F1/10

Autonomous Racing

1/10 the scale. 10 times the fun!

Advanced Path Planning and Obstacle Avoidance

Madhur Behl
(University of Virginia)

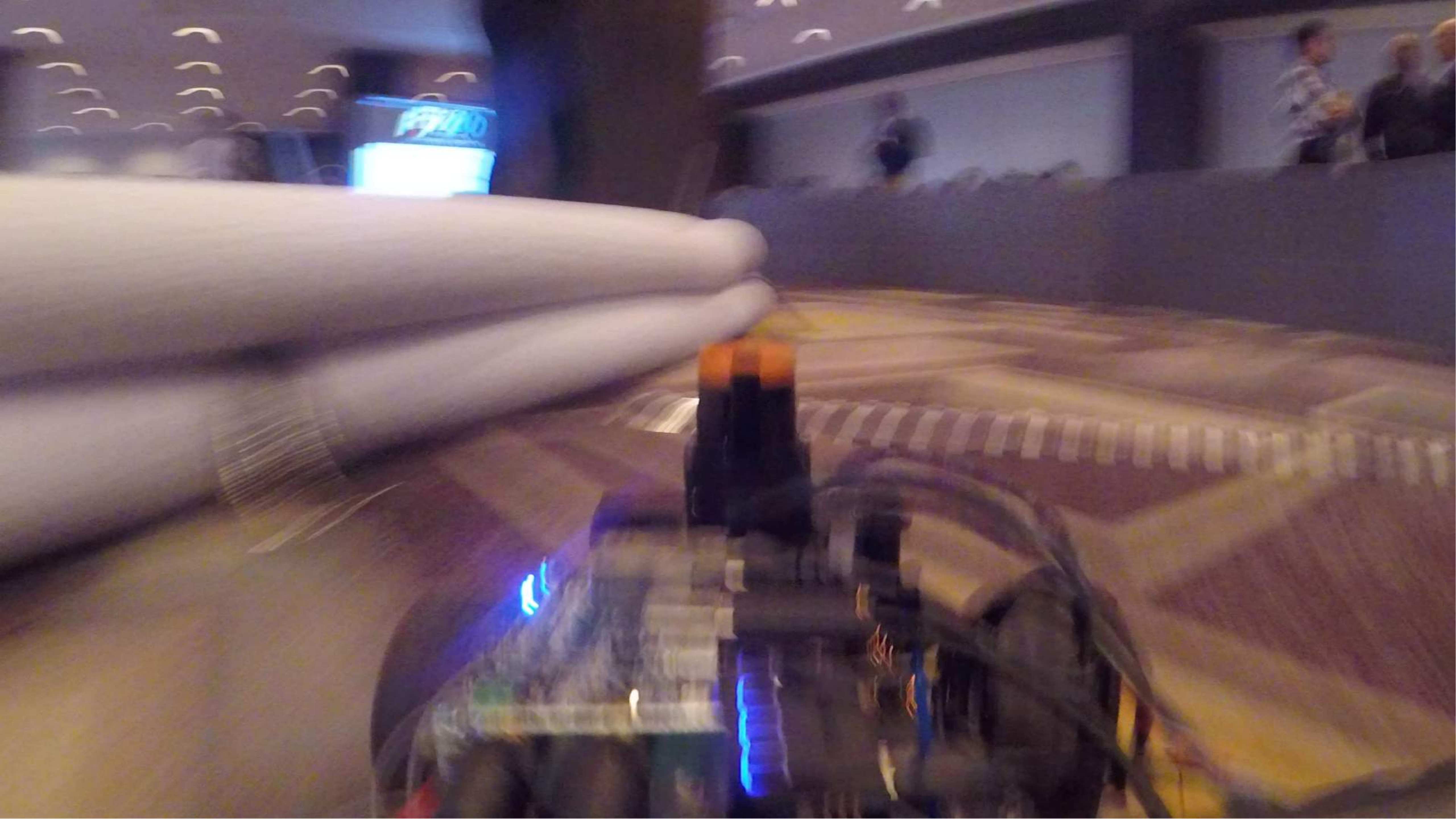
Assignment 4 Demo

- Wednesday, April 26 @ 2:00pm
- Rice 024
- Complete wall following laps
  - Extra Credits:
    - Launch file
    - Velocity PID
    - Collision Avoidance – stop when there is an obstacle in the front of the car.

Final Race

- Friday, May 10, from noon-2:00pm
- Link Lab Arena
    - New racetrack will be setup (slightly bigger and challenging)
    - Time trials – round robin.

4th F1/10 International Autonomous Racing Competition




Head-to-head Autonomous Racing
4th F1/10 International Autonomous Racing Competition
F1
TENTH
f1tenth.org


F1
TENTH
f1tenth.org

A simple path planning algorithm: **Gap Finding**

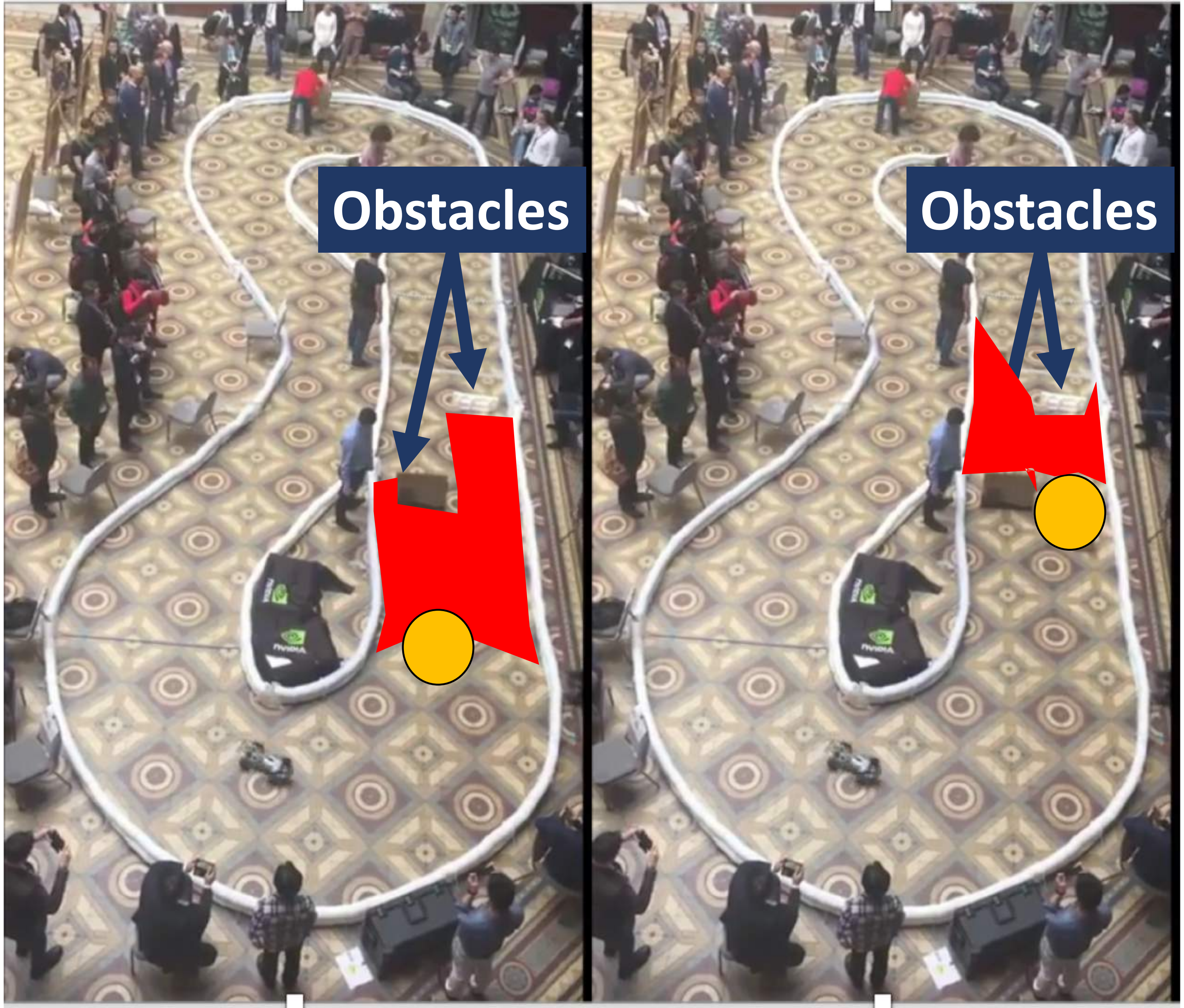
Conditions:

- Closed track
- Always move forward – no choices

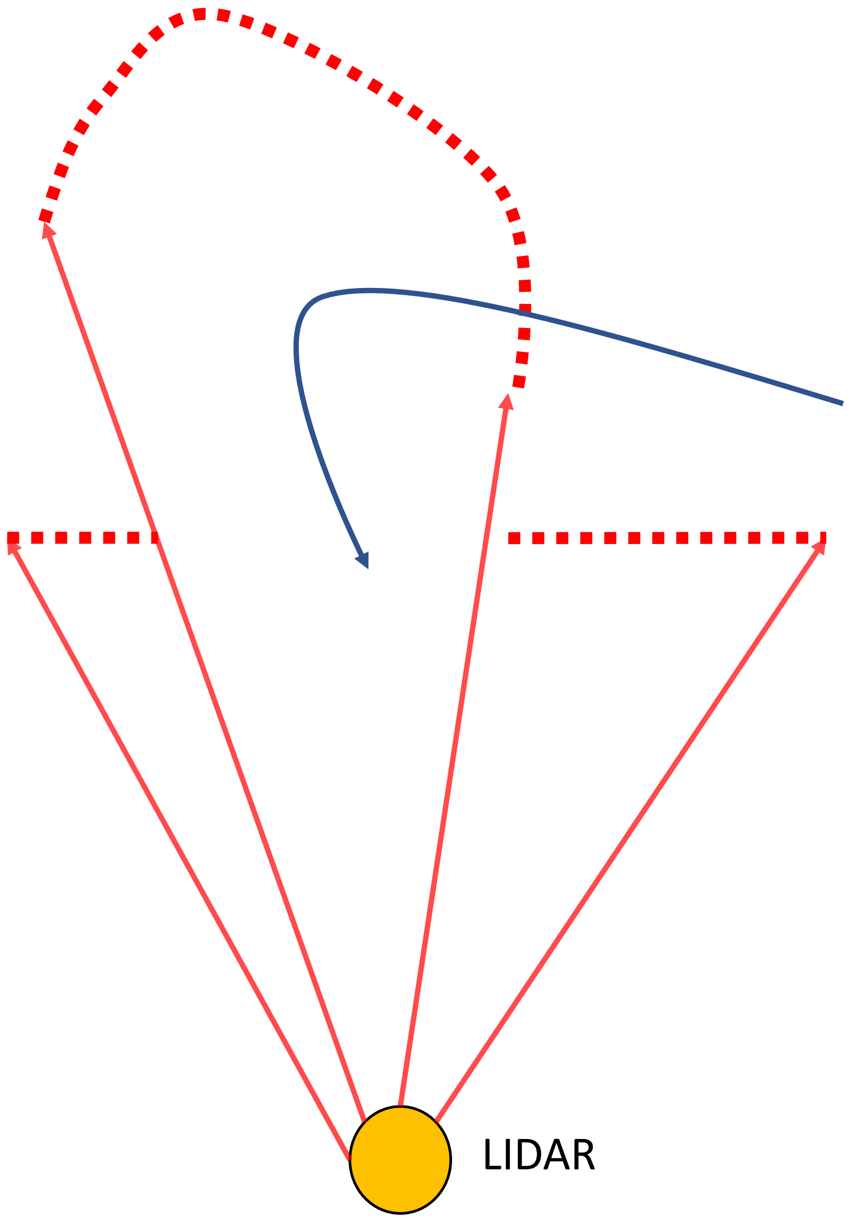
Idea: find the largest gap in front of the car and go through it

Competition track in Porto, April 2018

Find the gap



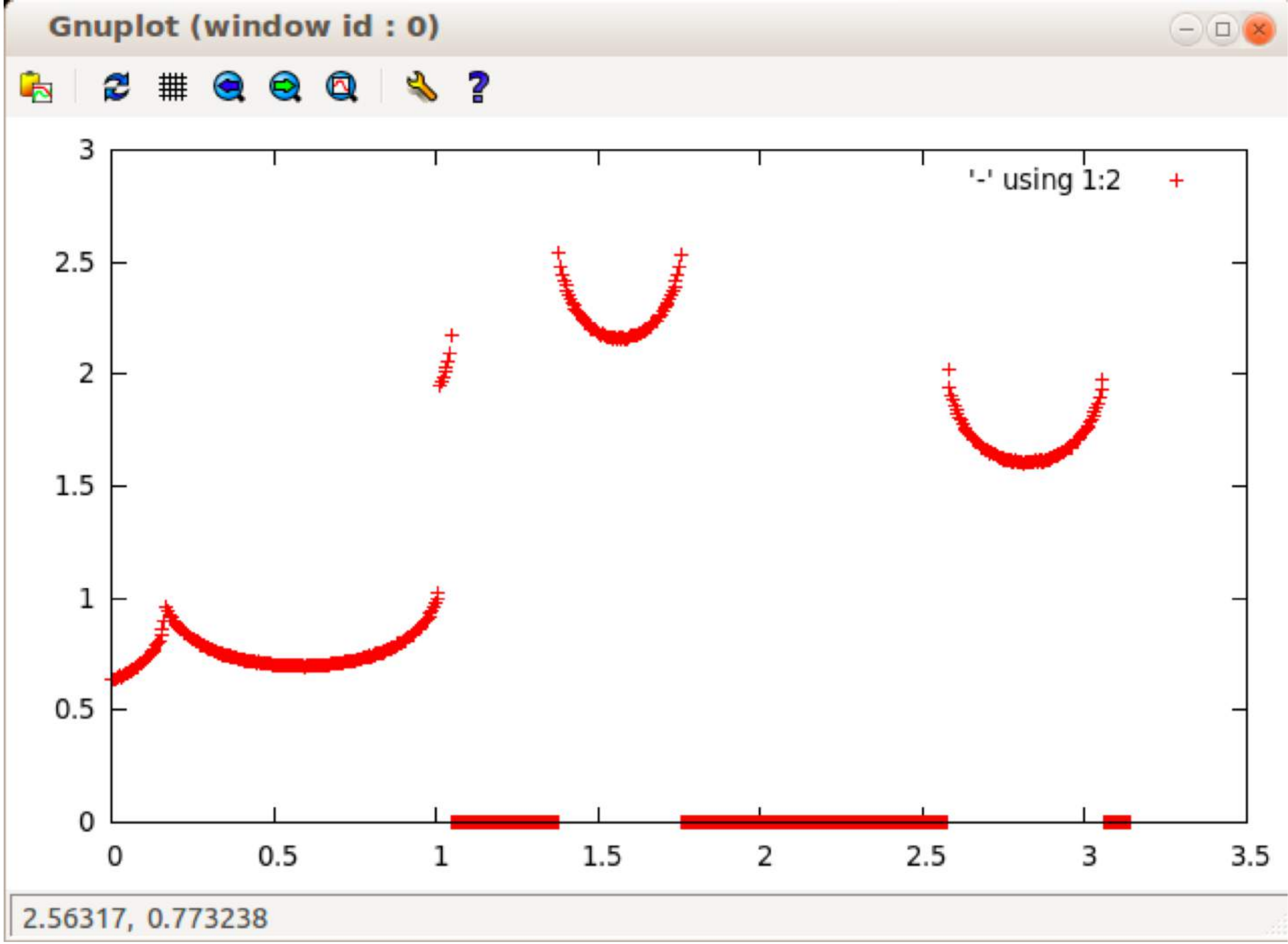
Find the gap



Find this gap and calculate its width

(this *has* to be the way forward)

Where should the car go ?



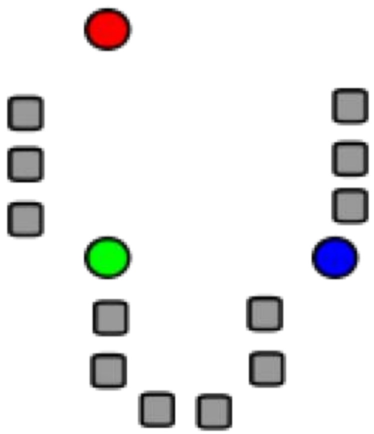
Gap finding

- **Find the gaps**
- Calculate the width of each gap
- Determine the widest gap
- Optional: Determine the “deepest” gap

Find the gap

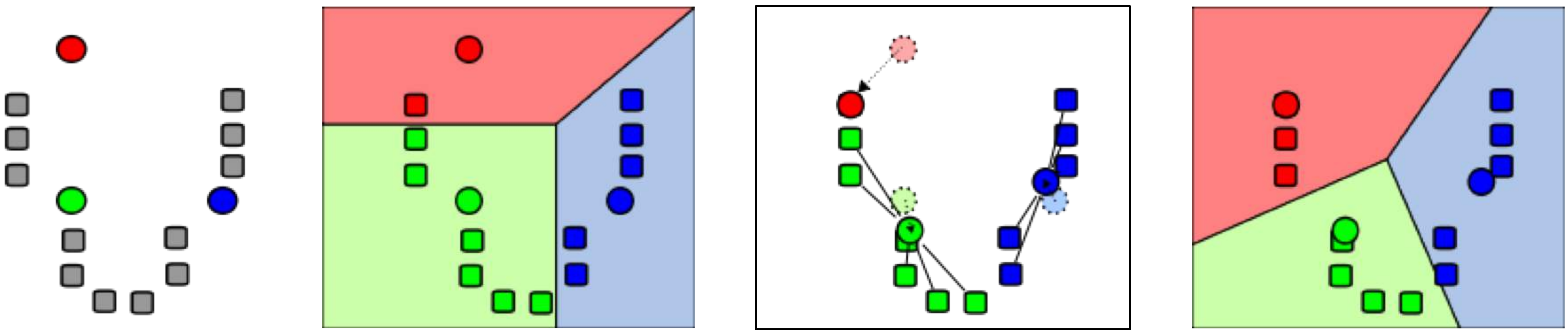
- What determines what a gap is?
- Intuitively, it's a sequence of adjacent ranges that have a similar value.
- We want to divide the 1080 ranges into *clusters* of values, such that values within one cluster are "close". Each cluster is a gap candidate.
  - How many clusters?

K-means clustering



2D example: each data item has 2 features

K-means clustering



Initialization:

Select a target number of clusters, k (here, k=3) (How?)

Select k cluster centers (How?)

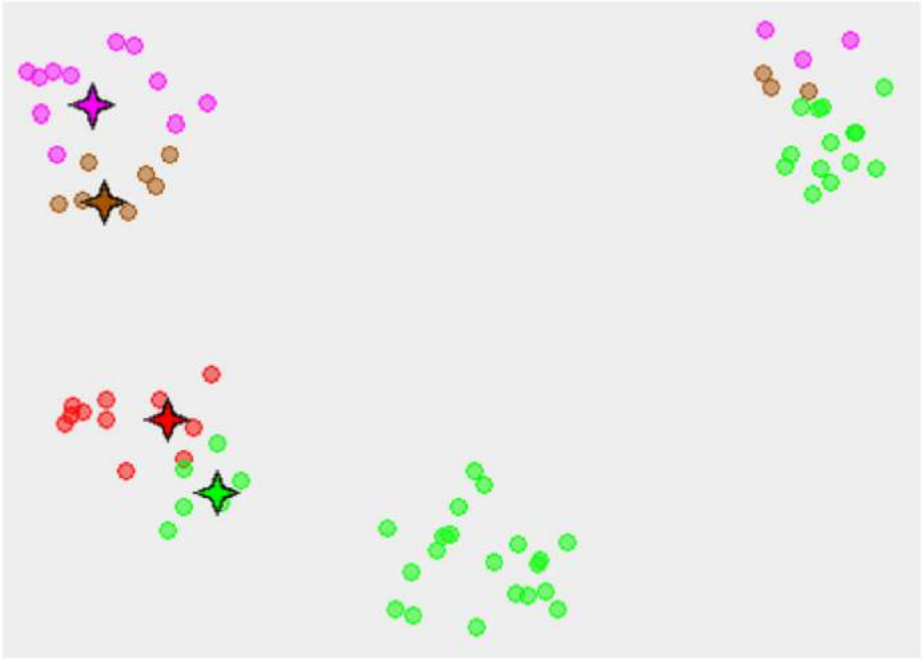
Assignment step:

Assign each data item to its nearest cluster center

Update step:

Set each cluster to be the average of the items assigned to its cluster

K-means clustering

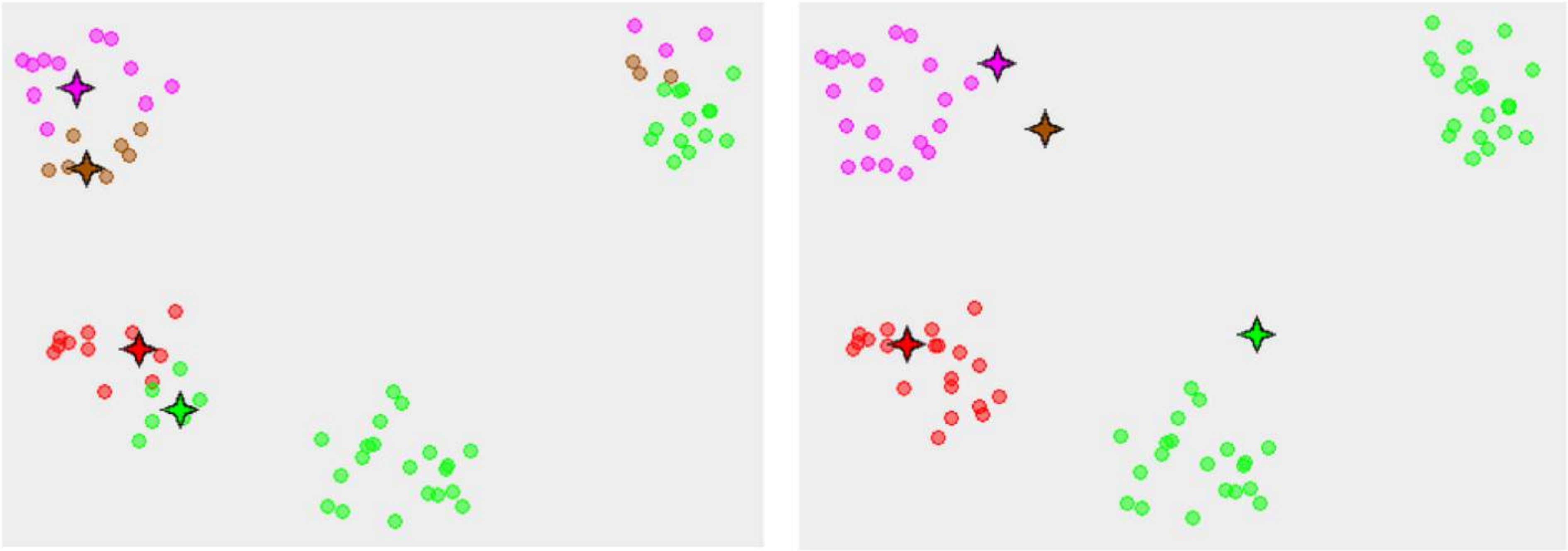


Initialization: Select a target number of clusters, k, and select k cluster centers
Assignment step: Assign each data item to its nearest cluster center
Update step: Set each cluster to be the average of the items assigned to its cluster

K-means clustering

Initialize and Assign

Update centers and re-assign



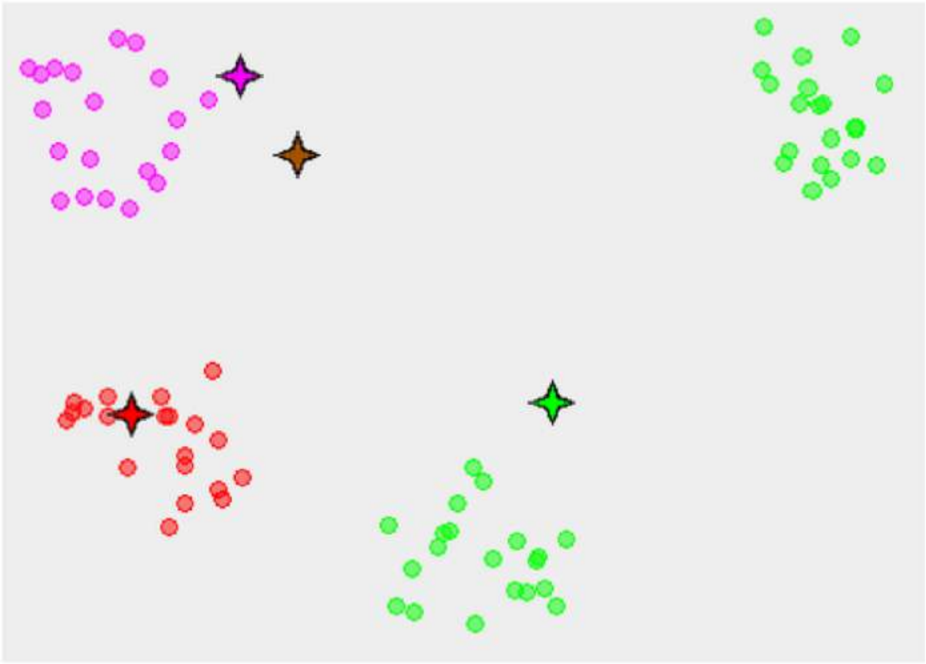
Initialization: Select a target number of clusters, k, and select k cluster centers

Assignment step: Assign each data item to its nearest cluster center

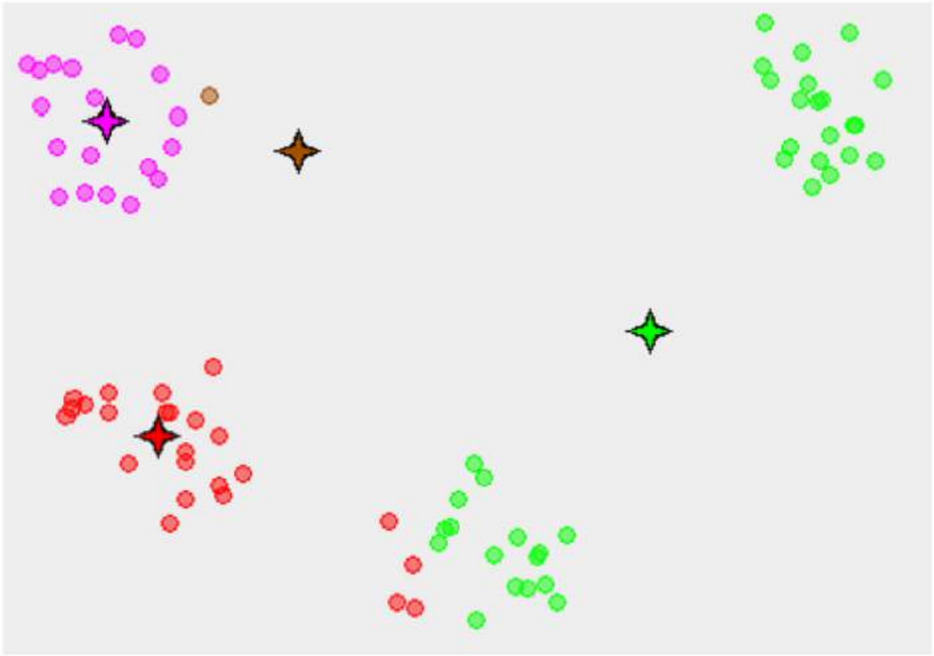
Update step: Set each cluster to be the average of the items assigned to its cluster

K-means clustering

(previous state)

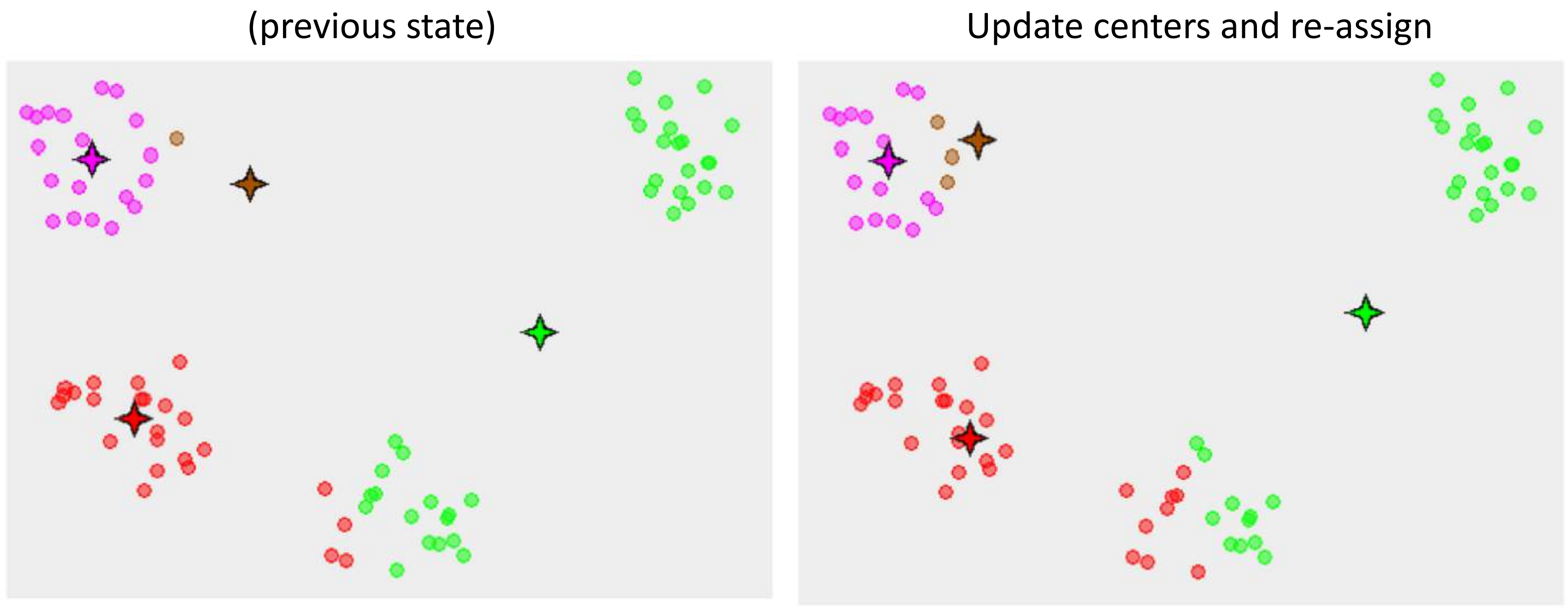


Update centers and re-assign



Initialization: Select a target number of clusters, k, and select k cluster centers
Assignment step: Assign each data item to its nearest cluster center
Update step: Set each cluster to be the average of the items assigned to its cluster

K-means clustering



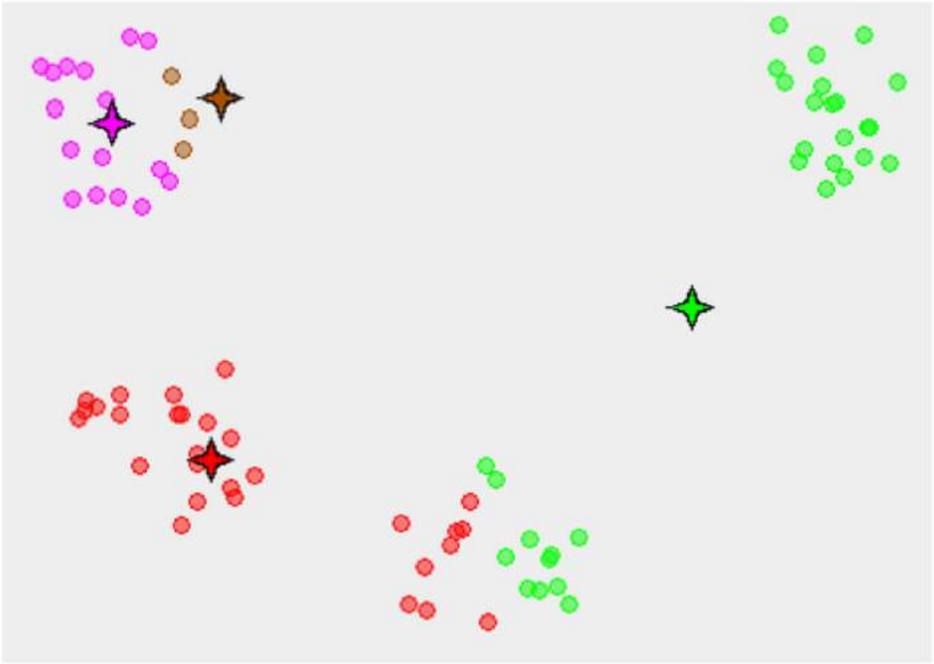
Initialization: Select a target number of clusters, k, and select k cluster centers

Assignment step: Assign each data item to its nearest cluster center

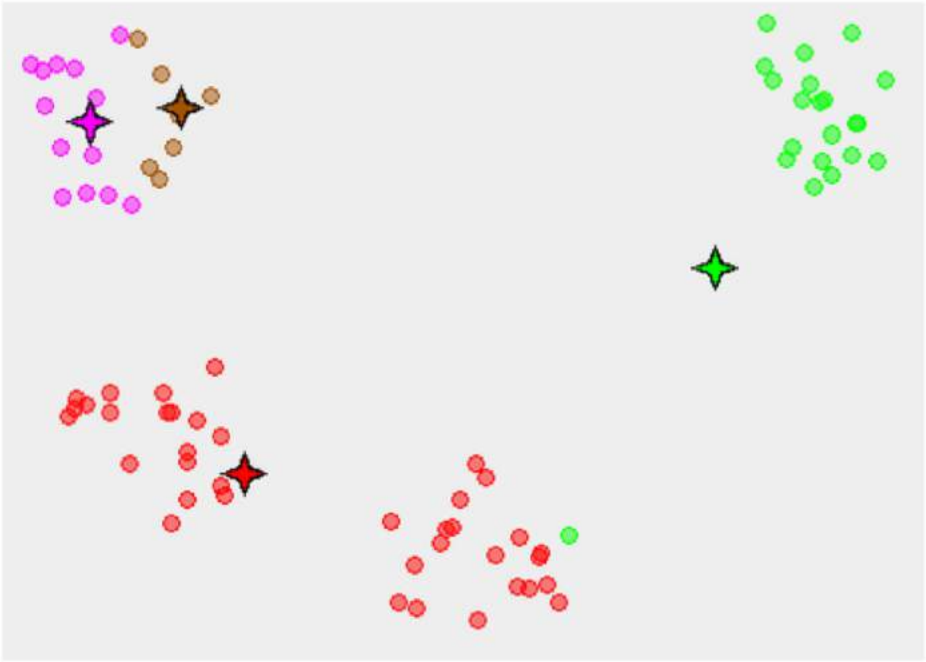
Update step: Set each cluster to be the average of the items assigned to its cluster

K-means clustering

(previous state)



Update centers and re-assign



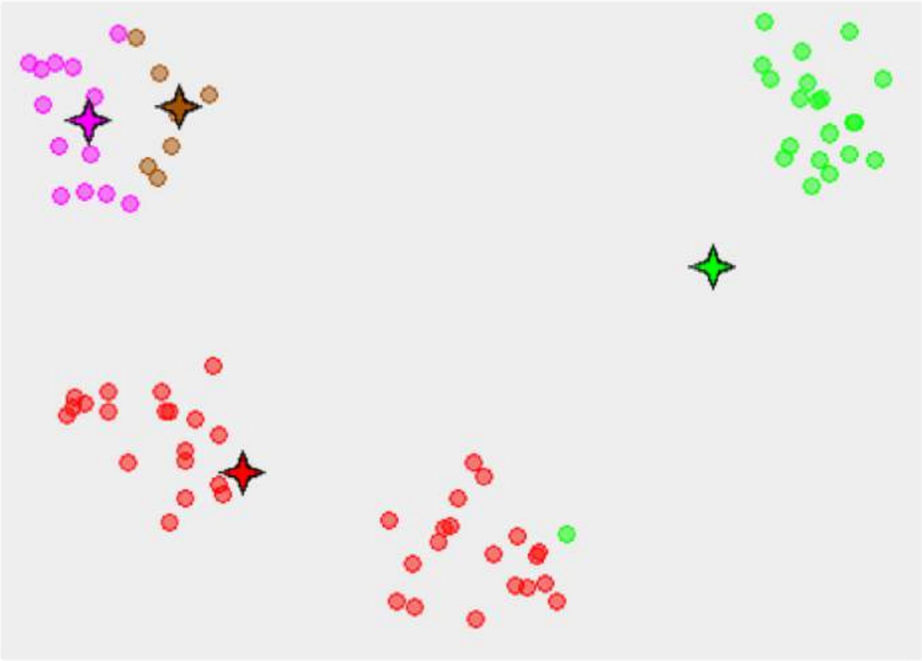
Initialization: Select a target number of clusters, k, and select k cluster centers

Assignment step: Assign each data item to its nearest cluster center

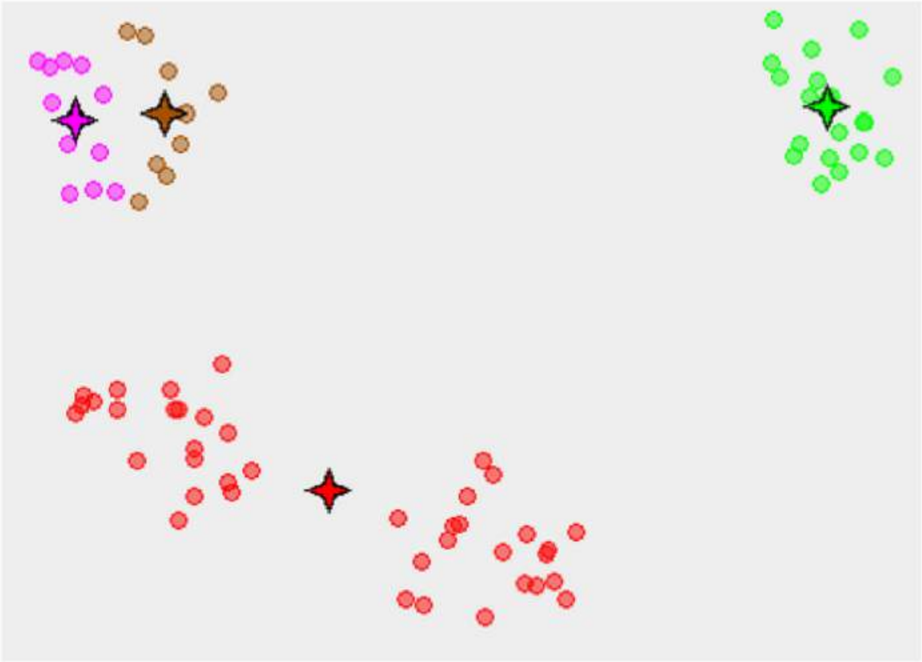
Update step: Set each cluster to be the average of the items assigned to its cluster

K-means clustering

(previous state)



Update centers and re-assign…not much movement → stop



Initialization: Select a target number of clusters, k, and select k cluster centers

Assignment step: Assign each data item to its nearest cluster center

Update step: Set each cluster to be the average of the items assigned to its cluster

K-means clustering

- K-means is minimizing the intra-cluster distances to generate compact clusters

$$\underset{\mathbf{S}}{\arg\min} \sum_{i=1}^{k} \sum_{\mathbf{x} \in S_i} \left\| \mathbf{x} - \boldsymbol{\mu}_i \right\|^2 = \underset{\mathbf{S}}{\arg\min} \sum_{i=1}^{k} |S_i| \operatorname{Var} S_i$$

Choice of clusters

i^{th} cluster

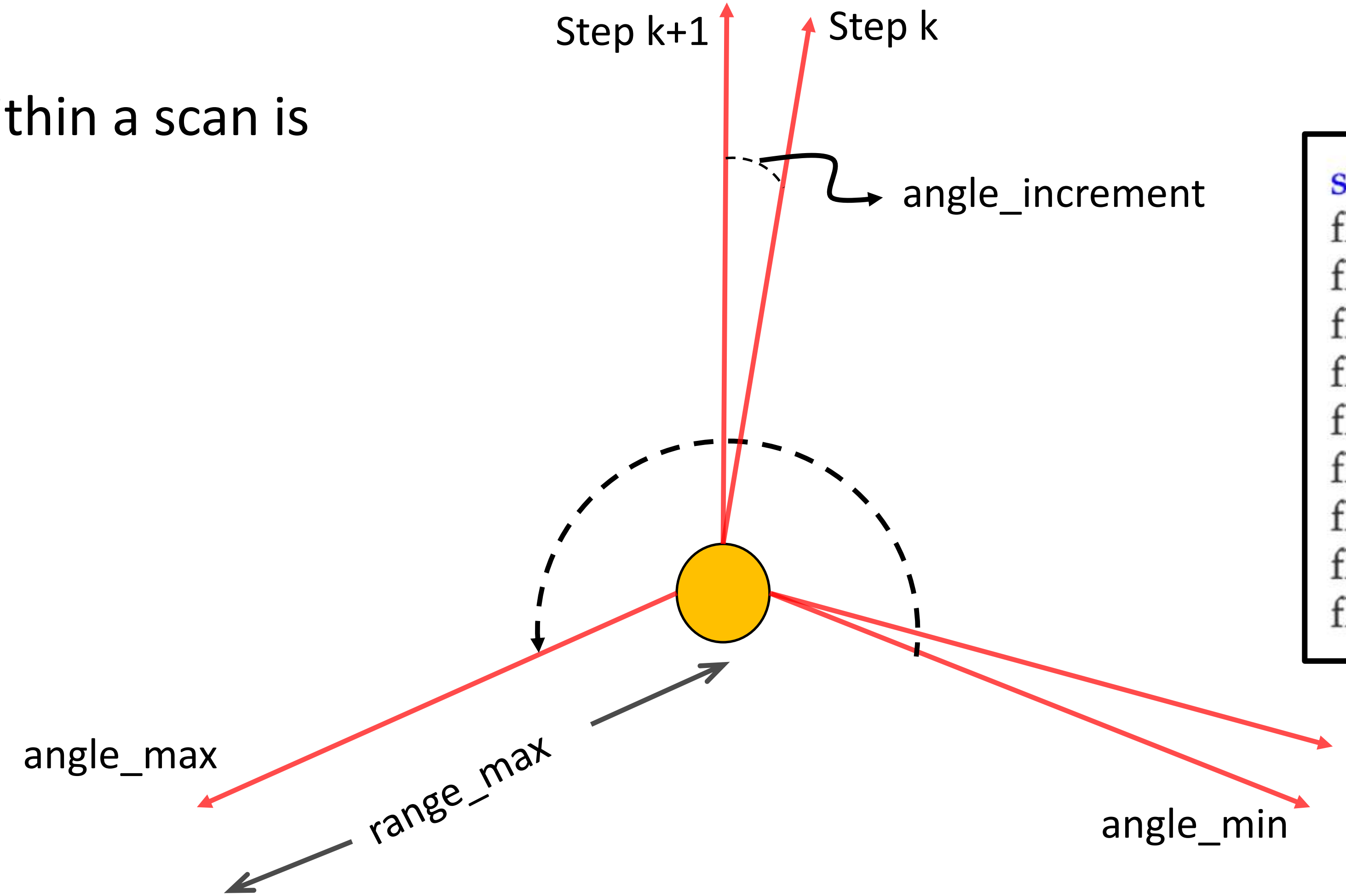
Data item

Centroid

Appropriate distance function

What features to use?

Each data item / measurement within a scan is characterized by
- Range
- Angle

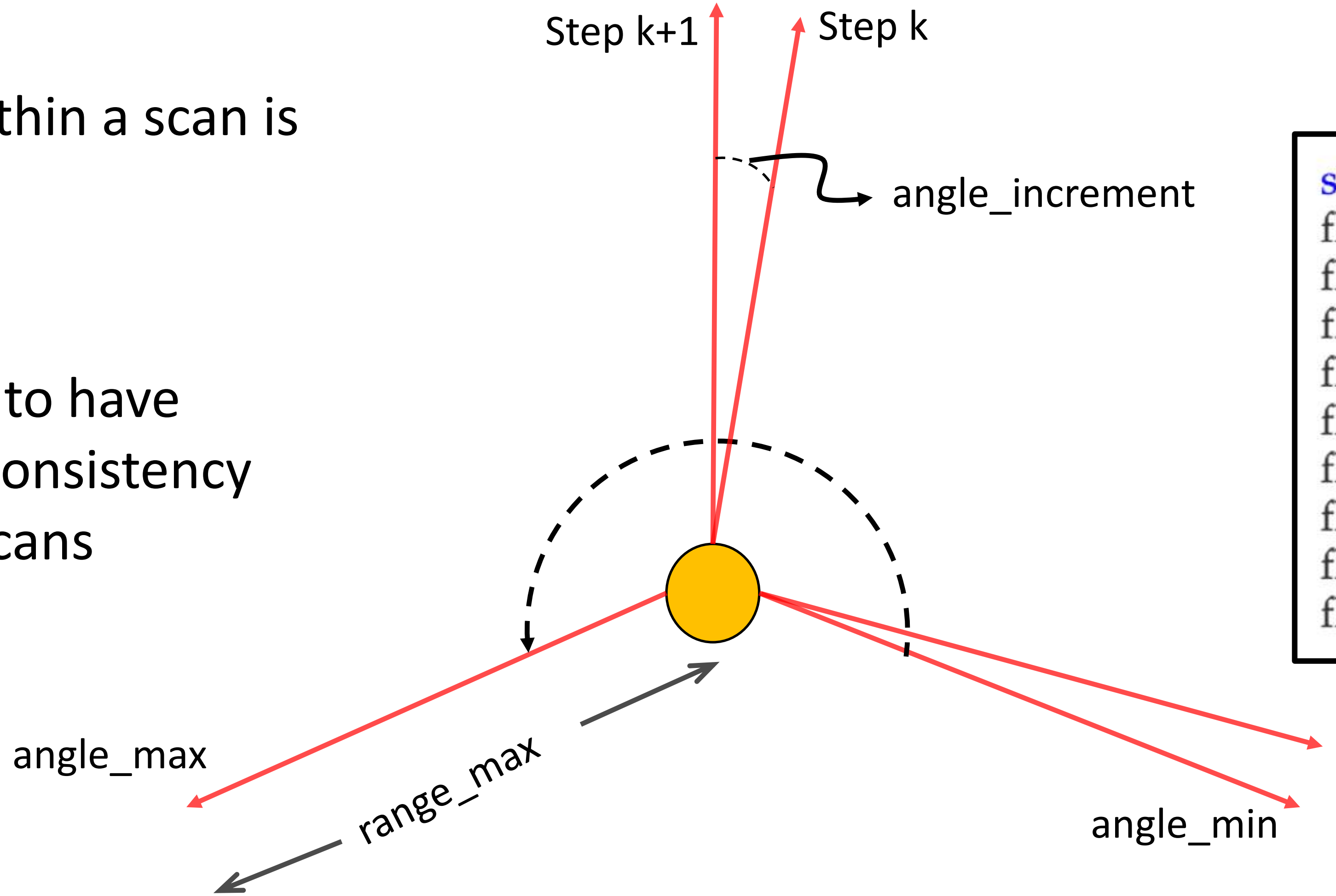



```
std_msgs/Header header
float32 angle_min
float32 angle_max
float32 angle_increment
float32 time_increment
float32 scan_time
float32 range_min
float32 range_max
float32[] ranges
float32[] intensities
```

What features to use?

Each data item / measurement within a scan is characterized by
- Range
- Angle

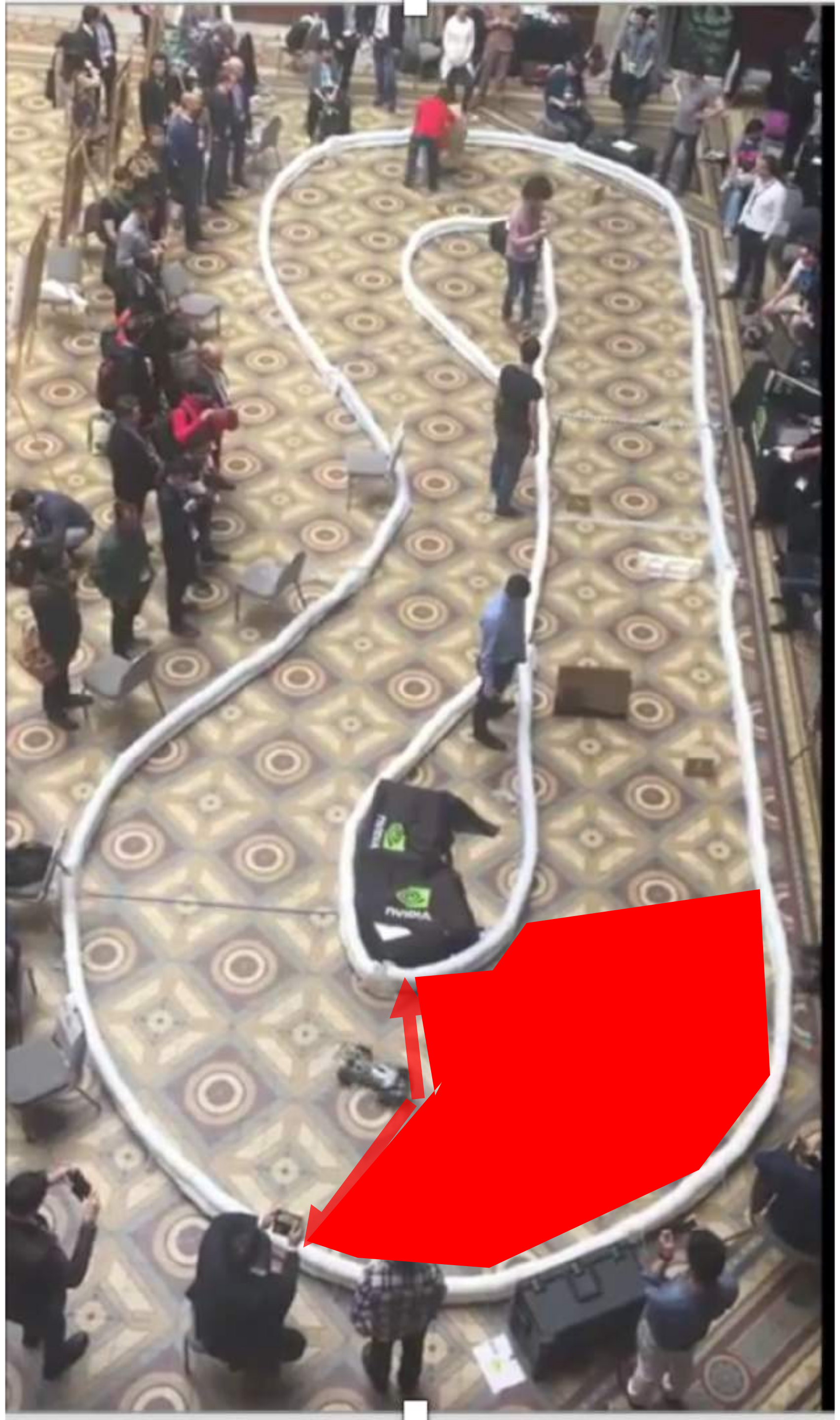
It probably helps to have some filtering = consistency check between scans




```
std_msgs/Header header
float32 angle_min
float32 angle_max
float32 angle_increment
float32 time_increment
float32 scan_time
float32 range_min
float32 range_max
float32[] ranges
float32[] intensities
```

Difficulties

Smooth curves and progressive openings
– where do you draw the line?



Dynamic Path Planning

Aim is of avoiding unexpected obstacles along the robot's trajectory to reach the goal.

Methods

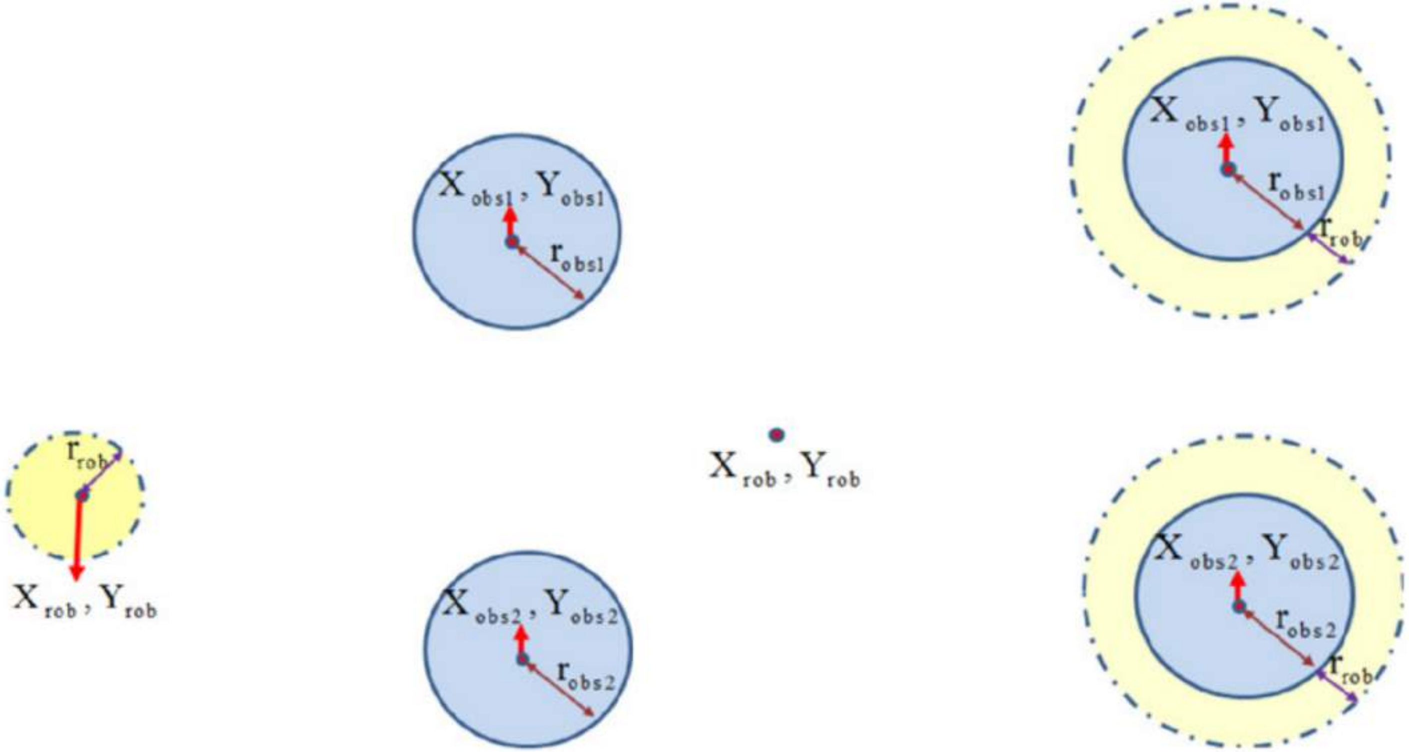
- Bug Algorithms
- Artificial Potential Field (APF) Algorithm
- Harmonic Potential Field (HPF) Algorithm
- Virtual Force Field (VFF) method
- Virtual Field Histogram (VFH) method
- Follow the Gap Method (FGM)

Some terms of concern

- Point Robot Approach
- Field of view of Robot
- Non-holonomic constraints

Point Robot Approach

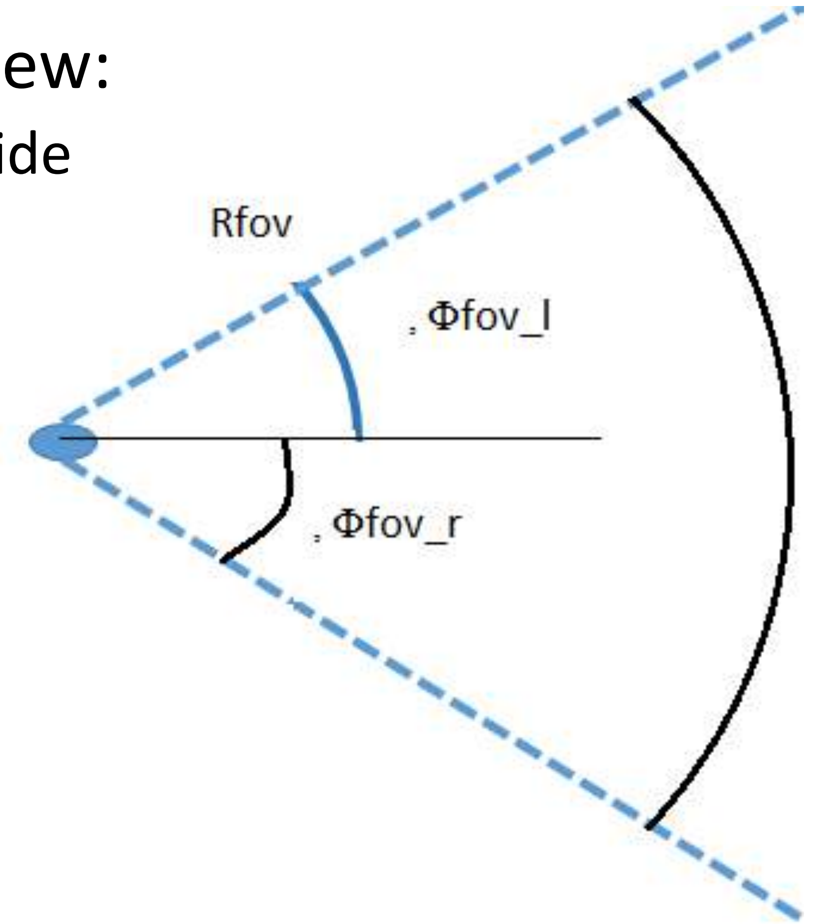
- Robot and Obstacles are assumed circular.
- Radius of robot is added to radius of obstacles
- The Robot is reduced to a point, while Obstacles are equally enlarged.



(a) Circular robot with circular obstacles. (b) Point robot with enlarged obstacles.

Field of view

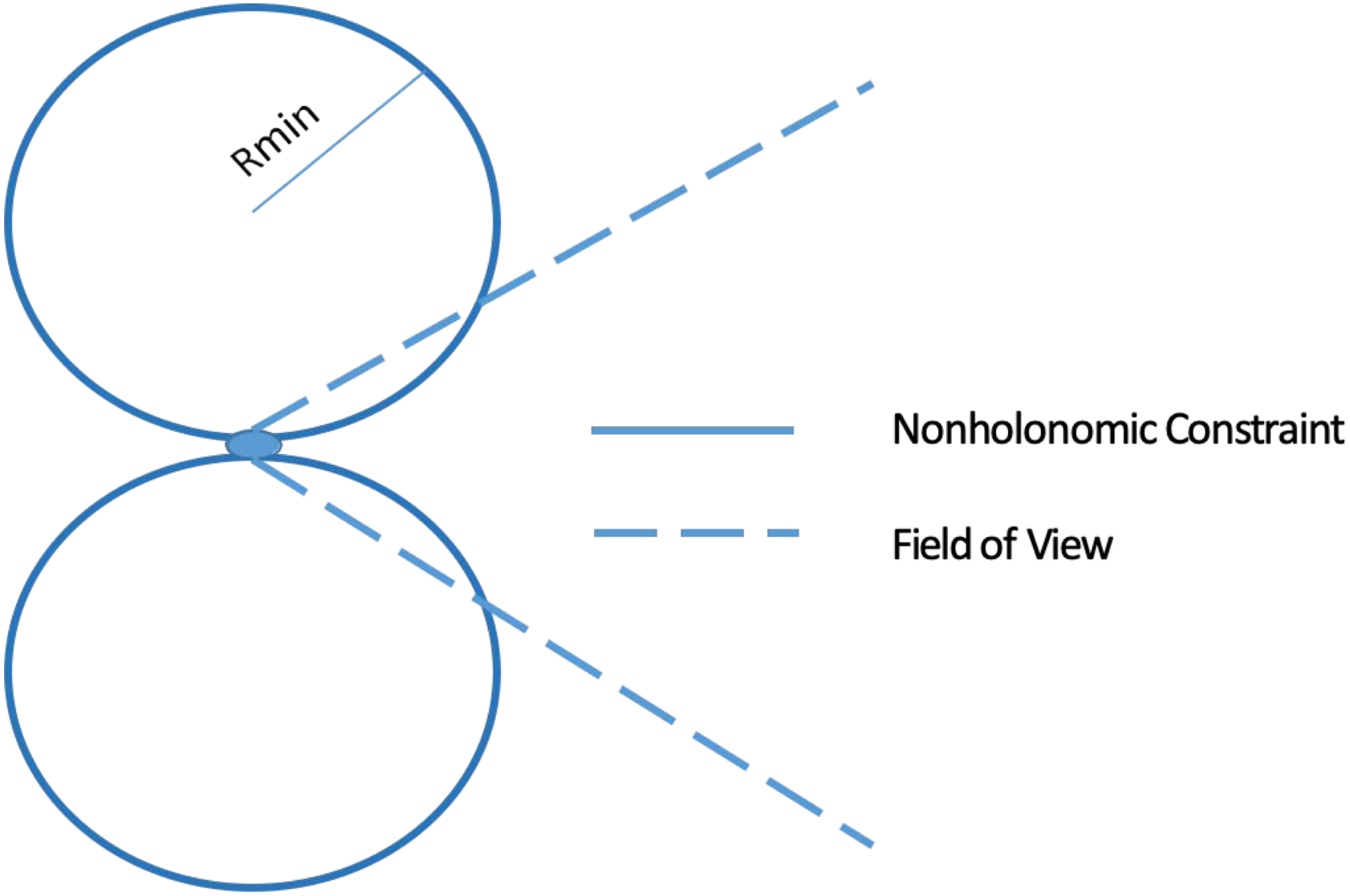
- The sector region within the range of robot's sensors to get information of environment.
- Two quantitative measures of field of view:
  - End angles of the sector on right and left side
  - Radius of the sector.



Nonholonomic Constraints

- If the vector space of the possible motion directions of a mechanical system is restricted
- And the restriction can not be converted into an algebraic relation between configuration variables.
- Can be visualized as, inability of a car like vehicle to move sideways, it is bound to follow an arc to reach a lateral co-ordinate.

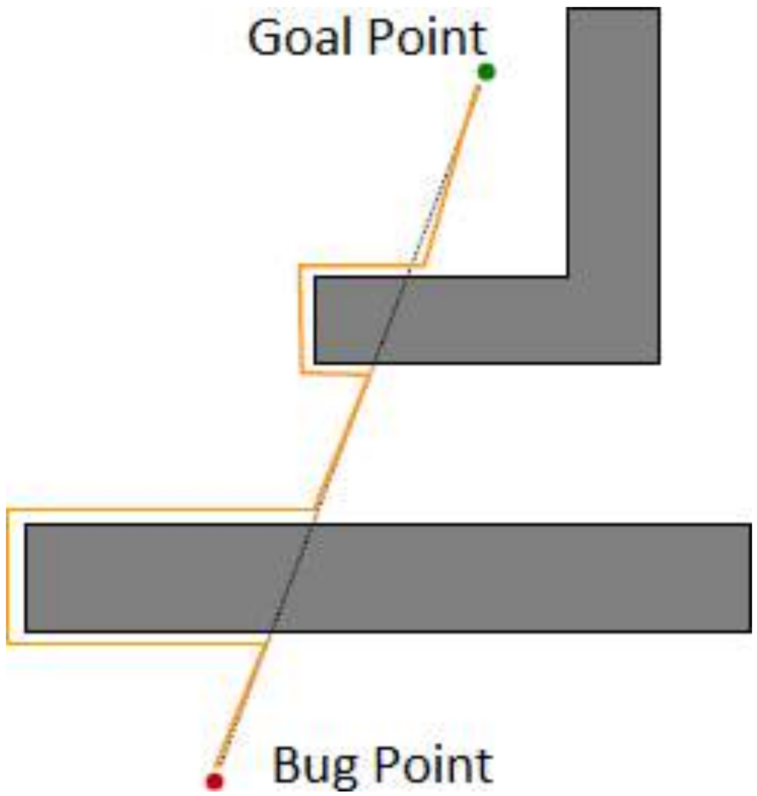
Nonholonomic Constraints and Field of View of Robot



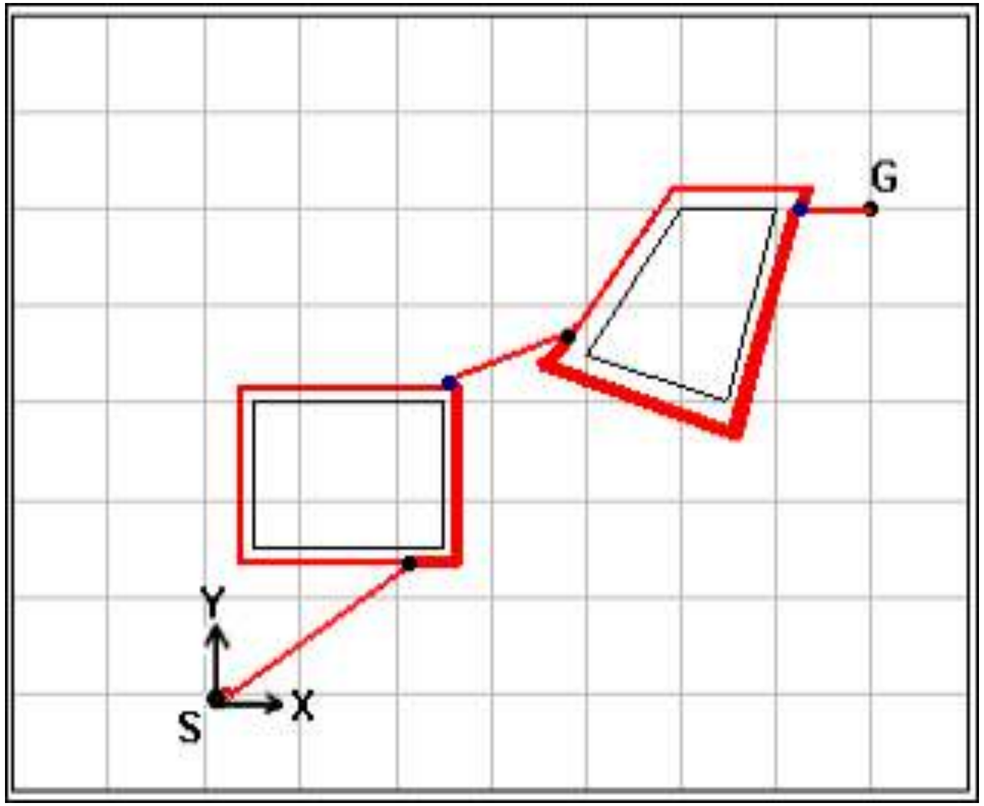
Bug Algorithms

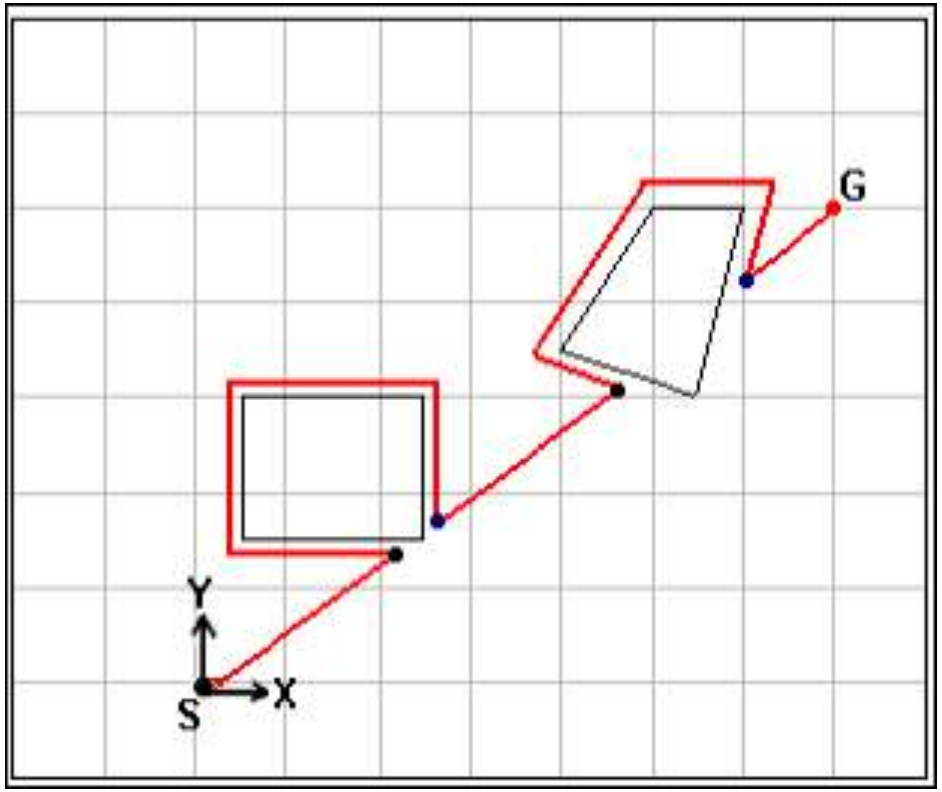
- Common sense approach of moving directly to goal.

- Contour the obstacle when found, until moving straight to goal is possible again.
- Path chosen – often too long
- Robot prone to move close to obstacles



Possible paths with Bug Algorithm





Follow the Gap Method (FGM)

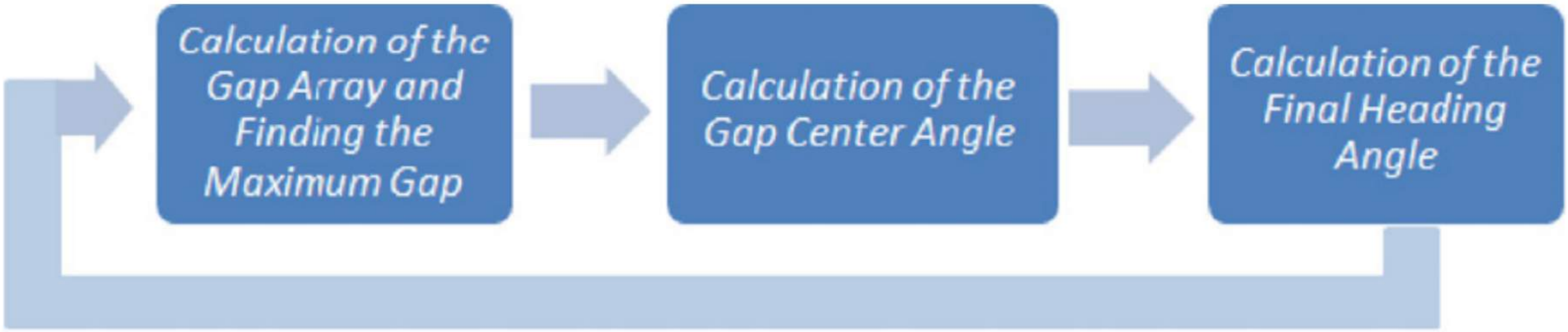


Fig. 3. Steps of the Follow the Gap method.

Follow the Gap Method (FGM)

- Point Robot Approach
- Obstacle representation
- Construction a gap array among obstacles.
- Determination of maximum gap, considering the Goal point location.
- Calculation of angle to Center of Maximum gap
- Robot proceeds to center of maximum gap.

Problem Definition

- The Algorithm
  - Should find a purely reactive heading to achieve goal co-ordinates
  - Should avoiding obstacles with as large distance as possible
  - Should consider measurement and nonholonomic constraints
  - for obstacle avoidance must collaborate with global planner

- Goal point – obtained from the global planner
- Obstacle co-ordinates - change with time

Point Robot Approach

Xrob = Abscissa of robot point

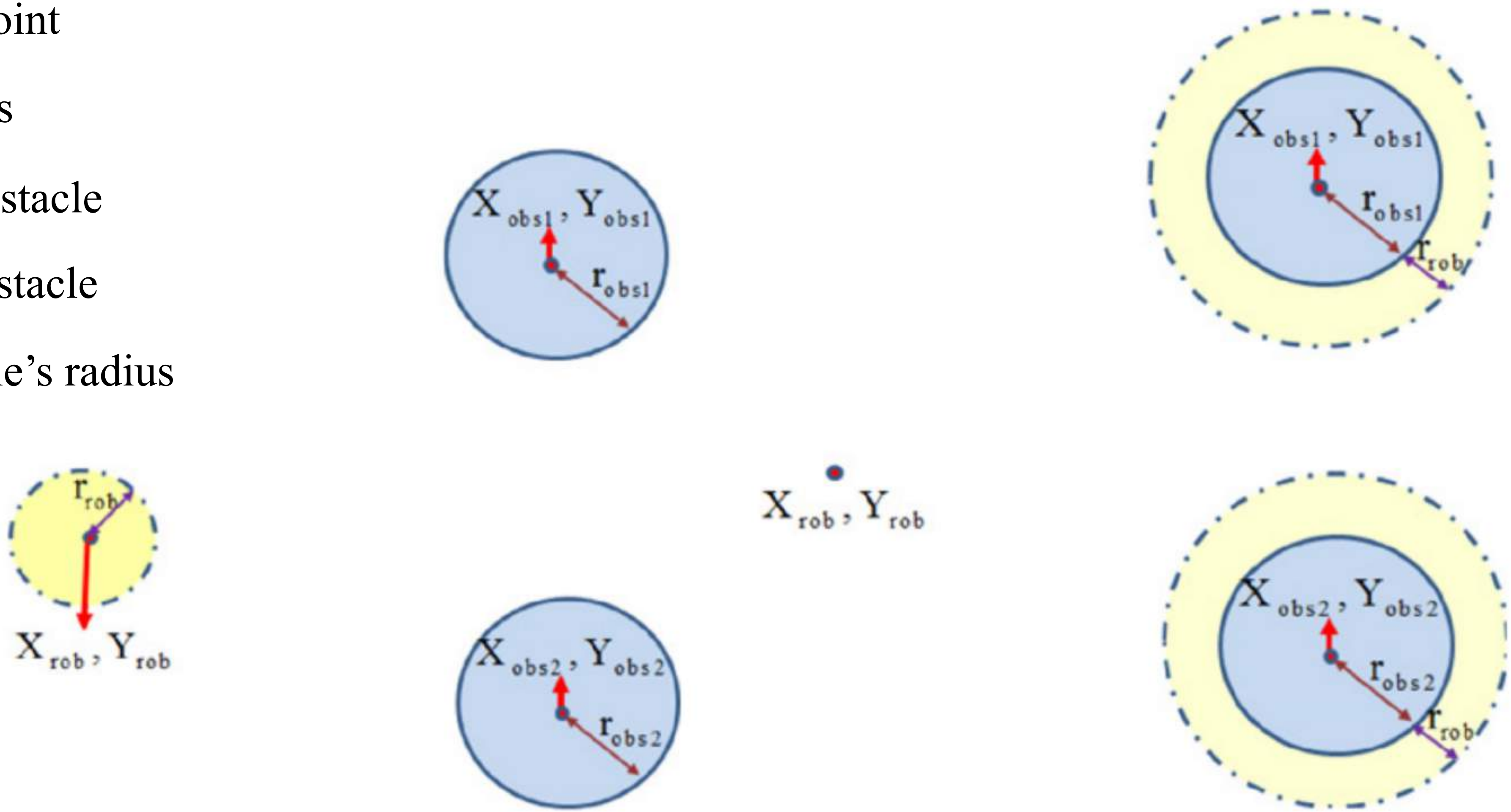
Yrob = Ordinate of robot point

Rrob = Robot circle's radius

Xobsn = Abscissa of nth obstacle

Yobsn = Ordinate of nth obstacle

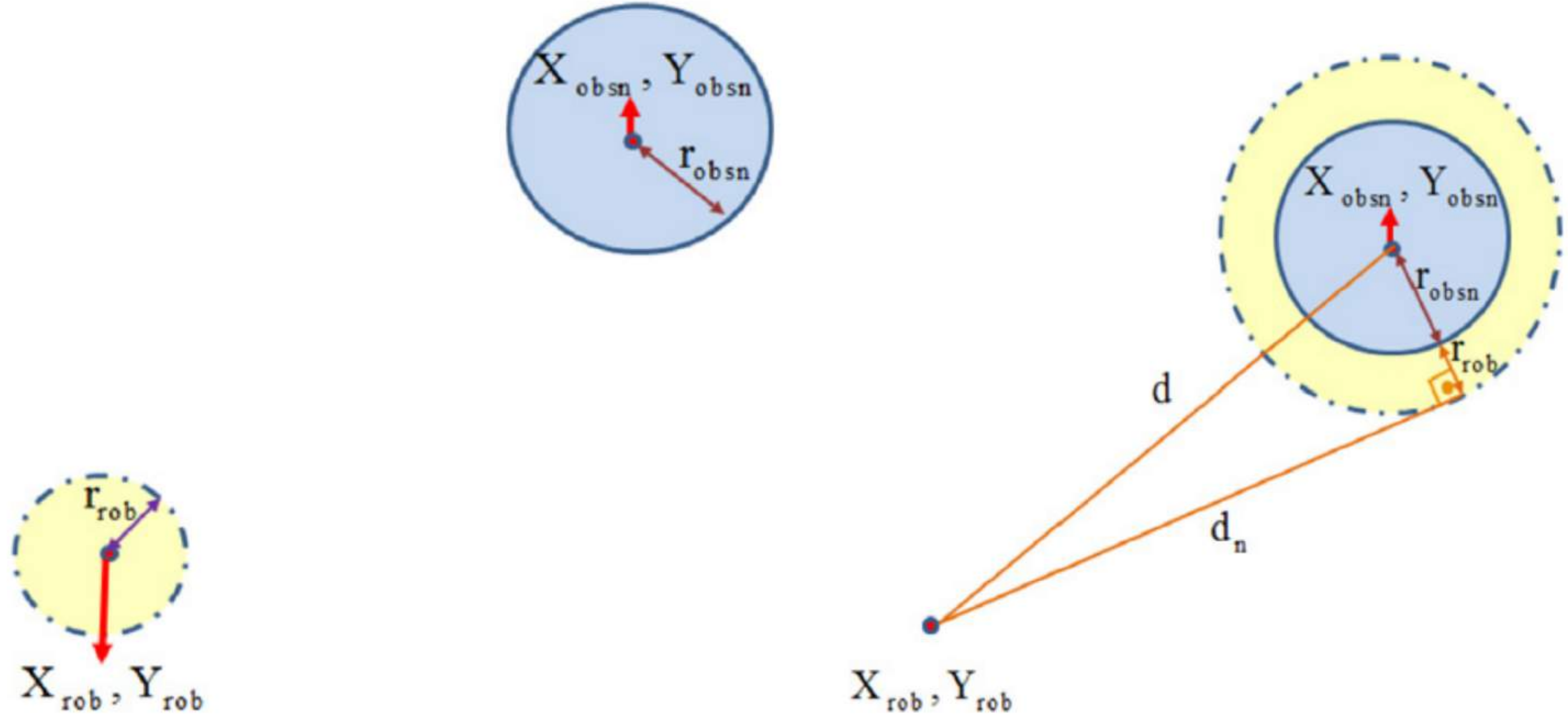
Robsn = nth obstacle's circle's radius



(a) Circular robot with circular obstacles. (b) Point robot with enlarged obstacles.

Distance to Obstacle

$$d = \sqrt{(X_{obsn} - X_{rob})^2 + (Y_{obsn} - Y_{rob})^2}$$
$$d_n^2 + (r_{obsn} + r_{rob})^2 = d^2$$
$$\Rightarrow d_n = \sqrt{(X_{obsn} - X_{rob})^2 + (Y_{obsn} - Y_{rob})^2 - (r_{obsn} + r_{rob})^2}.$$

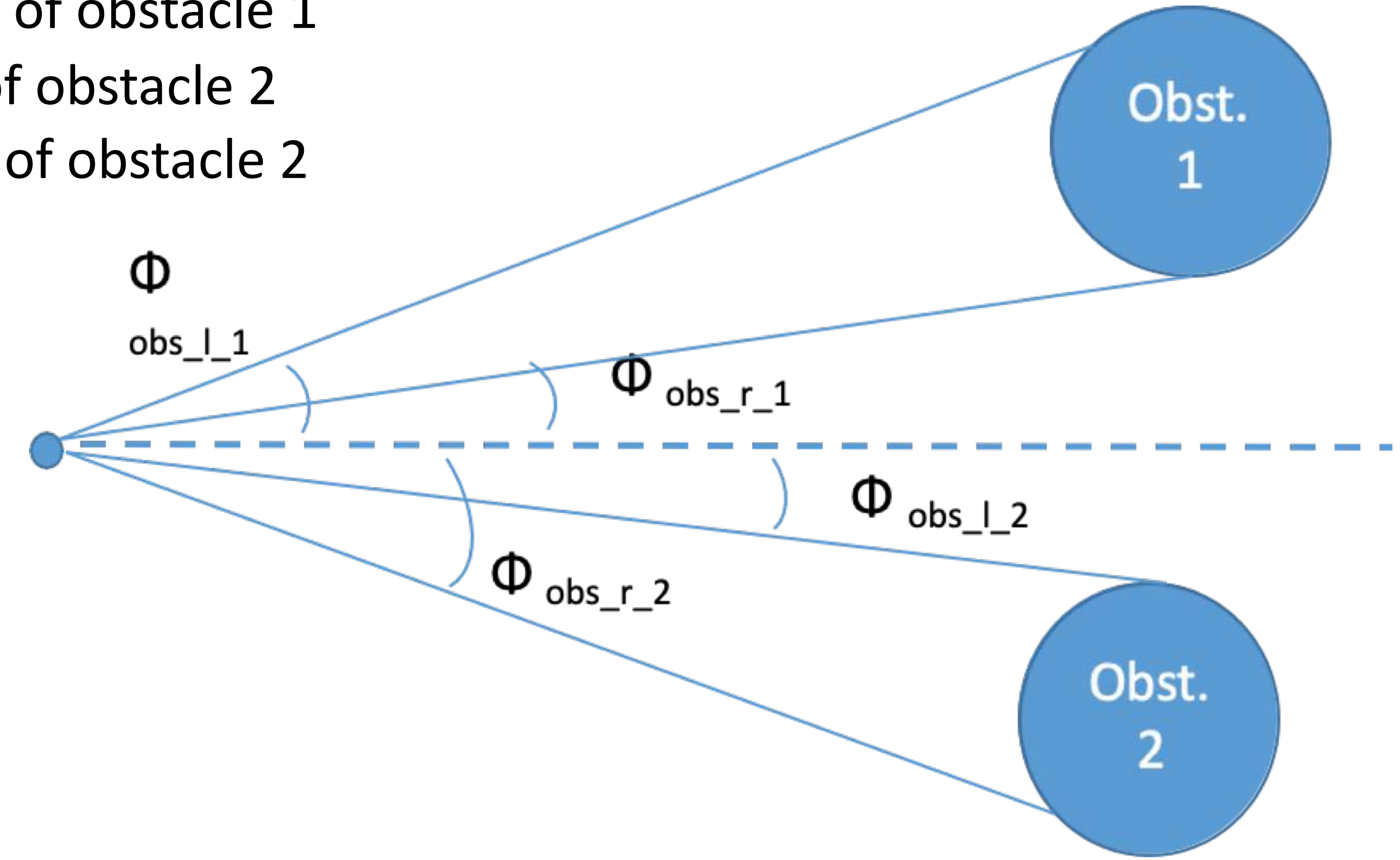


(a) Circular robot and circular obstacle parameters.

(b) Distance to obstacle geometry.

Obstacle Representation

- Two parameter representation
  - $\Phi_{obs\_l\_1}$ – Border left angle of obstacle 1
  - $\Phi_{obs\_r\_1}$ -- Border right angle of obstacle 1
  - $\Phi_{obs\_l\_1}$ – Border left angle of obstacle 2
  - $\Phi_{obs\_l\_1}$ – Border right angle of obstacle 2



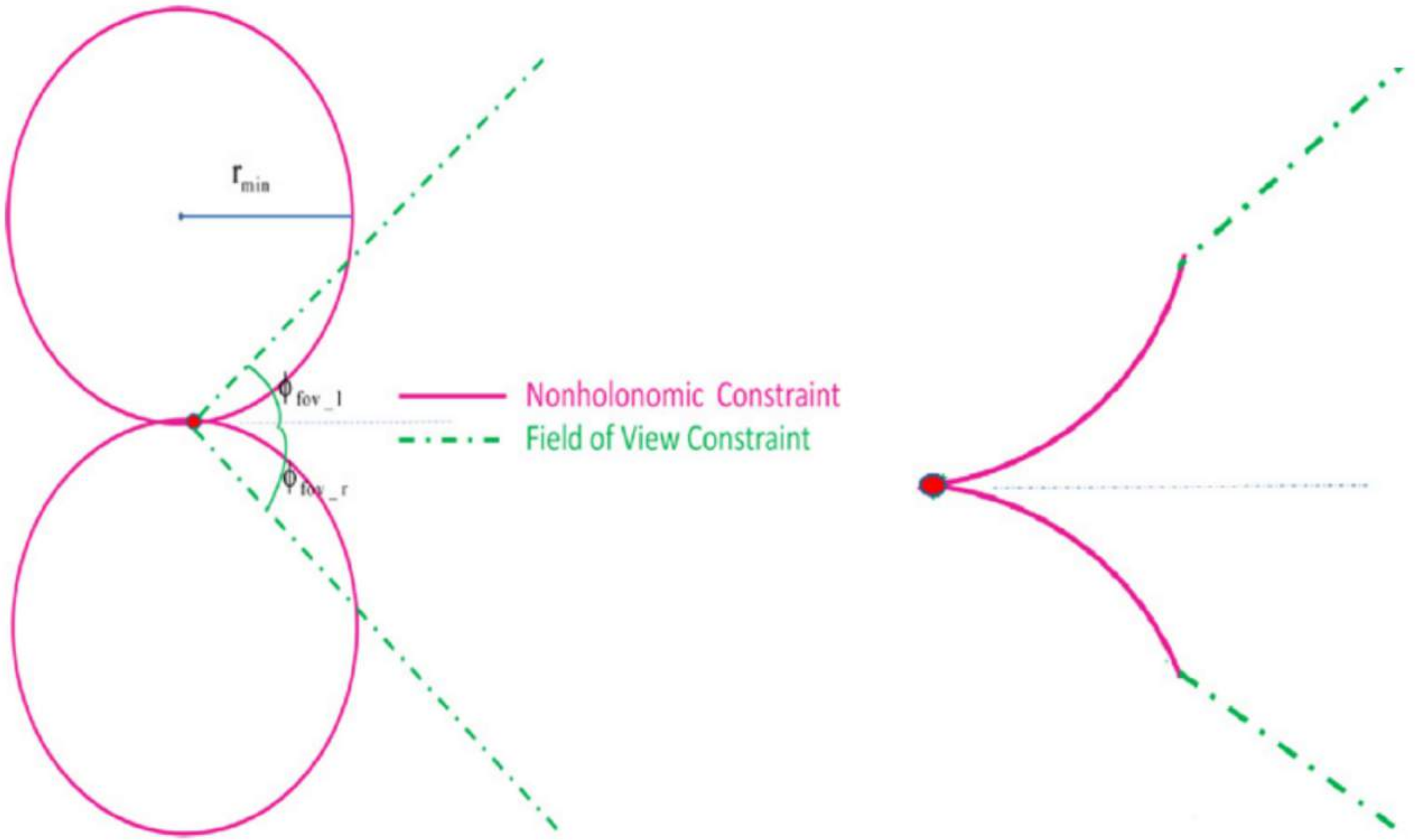
Gap Border Evaluation

In order to understand which boundary is active for a boundary obstacle, *decision rule* are illustrated as follows:

$$d_{nhol} < d_{fov} \Rightarrow \phi_{\lim} = \phi_{nhol}$$
$$d_{nhol} \geq d_{fov} \Rightarrow \phi_{\lim} = \phi_{fov}$$

where

$\phi_{\lim}$: Gap border angle. ($\phi_{\lim\_l}$



(a) Field of view and nonholonomic movement constraint separately.

(b) Gap border.

Gap boarder parameters

1. Φlim: Gap border angle

2. Φnhol: Border angle coming from nonholonomic constraint

3. Φfov: Border angle coming from field of view

4. dnhol: Nearest distance between nonholonomic constraint arc and obstacle border

5. dfov: Nearest distance between field of view line and obstacle border

Gap border parameters

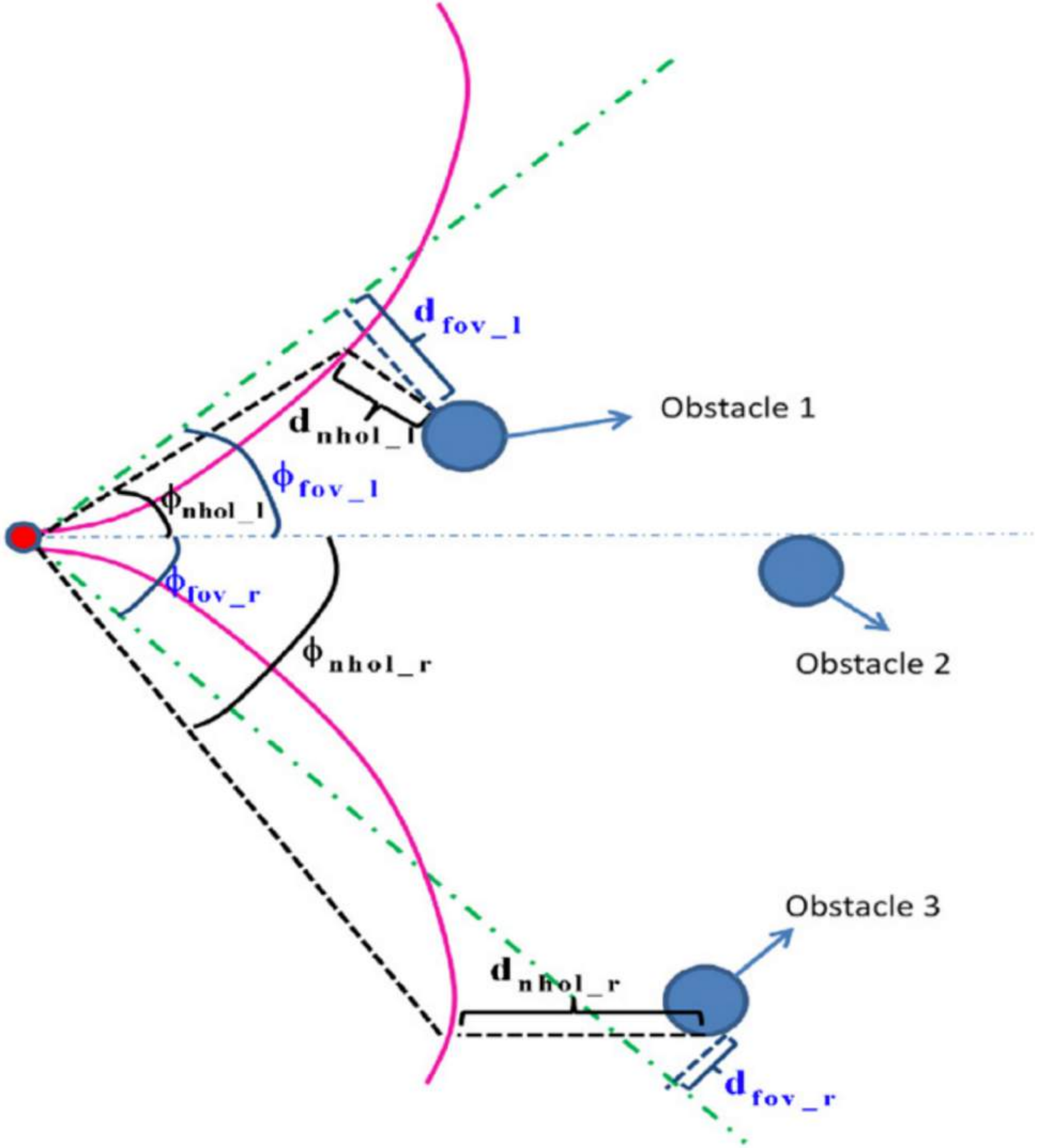
1. Φlim: Gap border angle

2. Φnhol: Border angle coming from nonholonomic constraint

3. Φfov: Border angle coming from field of view

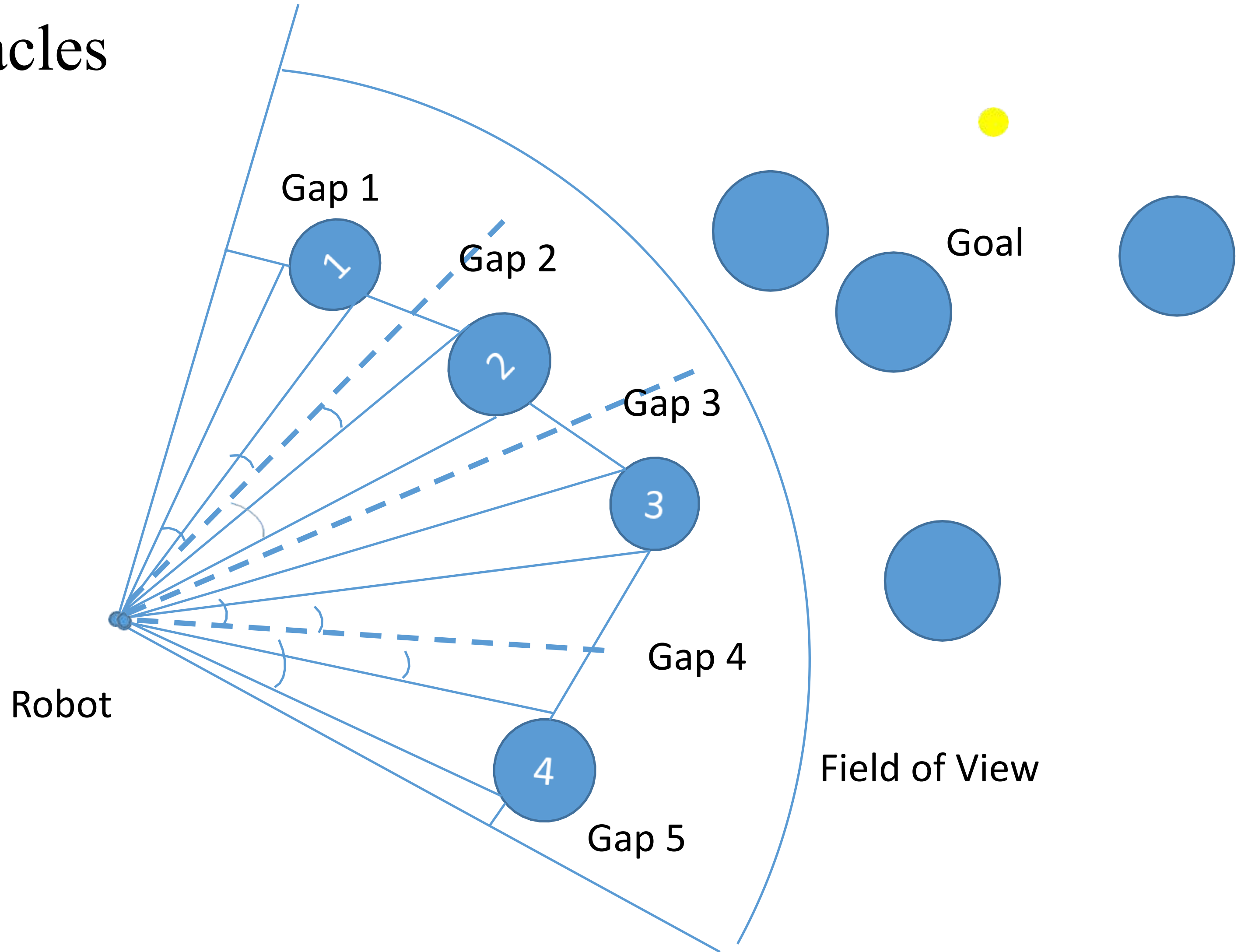
4. dnhol: Nearest distance between nonholonomic constraint arc and obstacle border

5. dfov: Nearest distance between field of view line and obstacle border



Construction of gap array

- N + 1 gaps for N obstacles

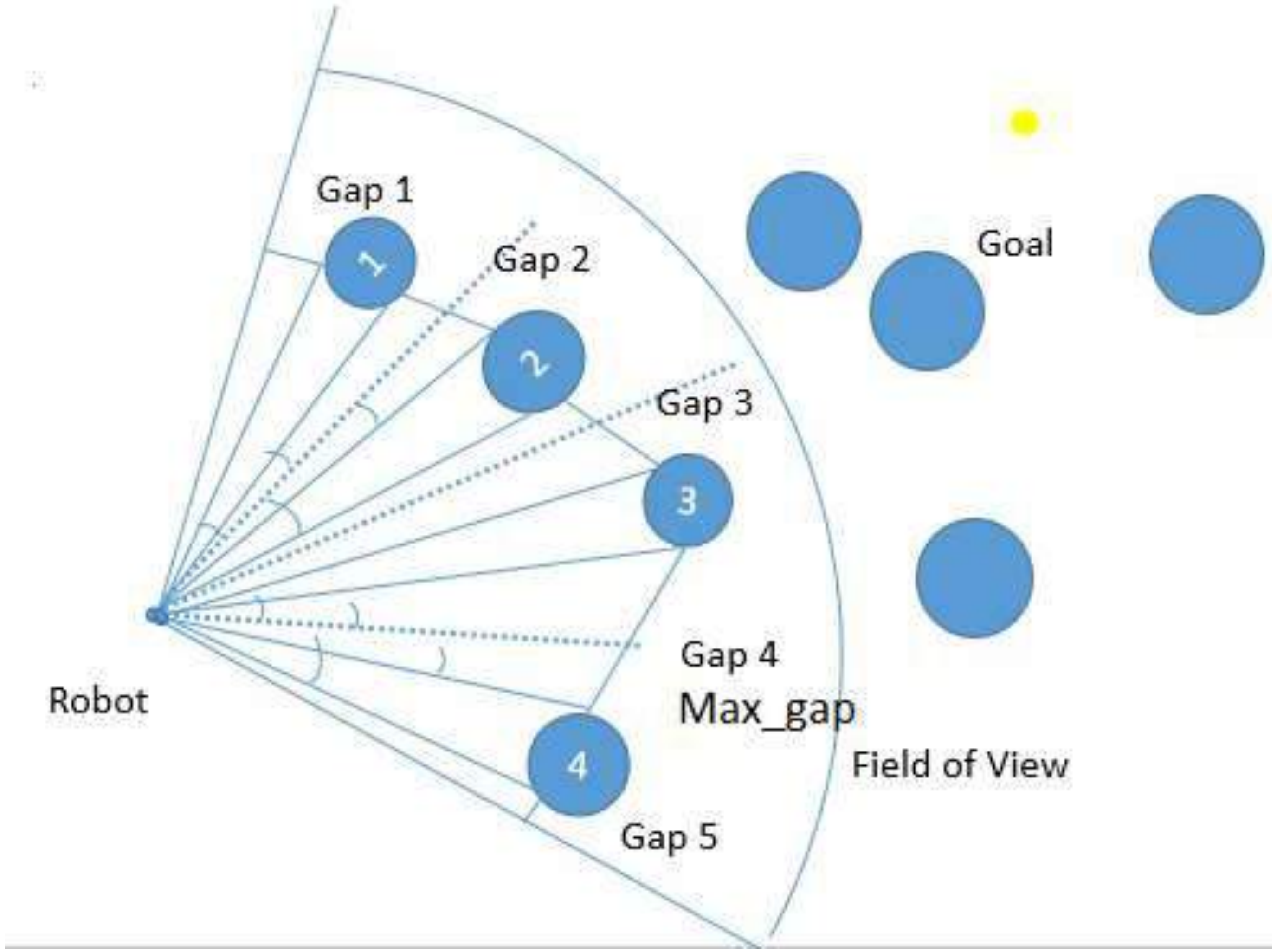


Gap array and Maximum Gap

- Gap[N+1] = [(Φlim_l – Φobs1_l)(Φobs1_r – Φobs2_l)……(Φobs(n- 1)_r –Φobs(n-1)_l)(Φobsn_r – Φlim_r)]

- Maximum gap is determined with a sorting algorithm in program.

Gap array and Maximum Gap



Follow the Gap Method (FGM)

- Point Robot Approach
- Obstacle representation
- Construction a gap array among obstacles.
- Determination of maximum gap, considering the Goal point location.
- Calculation of angle to Center of Maximum gap
- Robot proceeds to center of maximum gap.

Gap Center angle Calculation

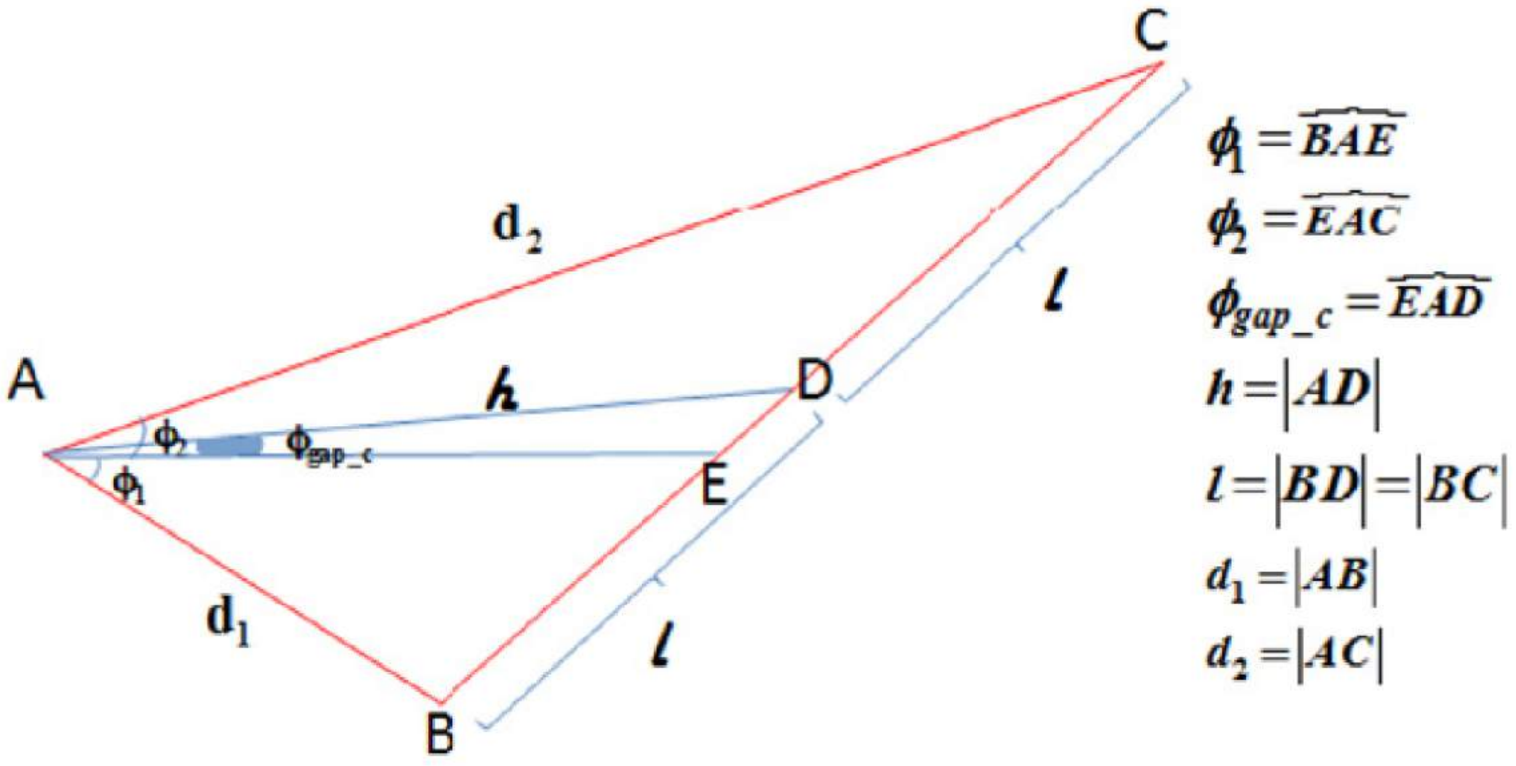


Fig. 8. Gap center angle parameterization.

Firstly, the Cosine Rule is applied to the ABC triangle:

$$(2l)^2 = d_1^2 + d_2^2 - 2d_1 d_2 \cos(\phi_1 + \phi_2)$$
$$l^2 = \frac{d_1^2 + d_2^2 - 2d_1 d_2 \cos(\phi_1 + \phi_2)}{4}.$$

After that, the Apollonius theorem is applied to the ABC triangle.

$$d_1^2 + d_2^2 = 2l^2 + 2h^2$$

Gap center angle

- The gap center angle (φgap_c) is found in terms of the measurable d1, d2, φ1, φ2 parameters

$$\phi_{gap\_c} = \arccos\left(\frac{d_1 + d_2\cos(\phi_1 + \phi_2)}{\sqrt{d_1^2 + d_2^2 + 2d_1d_2\cos(\phi_1 + \phi_2)}}\right) - \phi_1$$

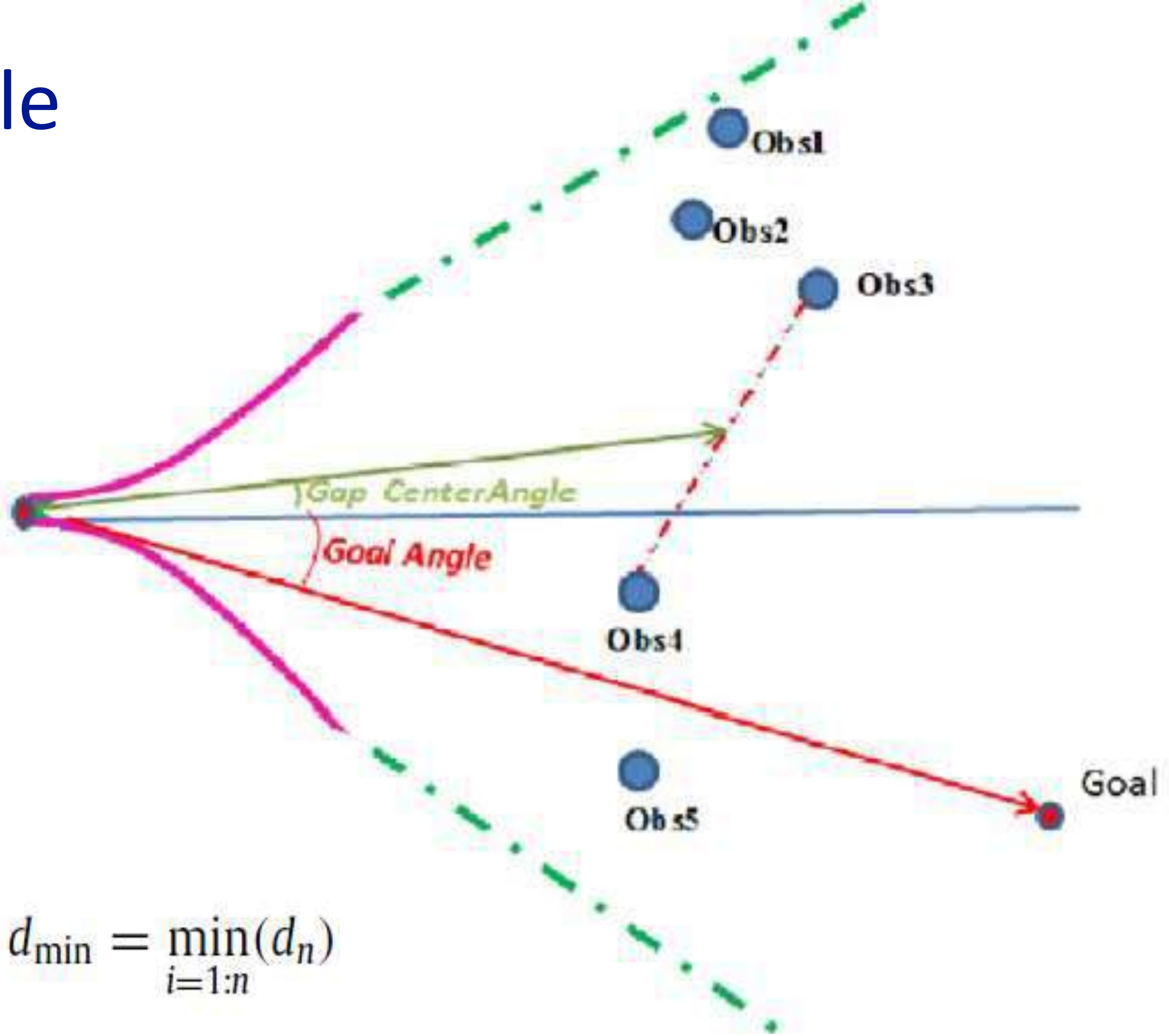
Follow the Gap Method (FGM)

- Point Robot Approach
- Obstacle representation
- Construction a gap array among obstacles.
- Determination of maximum gap, considering the Goal point location.
- Calculation of angle to Center of Maximum gap
- Robot proceeds to center of maximum gap.

Calculation of final heading angle

- Final angle is Combination of angle of center of maximum gap and Goal point angle.
- Determined by fusing weighted average function of gap center angle and goal angle.
- α is the weight to obstacle gap.
- α acts as tuning parameter for FGM.
- ß weight to goal point (assumed 1 for simplicity)
- dmin is minimum distance to the approaching obstacle.

Final Heading Angle



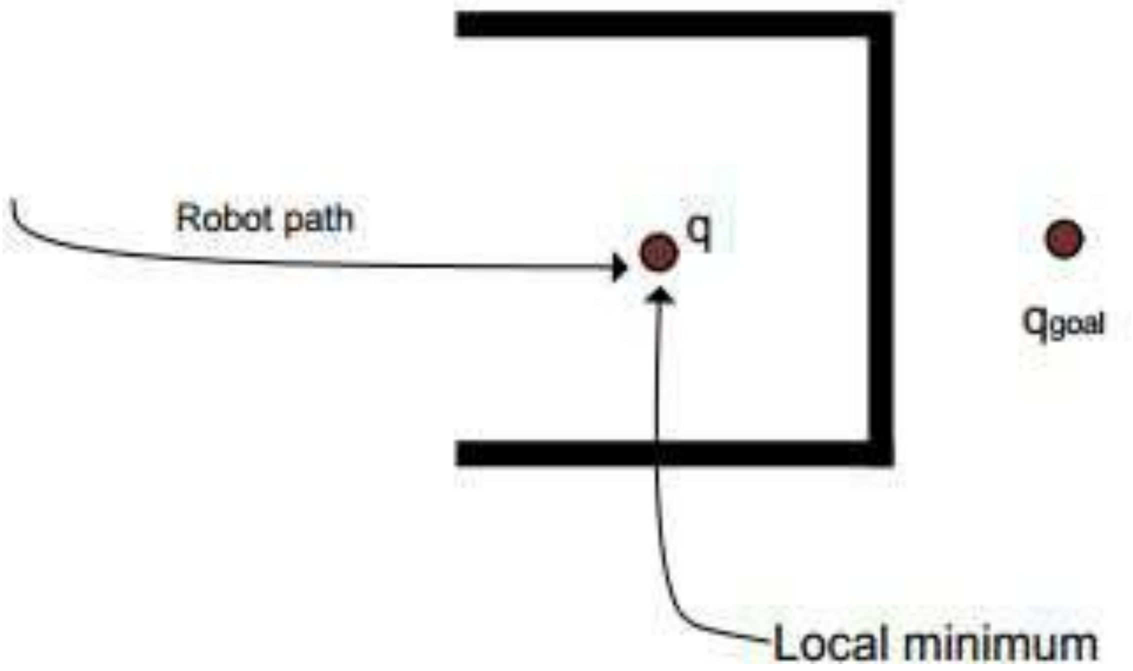
$$\phi_{final} = \frac{\frac{\alpha}{d_{\min}}\phi_{gap\_c} + \phi_{goal}}{\frac{\alpha}{d_{\min}} + 1} \quad \text{where } d_{\min} = \min_{i=1:n}(d_n)$$

Role of α value

- Weightage to gap angle is α/dmin
- α makes the path goal oriented or gap oriented.
- For α= 0, ϕfinal is equal to ϕgoal
- Increasing values of alpha brings ϕfinal closer to ϕgap_c and vice versa

Dead end Scenario

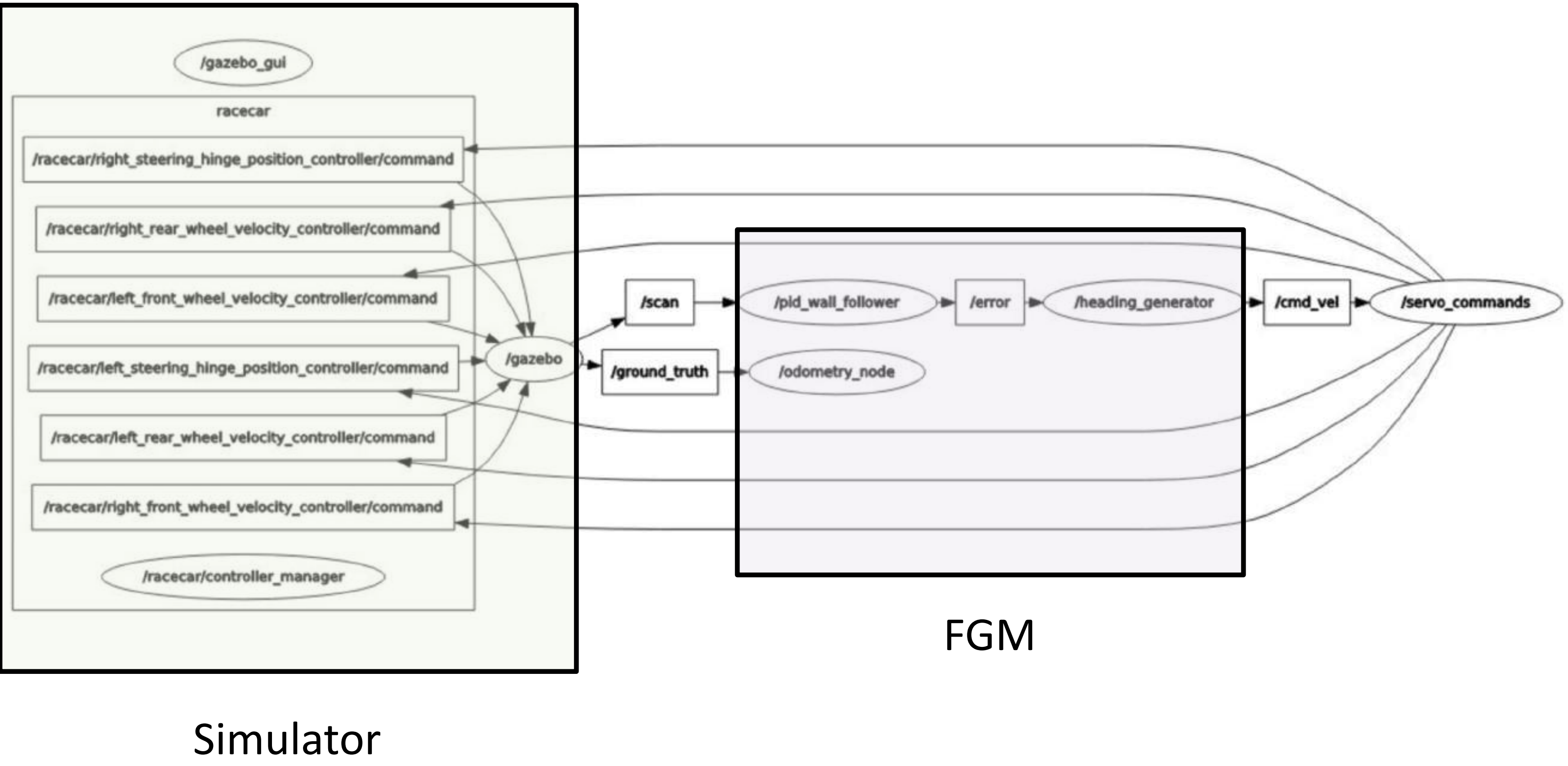
- A dead-end scenario of U-shaped obstacles is a problem for FGM as it is for APF as both are more sort of local planners.
- It needs upper level of intelligence.
- Can be solved by approaches like Virtual Obstacle Method, Multiple Goal Point method etc.



Advantages of FGM

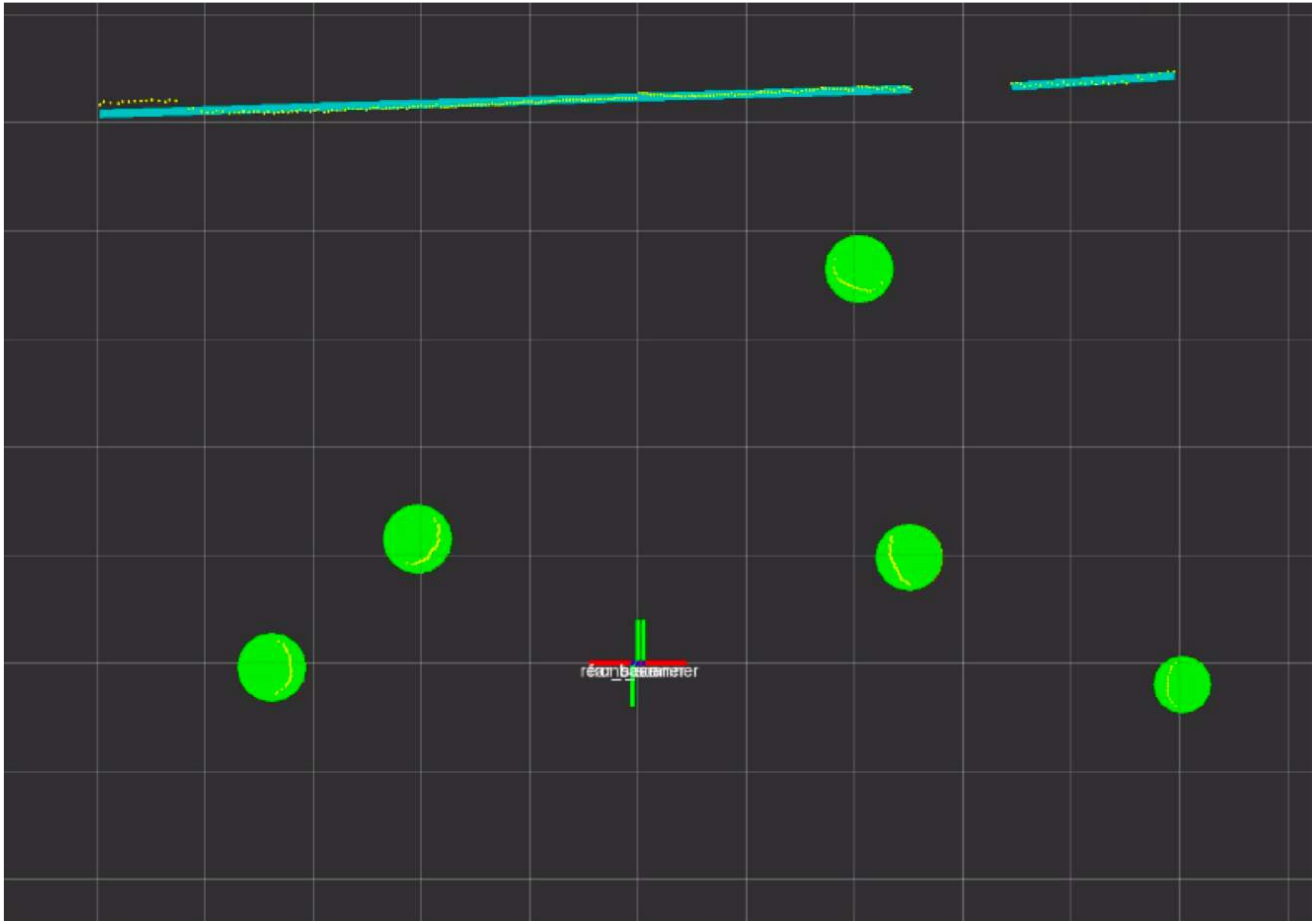
- Single tuning parameter (α) in weightage to gap center angle (α/dmin)
- Considers nonholonomic constraints for the robot.
- Only feasible trajectories are generated, lesser ambiguity to decision, lesser computation time.
- Field of view of robot is taken into account.
- Robot does not move in unmeasured directions.
- Passage through maximum gap center – Safest path.

Follow the Gap navigation and planning on the F1/10 Car



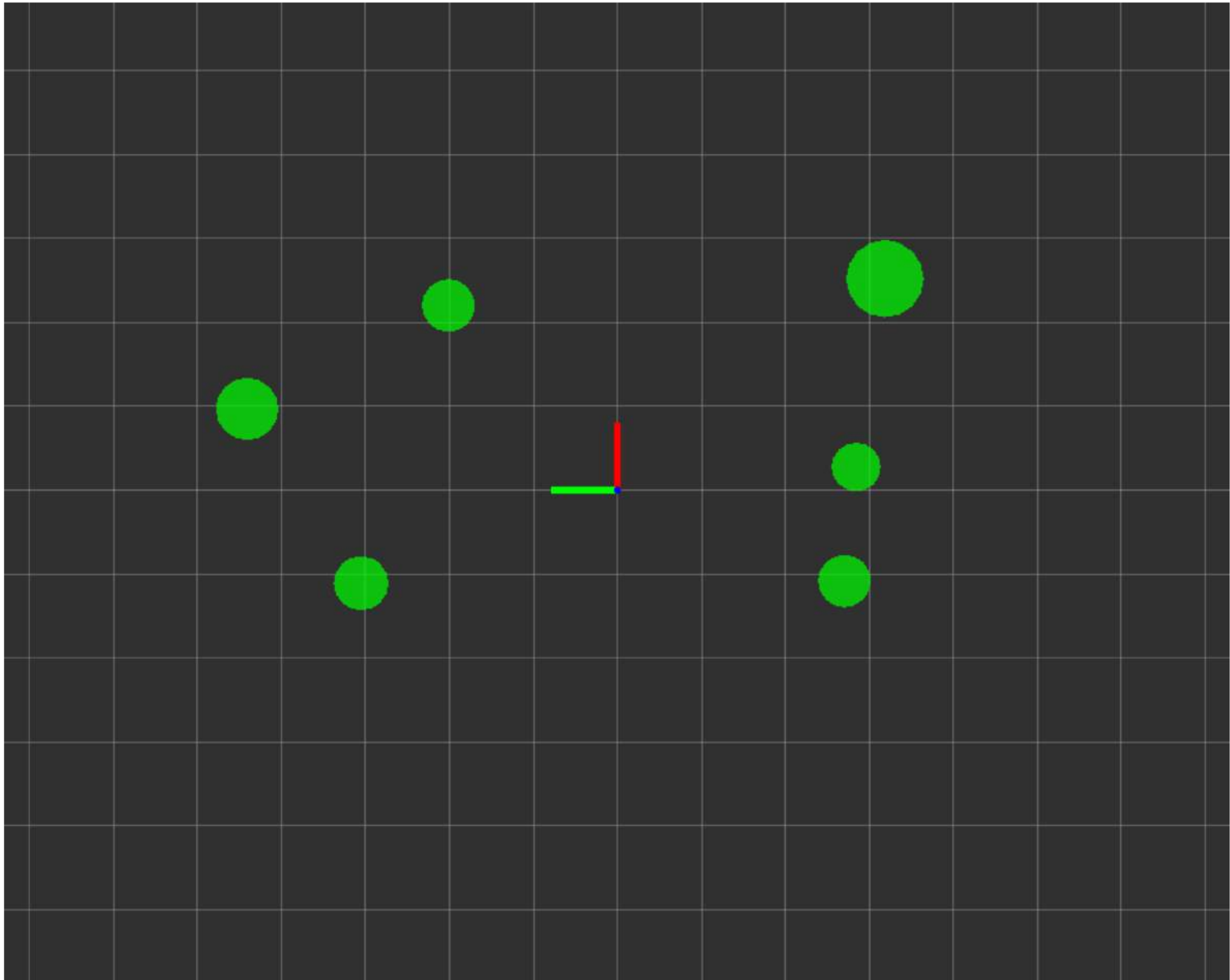
https://github.com/tysik/obstacle_detector

Fig. 1. Visual example of obstacle detector output.



https://github.com/tysik/obstacle_detector

Visual example of `obstacle_tracker` output.




Head-to-head Autonomous Racing
4th F1/10 International Autonomous Racing Competition
F1
TENTH
f1tenth.org