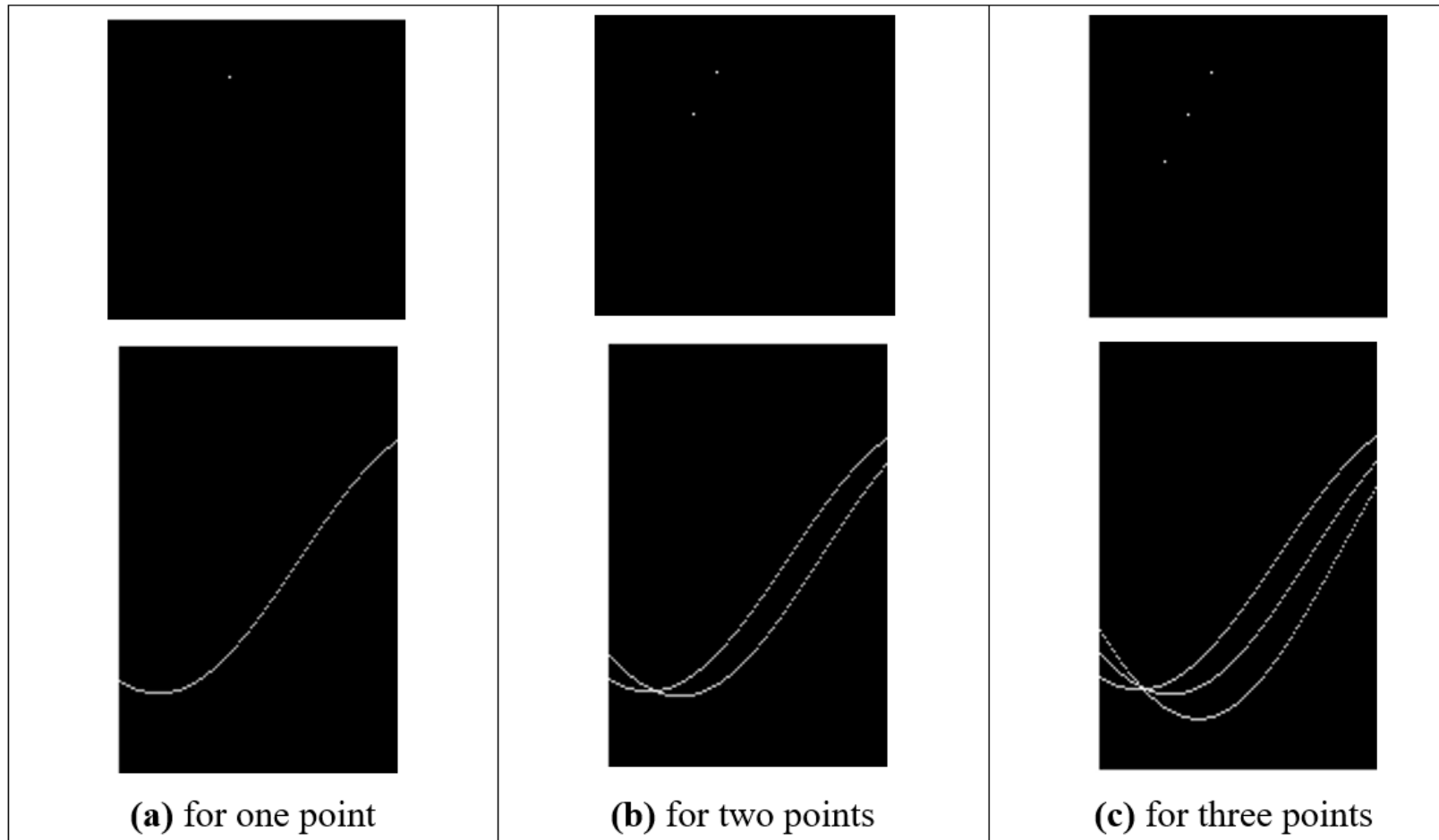


Image containing line



Images and the accumulator space of the polar Hough transform



Applying the Hough transform

