HOUGH TRANSFORM

Circles

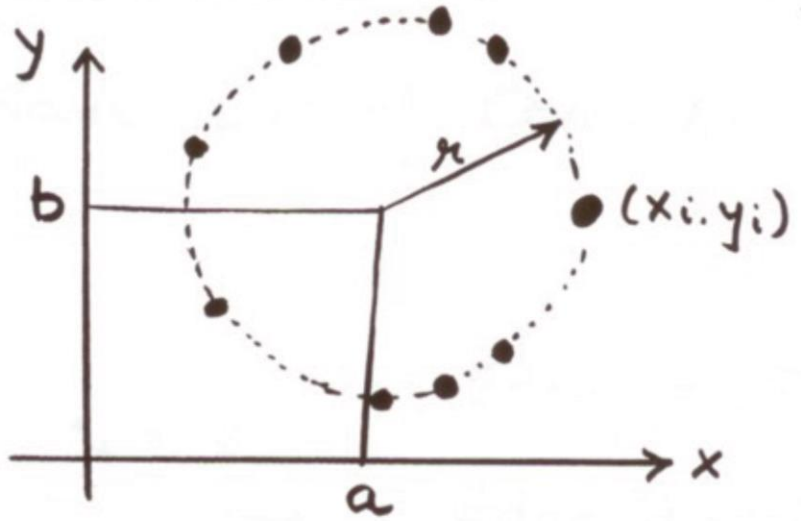
Finding Circles by Hough Transform

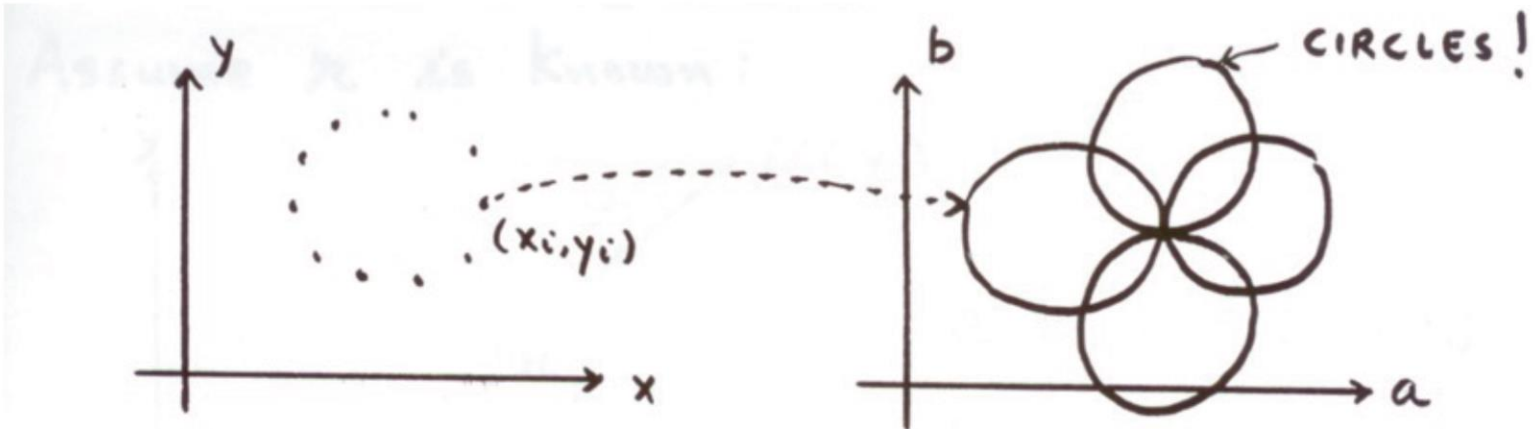
Equation of Circle:

$$(x_i - a)^2 + (y_i - b)^2 = r^2$$

If radius is known: (2D Hough Space)

Accumulator Array $A(a,b)$





parameters

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

variables

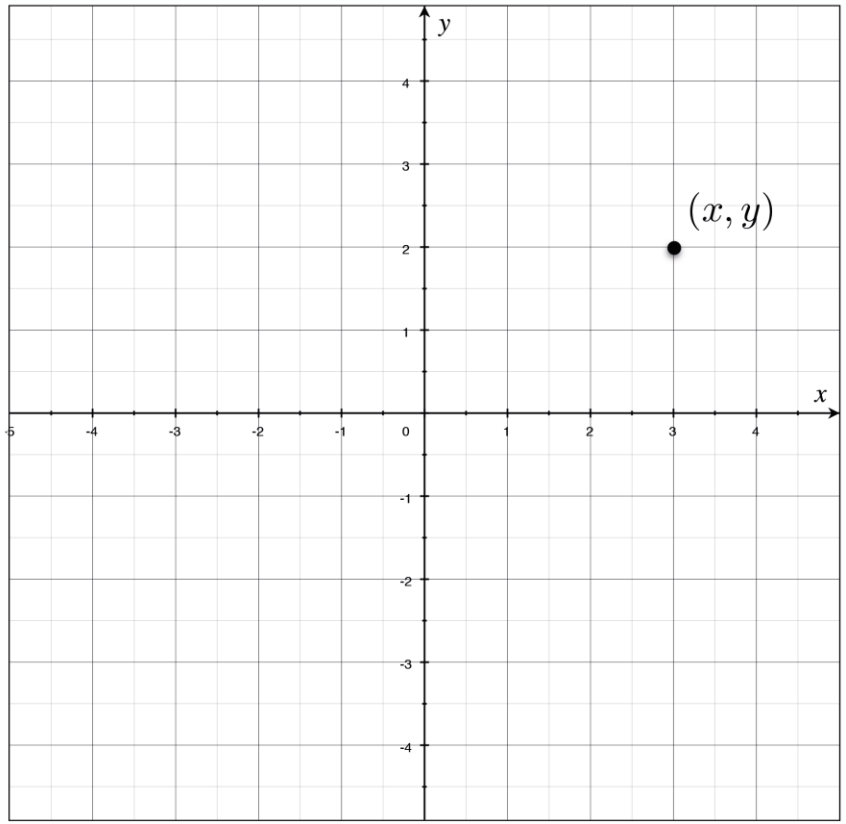
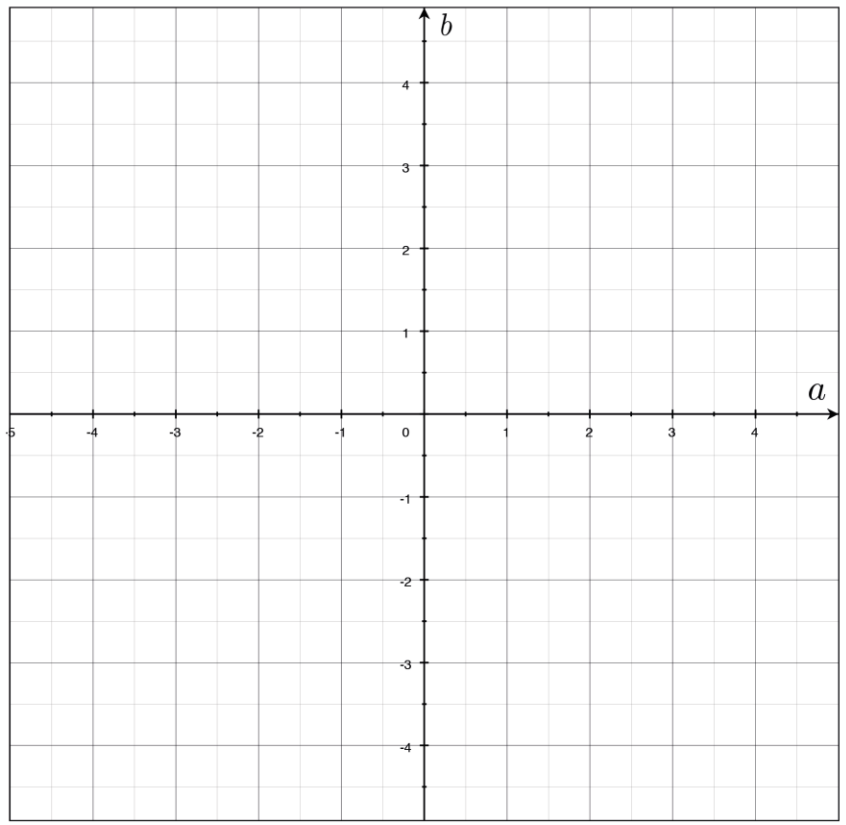


Image space

parameters

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

variables



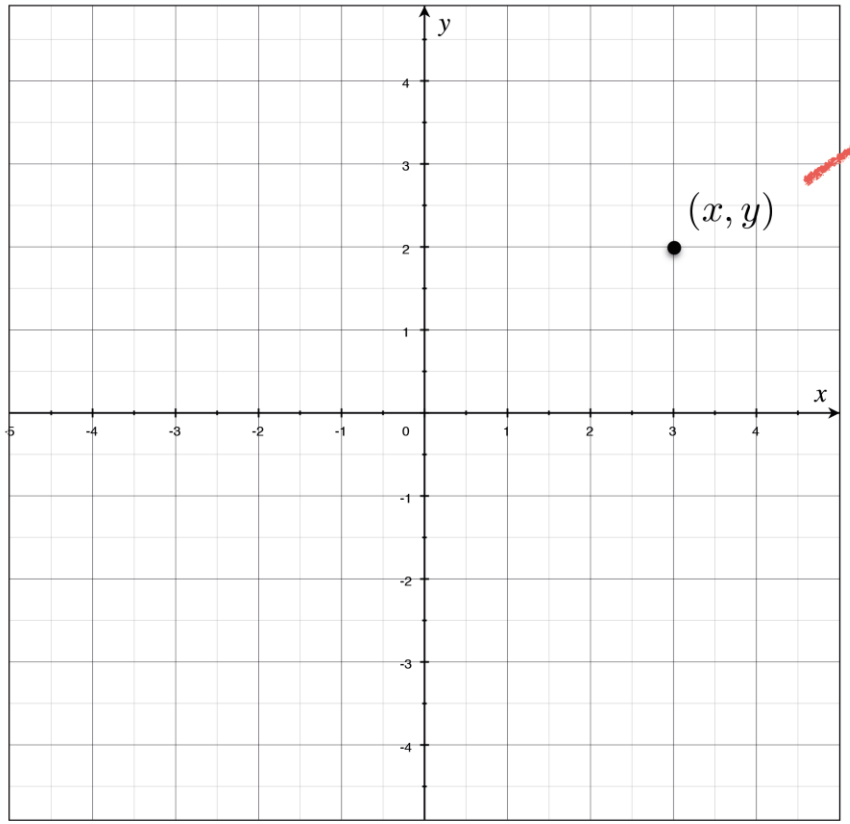
Parameter space

tekomoro

parameters

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

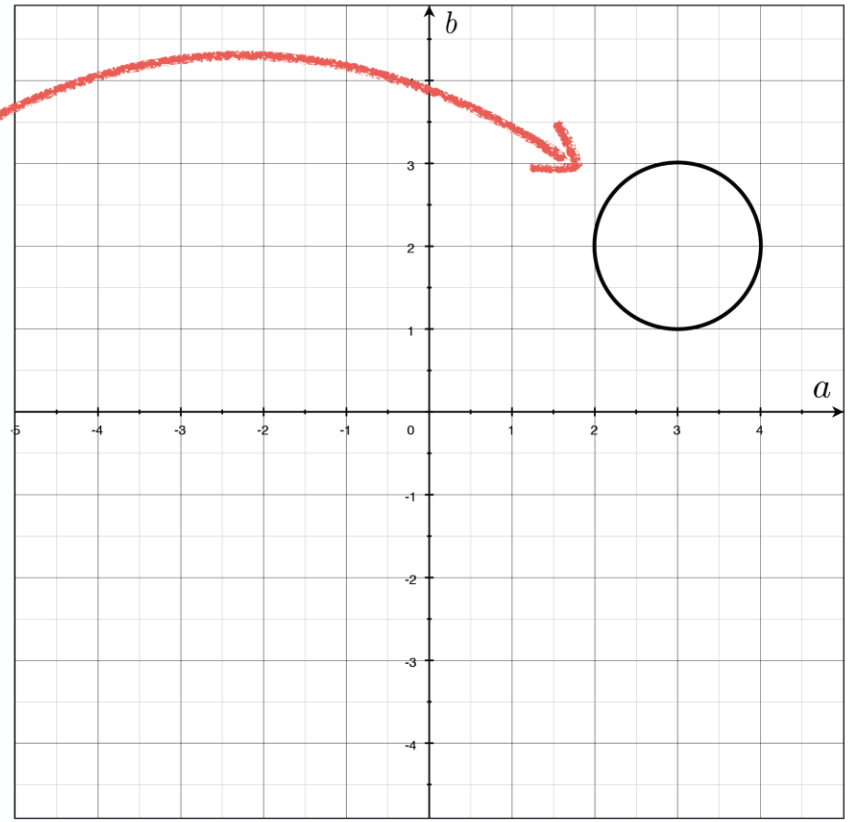
variables



parameters

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

variables

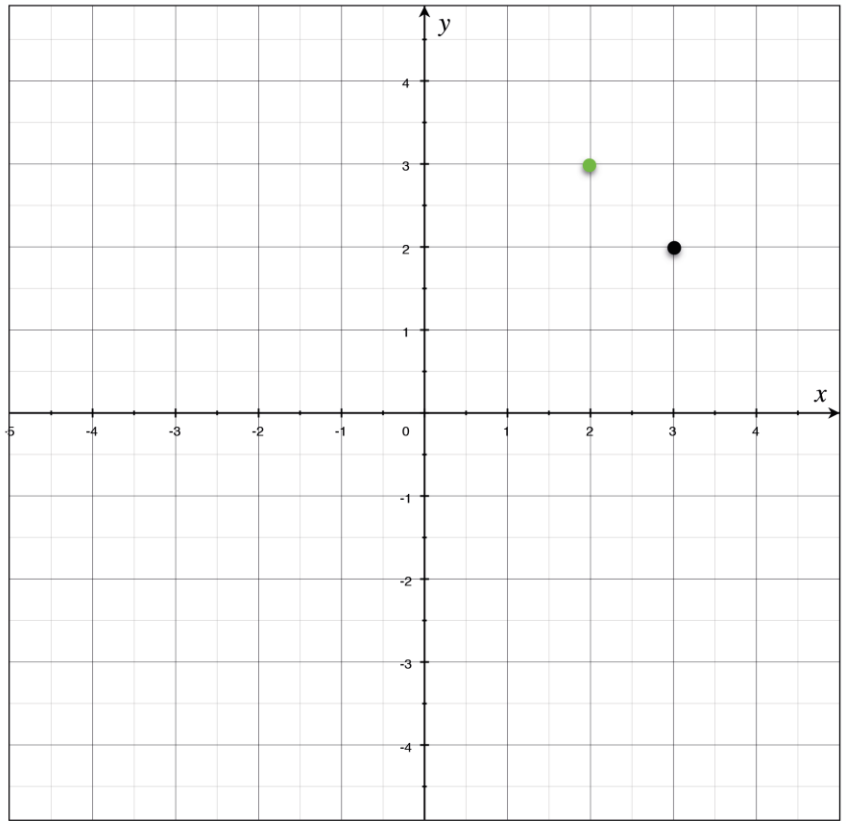


tekomoro

parameters

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

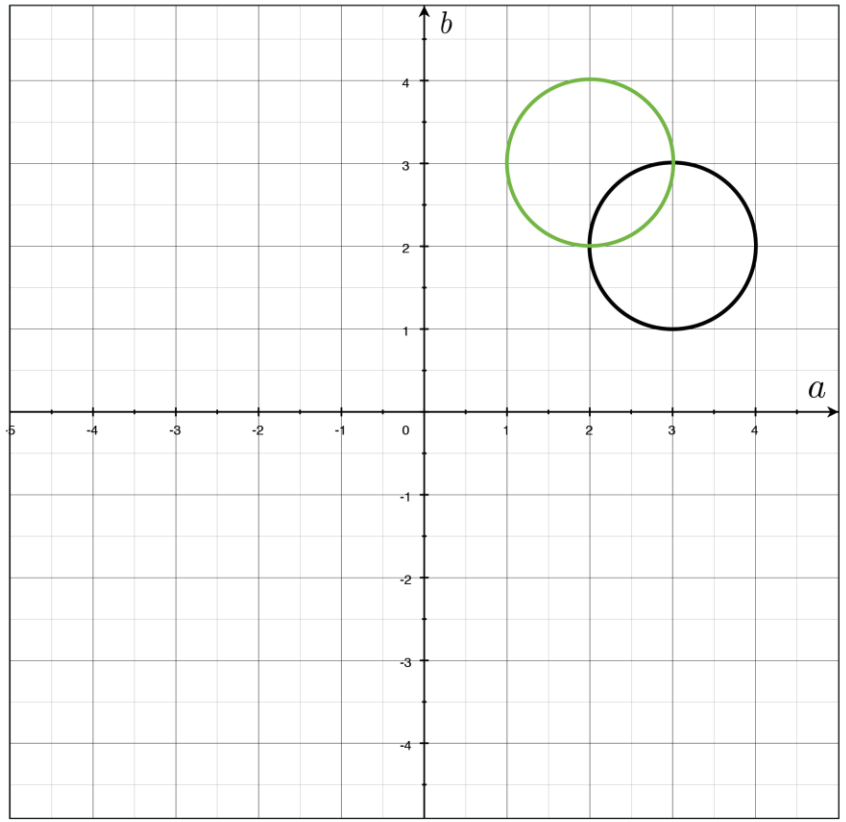
variables



parameters

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

variables

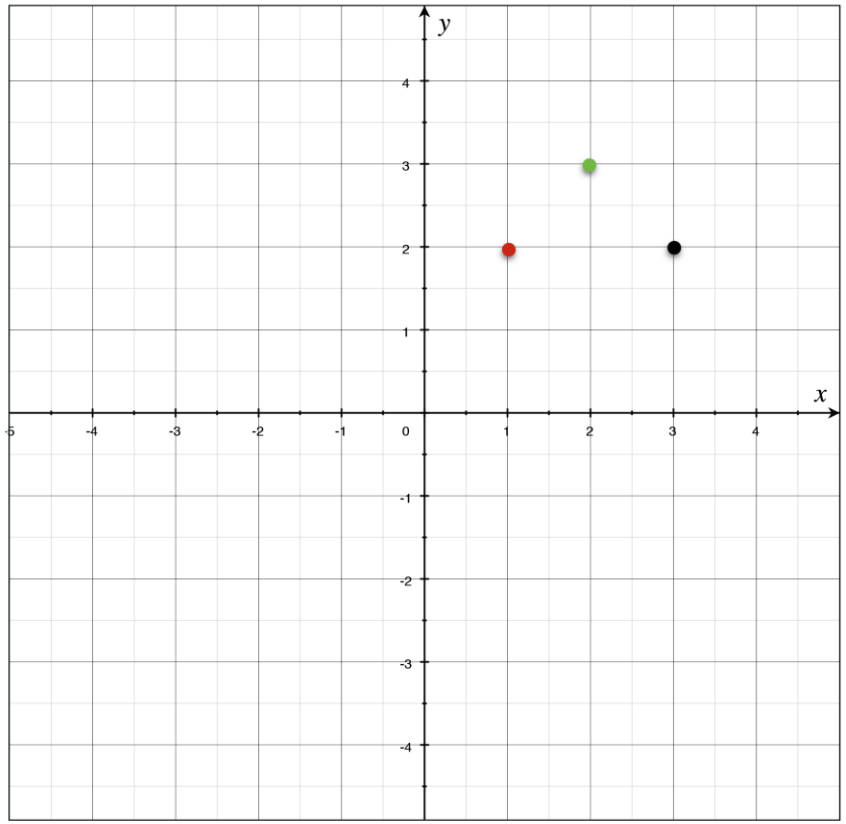


tekomoro

parameters

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

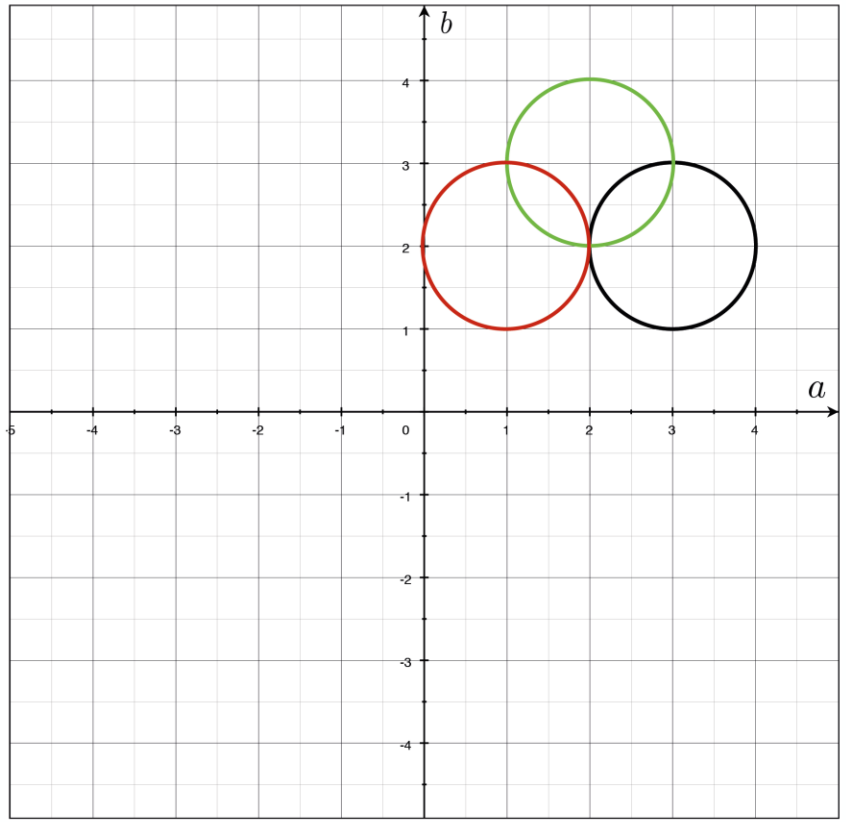
variables



parameters

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

variables

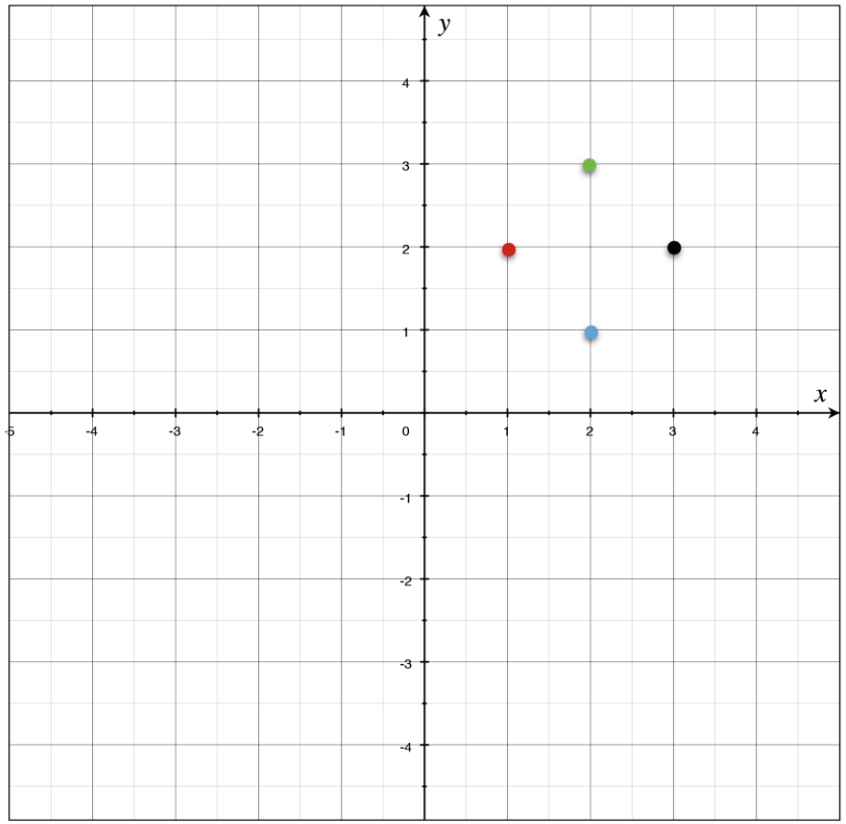


tekomoro

parameters

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

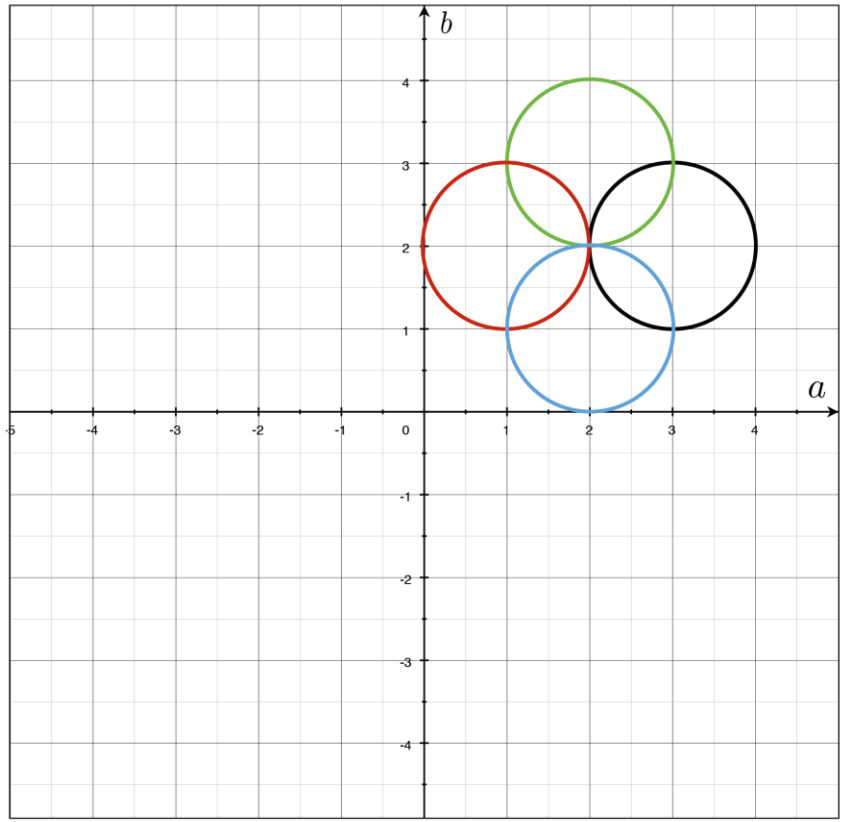
variables



parameters

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

variables



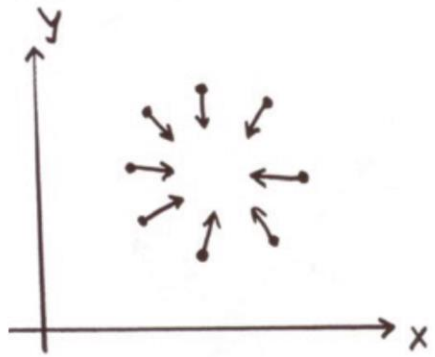
tekomoro

Using Gradient Information

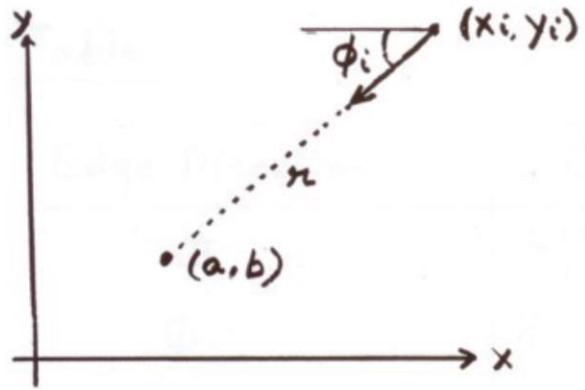
Gradient information can save lot of computation:

Edge Location (x_i, y_i)

Edge Direction ϕ_i



Assume radius is known:



$$a = x - r\cos\phi$$

$$b = y - r\sin\phi$$

Need to increment only one point in accumulator!!

tekomoro

parameters

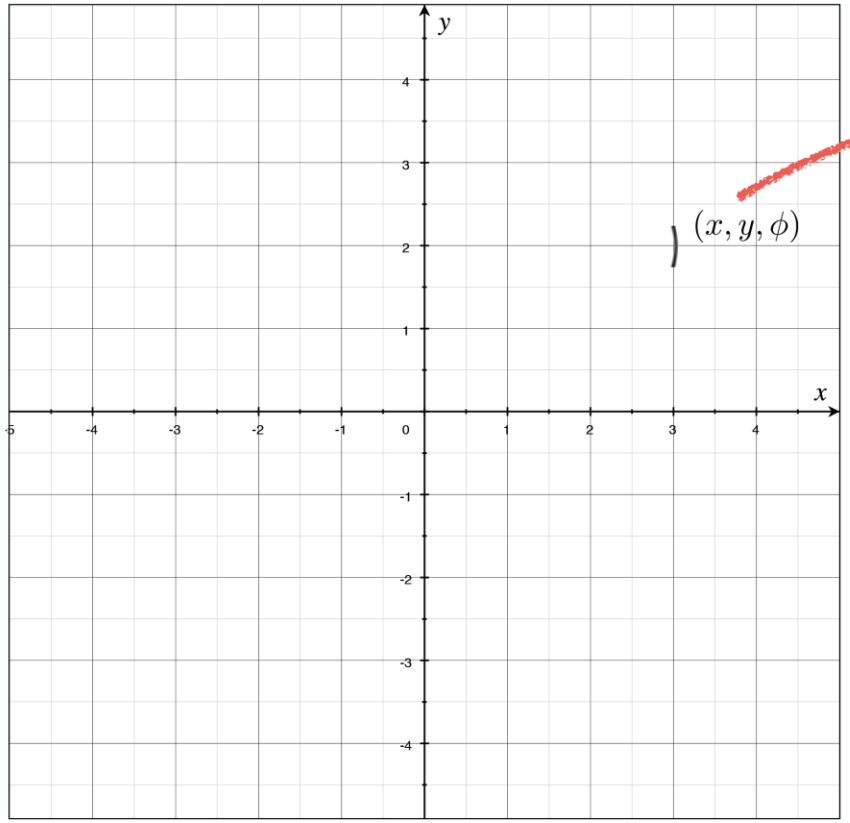
$$(x-a)^2+(y-b)^2=r^2$$

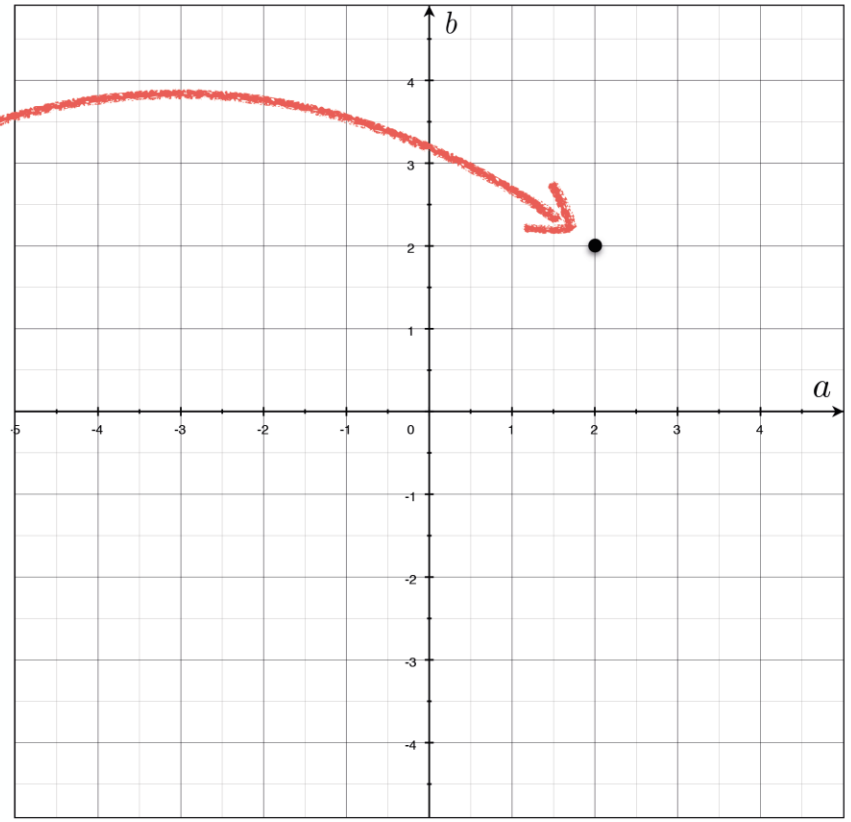
variables

parameters

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

variables



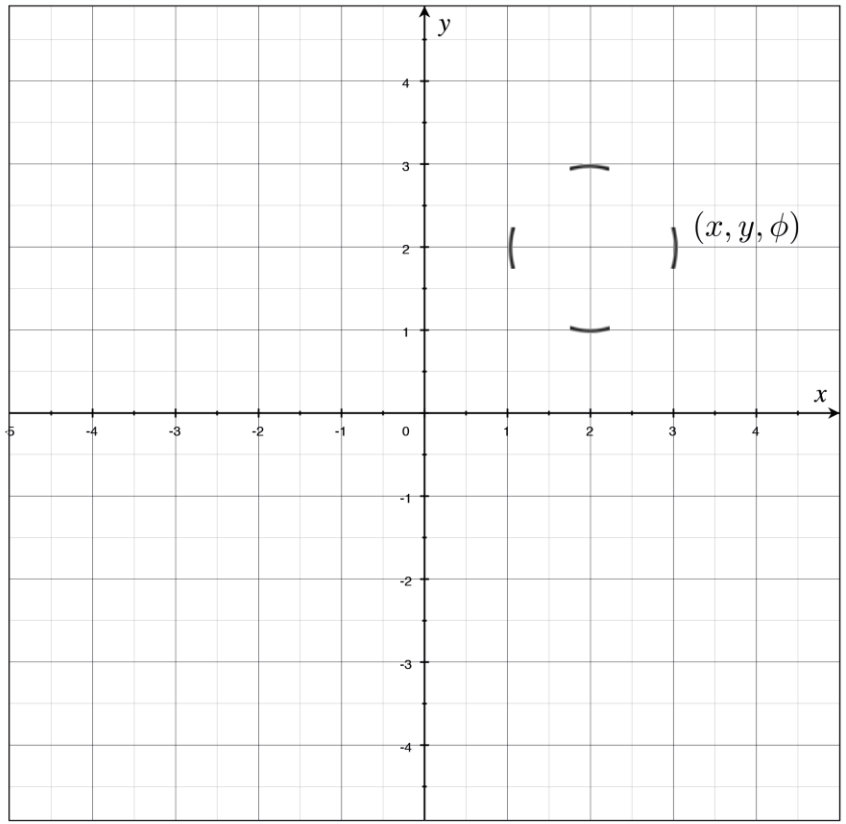


tekomoro

parameters

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

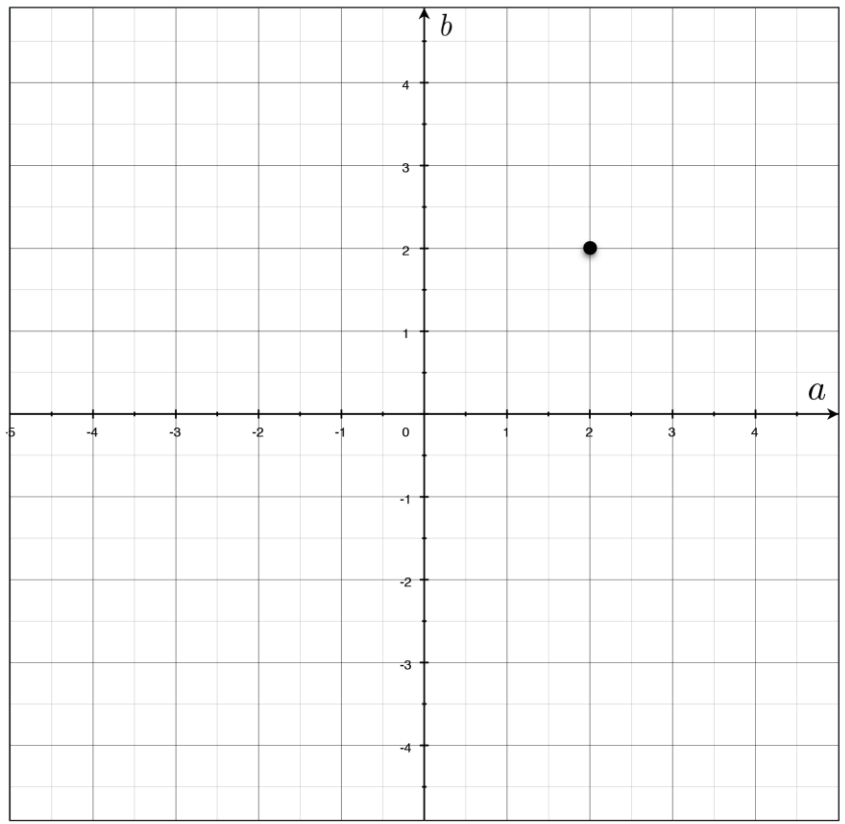
variables



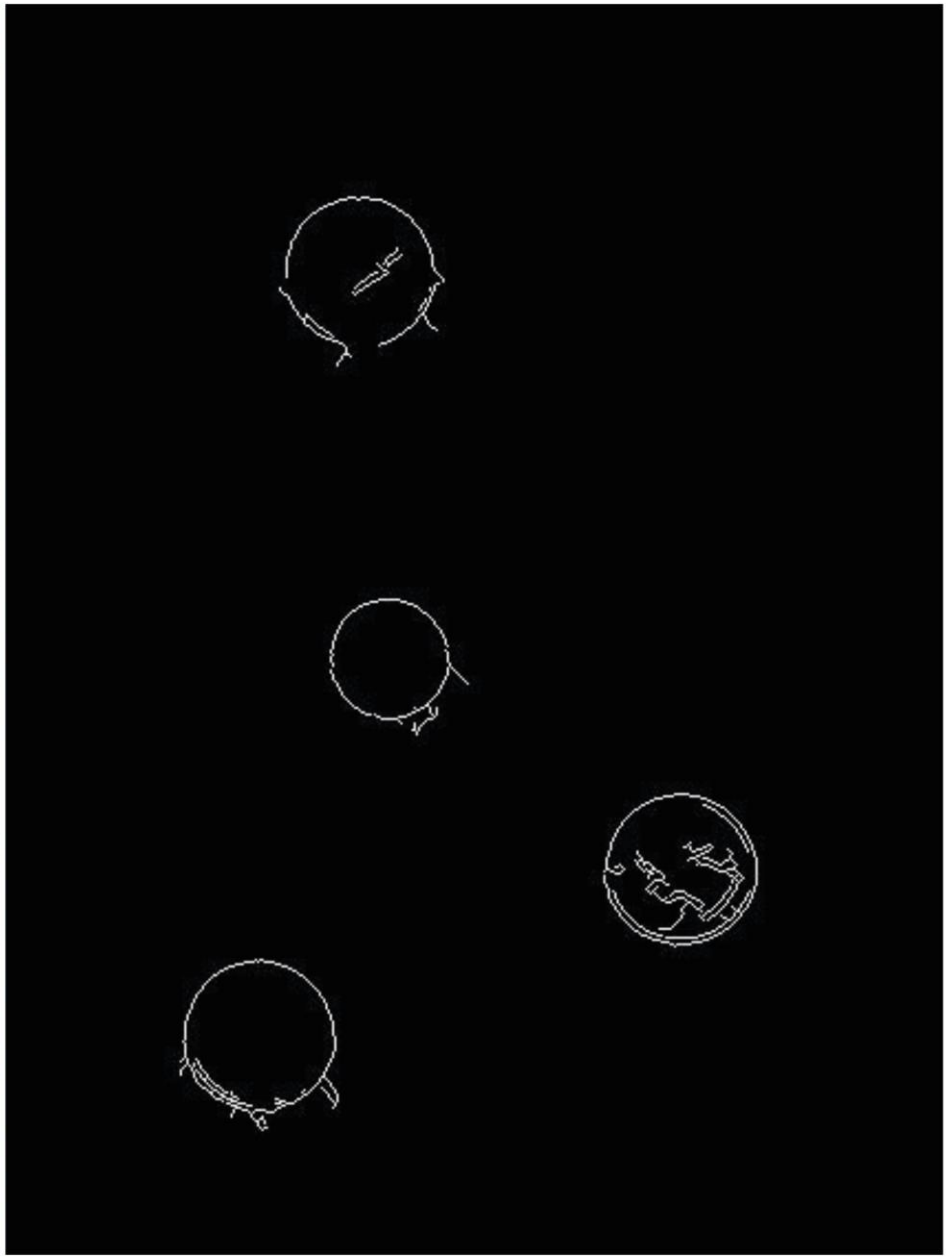
parameters

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

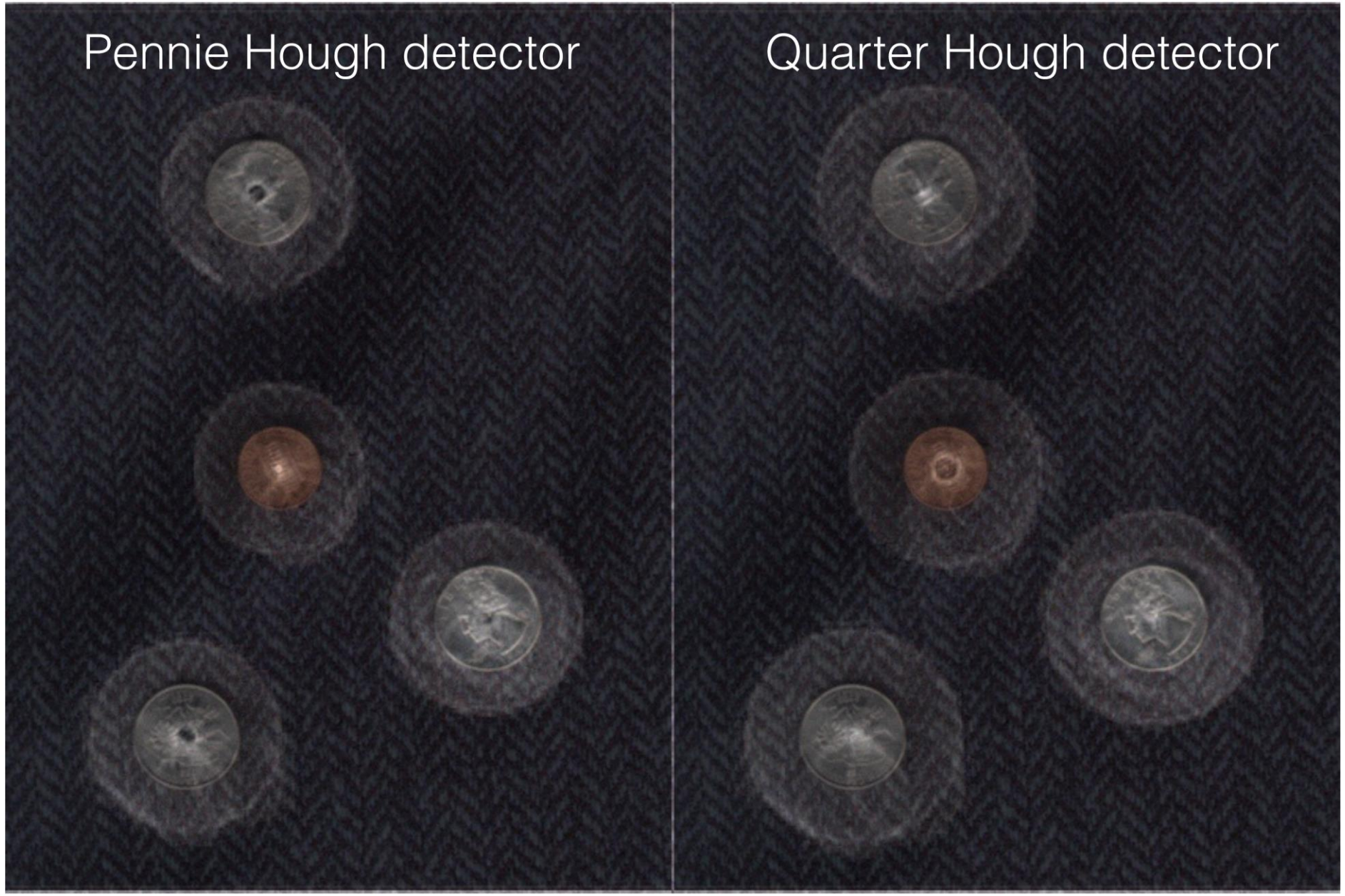
variables



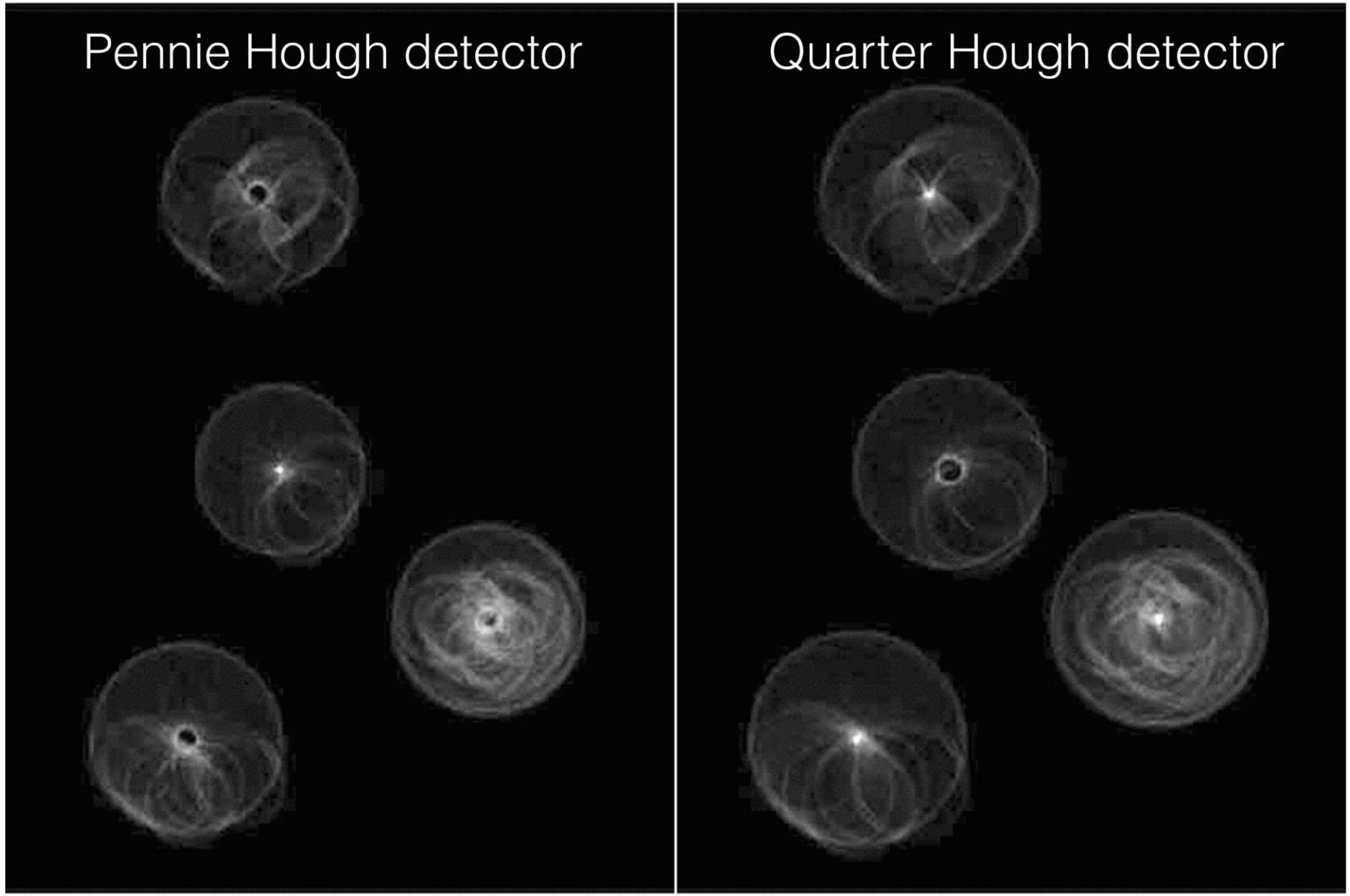
tekomoro



tekomoro



tekomoro

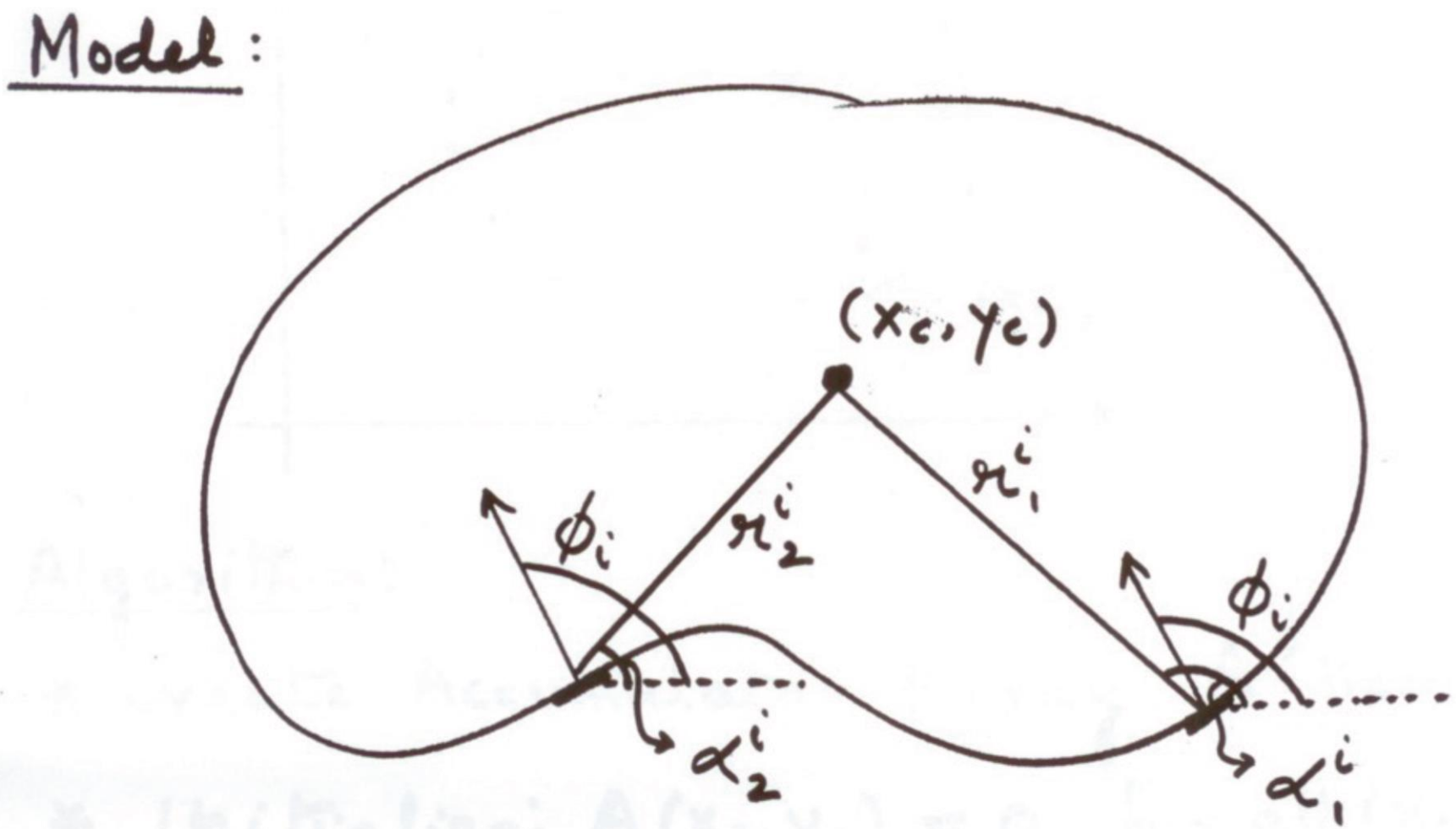


tekomoro

Can you use Hough Transforms for other objects,
beyond lines and circles?

tekomoro

Generalized Hough Transform



tekomoro

ϕ - Table

Edge Direction	$\bar{r} = (r, \alpha)$
ϕ_1	$\bar{r}^1_1, \bar{r}^1_2, \bar{r}^1_3$
ϕ_2	$\bar{r}^2_1, \bar{r}^2_2$
ϕ_i	$\bar{r}^i_1 \vdots \bar{r}^i_2$
ϕ_n	$\bar{r}^n_1, \bar{r}^n_2$

Generalized Hough Transform

Find Object Center (x_c, y_c) given edges (x_i, y_i, ϕ_i)

Create Accumulator Array $A(x_c, y_c)$

Initialize: $A(x_c, y_c) = 0 \quad \forall (x_c, y_c)$

For each edge point (x_i, y_i, ϕ_i)

For each entry $\bar{r}_k^i$ in table, compute:

$$x_c = x_i + r_k^i \cos\alpha_k^i$$

$$y_c = y_i + r_k^i \sin\alpha_k^i$$

Increment Accumulator: $A(x_c, y_c) = A(x_c, y_c) + 1$

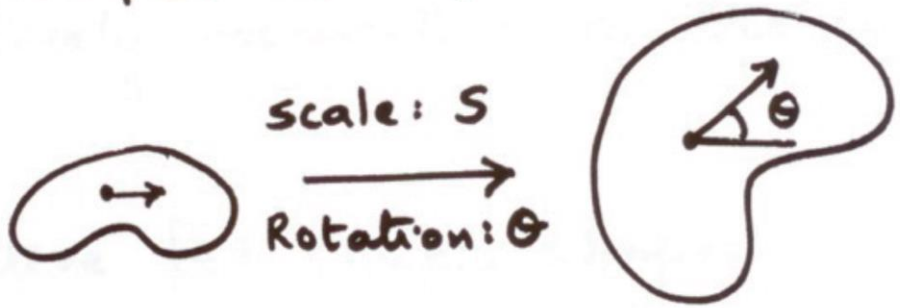
Find Local Maxima in $A(x_c, y_c)$

tekomoro

Scale & Rotation:

Use Accumulator Array:

$$A[x_c, y_c, S, \theta]$$



Use:

$$x_c = x_i + r_k^i S \cos(\alpha_k^i + \theta)$$

$$y_c = y_i + r_k^i S \sin(\alpha_k^i + \theta)$$

$$A(x_c, y_c, S, \theta) = A(x_c, y_c, S, \theta) + 1 .$$

Do you have to use edge detectors
to vote in Hough Space?

tekomoro

A. Train phase:

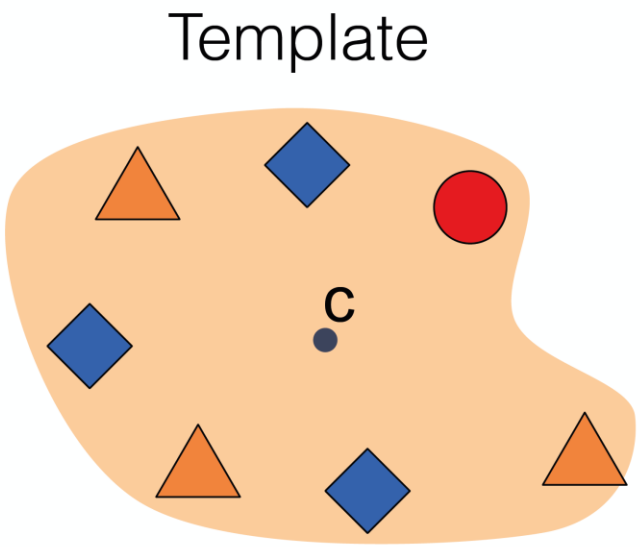
   1. Get features

   2. Store all displacements of feature from center

B. Test phase:

   1. Get features & lookup displacements

   2. Vote for center location



tekomoro

A. Train phase:

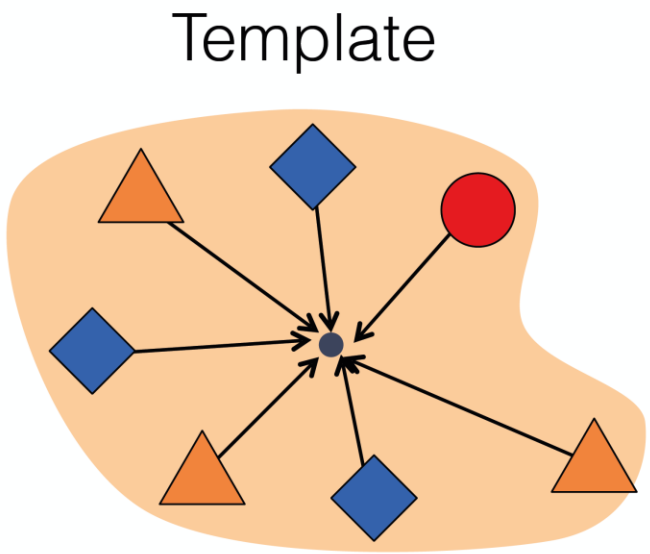
   1. Get features

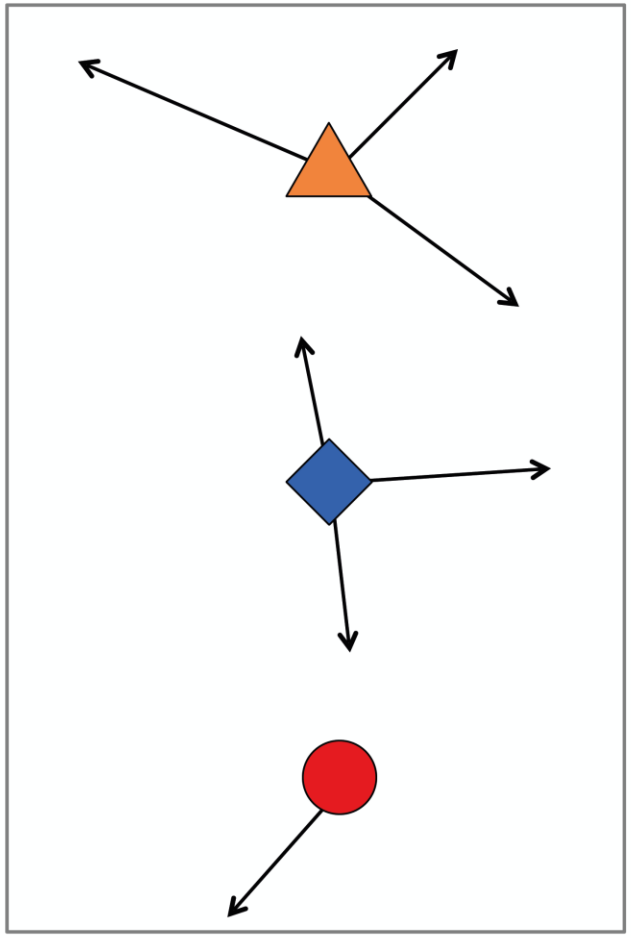
   2. Store all displacements of feature from center

B. Test phase:

   1. Get features & lookup displacements

   2. Vote for center location





tekomoro

A. Train phase:

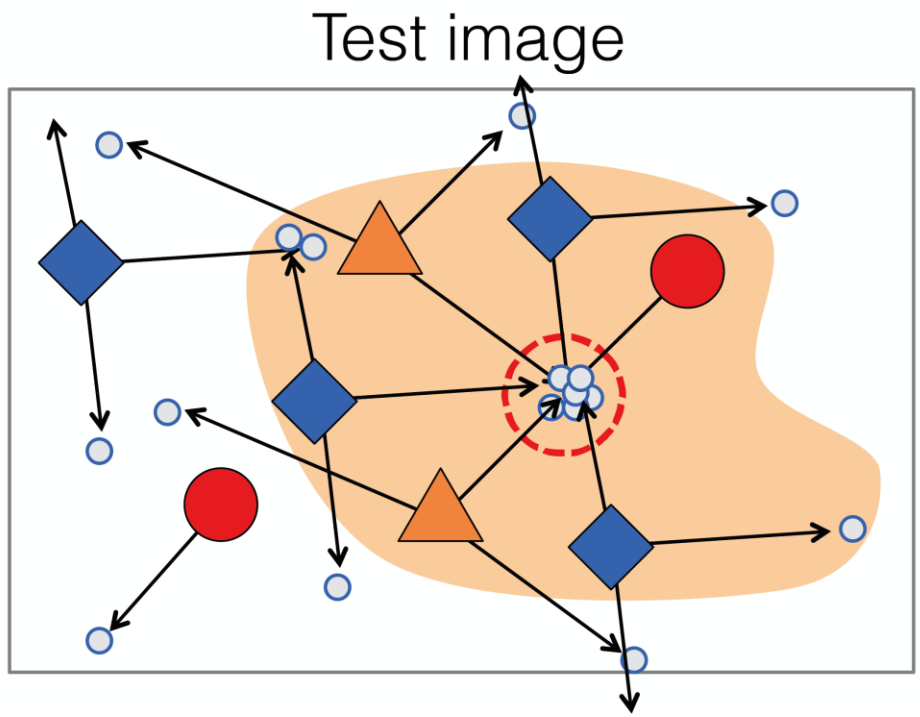
   1. Get features

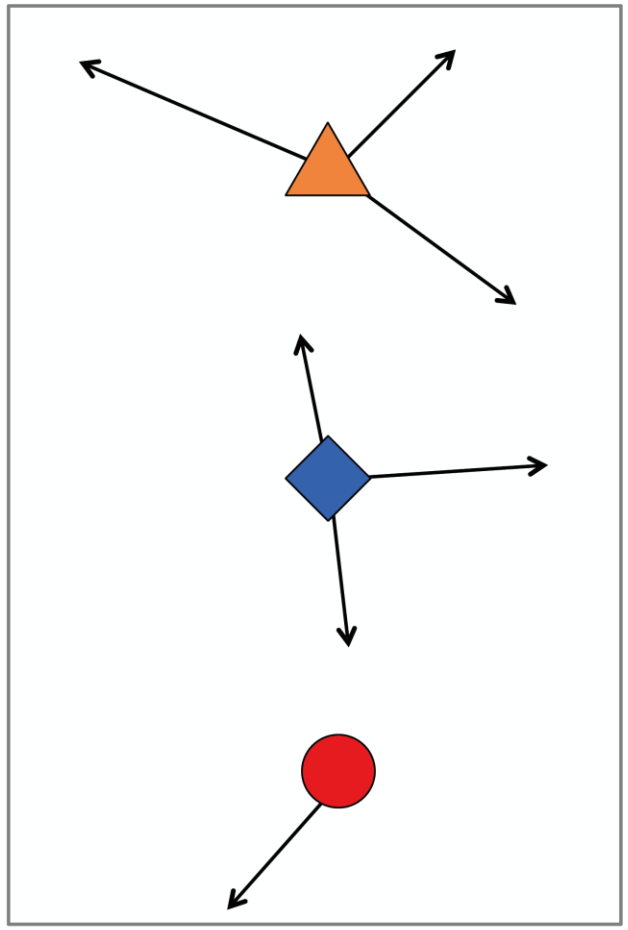
   2. Store all displacements of feature from center

B. Test phase:

   1. Get features & lookup displacements

   2. Vote for center location



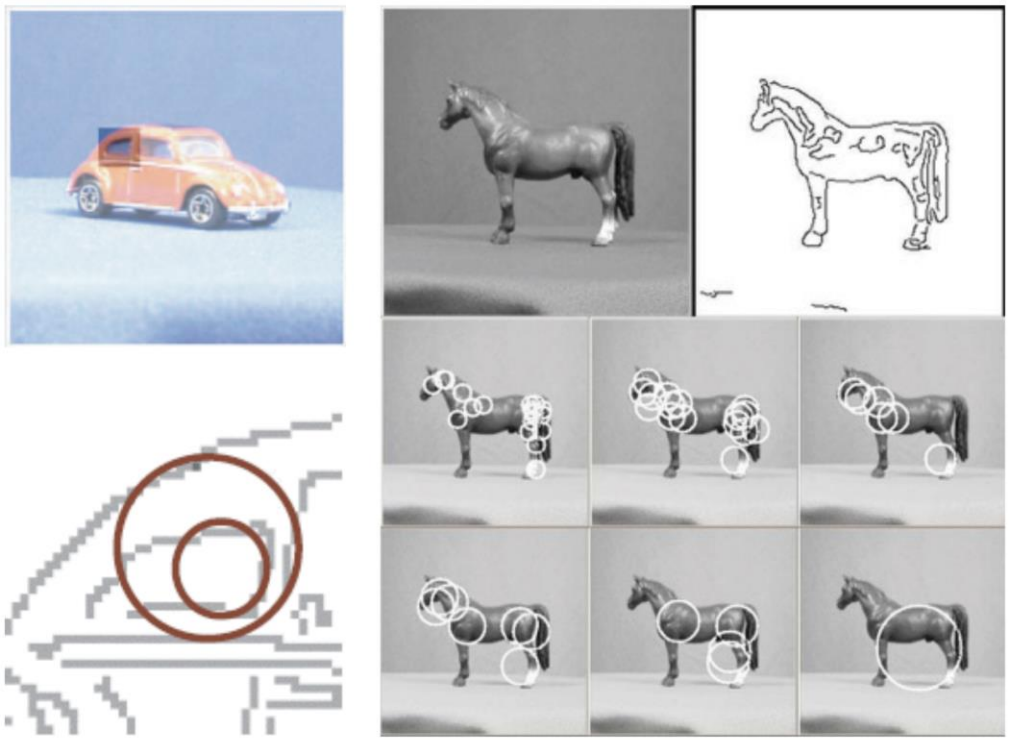


tekomoro

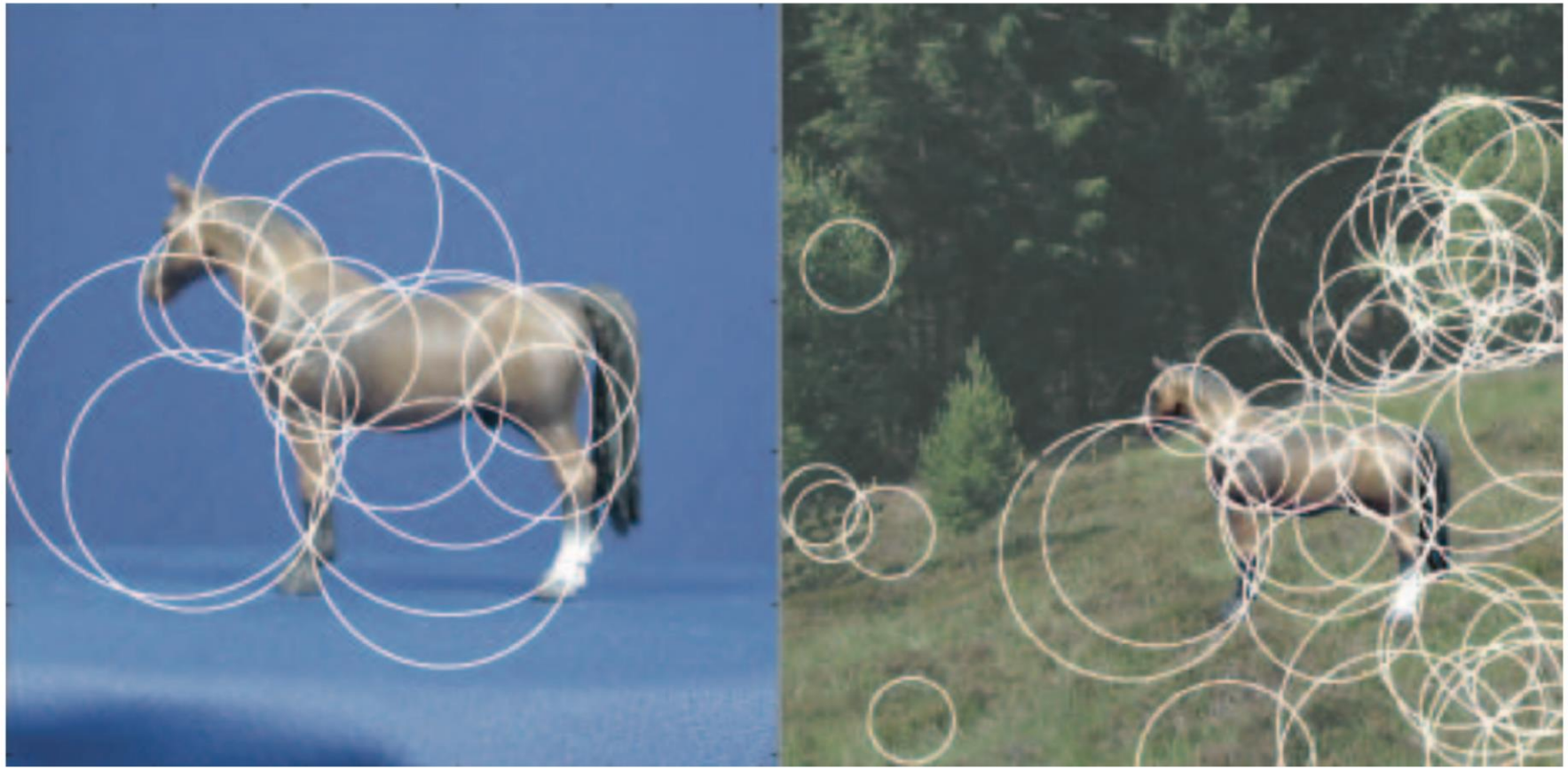
Application of Hough Transforms

tekomoro

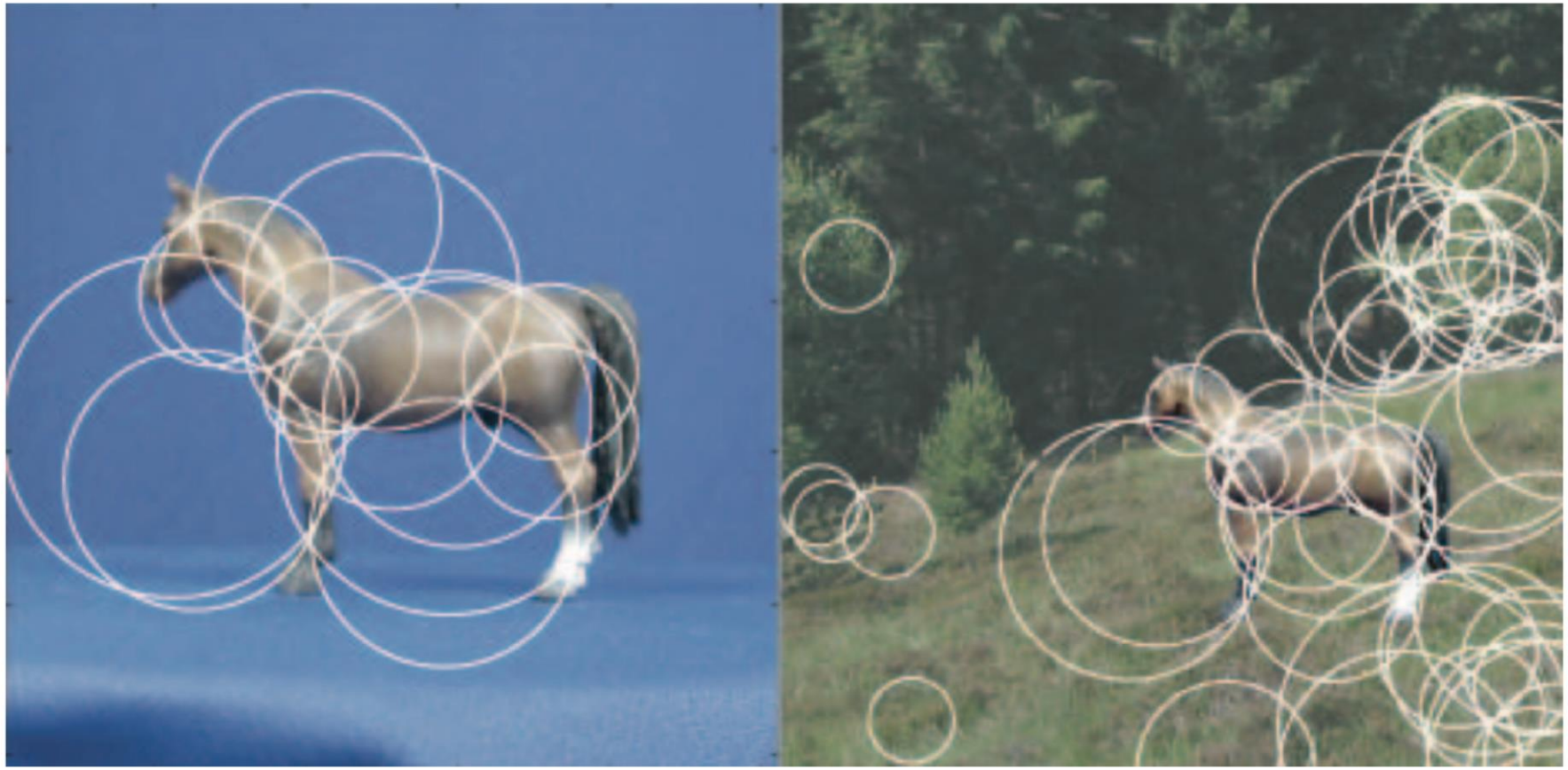
Detecting shape features



F. Jurie and C. Schmid, Scale-invariant shape features for recognition of object categories, CVPR 2004

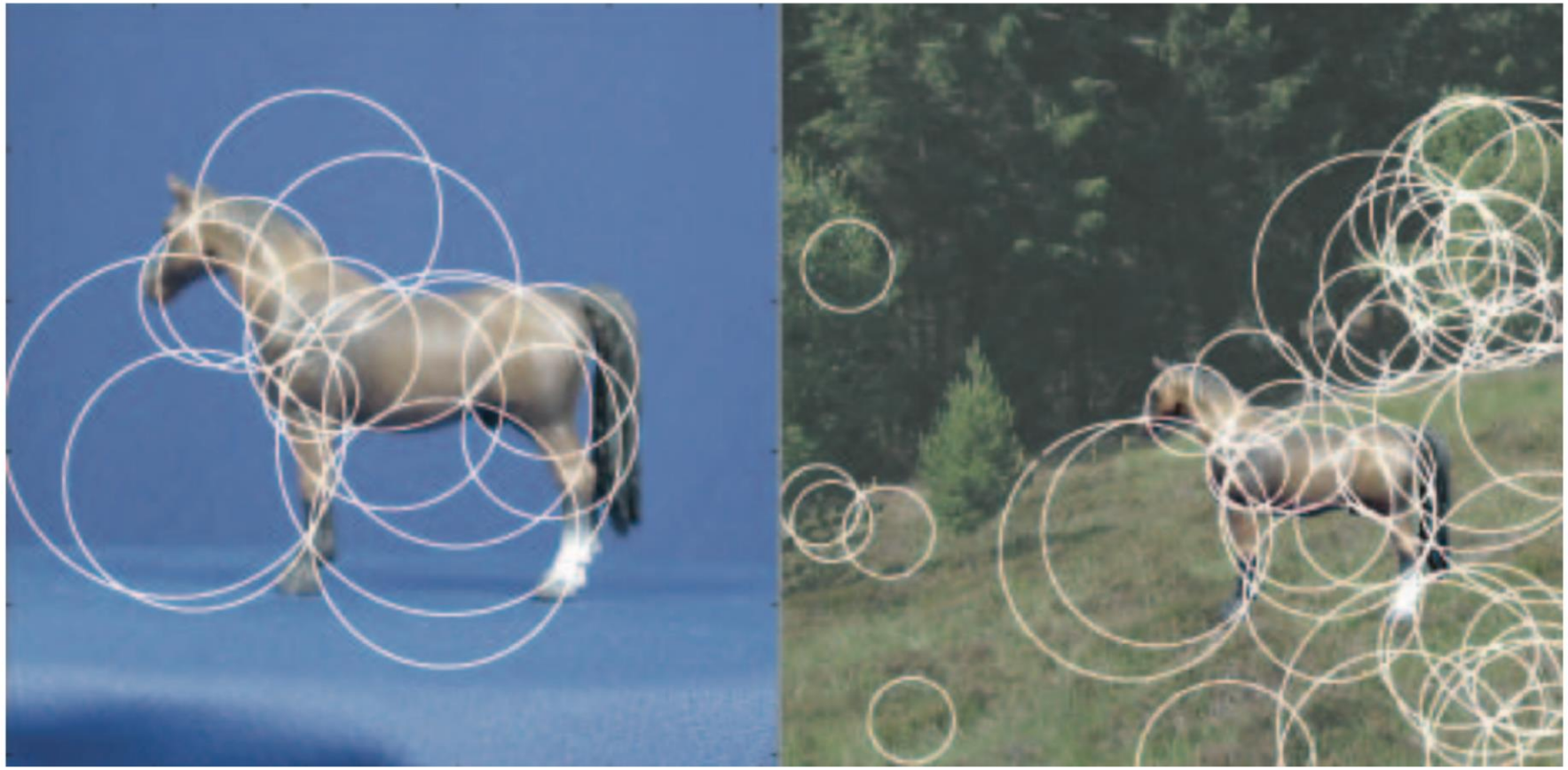


Robustness to scale and clutter



Robustness to scale and clutter

tekomoro



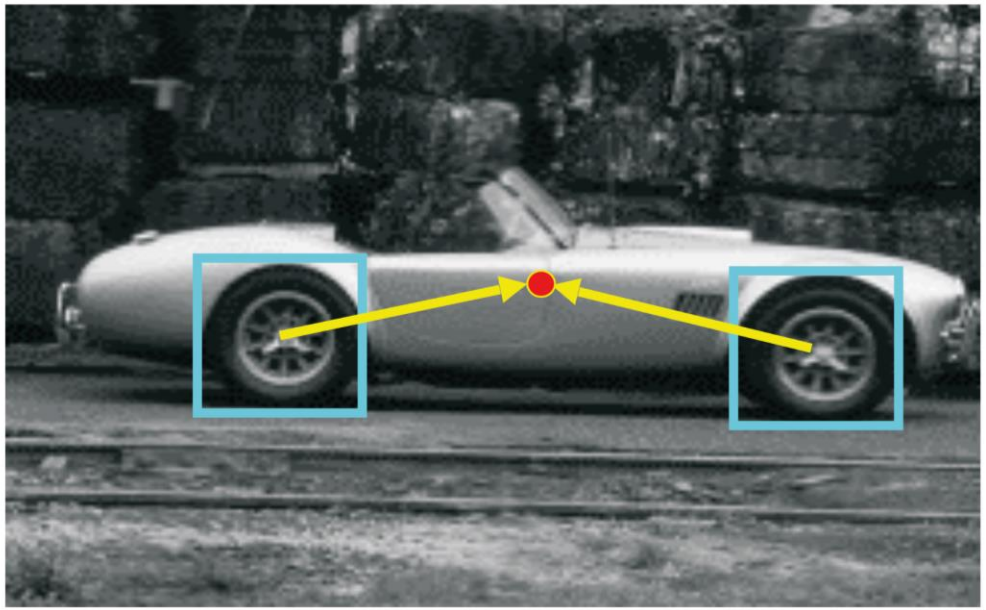
Robustness to scale and clutter

tekomoro

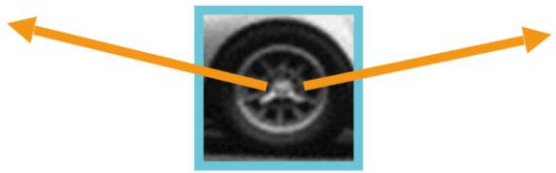
Clutter: When the object and the background are similar.

Object detection

Index displacements by “visual codeword”

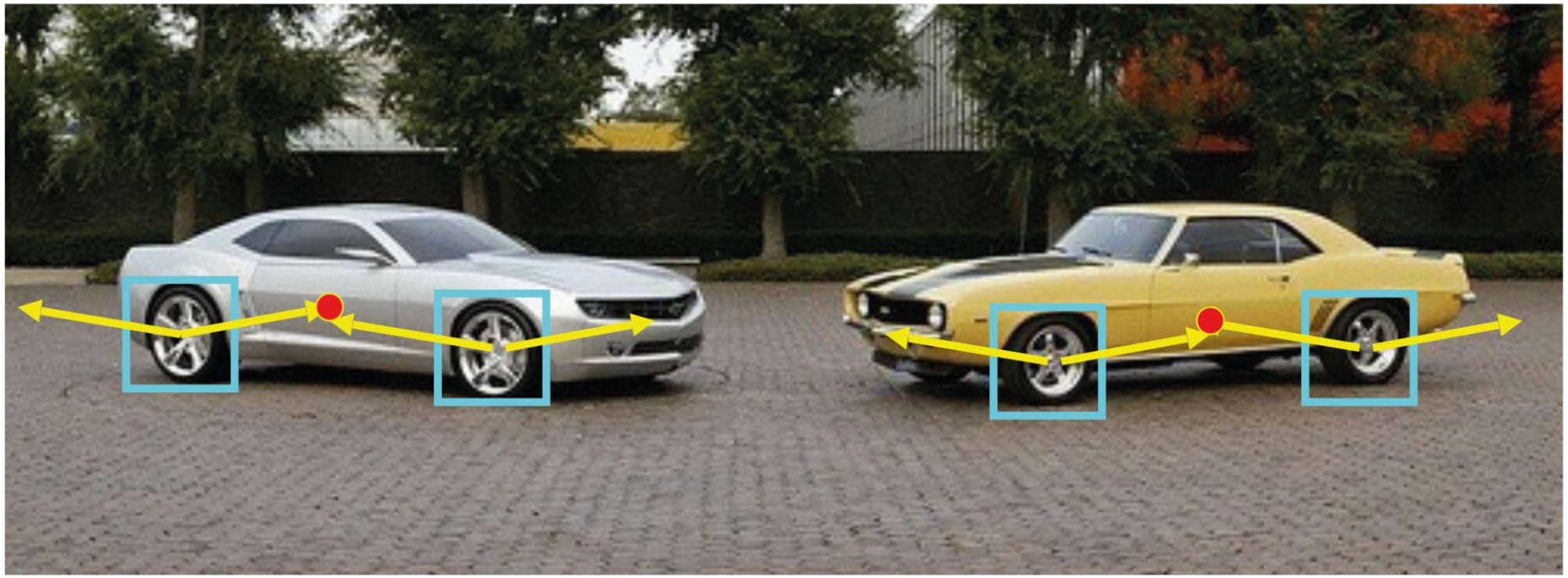


training image



visual codeword with displacement vectors

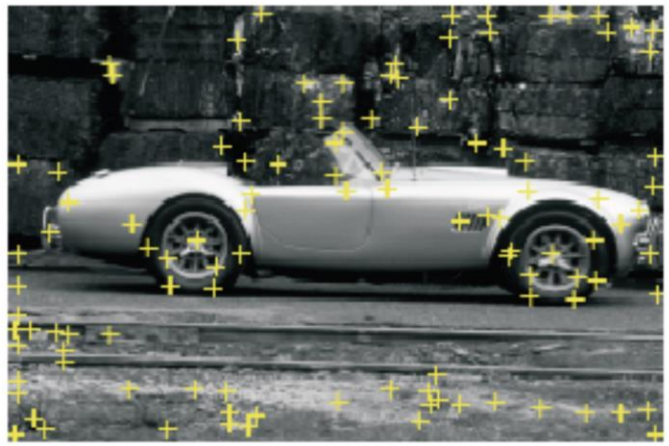
B. Leibe, A. Leonardis, and B. Schiele, Combined Object Categorization and Segmentation with an Implicit Shape Model,
ECCV Workshop on Statistical Learning in Computer Vision 2004

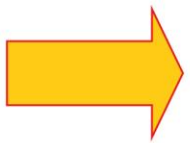


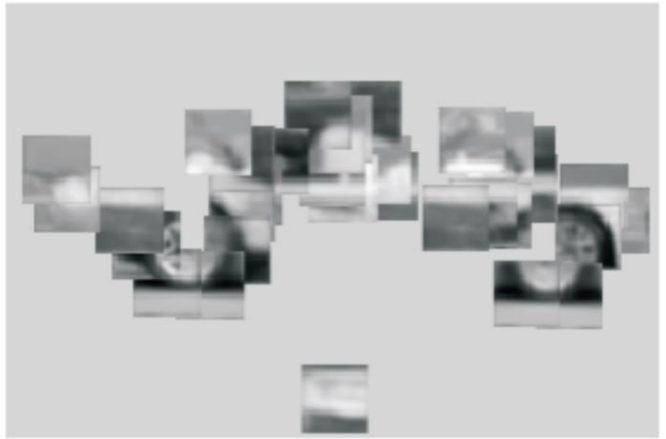
tekomoro

Train phase

1. get features



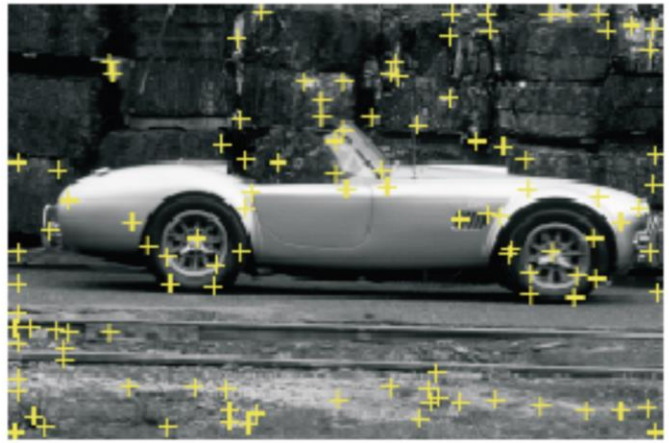


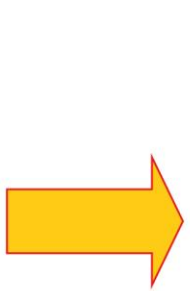


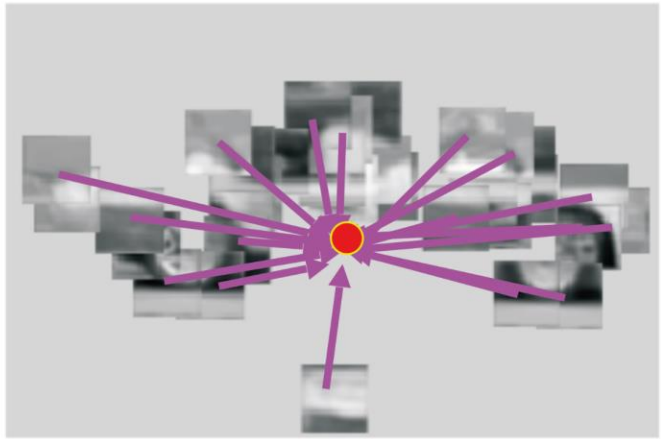
tekomoro

Train phase

2. store displacements

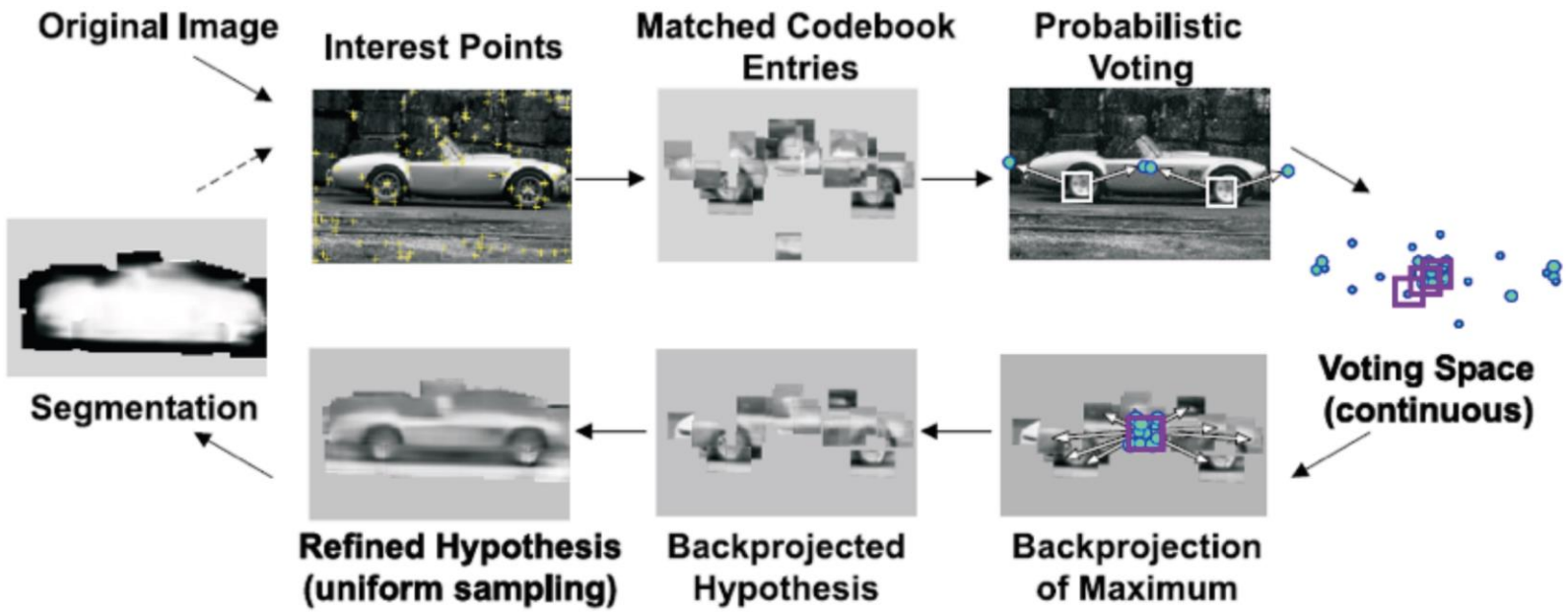




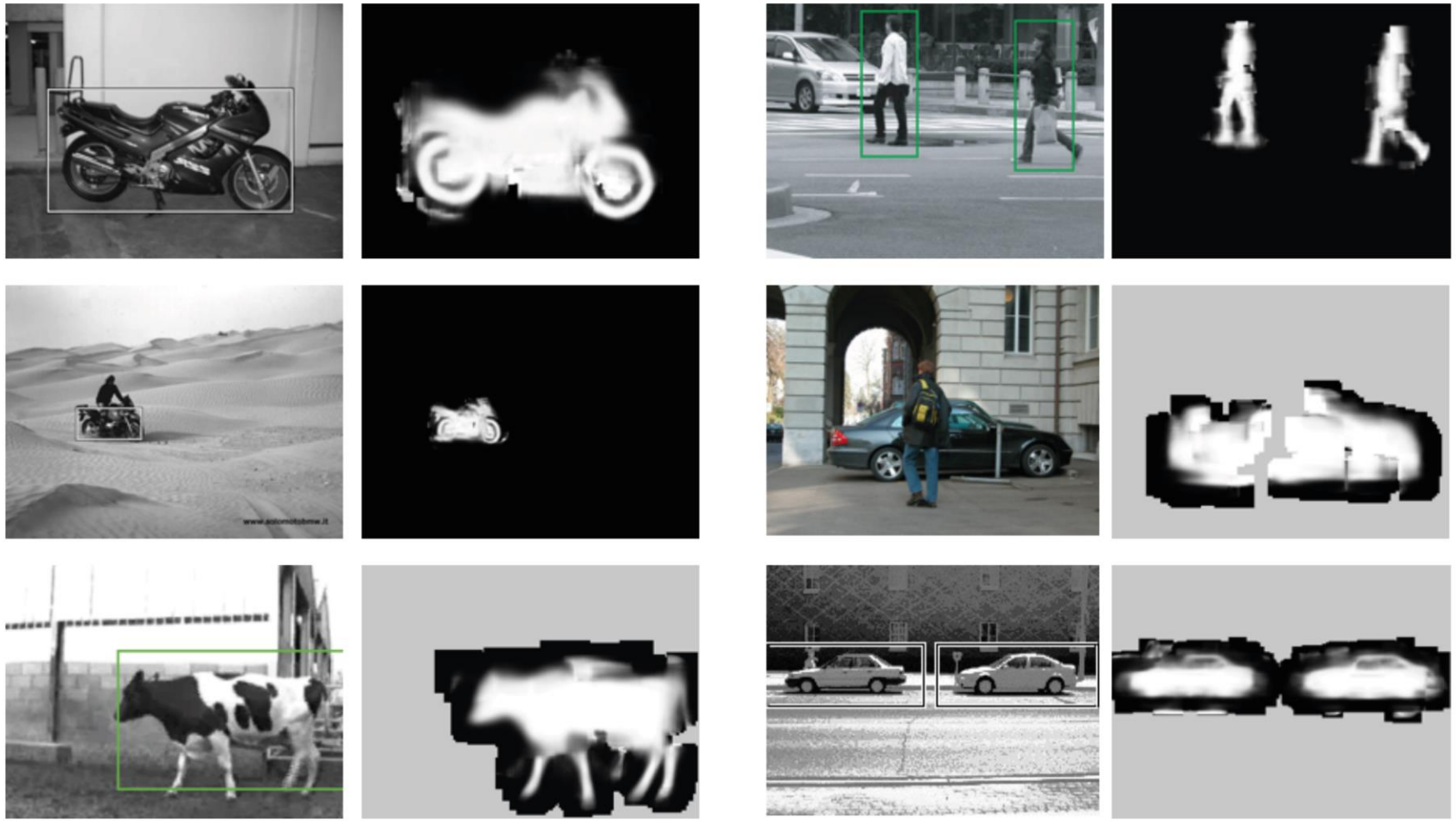


tekomoro

Test phase



tekomoro



tekomoro

The Hough transform …

Deals with occlusion well? 

Detects multiple instances? 

Robust to noise? 

Good computational complexity? 

Easy to set parameters? 

tekomoro

The Hough transform …

Deals with occlusion well? 

Detects multiple instances? 

Robust to noise? 

Good computational complexity? 

Easy to set parameters? 

tekomoro

Occlusion: When only part of the object we need to detect is there in the image.

Thank You

^_^

tekomoro