Fast Soccer Ball Detection using Deep Learning

Spring 2017

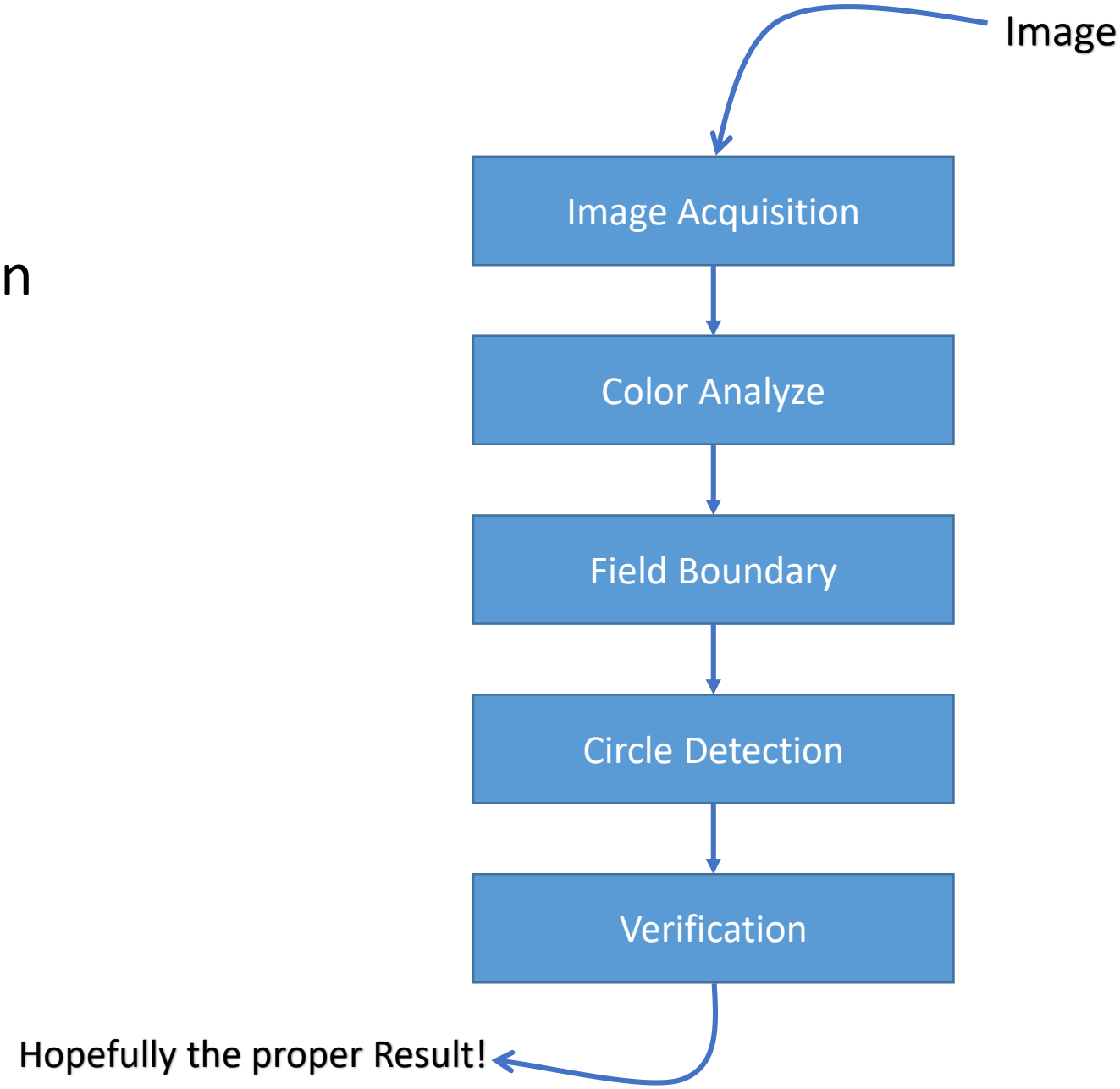
Slides by Aref Moqadam Mehr

Problem Statement

- RoboCup Soccer League
  - NAO
  - HSL
- Field is no longer Color-Coded
- Unspecified Patterns (in HSL)
- NAO white robot in addition to all the problems

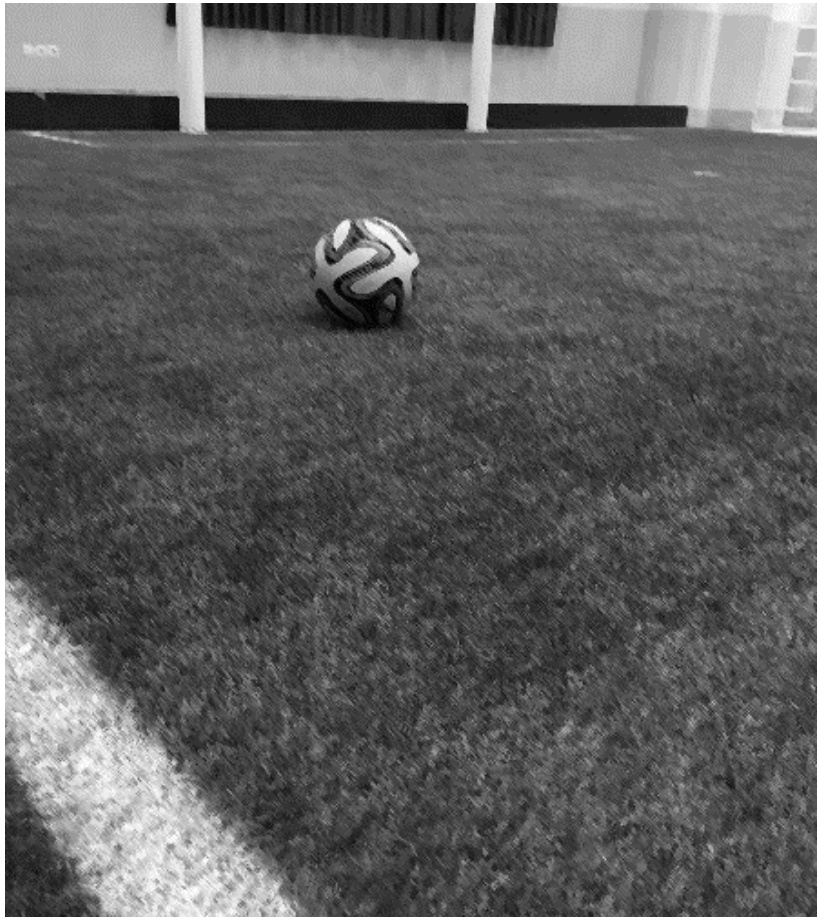
Intuition

- Image Acquisition
- Color Analysis
- Field Boundary
- Circle Detection
- Verification



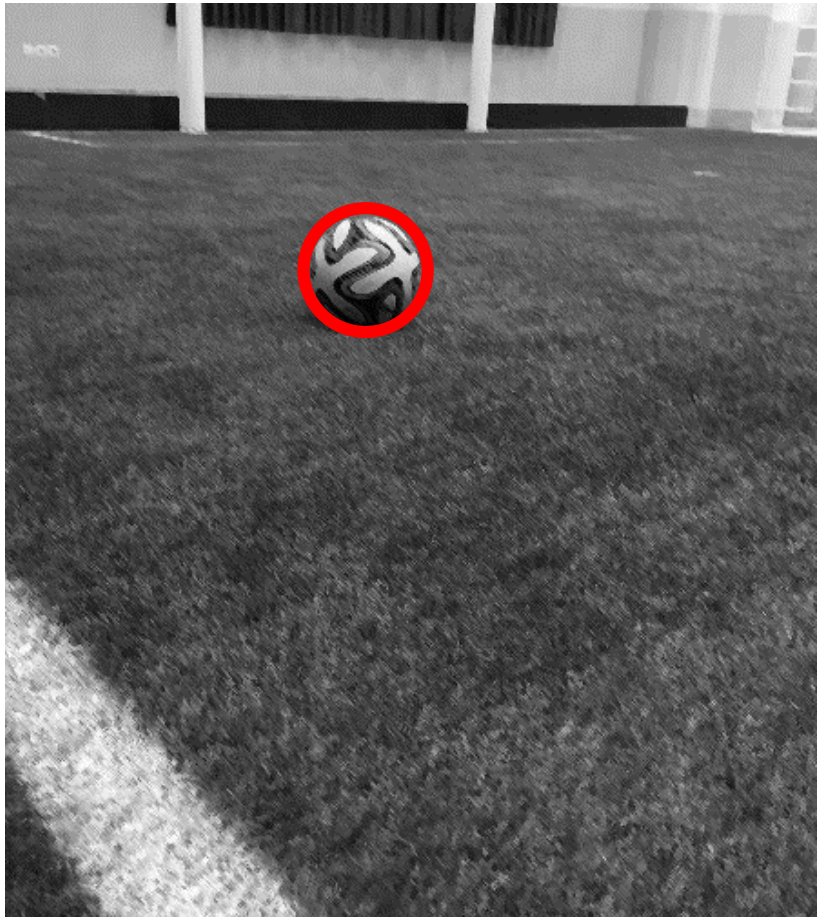
First Things First!

- Circle Detection

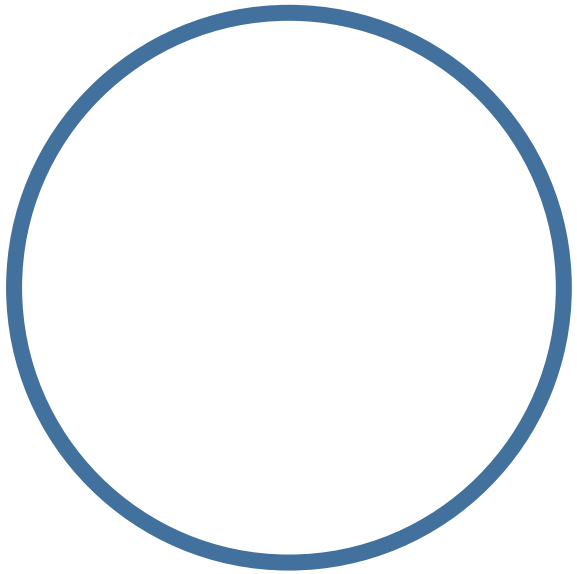


First Things First!

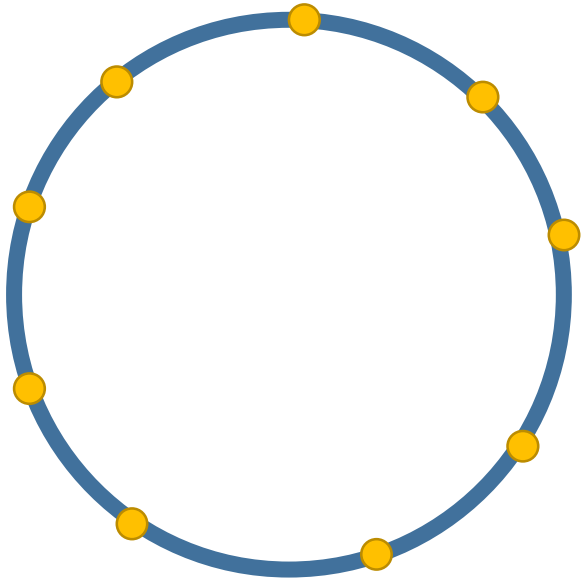
- Circle Detection
  - CHT
  - RHT
  - FRHT



Circle Hough Transform

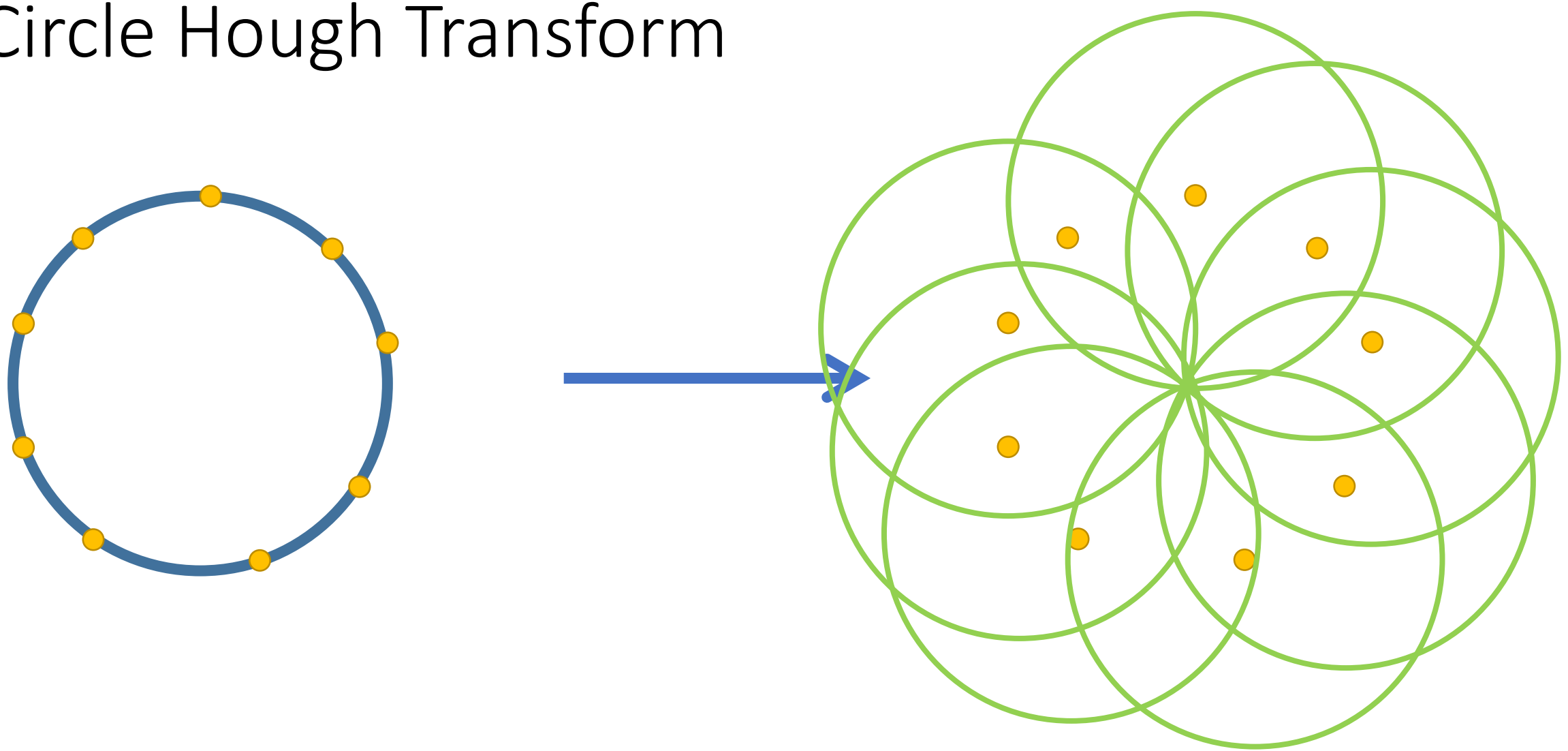


Circle Hough Transform

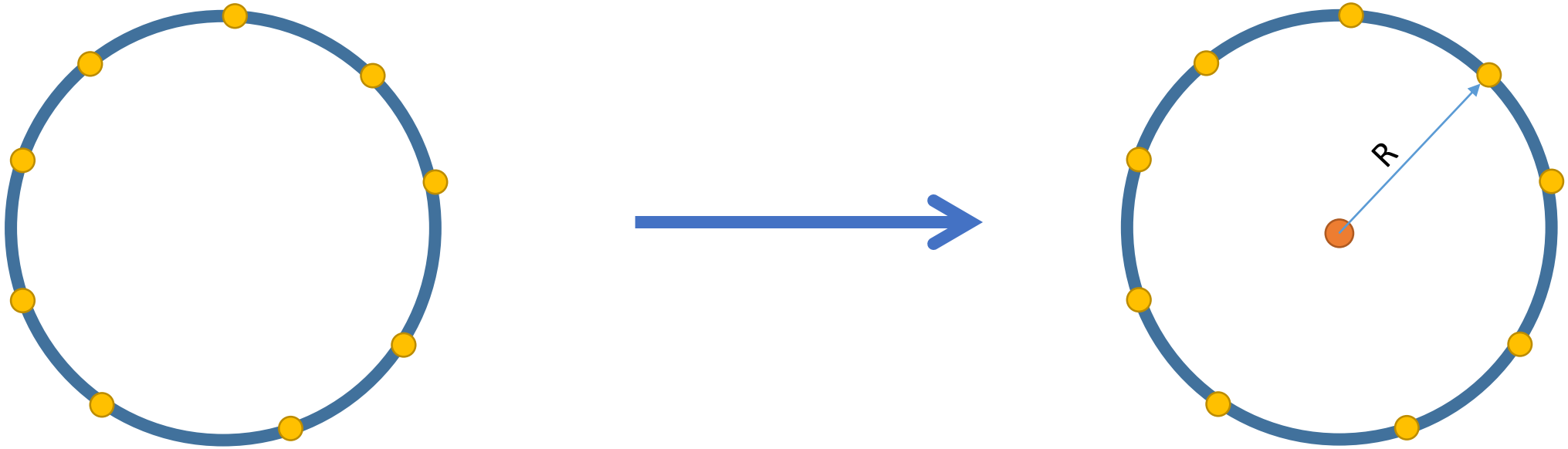


Circle Hough Transform

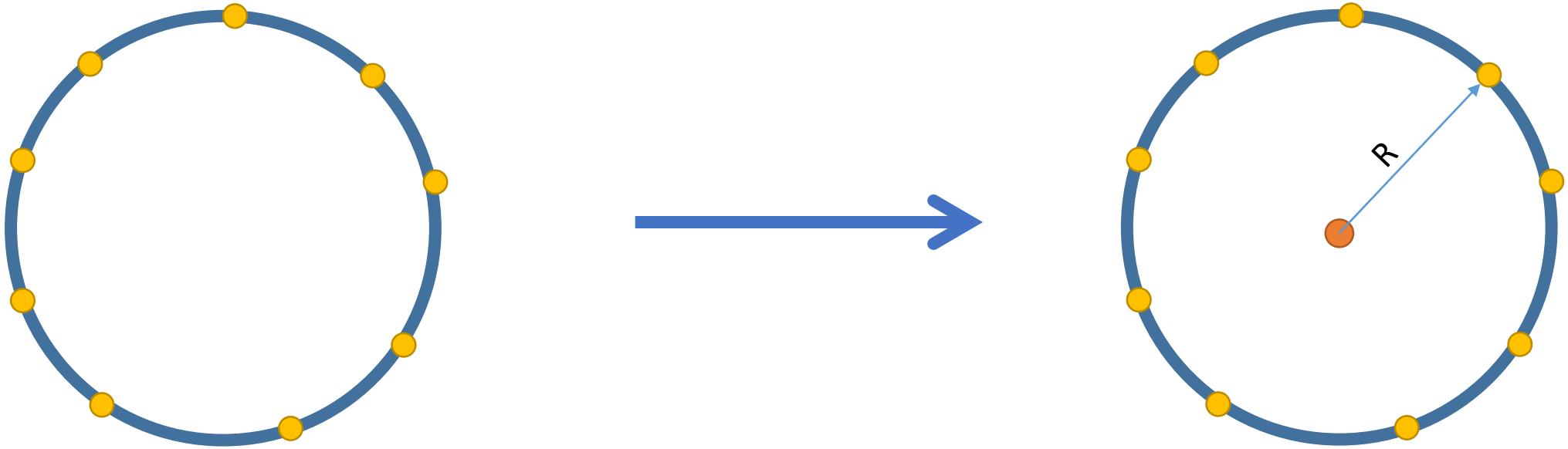
Circle Hough Transform



Circle Hough Transform



Circle Hough Transform



But What is ‘R’?!!

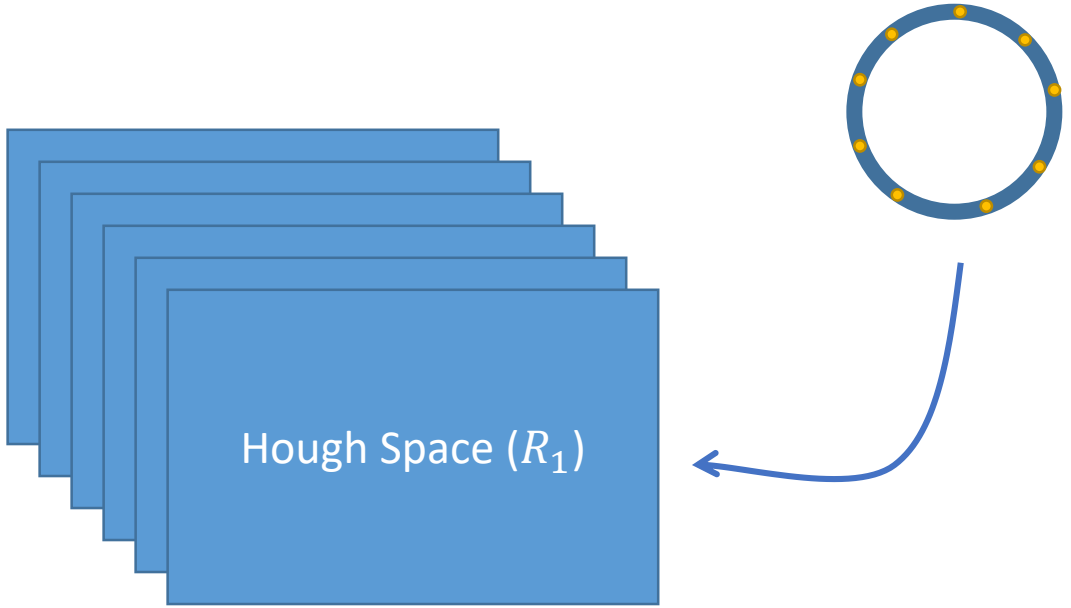
Circle Hough Transform

Since distance effect on the ‘R’ value

But What is ‘R’?!!

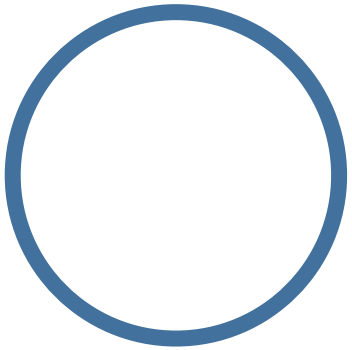
Multi Radius Hough Detection

- **Very inefficient** both in terms of **processing power** and **memory**



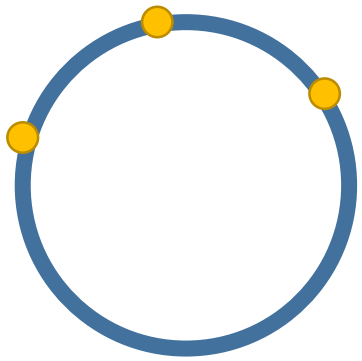
Randomized Hough Transform

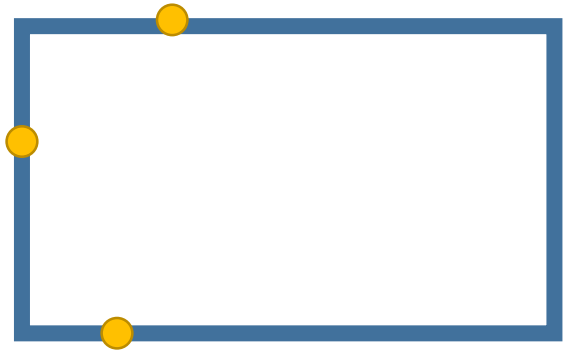
- How it works?



Randomized Hough Transform

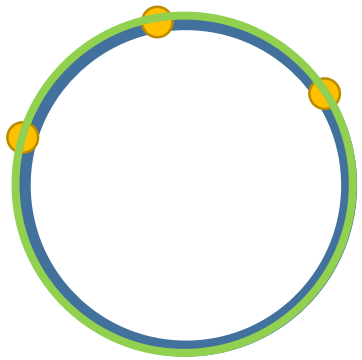
- How it works?

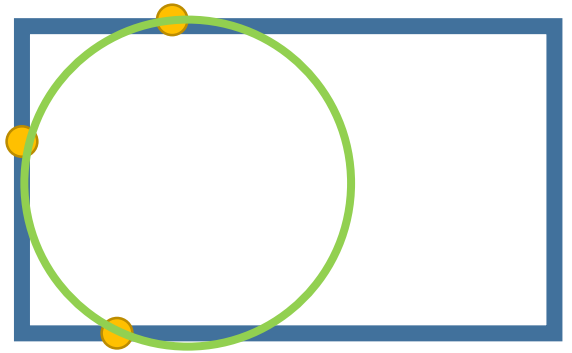




Randomized Hough Transform

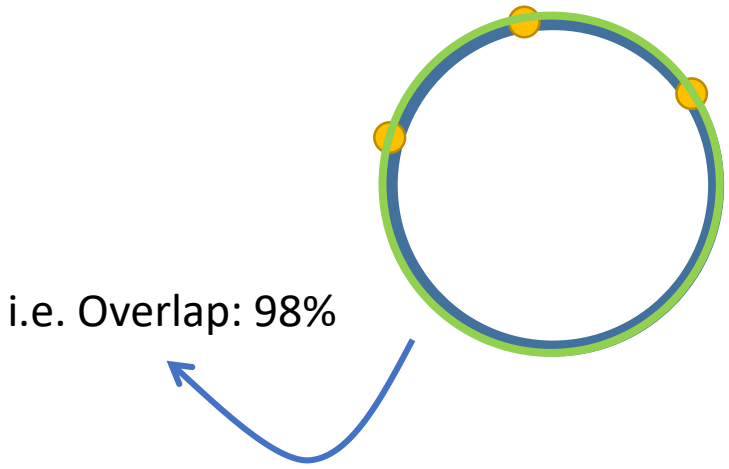
- How it works?

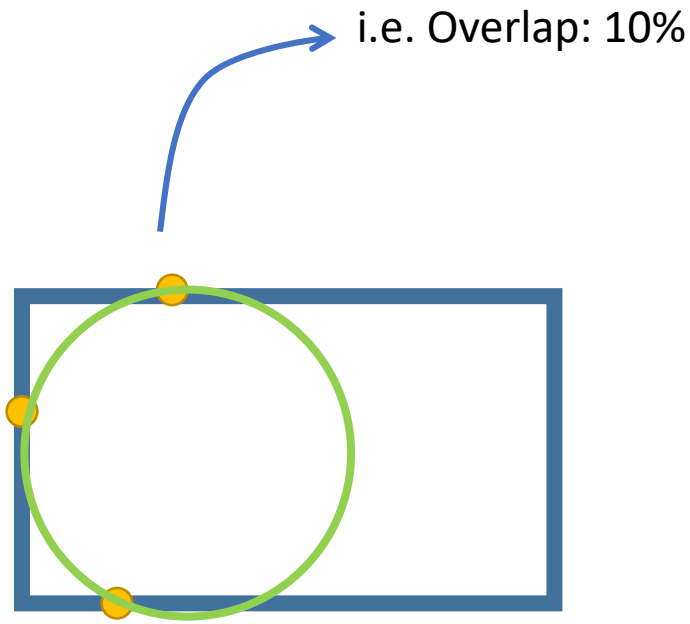




Randomized Hough Transform

- How it works?





Randomized Hough Transform

- Much Faster
- But does not have enough accurate
- Require lots of iteration to guarantee an acceptable result.

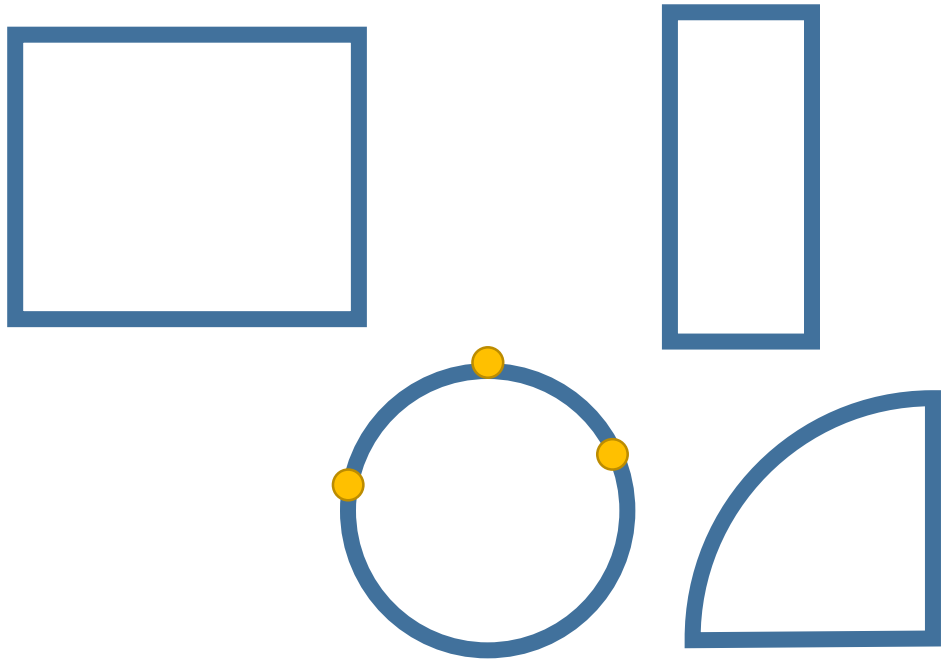
Randomized Hough Transform

Probability of point on the circle: $$P_r = \frac{EdgeArea(circle)}{EdgeArea(total\ edges)}$$

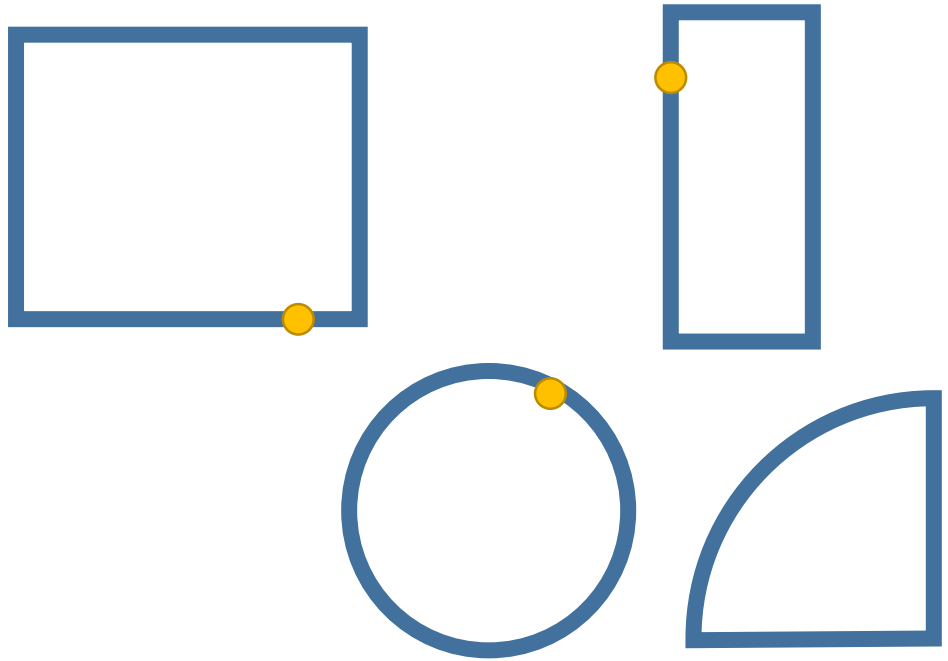
Probability of three points on the circle: $$P_c = P_1 * P_2 * P_3$$

Probability of three everywhere else: $$P_e = 1 - P_c$$

Randomized Hough Transform



Less Probable (i.e. 0.01%)

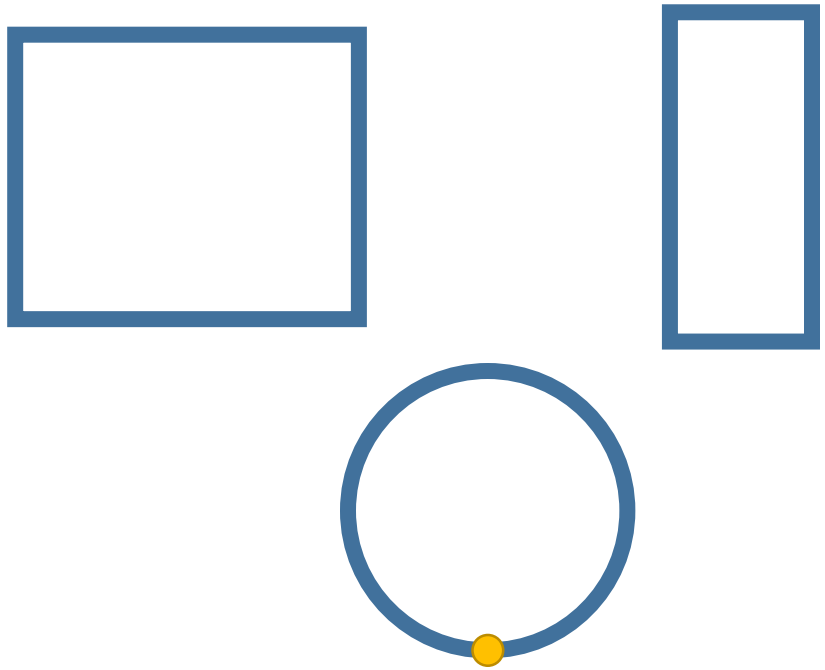


More Likely to happen (i.e. 99.99%)

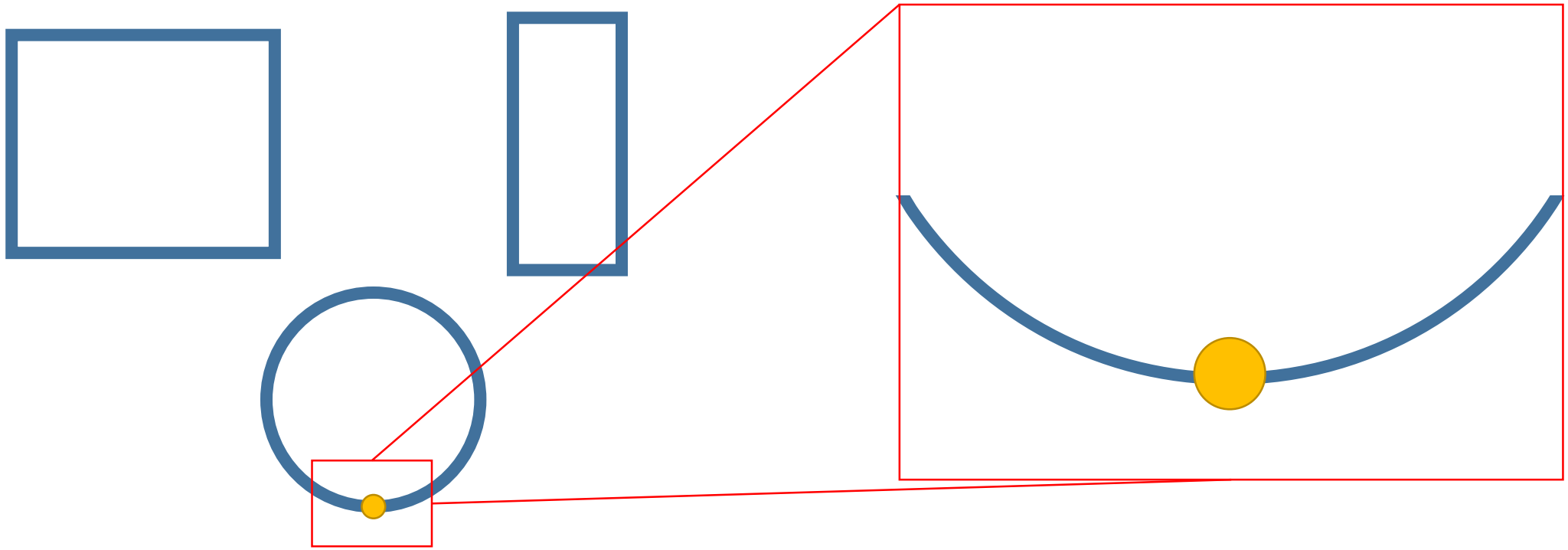
Fast Randomized Hough Transform

- To solve the accuracy.
- Hence, reduce the maximum iteration required
- Thus, become even faster!

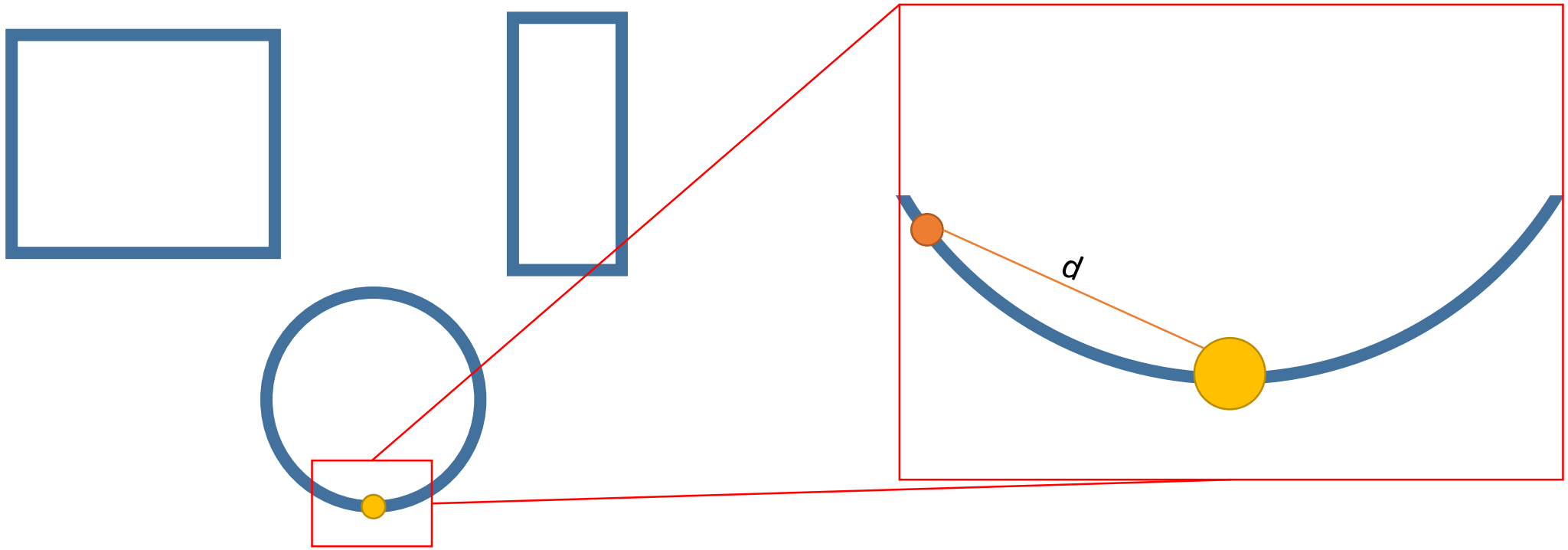
Fast Randomized Hough Transform



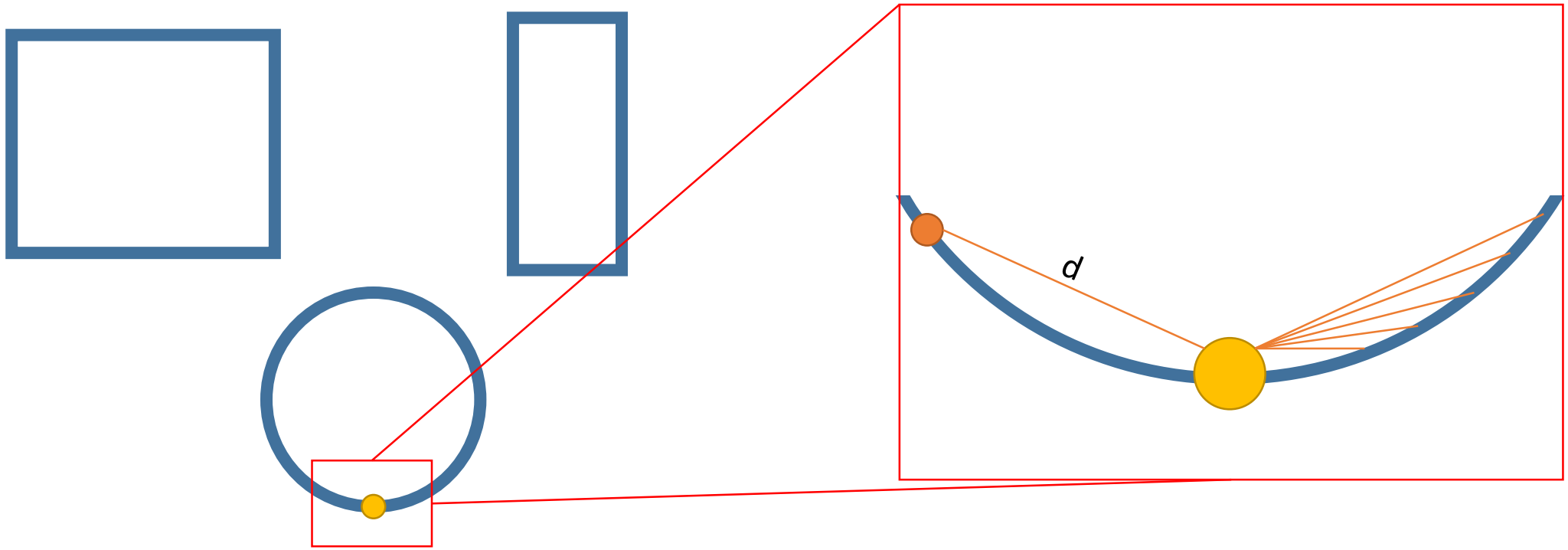
Fast Randomized Hough Transform



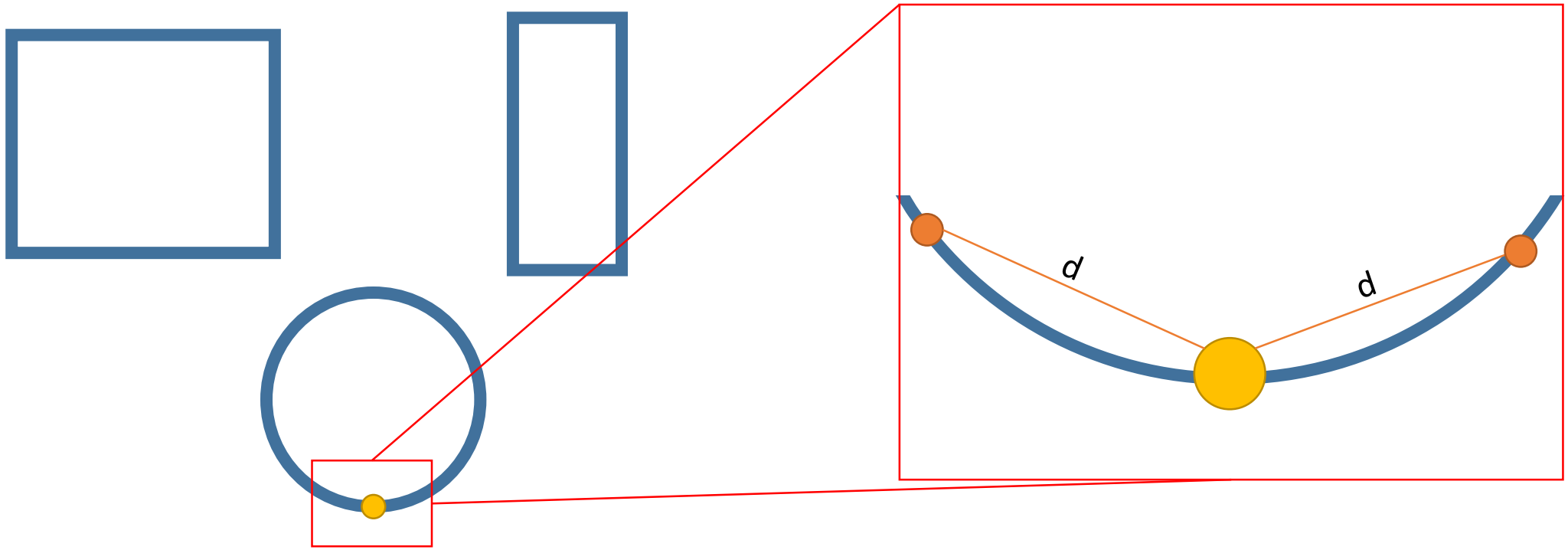
Fast Randomized Hough Transform



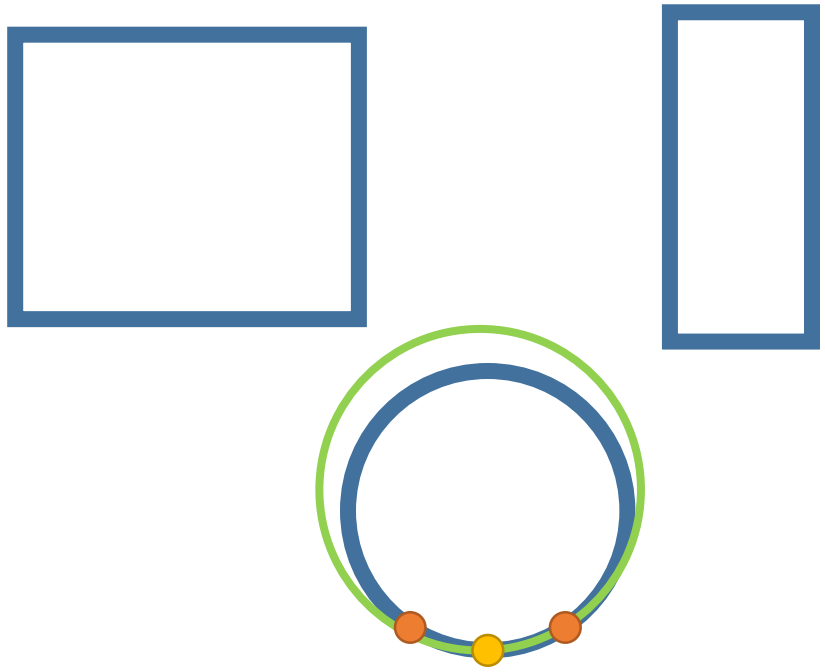
Fast Randomized Hough Transform



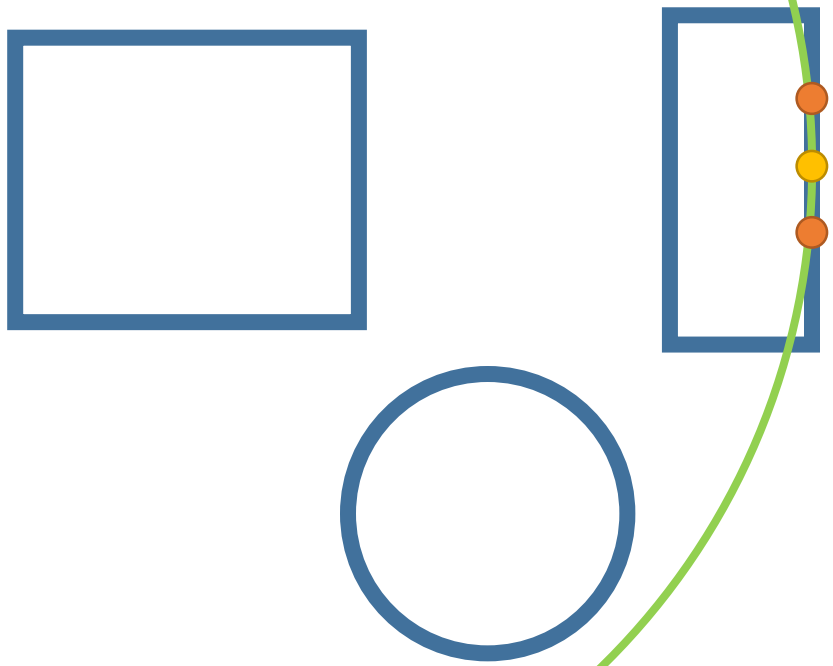
Fast Randomized Hough Transform



Fast Randomized Hough Transform



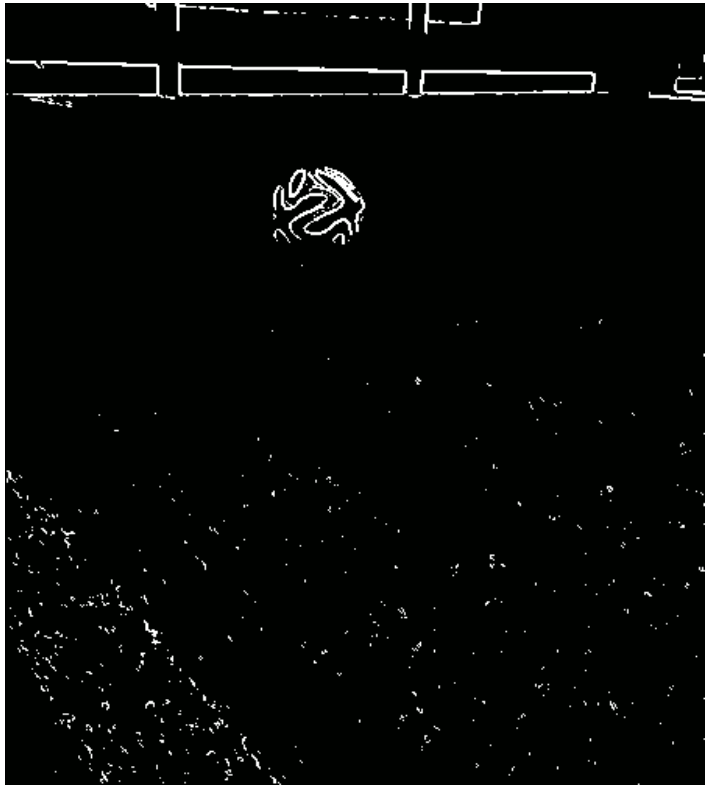
Fast Randomized Hough Transform



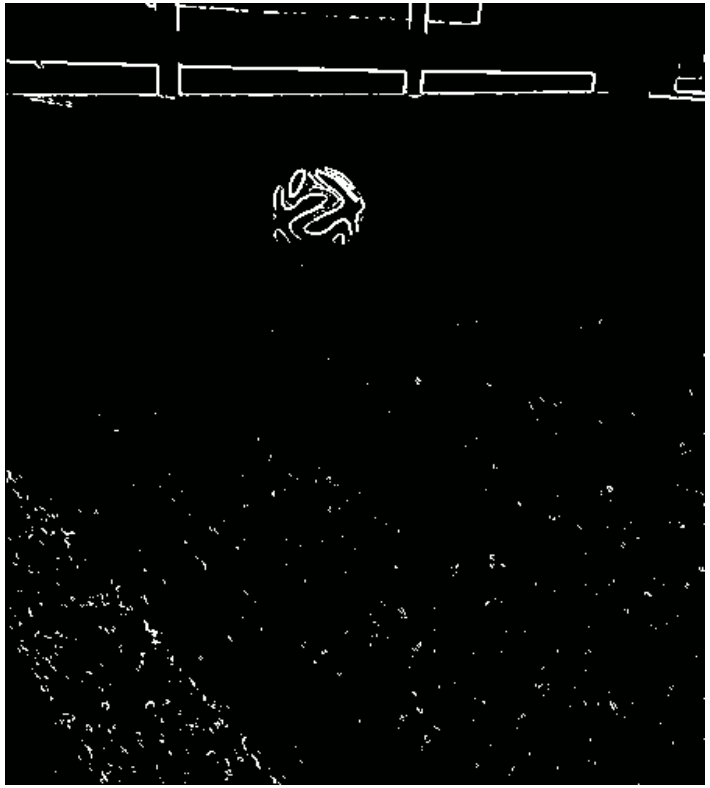
Fast Randomized Hough Transform

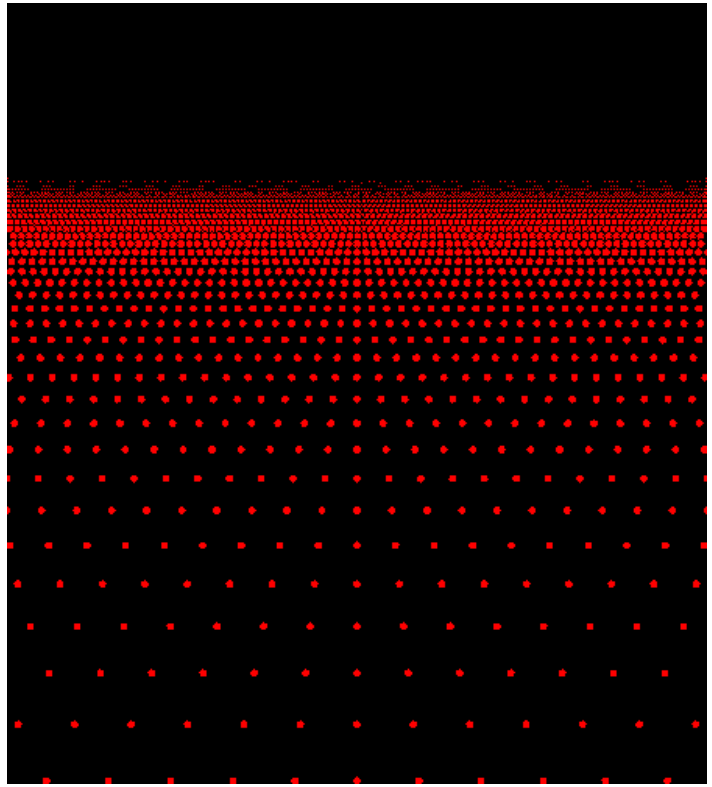
- There is an extracted circle for each iteration
- Here in our project:
  - Filter by size
  - Filter by green and non-green pixel percentage inside the circle

Edge Detection

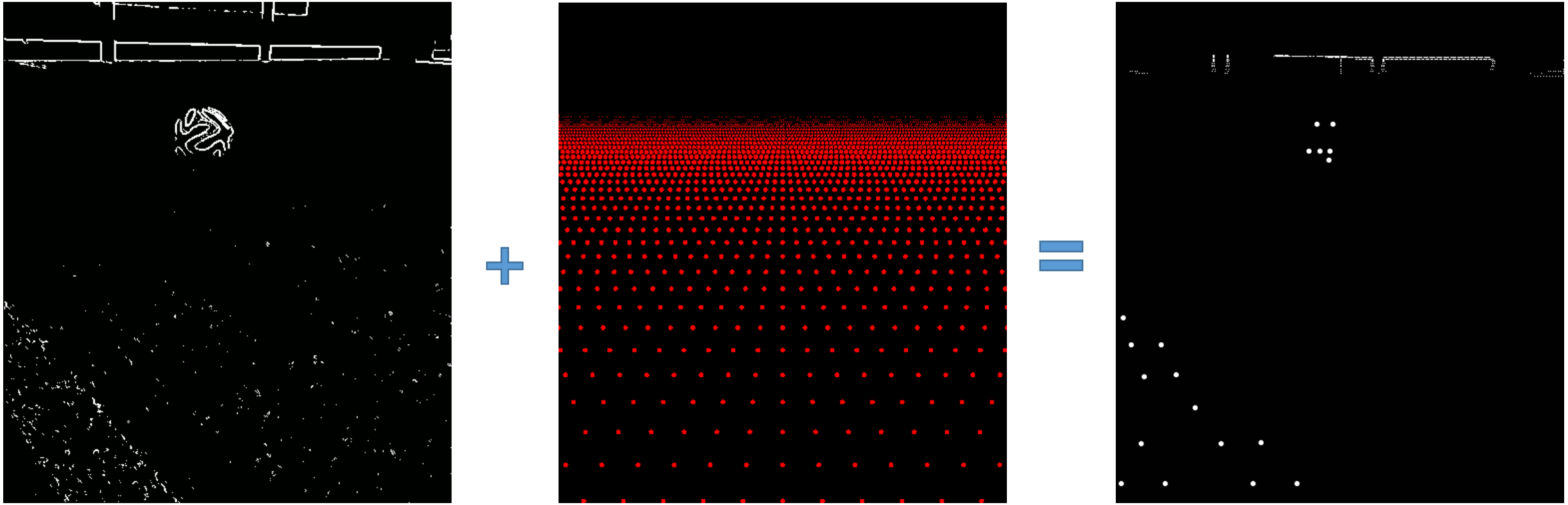


Edge Detection



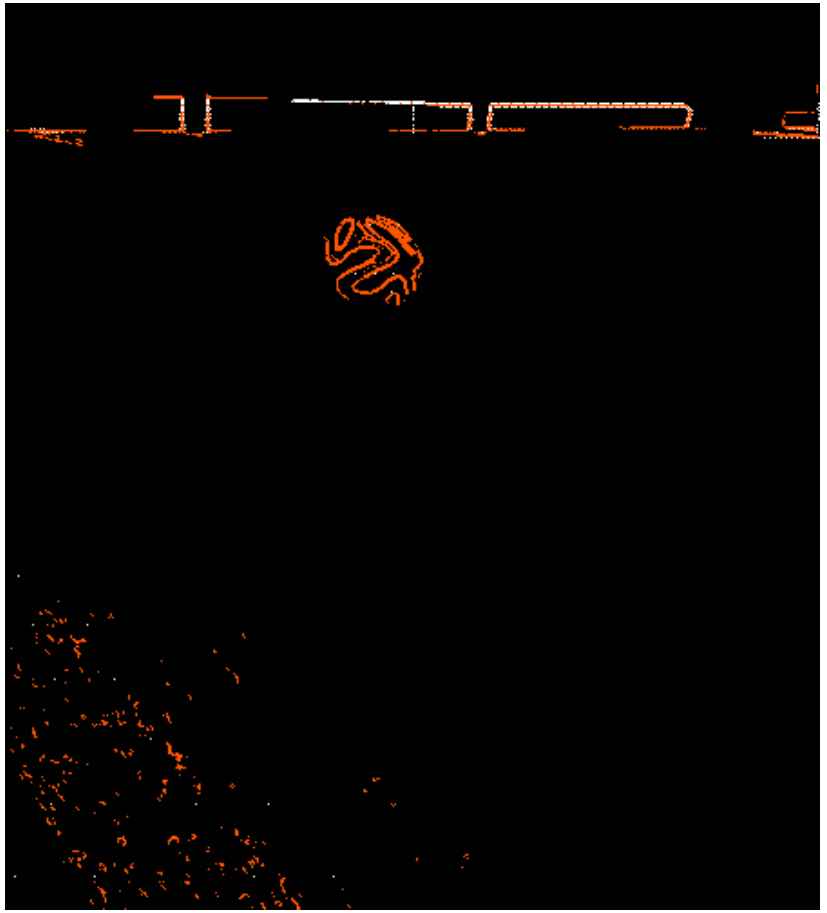


Edge Detection



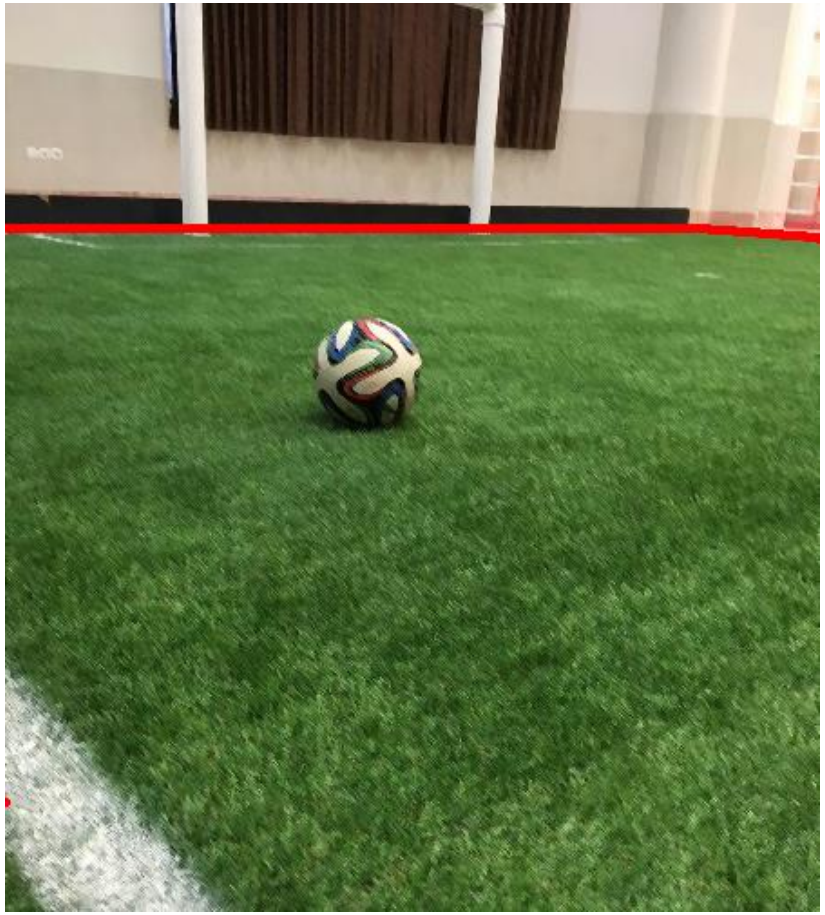
Here is the magic

- Distributed of Random Points
- Very Efficient



Let’s back-up a little bit!

- Field Boundary Detection
  - Ball is always inside the field
  - Outside of the field is usually crowded.



Field-Green Color

- When the robot is inside the field *usually* it see *mostly* green (When the image is cropped below the horizon).

Thomas Reinhardt

Horizon

Horizon

- “In graphical perspective, a vanishing point is a point in the image plane where the projections of a set of parallel lines in space intersect.” -Wikipedia
- Horizon is the projection of the infinity (or a very distanced point)

Photo by: http://wide-wallpapers.net

In our work

- The highest pixel for field boundary in image is when robot is at one corner and observing the diagonal corner of the field.

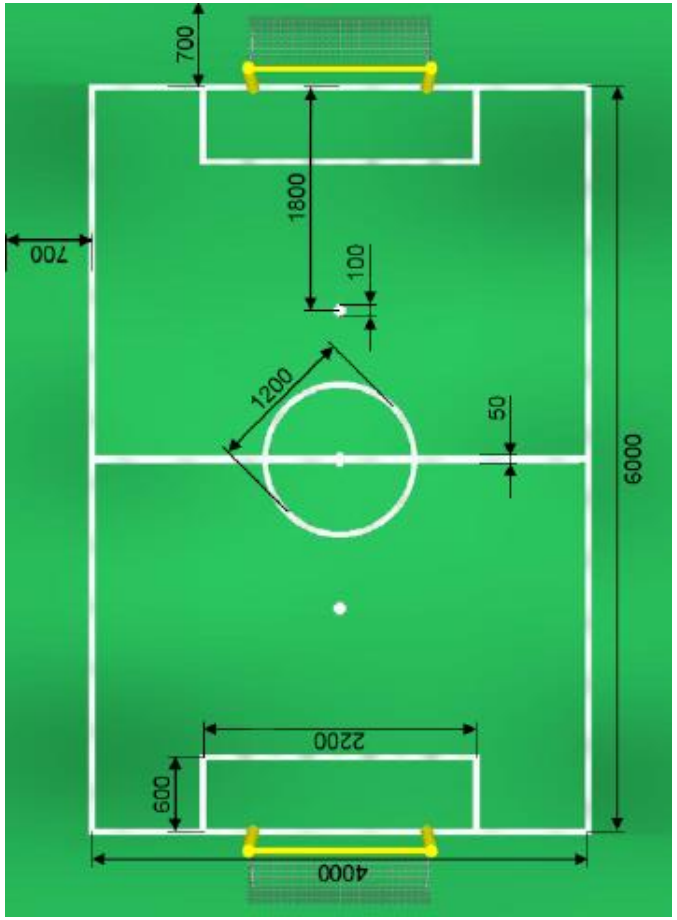
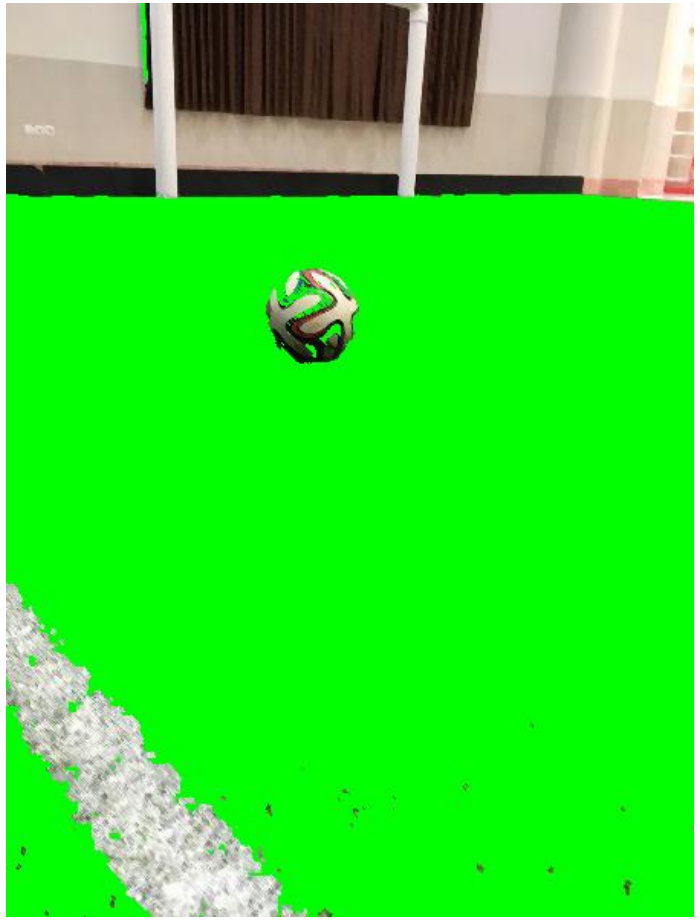


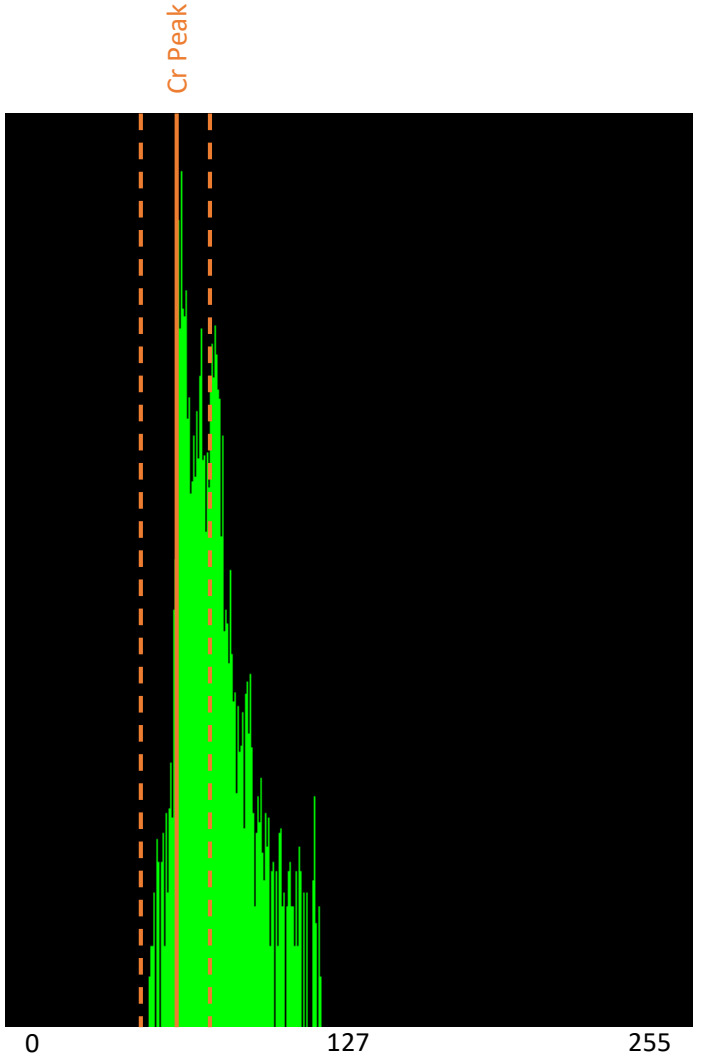
Photo by: http://tzi.de/spl/

Again Field-Green Color

$p = peak(histogram)$

$isGreen(P) = |P - p| < \delta$





Field Boundary

Run Ups Results

Andrew's monotone Results

Finally Filters

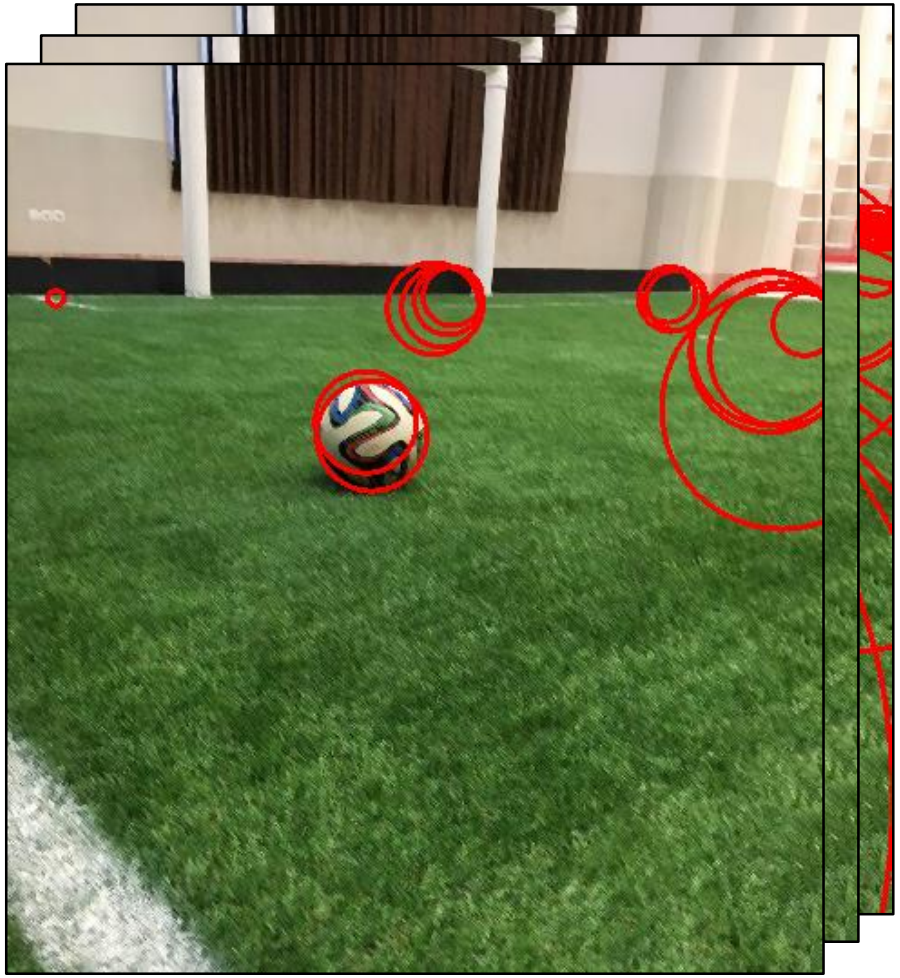
- Size
- White Pixels Maximum
- Non-Green Pixels Limit
- Projected Size
- Pattern

Finally Filters

- Size
- White Pixels Maximum
- Non-Green Pixels Limit
- Projected Size
- Pattern

Finally Filters

- Size
- White Pixels Maximum
- Non-Green Pixels Limit
- Projected Size
- Pattern



Finally Filters

- Size
- White Pixels Maximum
- Non-Green Pixels Limit
- Projected Size
- Pattern

Thanks To

- MRL Humanoid (MRL-HSL)
- MRL Biped Lab. (MRL-SPL)
- MRL3D Soccer Simulation Team

Any Questions?

- The code can be found on my github at:
  - http://github.com/arefmq
- The documentation is available on my profile at:
  - http://mrl-spl.ir/~moqadam
- You can also reach my by email via:
  - a.moqadam@mrl-spl.ir