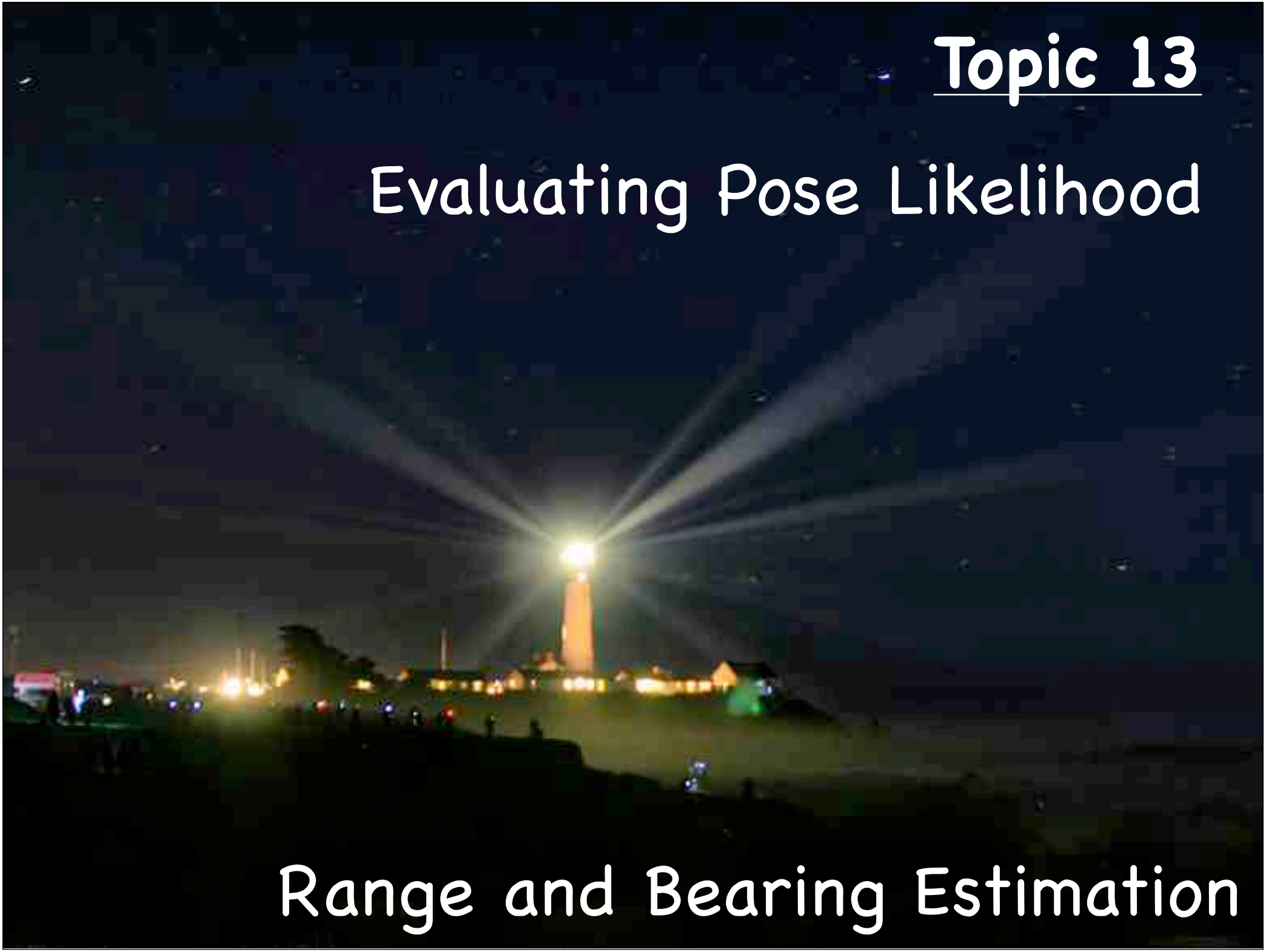

Topic 13
Evaluating Pose Likelihood
Range and Bearing Estimation

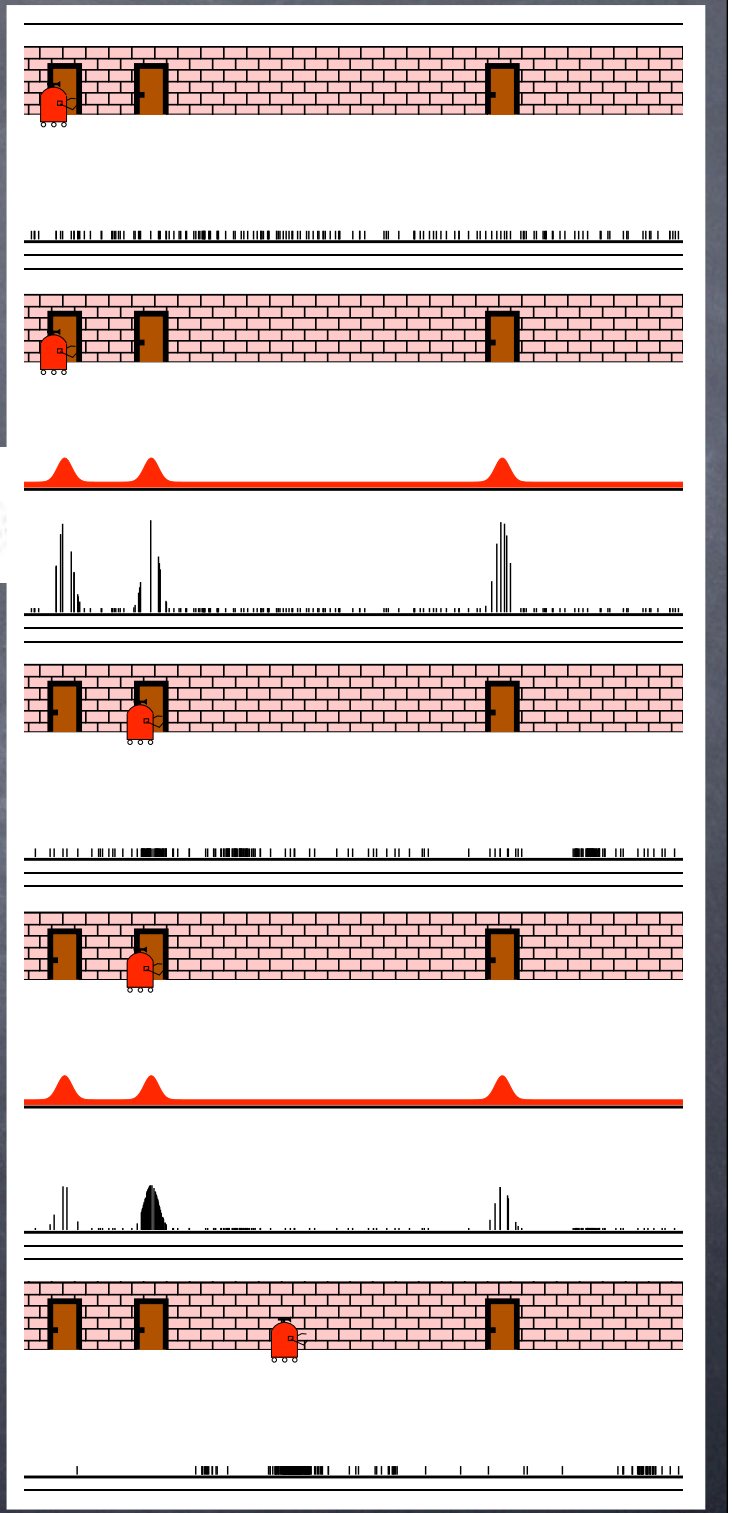
Particle filter recap

posterior likelihood dynamics prior

$$p(\mathbf{x}_k|\mathbf{Z}_k) = \alpha\, p(\mathbf{Z}_k|\mathbf{x}_k) \int p(\mathbf{x}_k|\mathbf{x}_{k-1}) p(\mathbf{x}_{k-1}|\mathbf{Z}_{k-1})$$

"belief at t=k" "update" "predict" "belief at t=k-1"

- Bayes filter model to estimate true pose from all possible poses
- recursive Markovian inference over time with predict-update loop
- Particle filter algorithms predict-update with sample set of poses



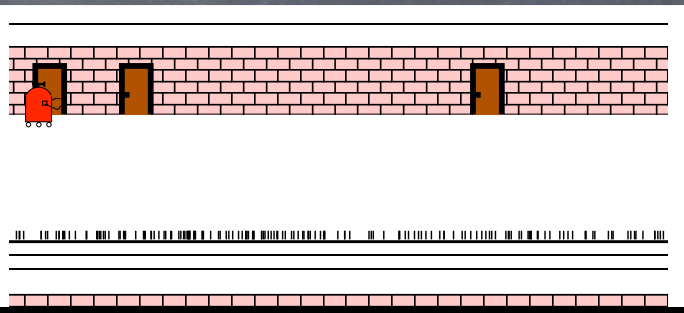
Particle filter recap

posterior likelihood dynamics prior

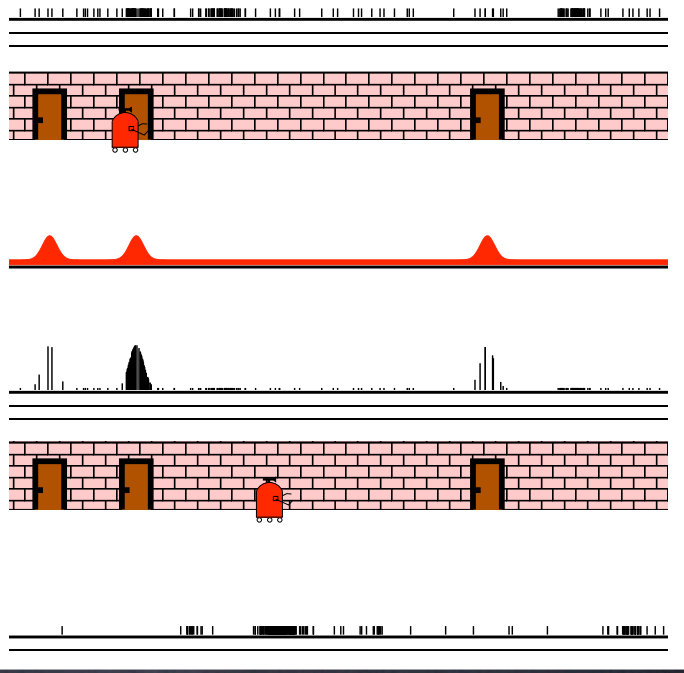
$$p(\mathbf{x}_k|\mathbf{Z}_k) = \alpha\, p(\mathbf{Z}_k|\mathbf{x}_k) \int p(\mathbf{x}_k|\mathbf{x}_{k-1}) p(\mathbf{x}_{k-1}|\mathbf{Z}_{k-1})$$

"belief at t=k" "update" "predict" "belief at t=k-1"

- Bayes filter model to estimate true pose from all possible poses
- recursive Markovian inference over time with predict-update loop
- Particle filter algorithms predict-update with sample set of poses



What are we still missing to compute localization?



Particle filter recap

posterior likelihood dynamics prior

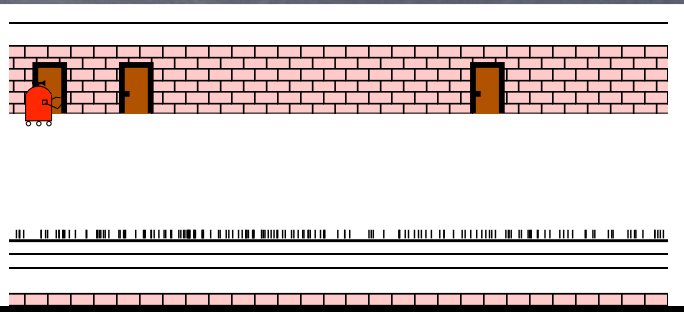
$$p(\mathbf{x}_k|\mathbf{Z}_k) = \alpha\, p(\mathbf{Z}_k|\mathbf{x}_k) \int p(\mathbf{x}_k|\mathbf{x}_{k-1})\, p(\mathbf{x}_{k-1}|\mathbf{Z}_{k-1})$$

"belief at t=k" "update" "predict" "belief at t=k-1"

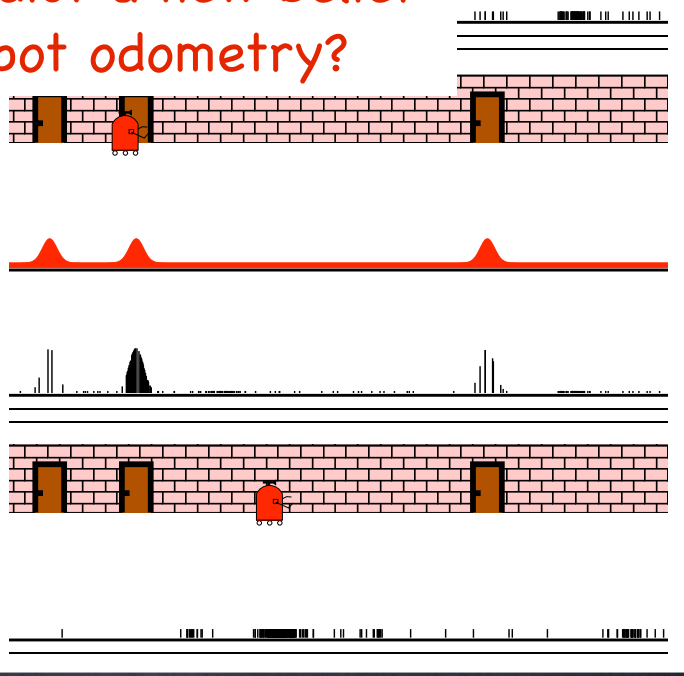
How to evaluate the likelihood of a pose given robot observations?

How to predict a new belief given robot odometry?

- recursive Markovian inference over time with predict-update loop
- Particle filter algorithms predict-update with sample set of poses



What are we still missing to compute localization?



Particle filter recap

posterior likelihood dynamics prior

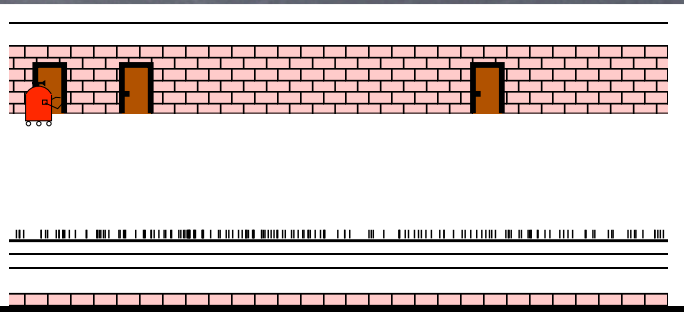
$$p(\mathbf{x}_k|\mathbf{Z}_k) = \alpha\, p(\mathbf{Z}_k|\mathbf{x}_k) \int p(\mathbf{x}_k|\mathbf{x}_{k-1})\, p(\mathbf{x}_{k-1}|\mathbf{Z}_{k-1})$$

"belief at t=k" "update" "predict" "belief at t=k-1"

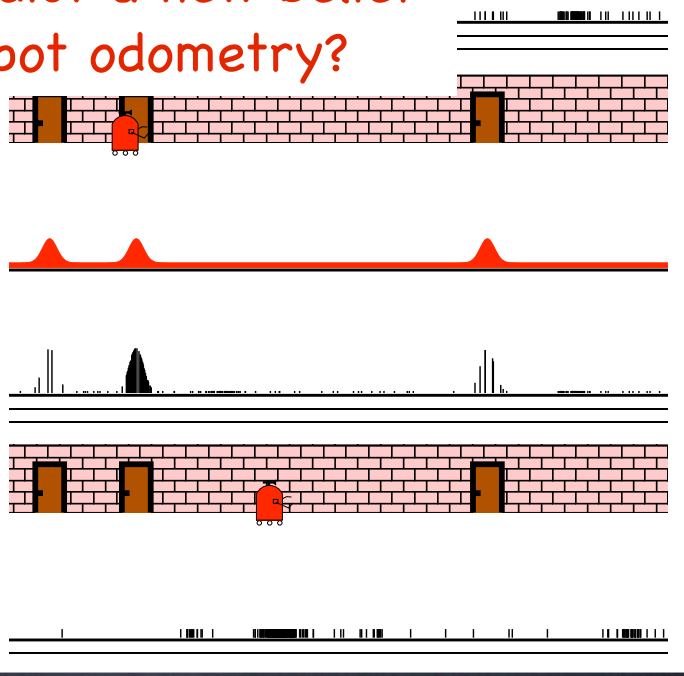
How to evaluate the likelihood of a pose given robot observations?

How to predict a new belief given robot odometry?

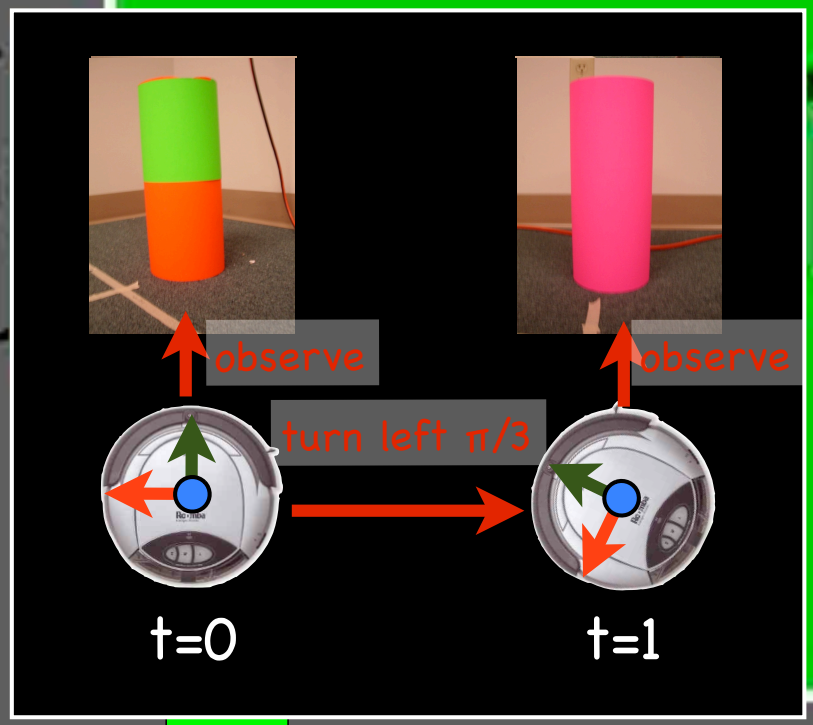
- recursive Markovian inference over time with predict-update loop
- Particle filter algorithms predict-update with sample set of poses



What are we still missing to compute localization?



suppose the robot starts by seeing this landmark, then turns left, and then sees another landmark



Particle filter steps

1.(Re)sample particles

2.Predict forward in time

3.Update particle weights

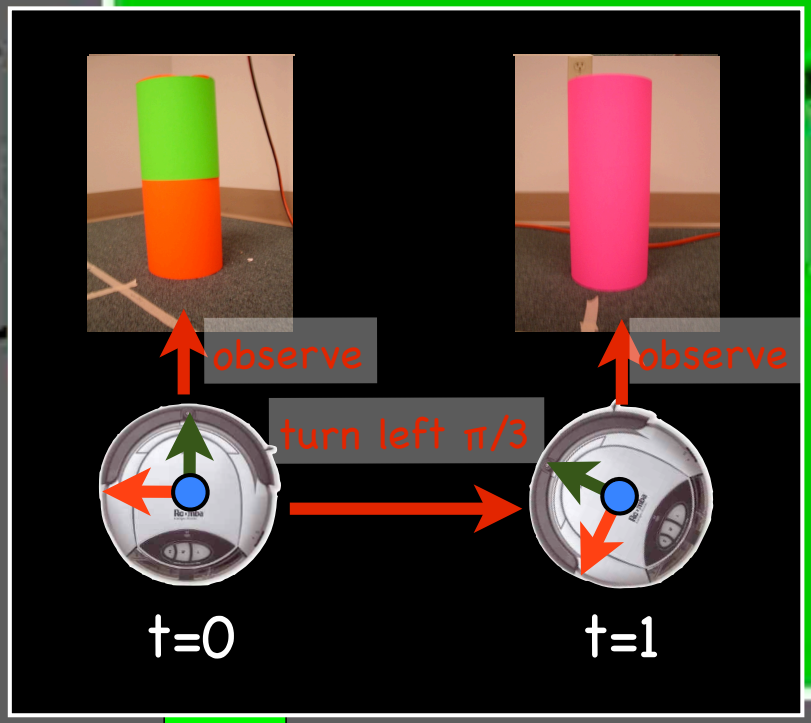
4.Repeat

70

260

suppose the robot starts by seeing this landmark, then turns left, and then sees another landmark

Initially, the robot has no observations



Particle filter steps

1.(Re)sample particles

2.Predict forward in time

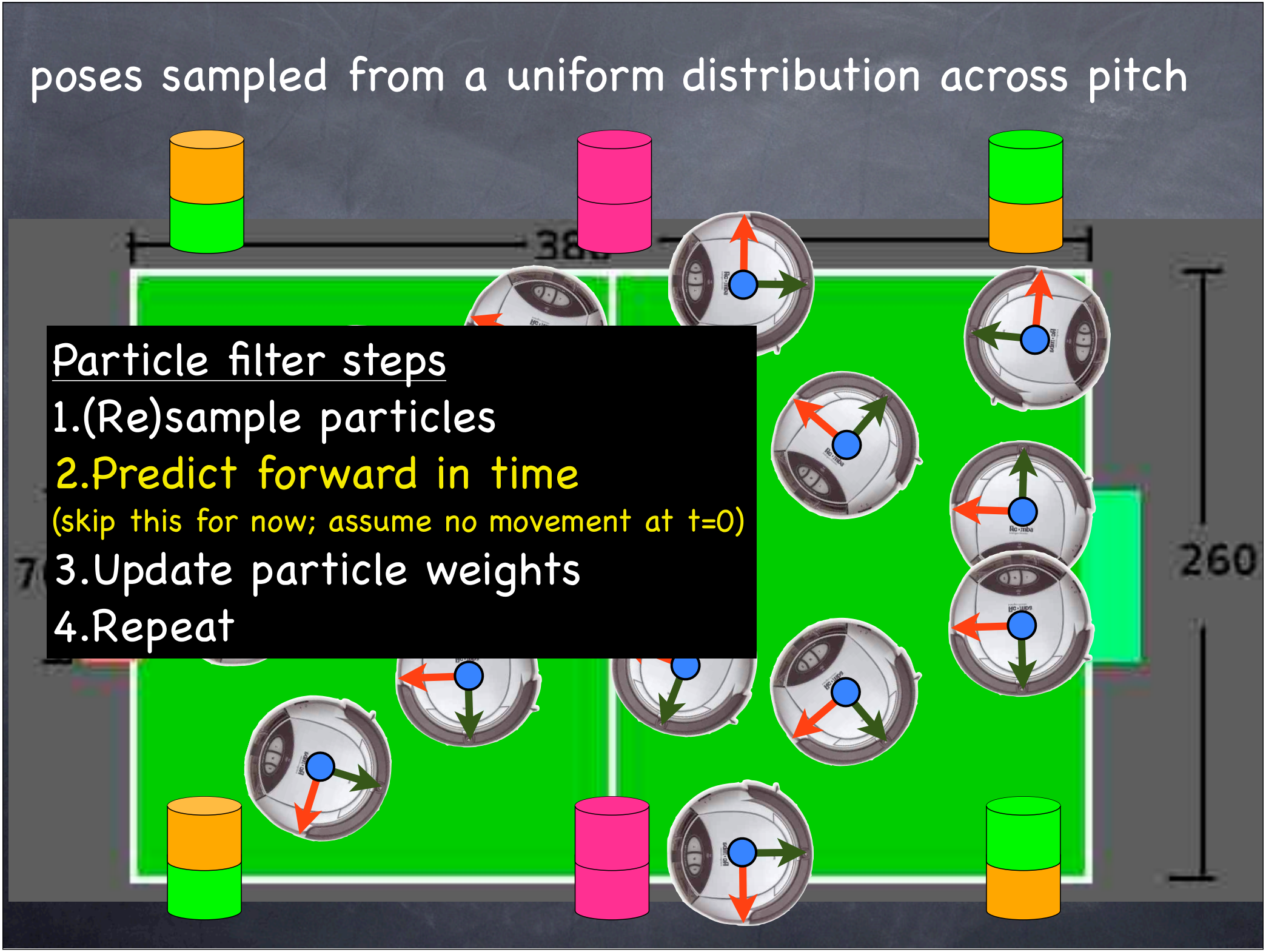
3.Update particle weights

4.Repeat

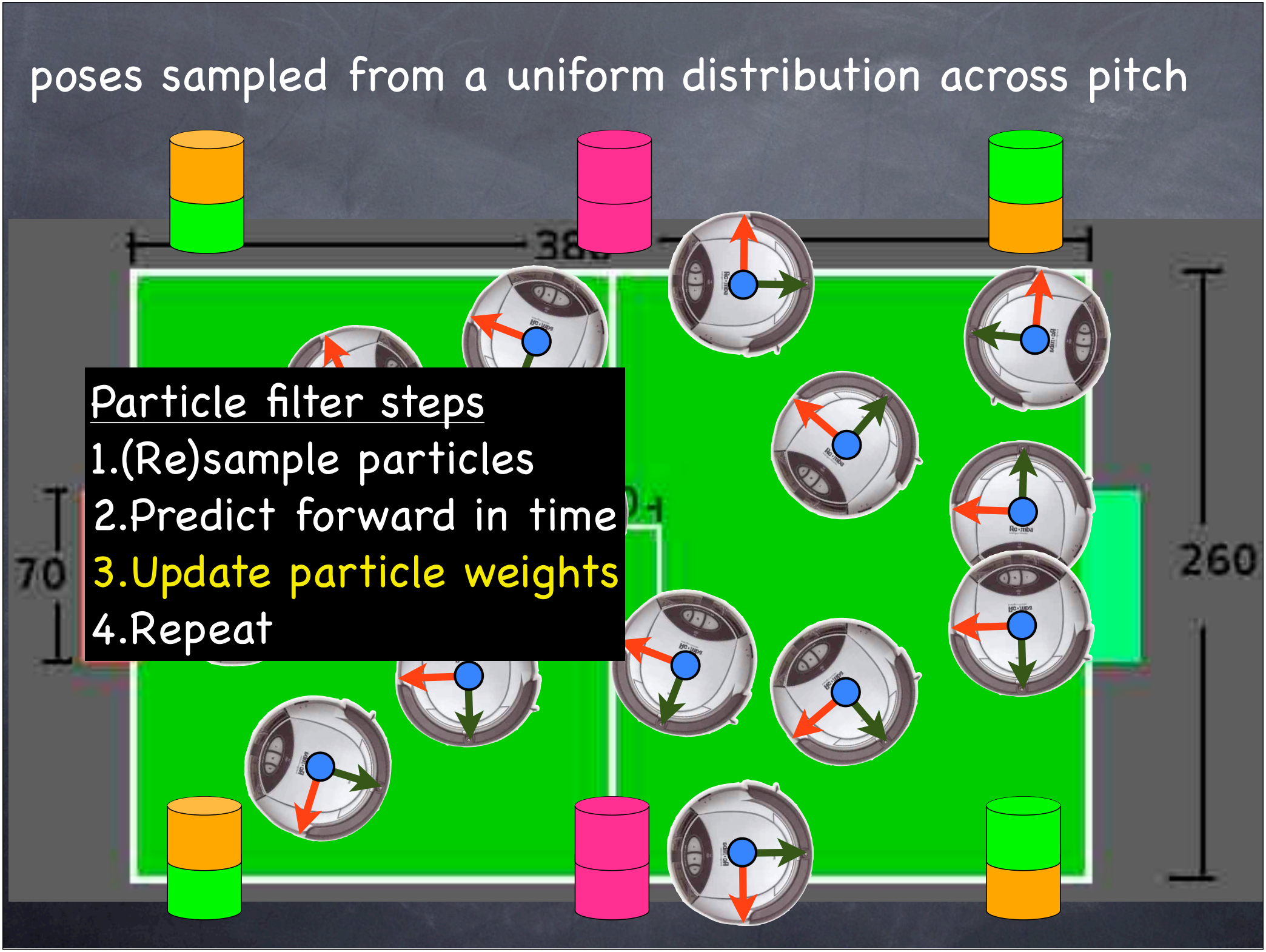
What should happen during initial sampling?


poses sampled from a uniform distribution across pitch
70
260

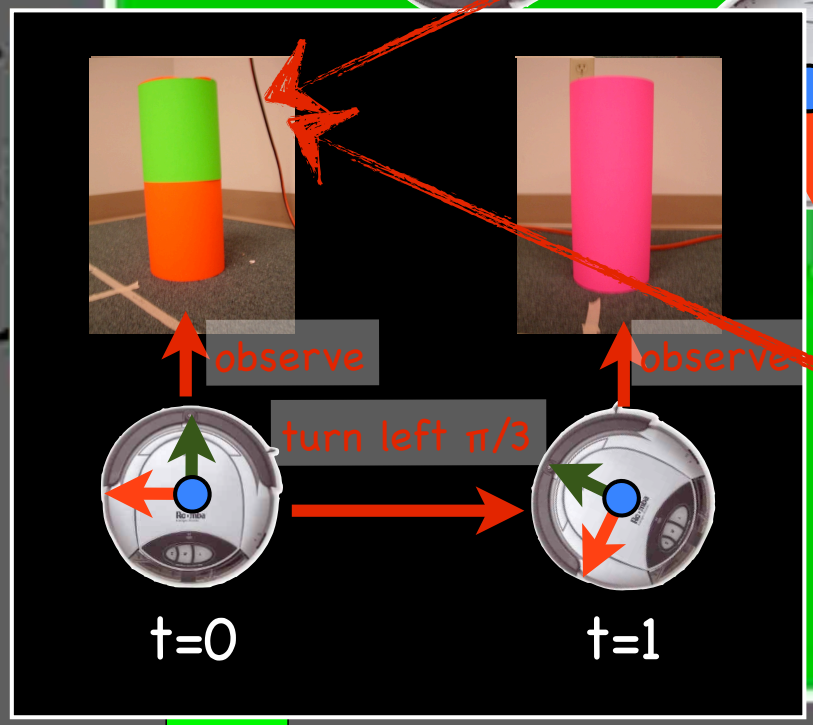
poses sampled from a uniform distribution across pitch



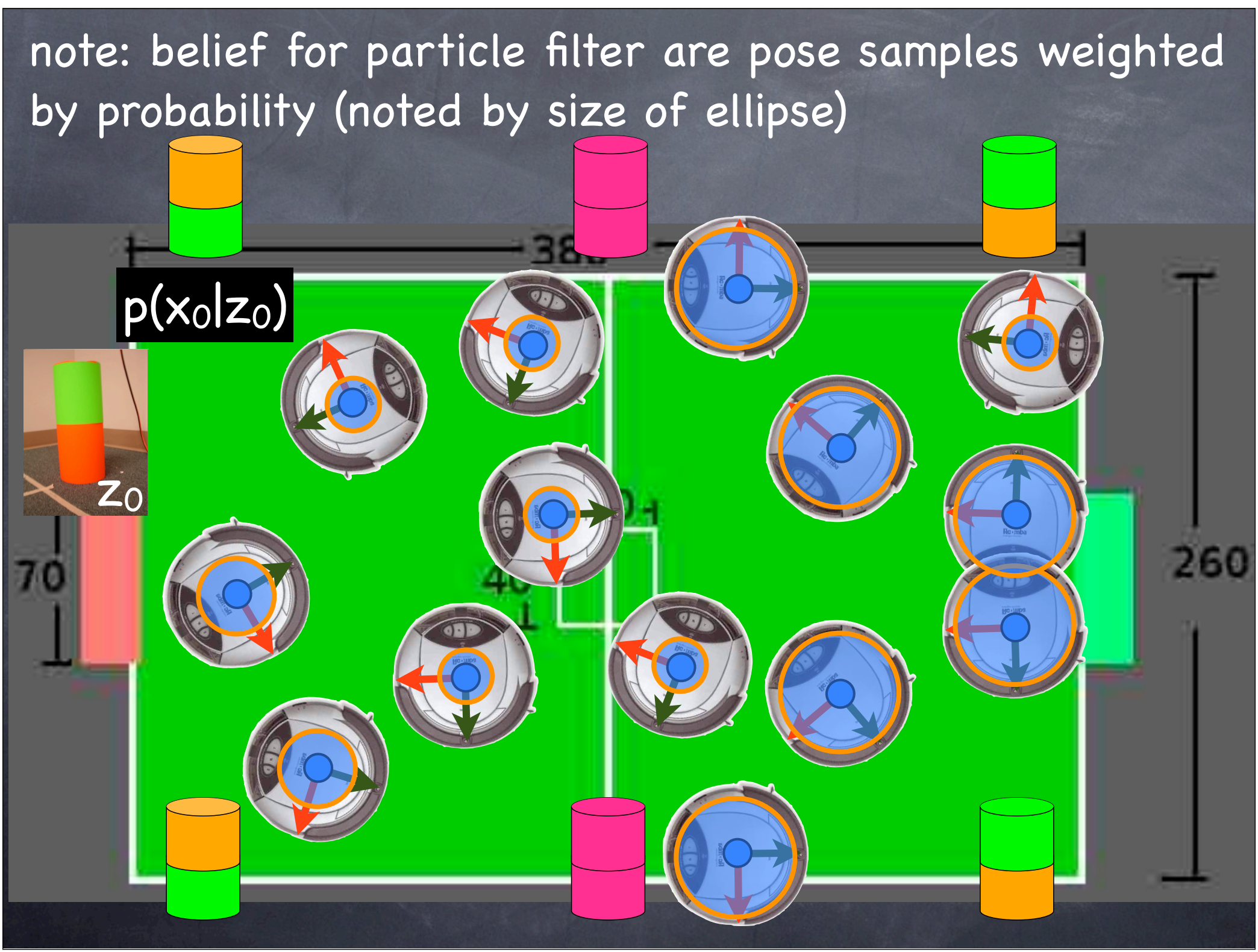
poses sampled from a uniform distribution across pitch



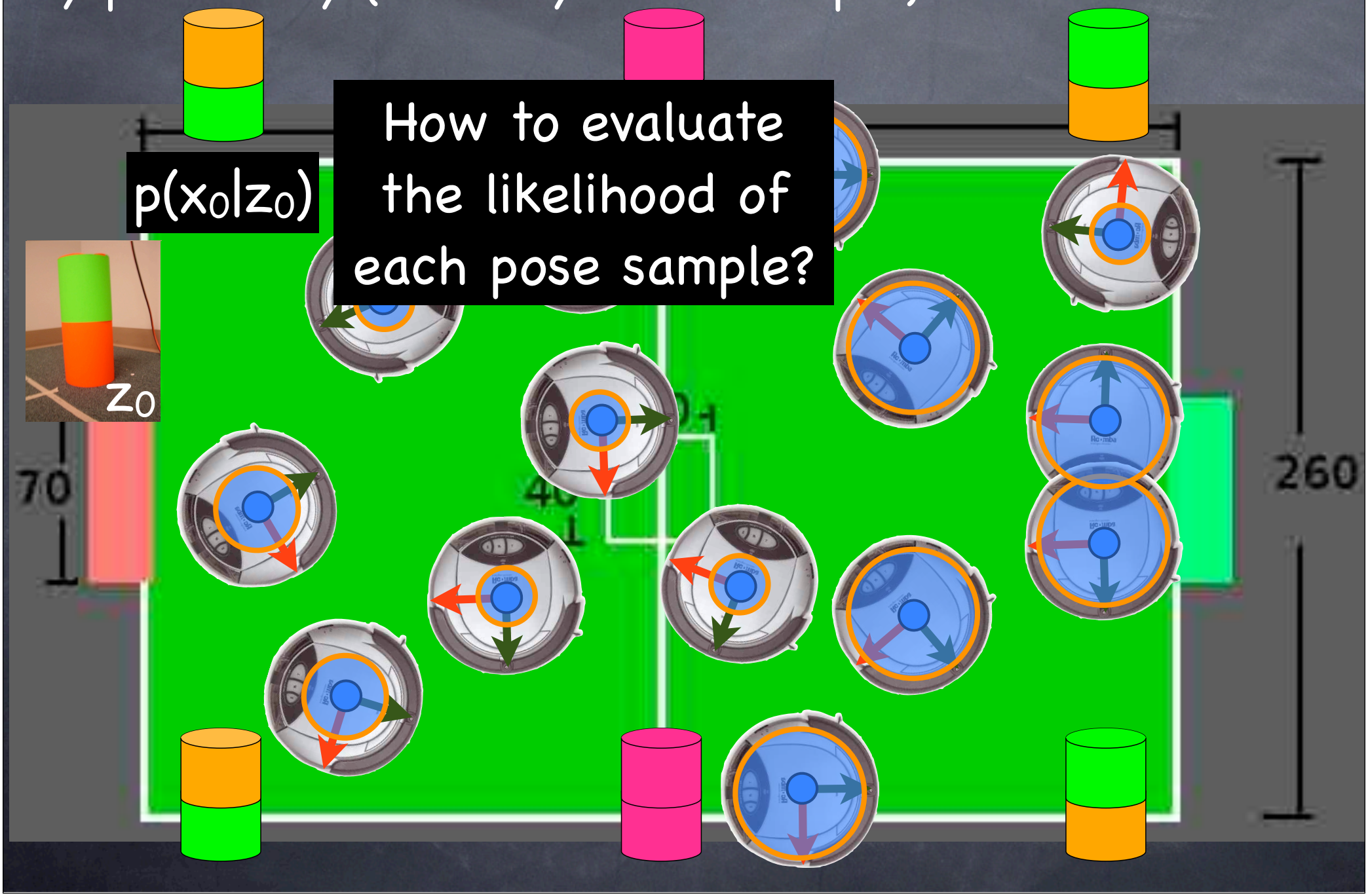
remember from previous example: poses seeing green/ orange landmark should receive higher weights



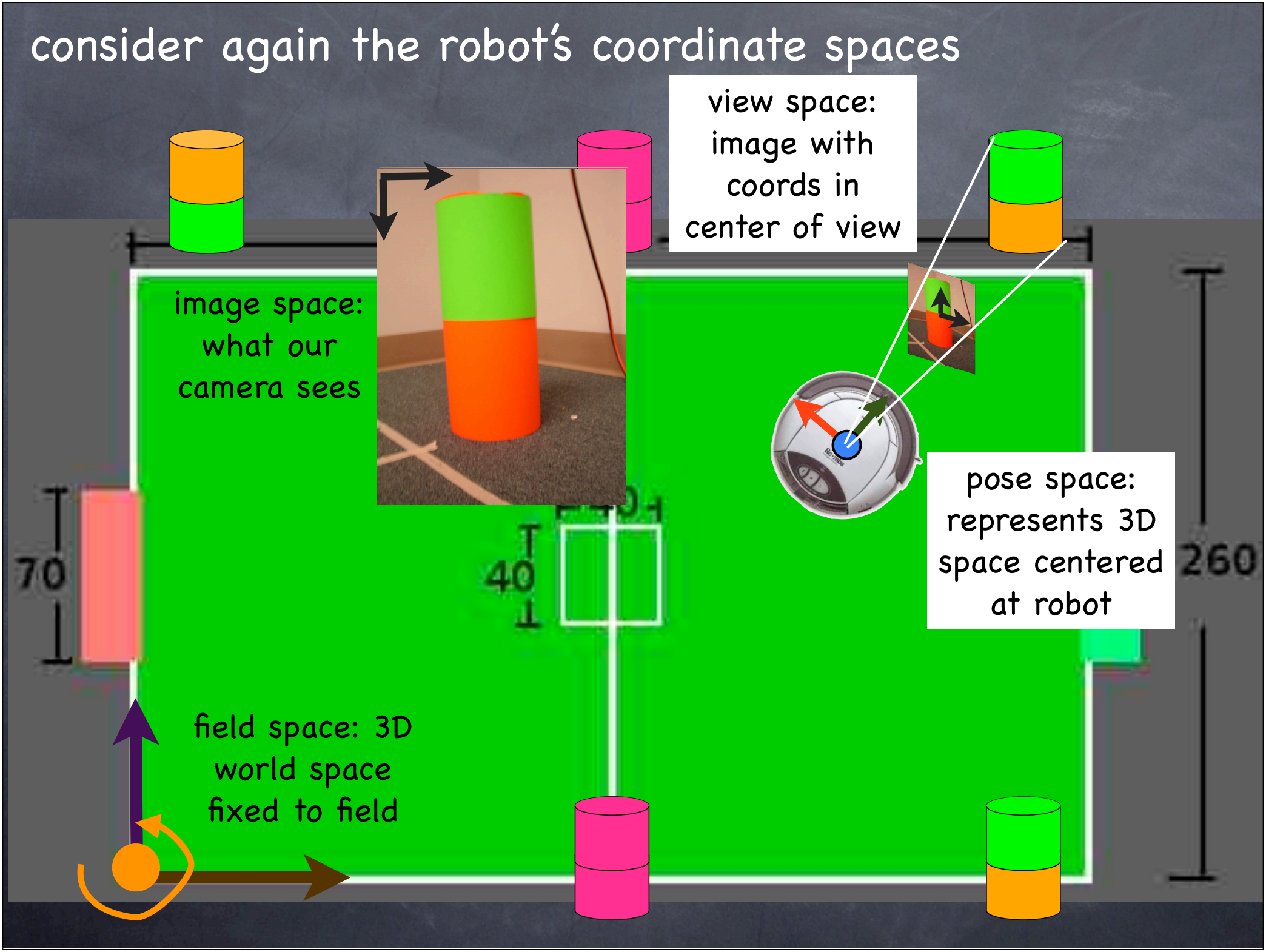
note: belief for particle filter are pose samples weighted by probability (noted by size of ellipse)



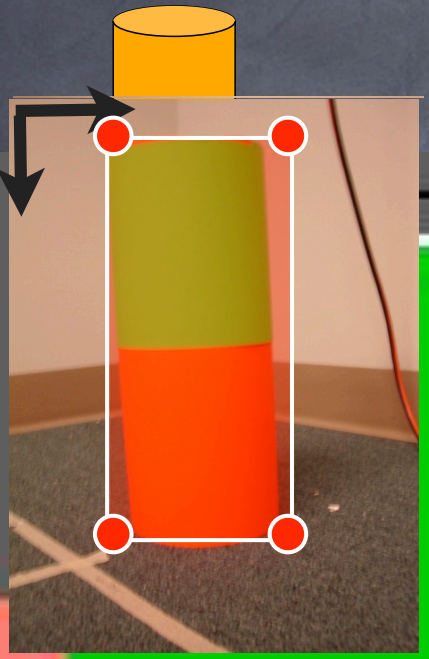
note: belief for particle filter are pose samples weighted by probability (noted by size of ellipse)

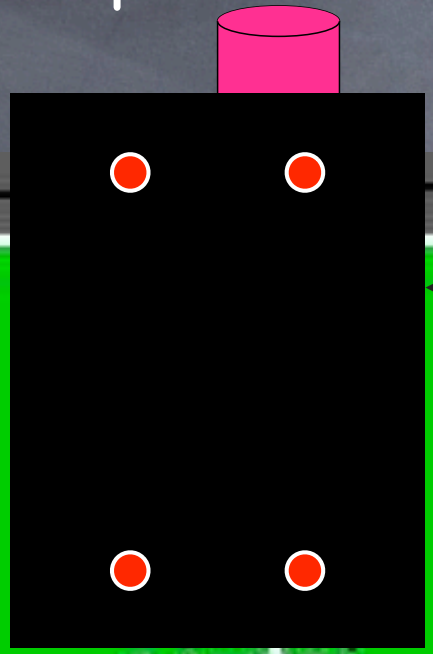


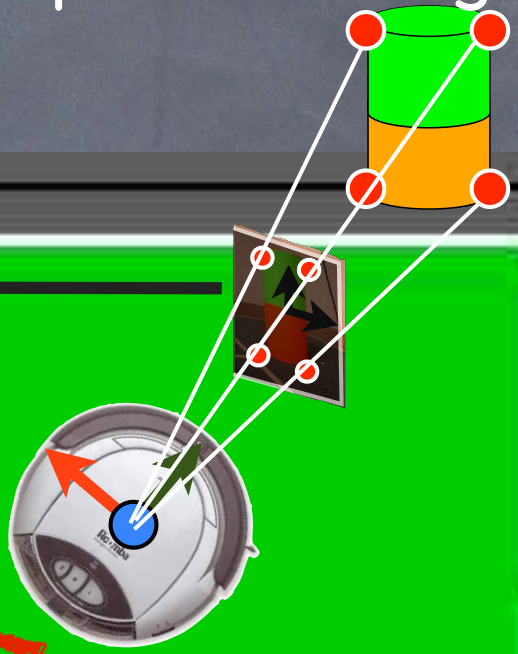
consider again the robot's coordinate spaces



To evaluate a pose, we could graphically render what should be seen at the pose and compare to image







compare rendered image with blob dimensions

project object corners into robot view

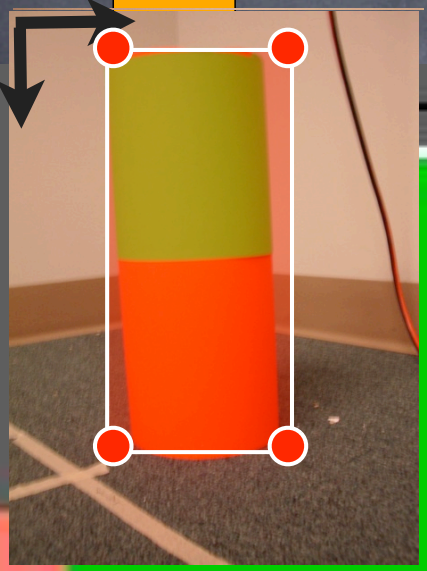
transform world into robot's perspective

70

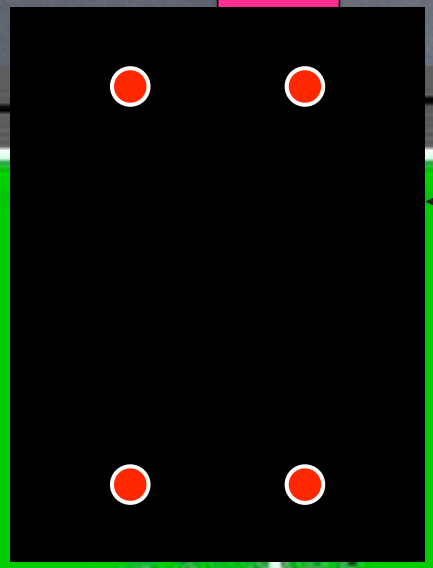
40

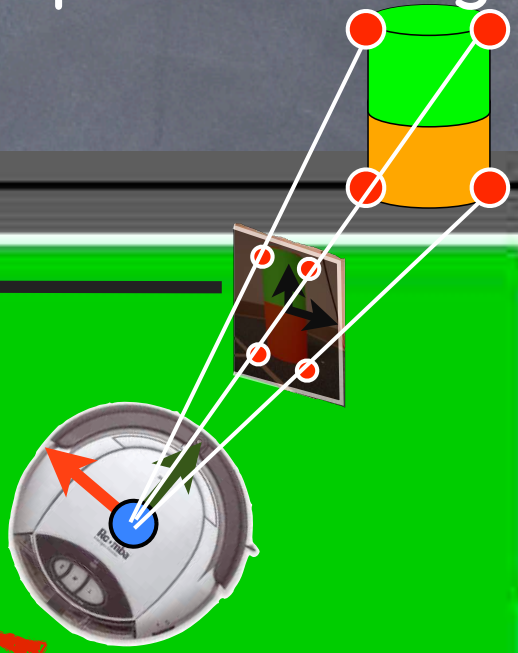
260

To evaluate a pose, we could graphically render what should be seen at the pose and compare to image



compare rendered image with blob dimensions





project object corners into robot view

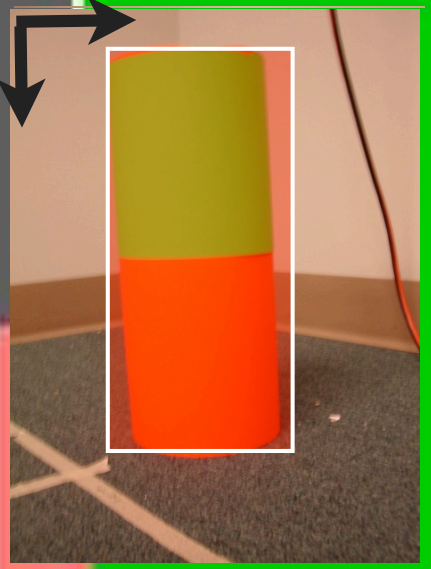
70

40

260

Good approach in general, but our robots have significant uncertainty due to low resolution, barrel distortion, etc.

Instead, we will compare the range and bearing of objects in robot pose coordinates

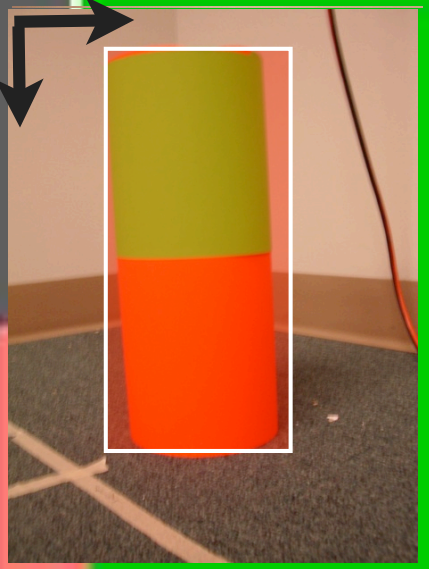


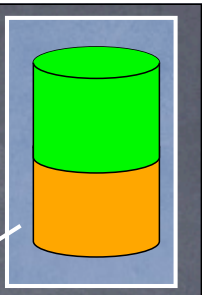
r: distance to object

Φ:angle to object

transform world into "robot's" perspective

Instead, we will compare the range and bearing of objects in robot pose coordinates



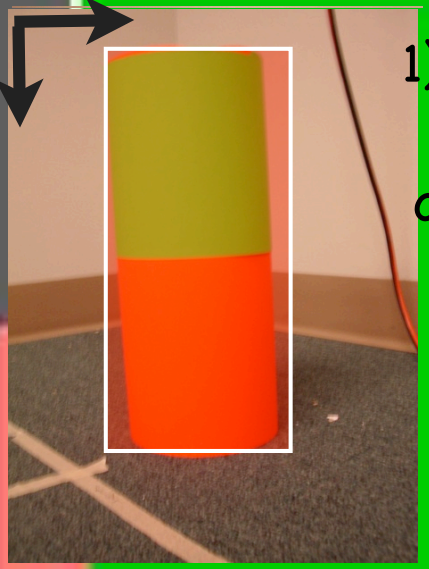


r: distance to object

Φ:angle to object

transform world into "robot's" perspective

Instead, we will compare the range and bearing of objects in robot pose coordinates



1) Estimate (r', Φ') from blob of observed object

r: distance to object

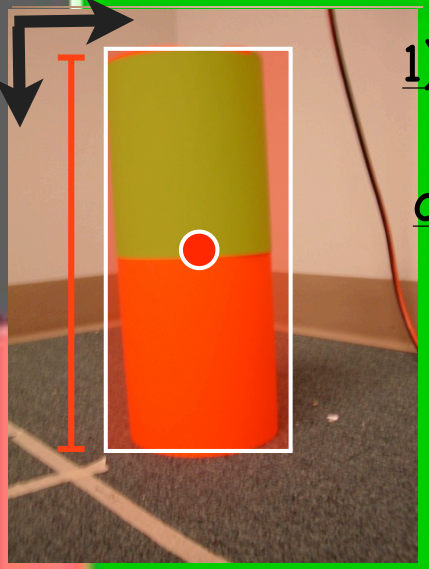
Φ:angle to object

2) calculate (r_x, Φ_x) expected by given a hypothesized pose x

3) compute weight from match between (r_x, Φ_x) and (r', Φ')

transform world into "robot's" perspective

Instead, we will compare the range and bearing of objects in robot pose coordinates



1) Estimate (r', Φ') from blob of observed object

r: distance to object

Φ:angle to object

2) calculate (r_x, Φ_x) expected by given a hypothesized pose x

3) compute weight from match between (r_x, Φ_x) and (r', Φ')

transform world into "robot's" perspective

40

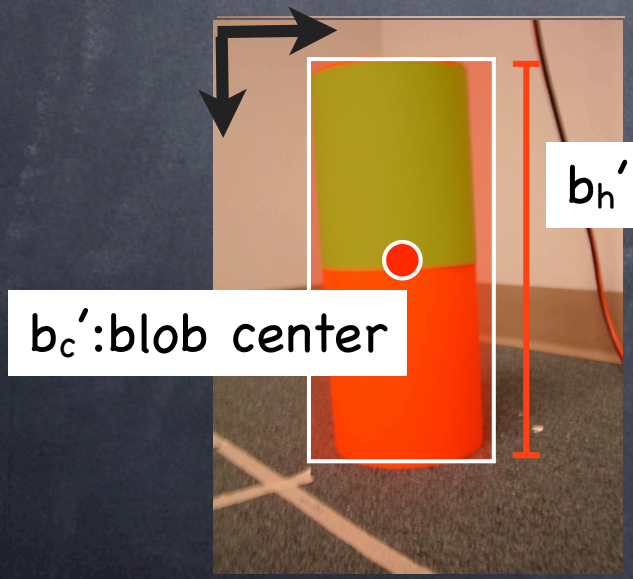
40

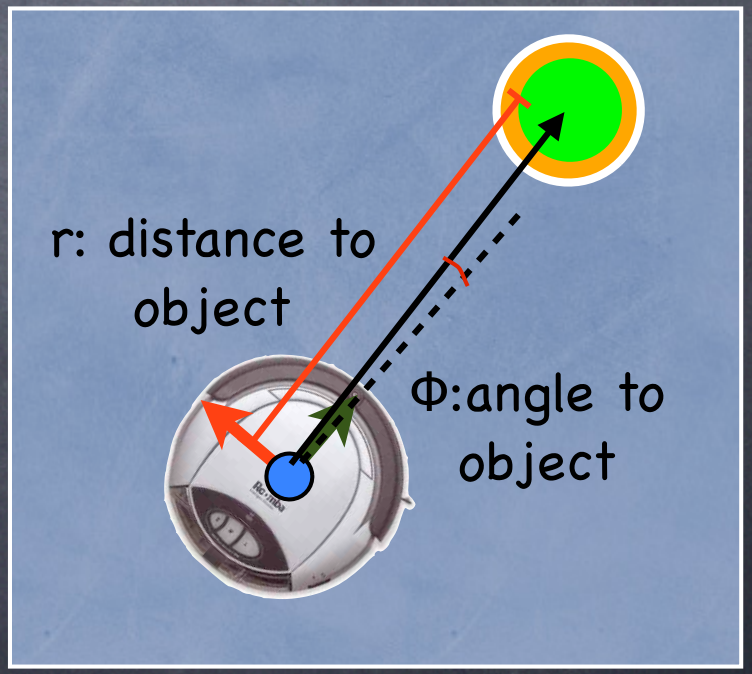
70

260

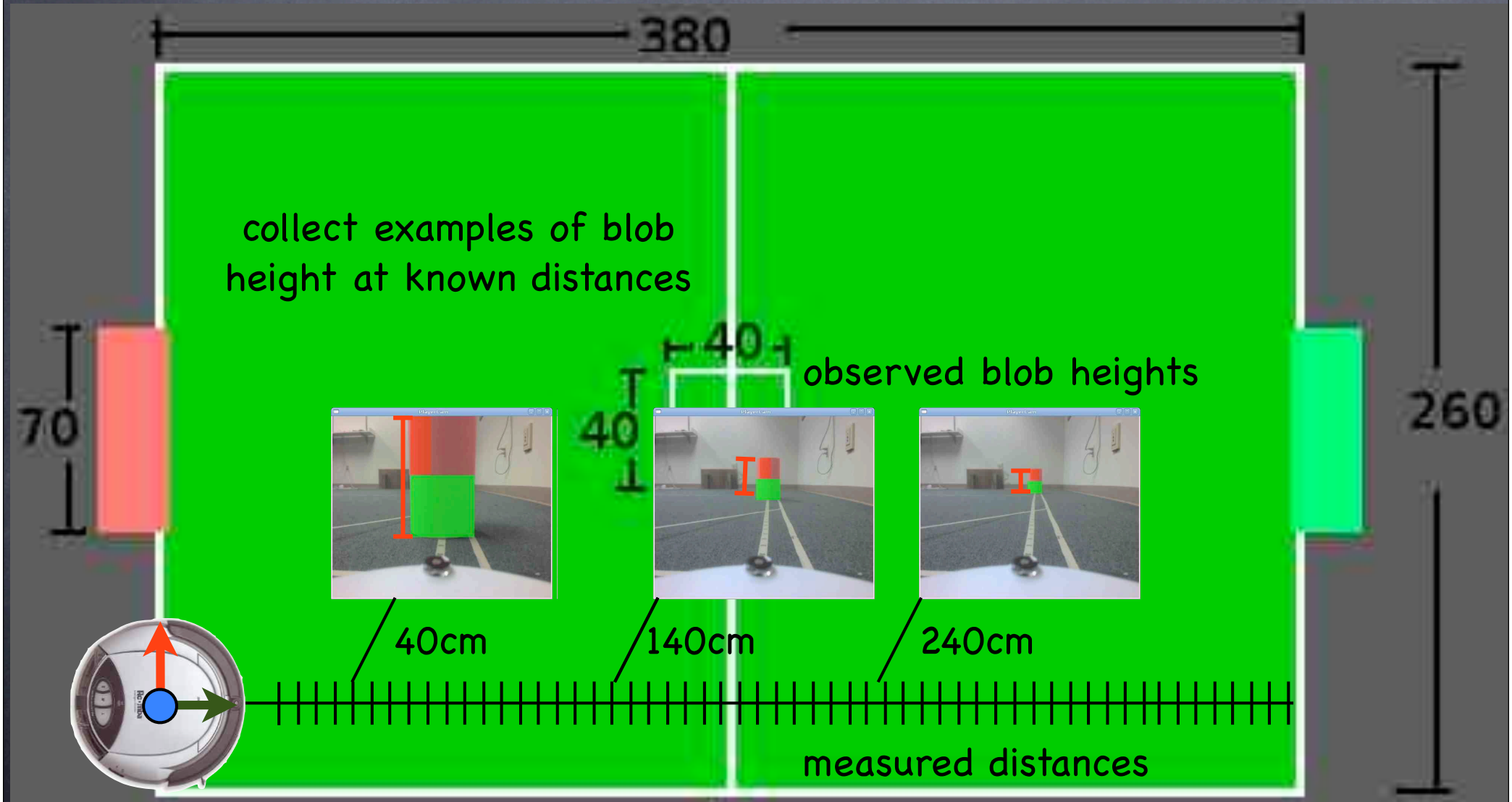
Estimating range and distance from blob

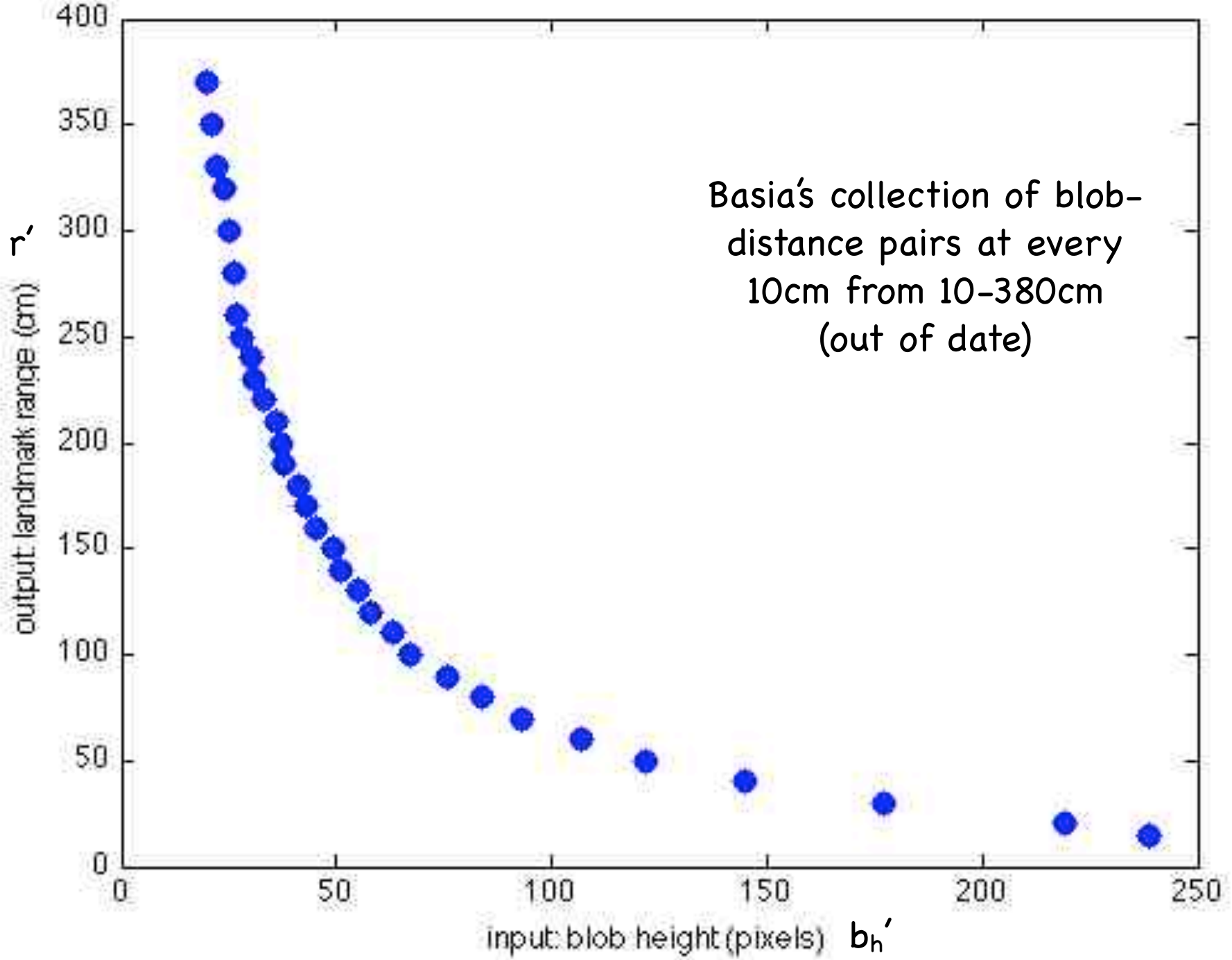
- Construct two functions:
  - Predict object distance from blob height
  - Predict object angle form blob center

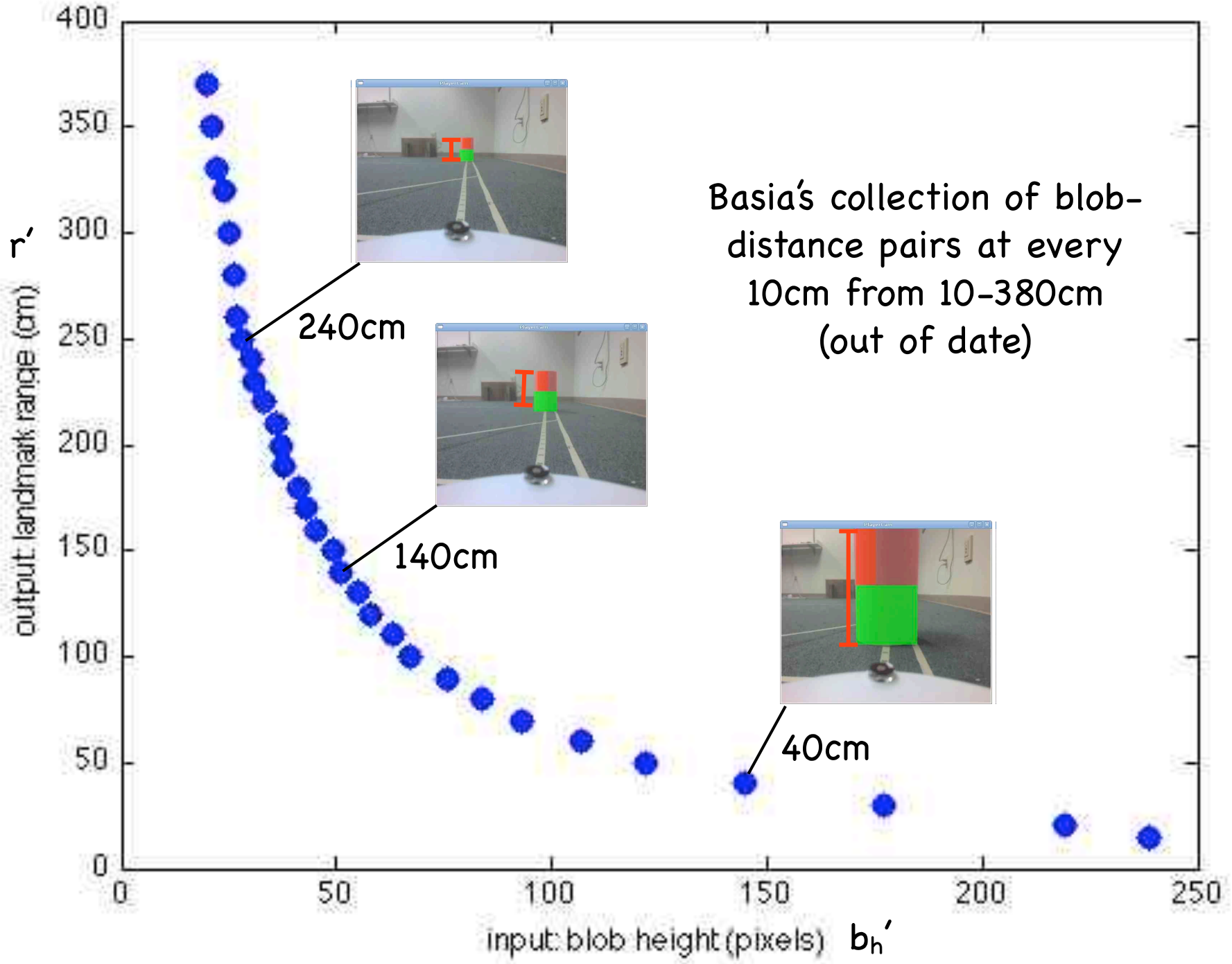


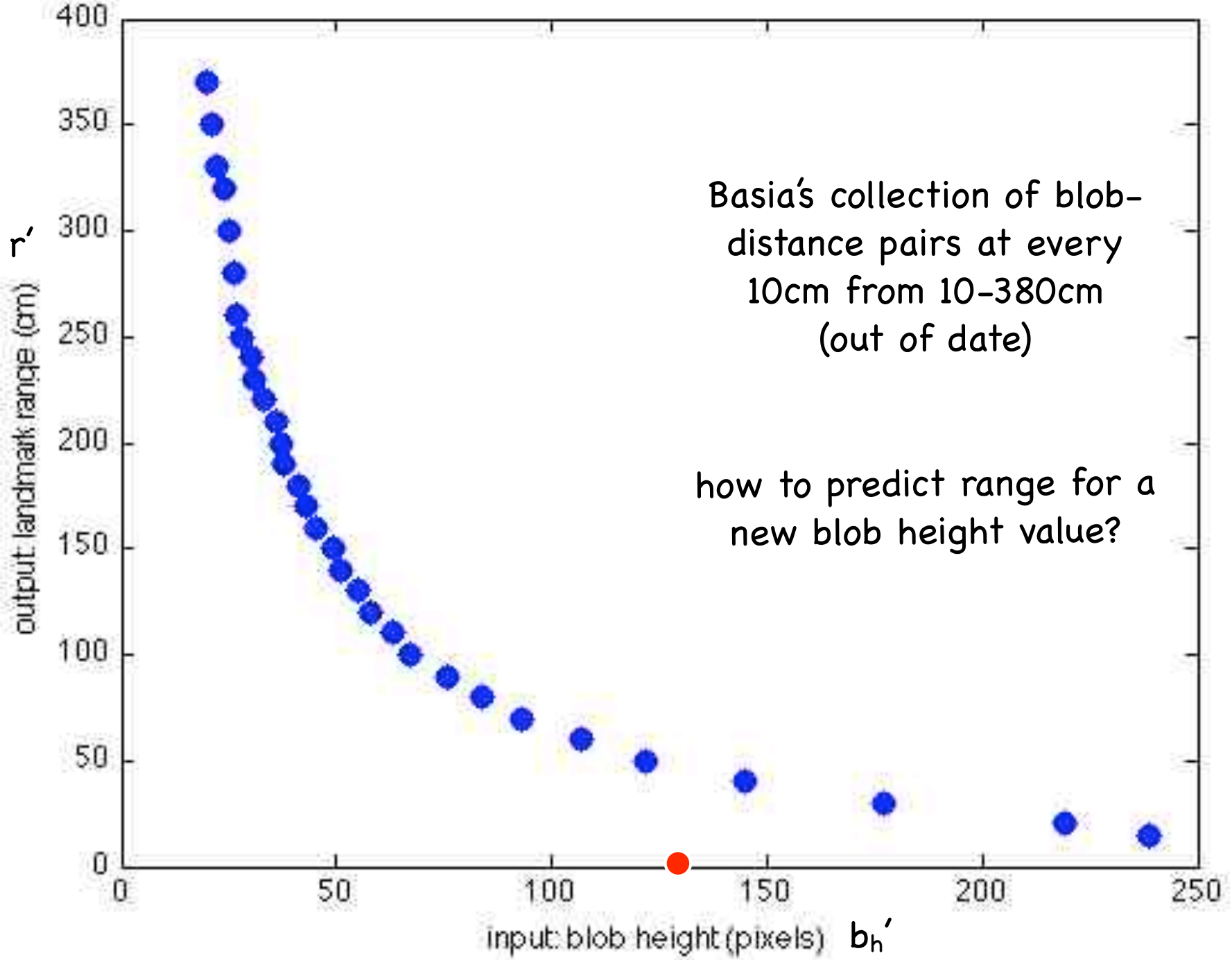


Estimating distance from examples



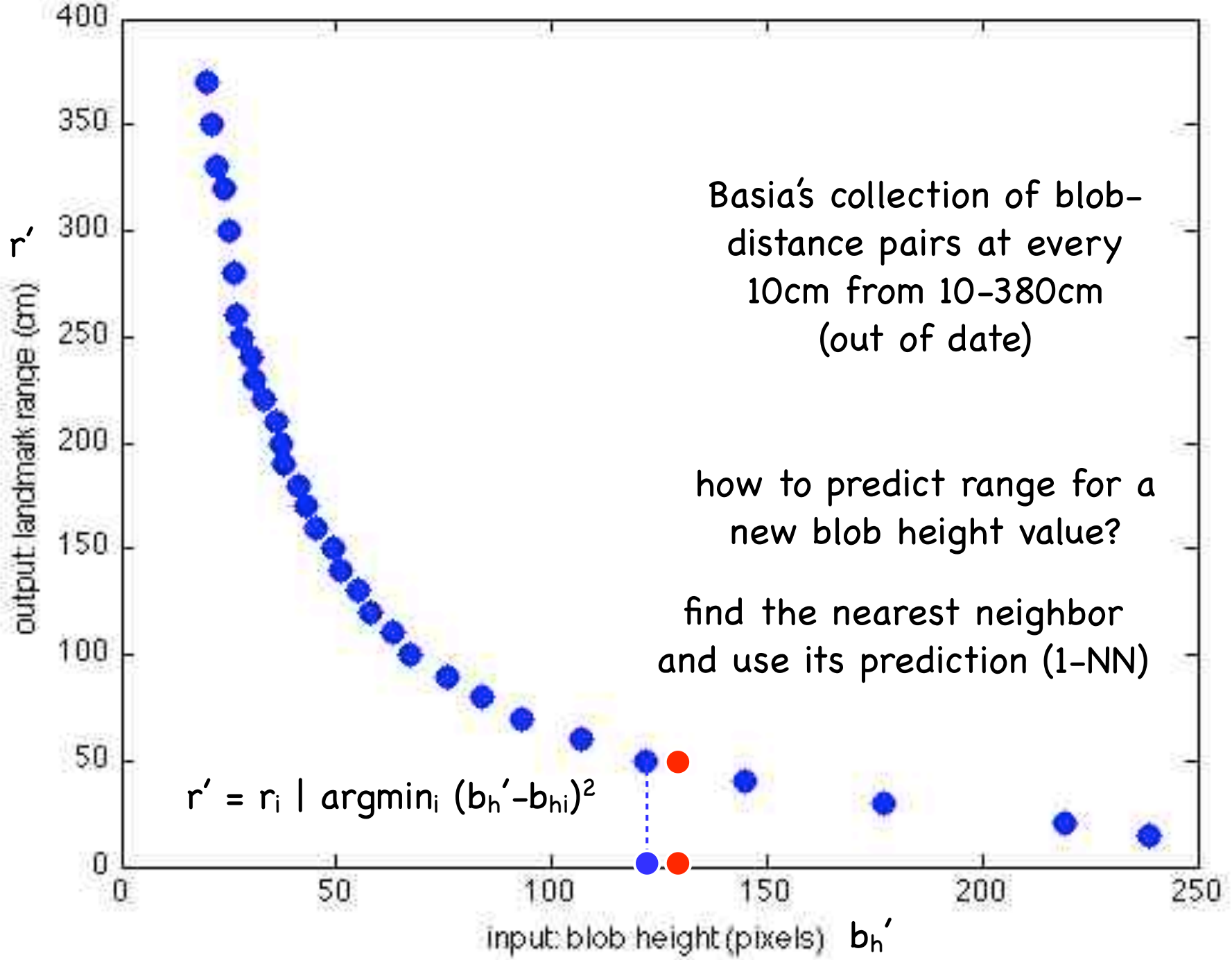

400
350
300
250
200
150
100
50
0
0
50
100
150
200
250
output: landmark range (cm)
r'
input: blob height (pixels)
b_h'
Basia's collection of blob-distance pairs at every 10cm from 10-380cm (out of date)

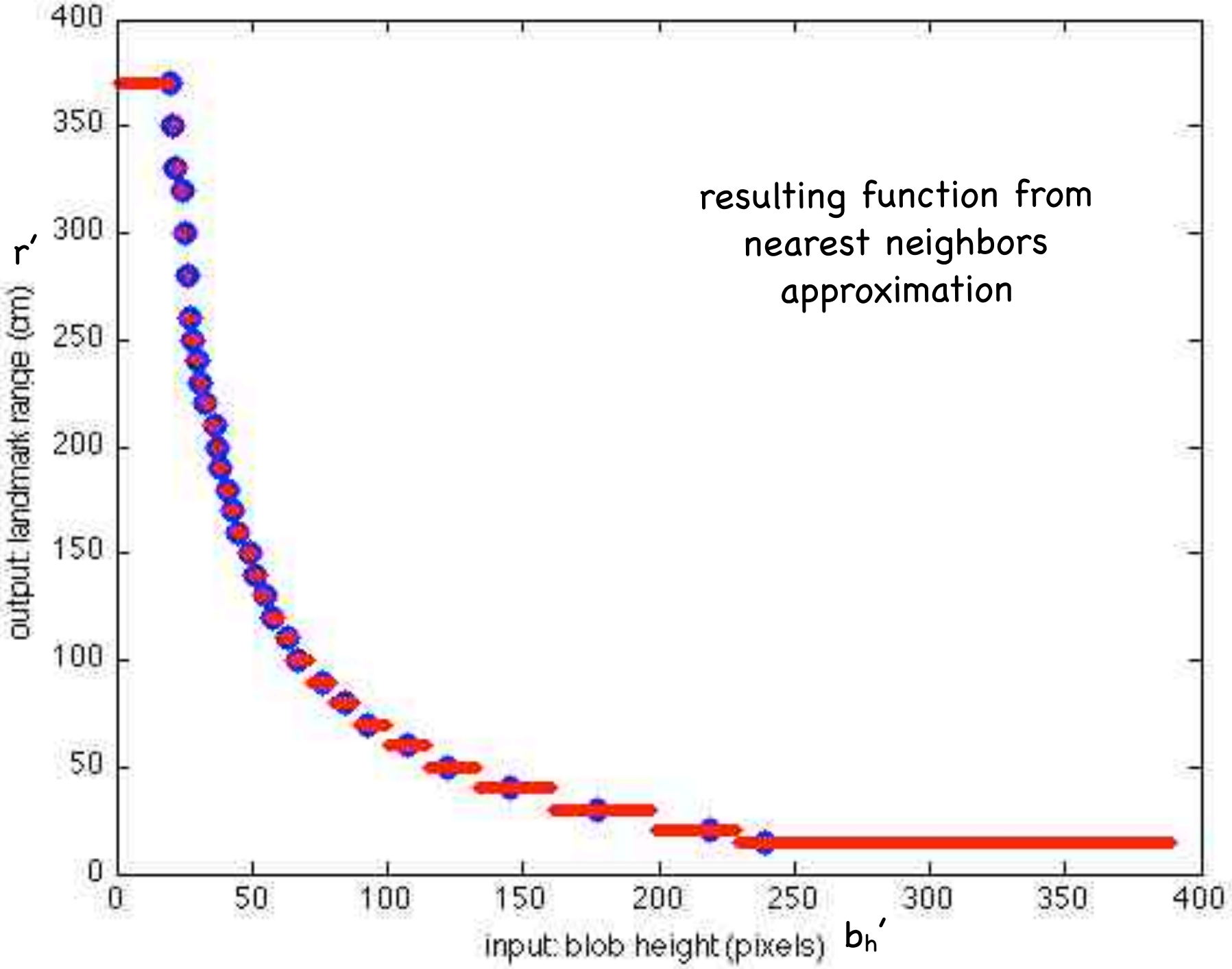

400
350
300
250
200
150
100
50
0
0
50
100
150
200
250
r′
output: landmark range (cm)
input: blob height (pixels) b_h'
240cm
140cm
40cm
Basia's collection of blob-distance pairs at every 10cm from 10-380cm (out of date)

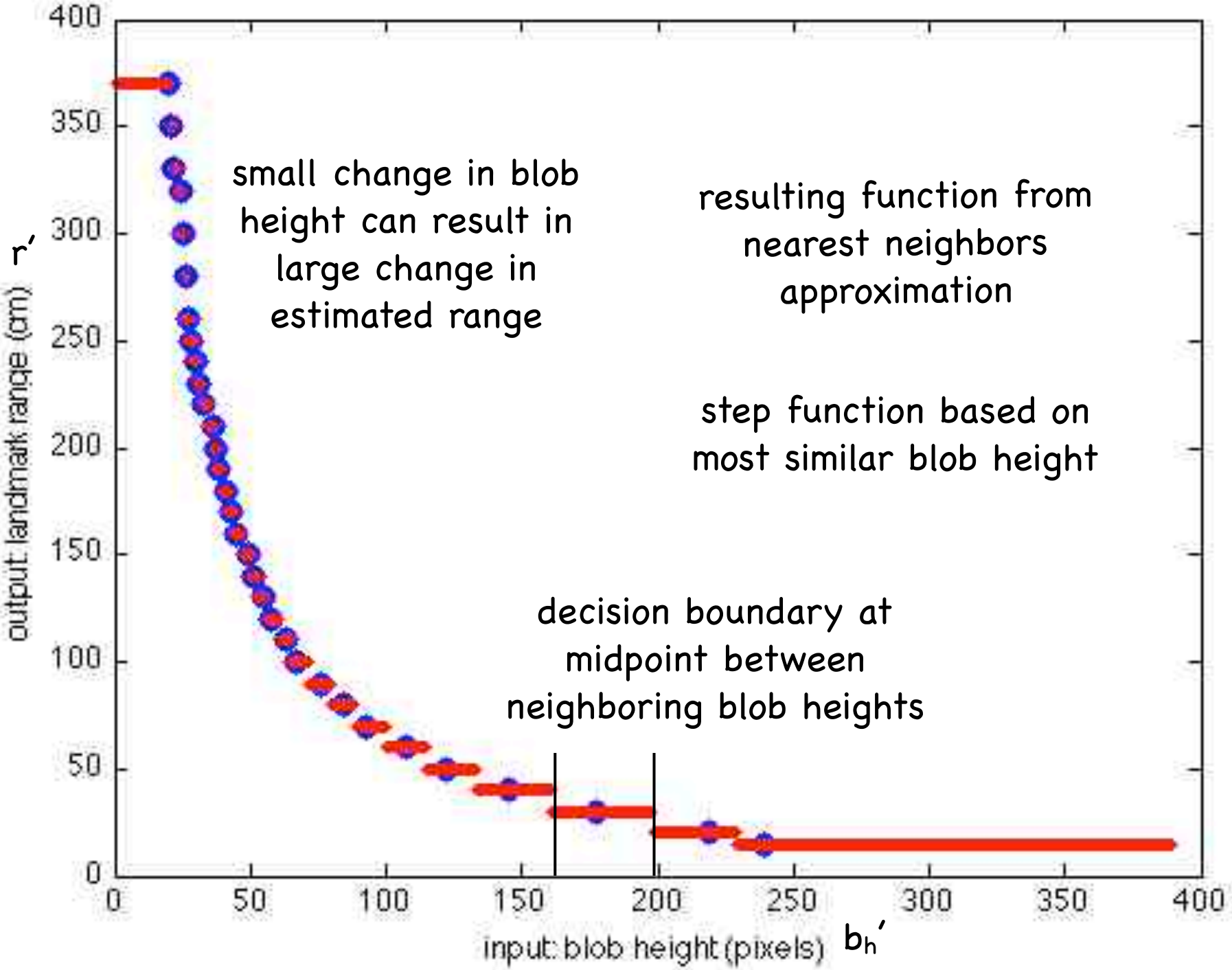


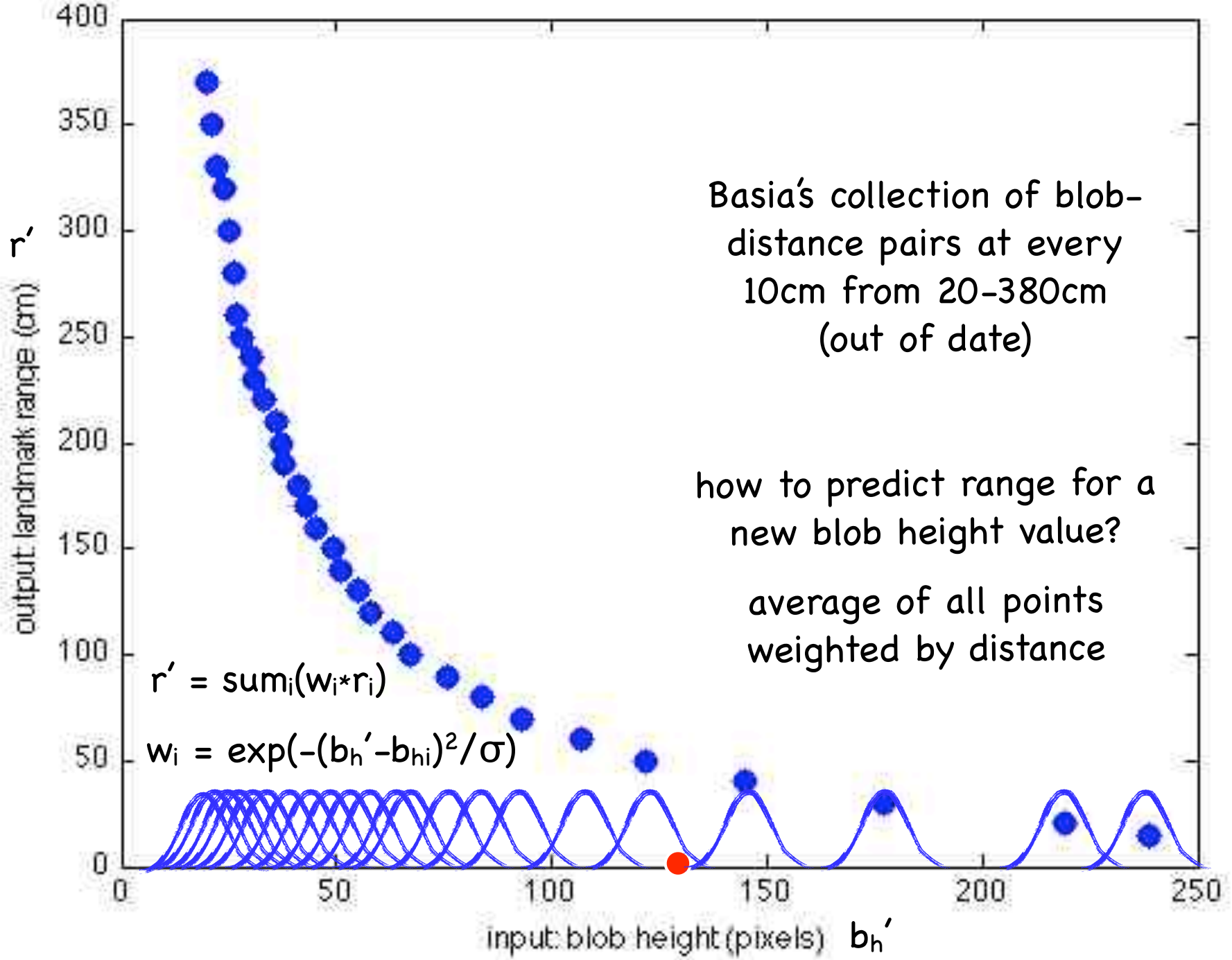
Basia's collection of blob-distance pairs at every 10cm from 10-380cm (out of date)

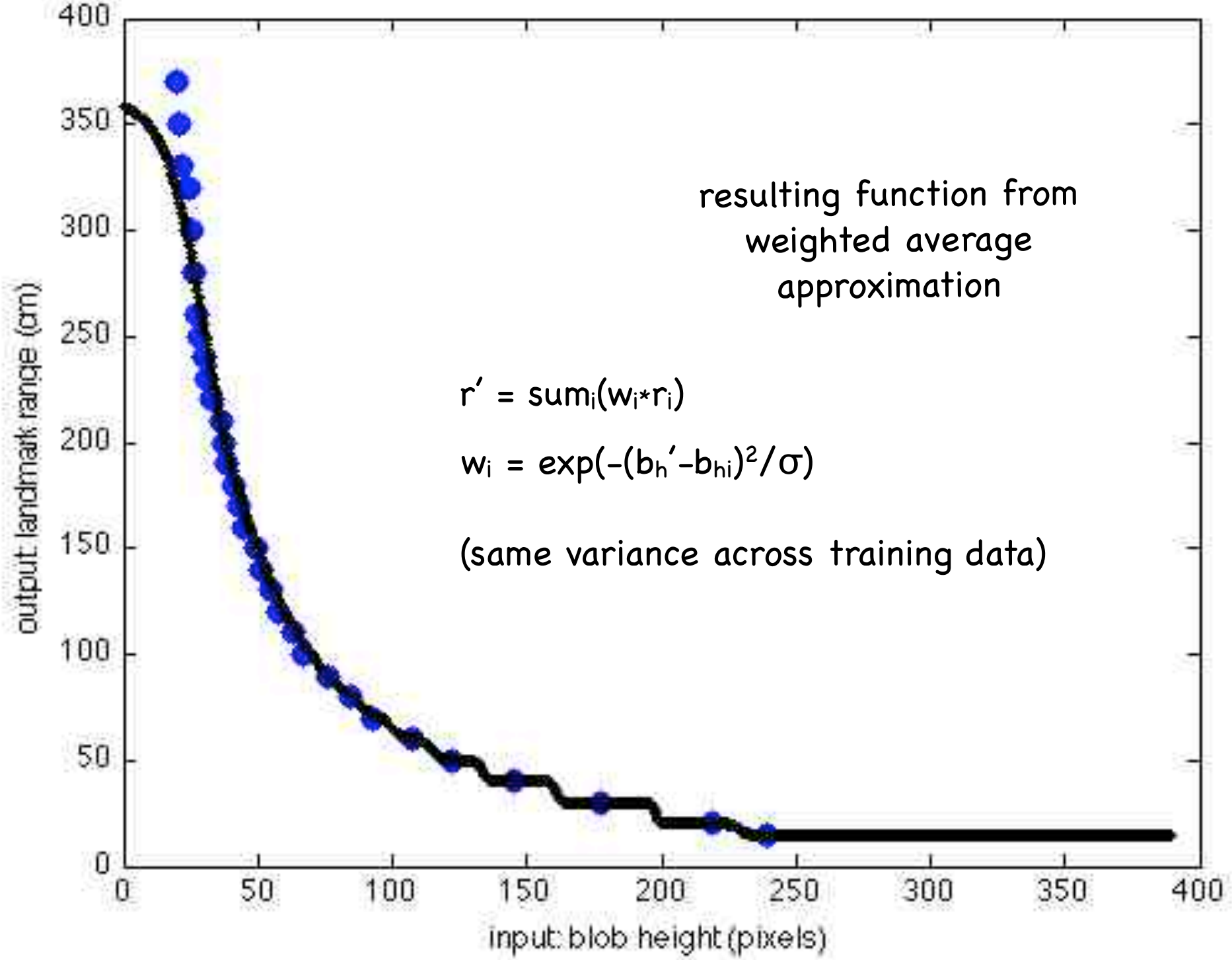
how to predict range for a new blob height value?

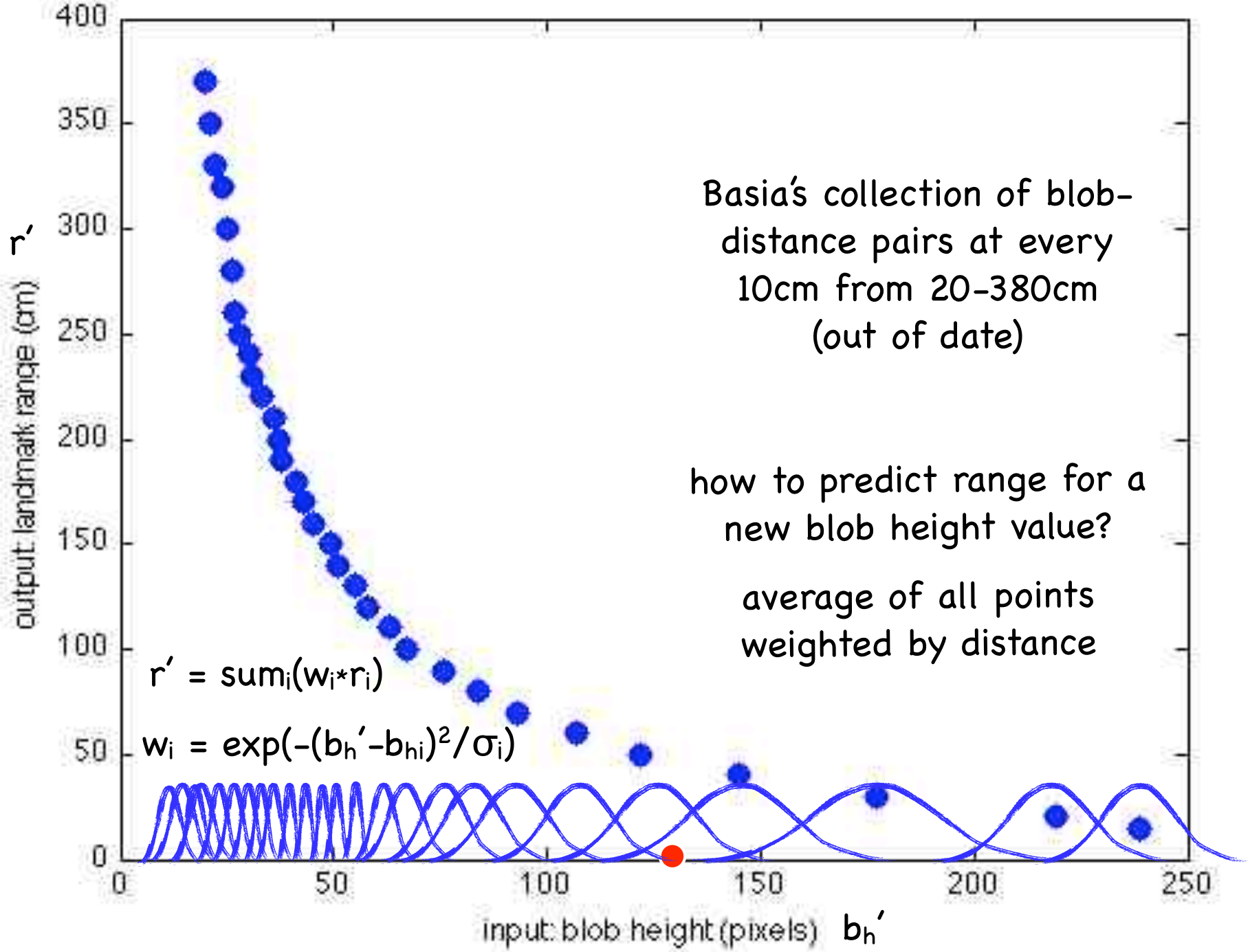

400
350
300
250
200
150
100
50
0
0
50
100
150
200
250
output: landmark range (cm)
r′
input: blob height (pixels)
b_h'
Basia's collection of blob-distance pairs at every 10cm from 10-380cm (out of date)
how to predict range for a new blob height value?
find the nearest neighbor and use its prediction (1-NN)
$r' = r_i \mid \text{argmin}_i\ (b_h'-b_{hi})^2$

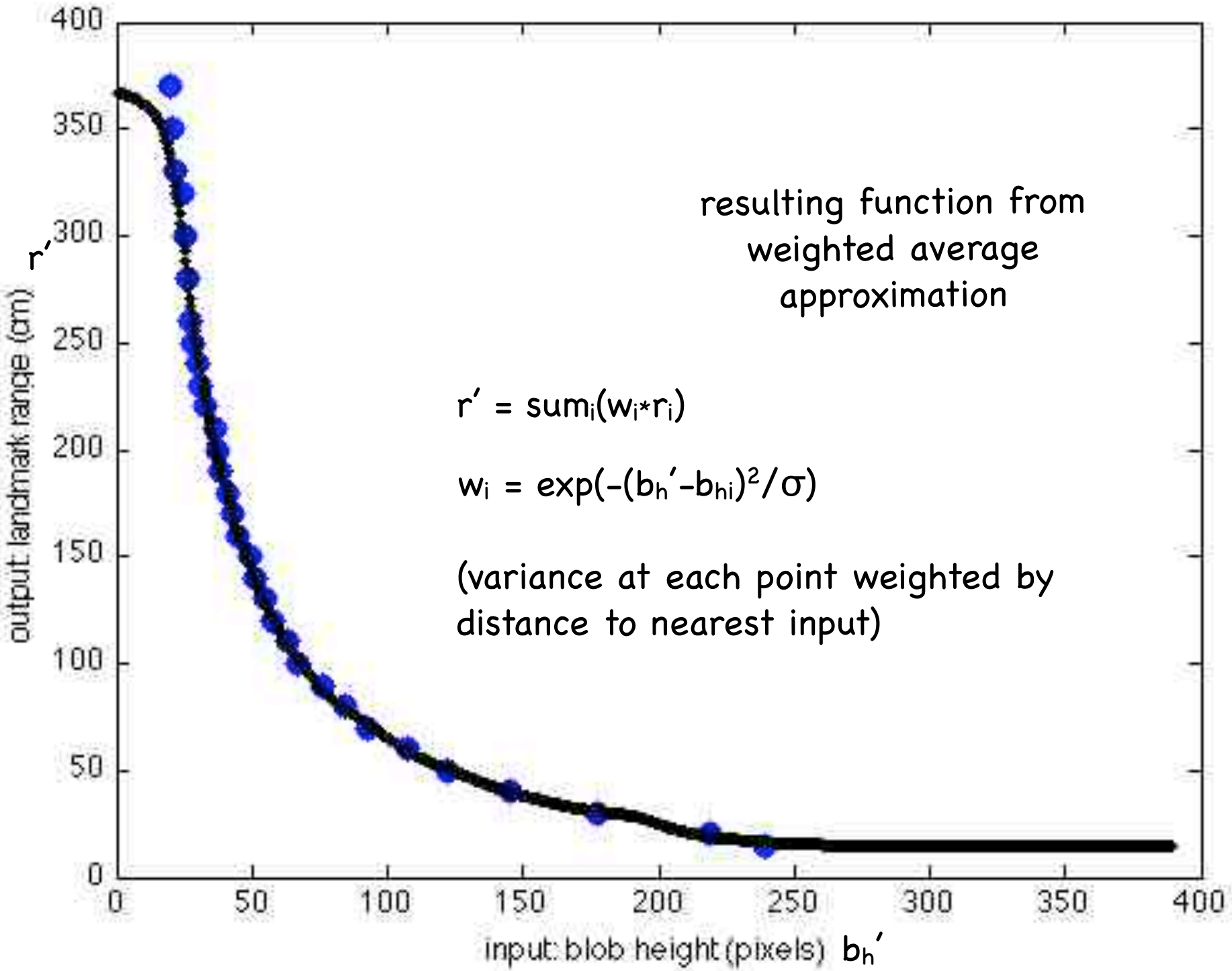

400
350
300
250
200
150
100
50
0
r′
output: landmark range (cm)
0
50
100
150
200
250
300
350
400
input: blob height (pixels)
b_h'
resulting function from
nearest neighbors
approximation


400
350
300
250
200
150
100
50
0
r′
output: landmark range (cm)
0
50
100
150
200
250
300
350
400
input: blob height (pixels)
bh′
small change in blob height can result in large change in estimated range
resulting function from nearest neighbors approximation
step function based on most similar blob height
decision boundary at midpoint between neighboring blob heights


400
350
300
250
200
150
100
50
0
0
50
100
150
200
250
output: landmark range (cm)
r′
input: blob height (pixels)
b_h'
Basia's collection of blob-distance pairs at every 10cm from 20-380cm (out of date)
how to predict range for a new blob height value?
average of all points weighted by distance
$r' = sum_i(w_i * r_i)$
$w_i = exp(-(b_h' - b_{hi})^2/\sigma)$


400
350
300
250
200
150
100
50
0
0
50
100
150
200
250
300
350
400
output: landmark range (cm)
input: blob height (pixels)
resulting function from
weighted average
approximation
r′ = sum$_i$(w$_i$*r$_i$)
w$_i$ = exp(-(b$_h$′-b$_{hi}$)^2/σ)
(same variance across training data)


400
350
300
250
200
150
100
50
0
0
50
100
150
200
250
output: landmark range (cm)
r′
input: blob height (pixels)
b_h'
Basia's collection of blob-distance pairs at every 10cm from 20-380cm (out of date)
how to predict range for a new blob height value?
average of all points weighted by distance
$r' = \text{sum}_i(w_i * r_i)$
$w_i = \exp(-(b_h' - b_{hi})^2/\sigma_i)$

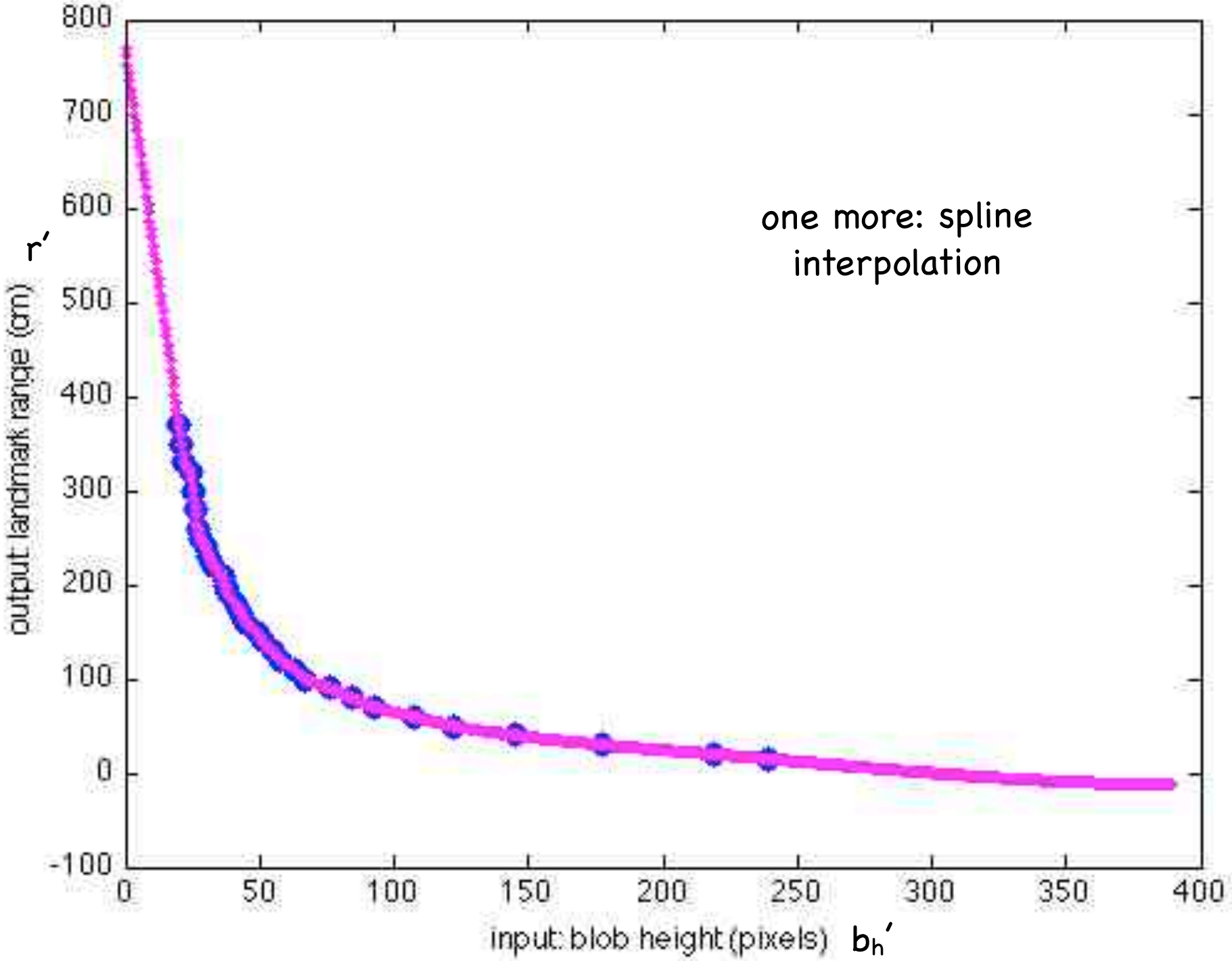


resulting function from weighted average approximation

$r' = \text{sum}_i(w_i * r_i)$

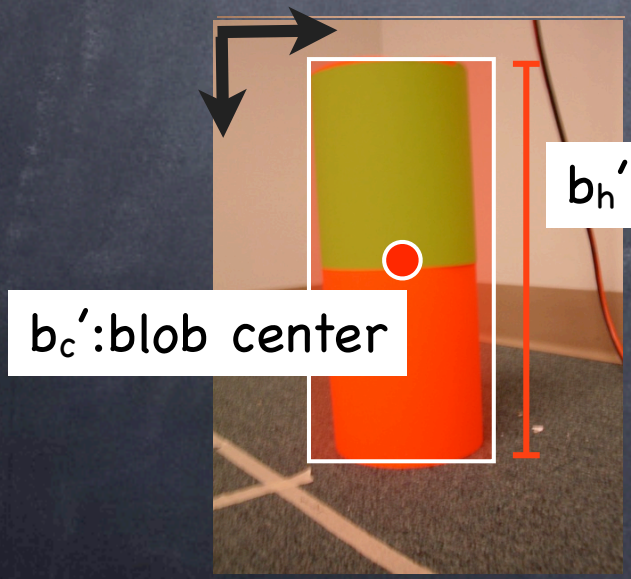
$w_i = \exp(-(b_h' - b_{hi})^2/\sigma)$

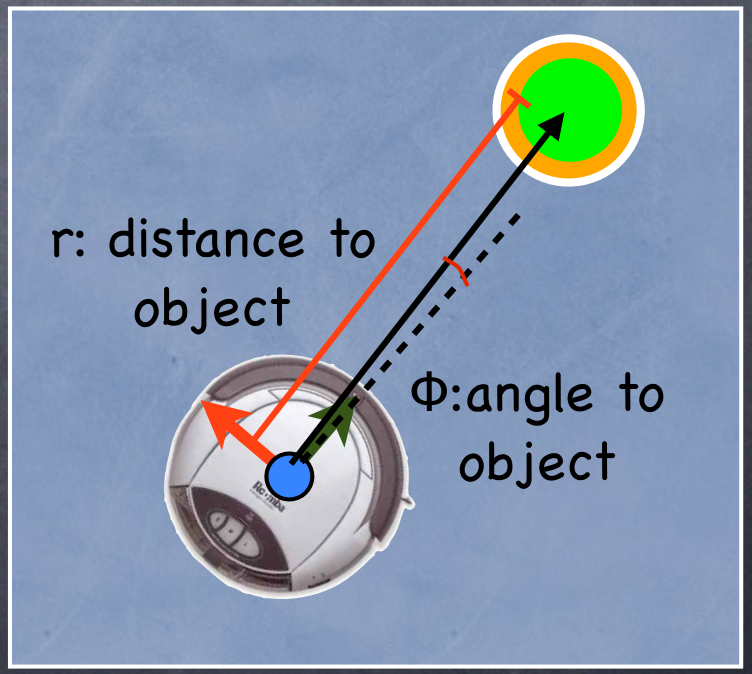
(variance at each point weighted by distance to nearest input)


800
700
600
500
400
300
200
100
0
-100
r′
output: landmark range (cm)
0
50
100
150
200
250
300
350
400
input: blob height (pixels)
b_h'
one more: spline
interpolation

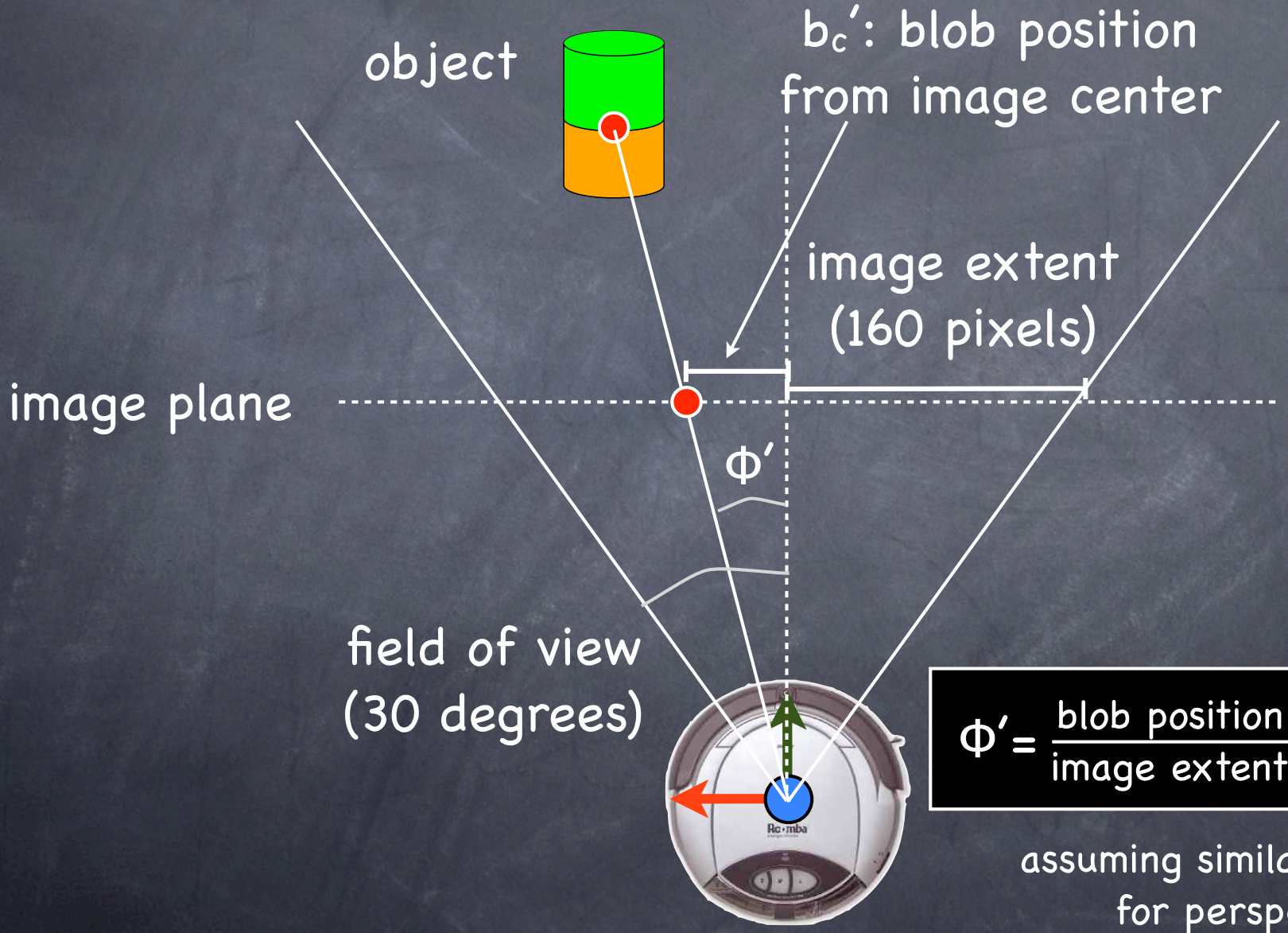
Estimating range and distance from blob

- Construct two functions:
  - Predict object distance from blob height
  - Predict object angle form blob center





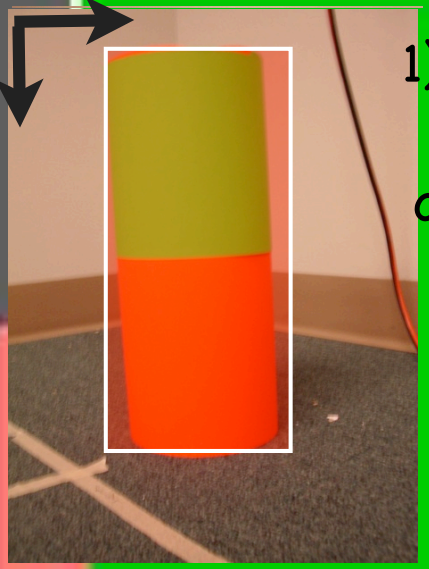
Approximating bearing



$$\Phi' = \frac{\text{blob position}}{\text{image extent}} * \text{field of view}$$

assuming similar triangles
for perspective

Instead, we will compare the range and bearing of objects in robot pose coordinates



1) Estimate (r′,Φ′) from blob of observed object

r: distance to object

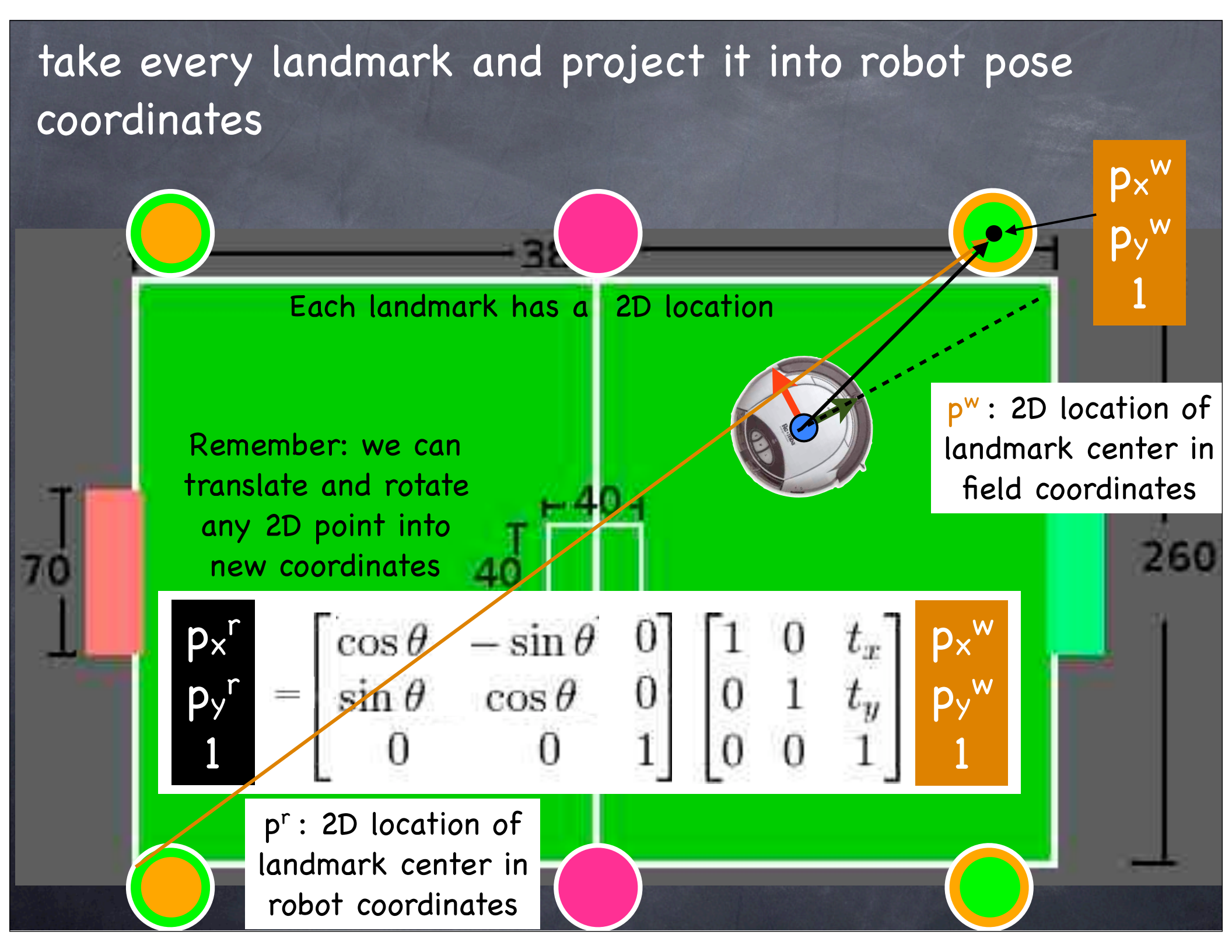
Φ:angle to object

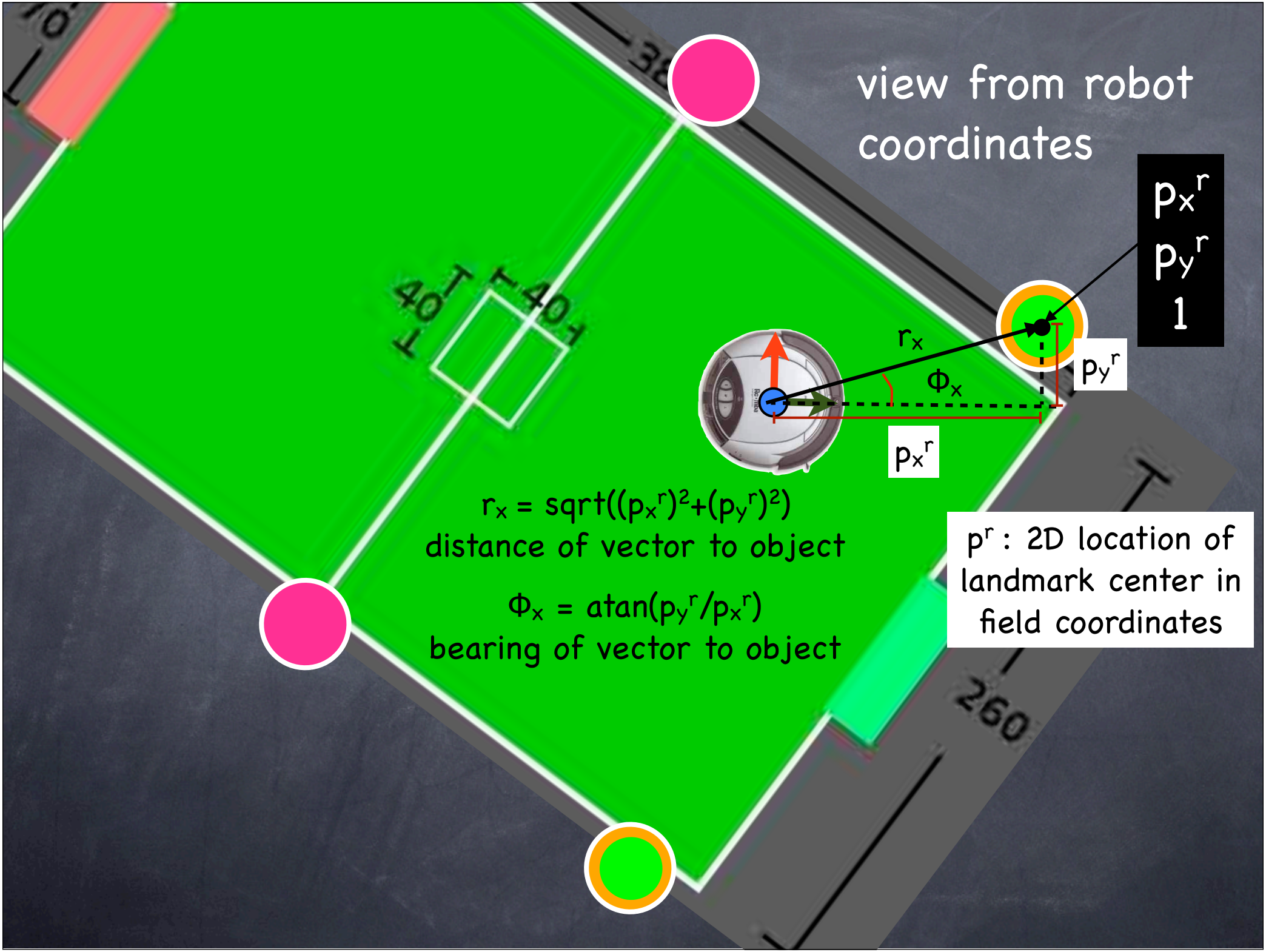
2) calculate (r_x,Φ_x) expected by given a hypothesized pose x

3) compute weight from match between (r_x,Φ_x) and (r′,Φ′)

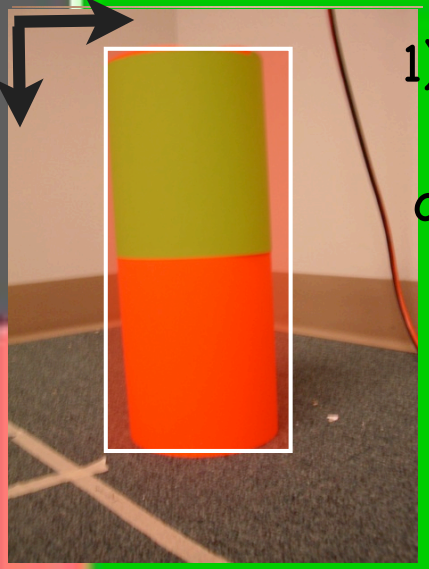
transform world into "robot's" perspective

take every landmark and project it into robot pose coordinates




view from robot coordinates
p_x^r
p_y^r
1
r_x
Φ_x
p_y^r
p_x^r
$r_x = \text{sqrt}((p_x^r)^2+(p_y^r)^2)$
distance of vector to object
$\Phi_x = \text{atan}(p_y^r/p_x^r)$
bearing of vector to object
p^r : 2D location of landmark center in field coordinates
260

Instead, we will compare the range and bearing of objects in robot pose coordinates



1) Estimate (r', Φ') from blob of observed object

r: distance to object

Φ:angle to object

2) calculate (r_x, Φ_x) expected by given a hypothesized pose x

transform world into "robot's" perspective

3) compute weight from match between (r_x, Φ_x) and (r', Φ')

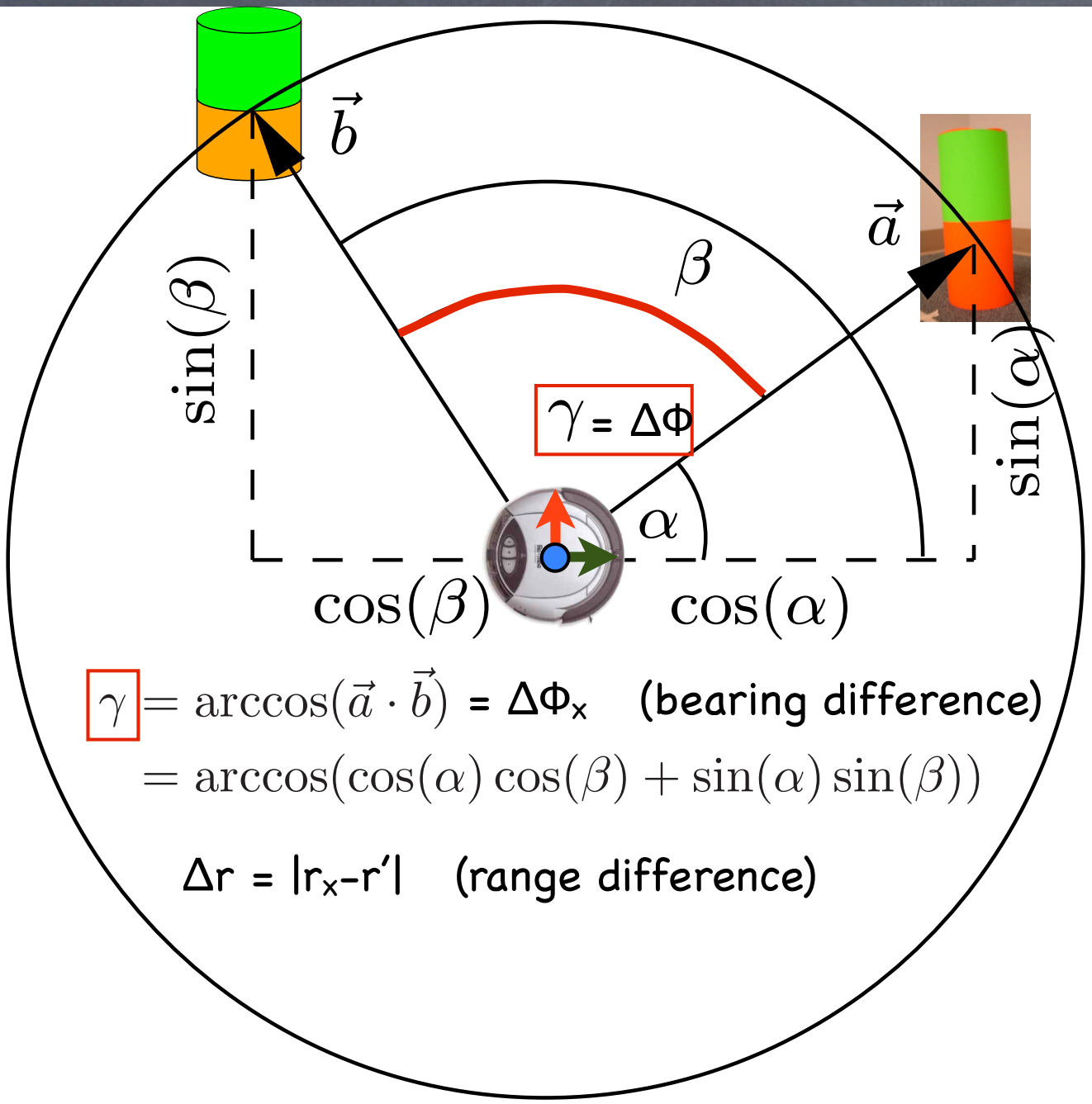
380

260

70

40

40



$l(z=$  $|x) \approx$

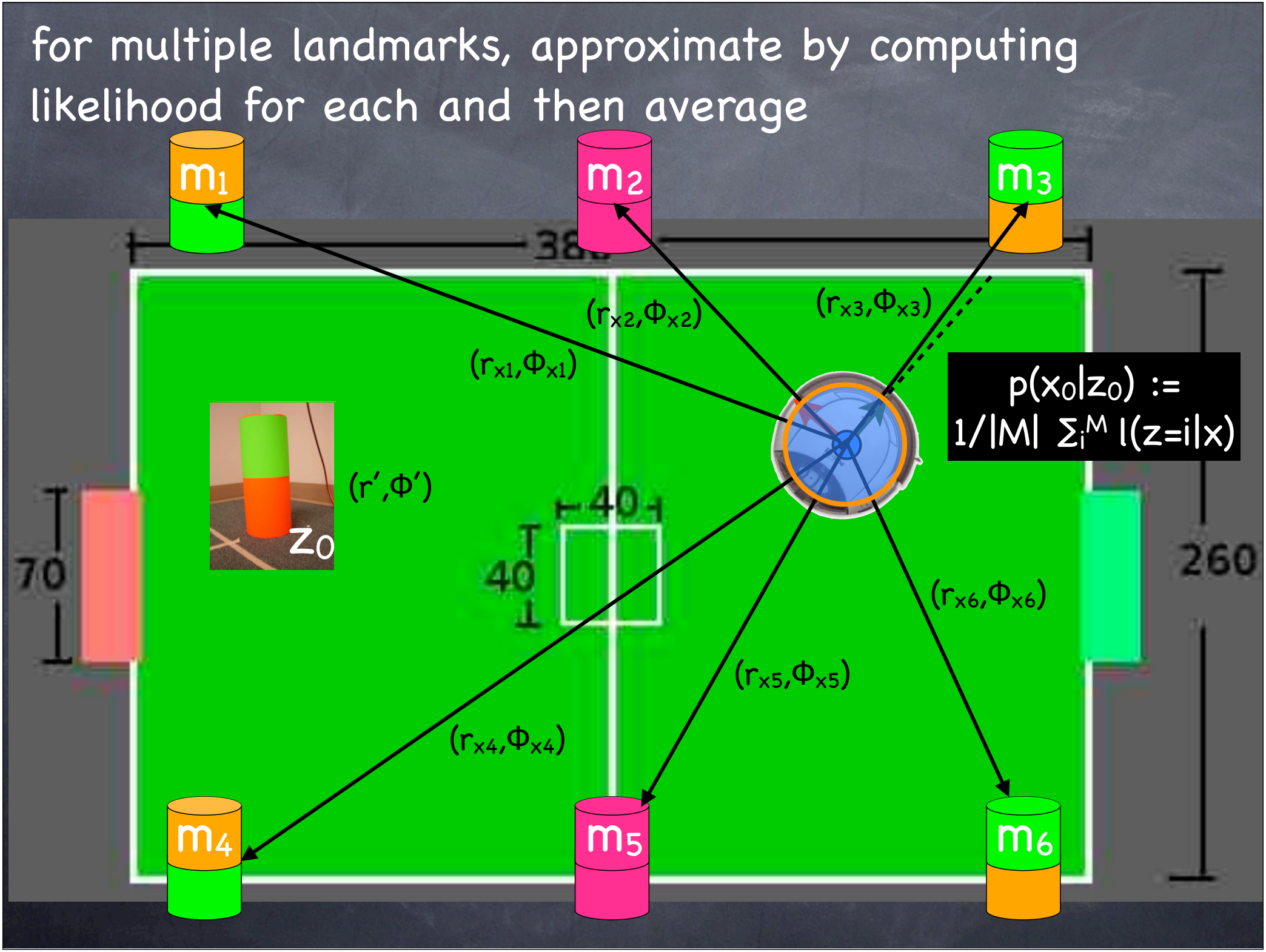

"low", if out view and $-30° < \Phi_x < 30°$


"high", if in view and $-30° < \Phi_x < 30°$

$\exp(-\Delta\Phi * \Delta r)$,

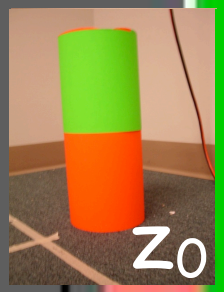
otherwise

for multiple landmarks, approximate by computing likelihood for each and then average



evaluate every pose sample

$$p(x_0|z_0) := 1/|M| \sum_i^M l(z=i|x)$$



m_1 m_2 m_3 m_4 m_5 m_6

70

260

Particle filter recap

posterior likelihood dynamics prior

$$p(\mathbf{x}_k|\mathbf{Z}_k) = \alpha\, p(\mathbf{Z}_k|\mathbf{x}_k) \int p(\mathbf{x}_k|\mathbf{x}_{k-1})\, p(\mathbf{x}_{k-1}|\mathbf{Z}_{k-1})$$

"belief at t=k" "update" "predict" "belief at t=k-1"

How to evaluate the likelihood of a pose given robot observations?

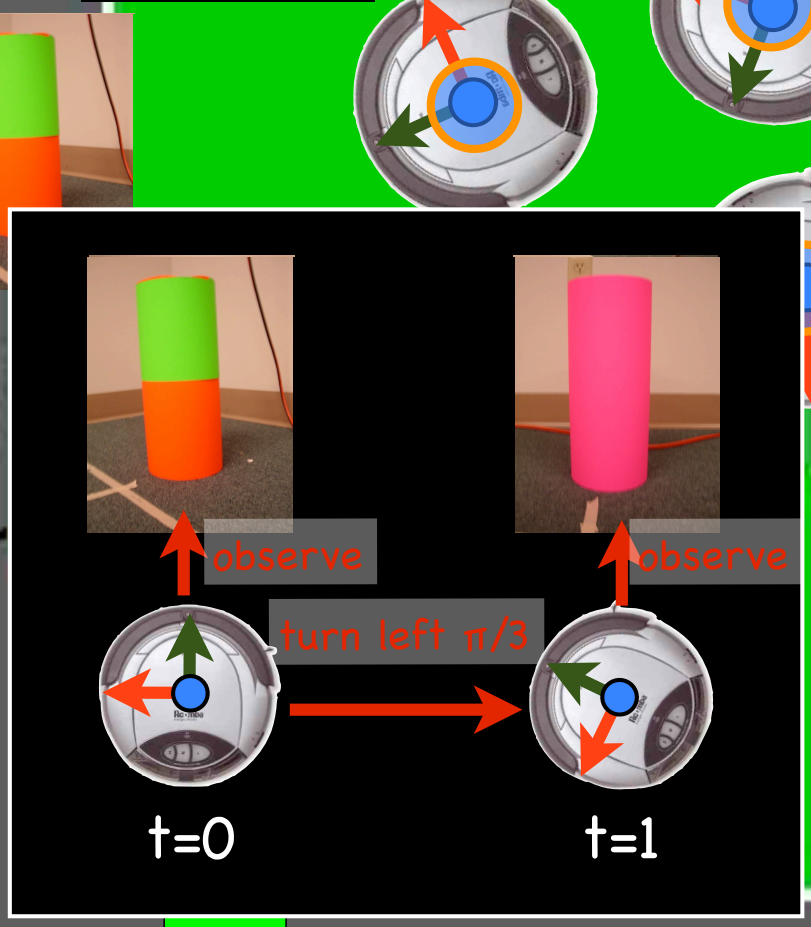
How to predict a new belief given robot odometry?

- recursive Markovian inference over time with predict-update loop
- Particle filter algorithms predict-update with sample set of poses

What are we still missing to compute localization?

note: belief for particle filter are pose samples weighted by probability (noted by size of ellipse)

$p(x_0|z_0)$



Particle filter steps

1.(Re)sample particles

2.Predict forward in time

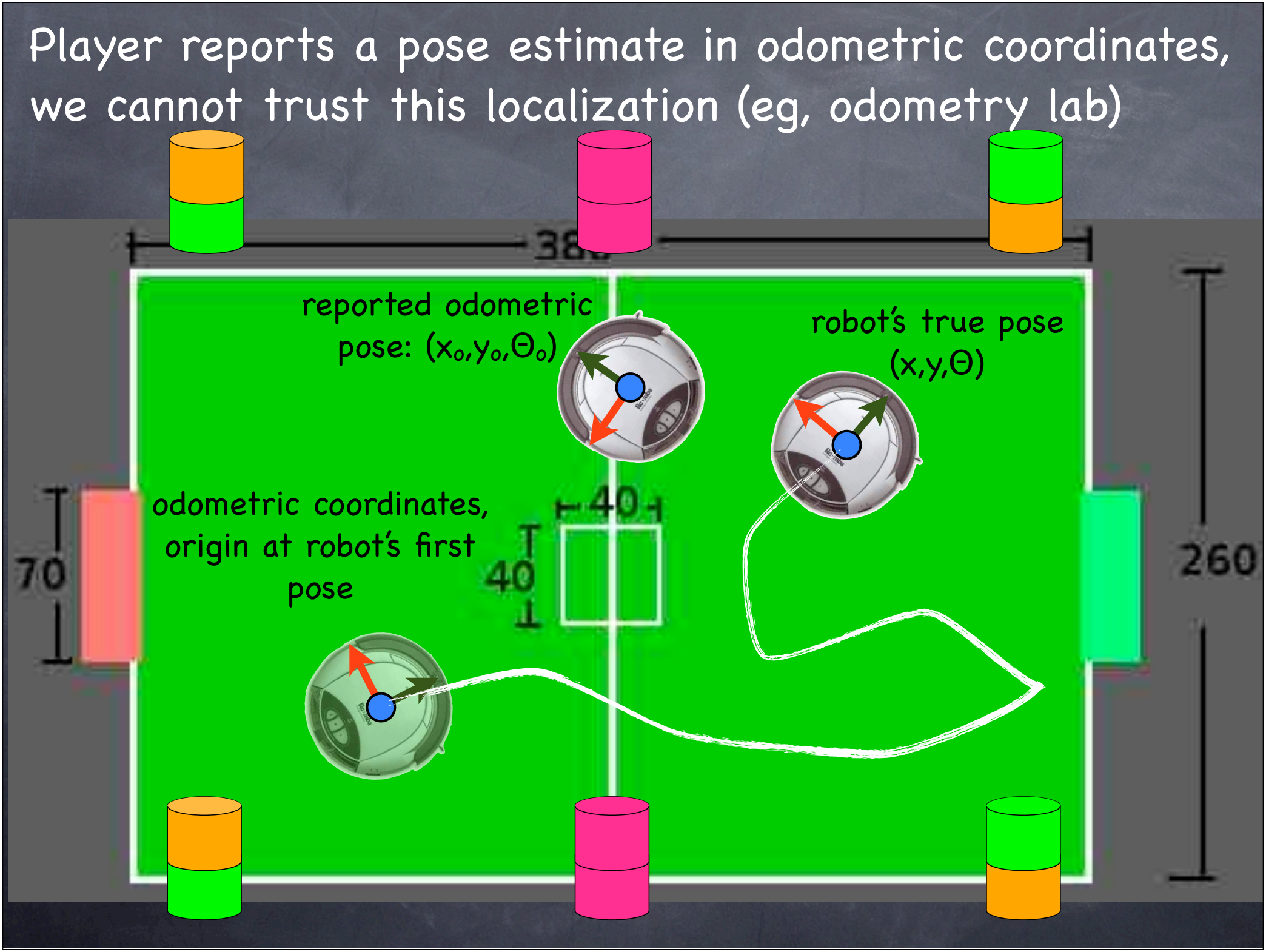
3.Update particle weights

4.Repeat

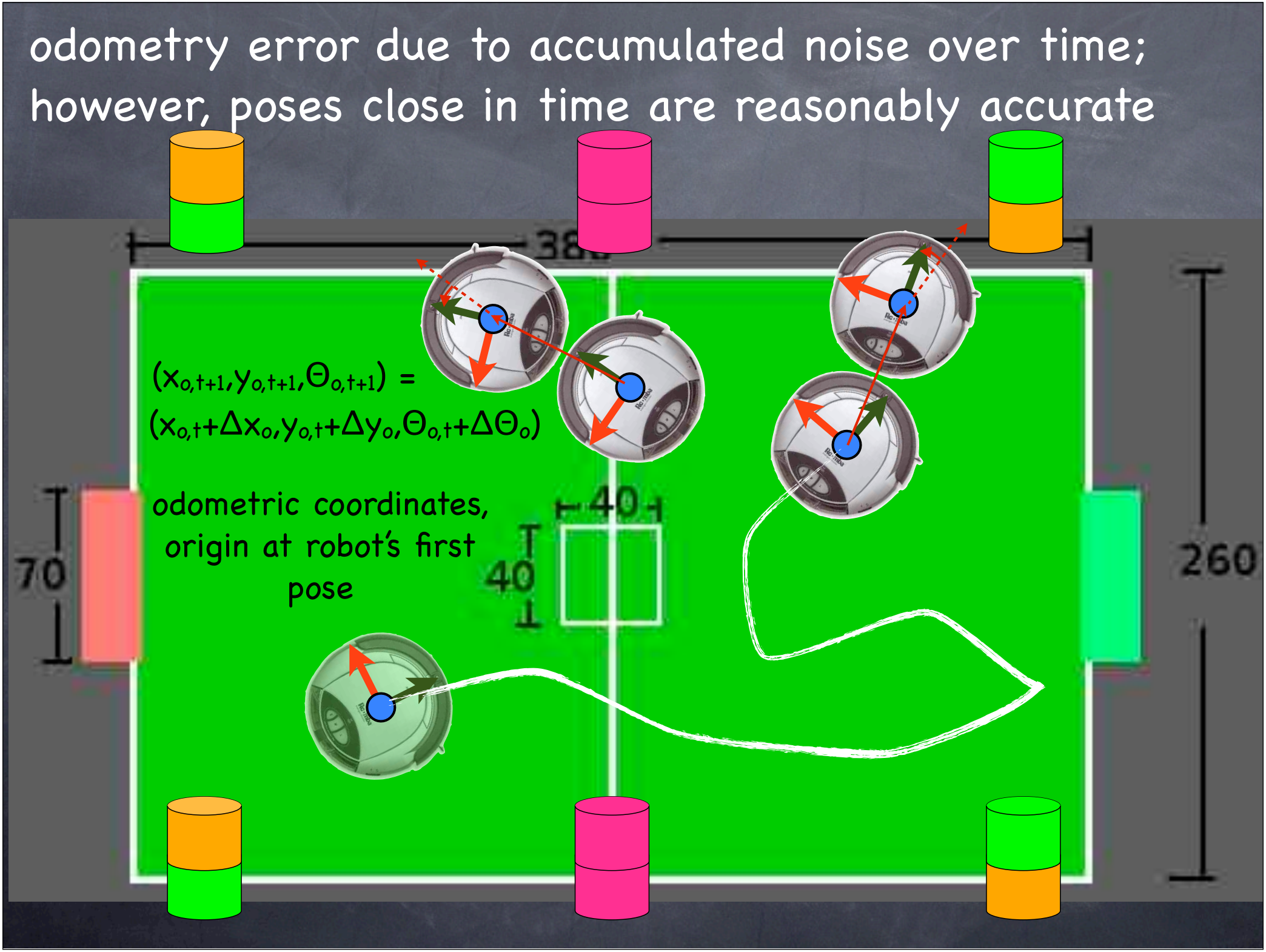
70

260

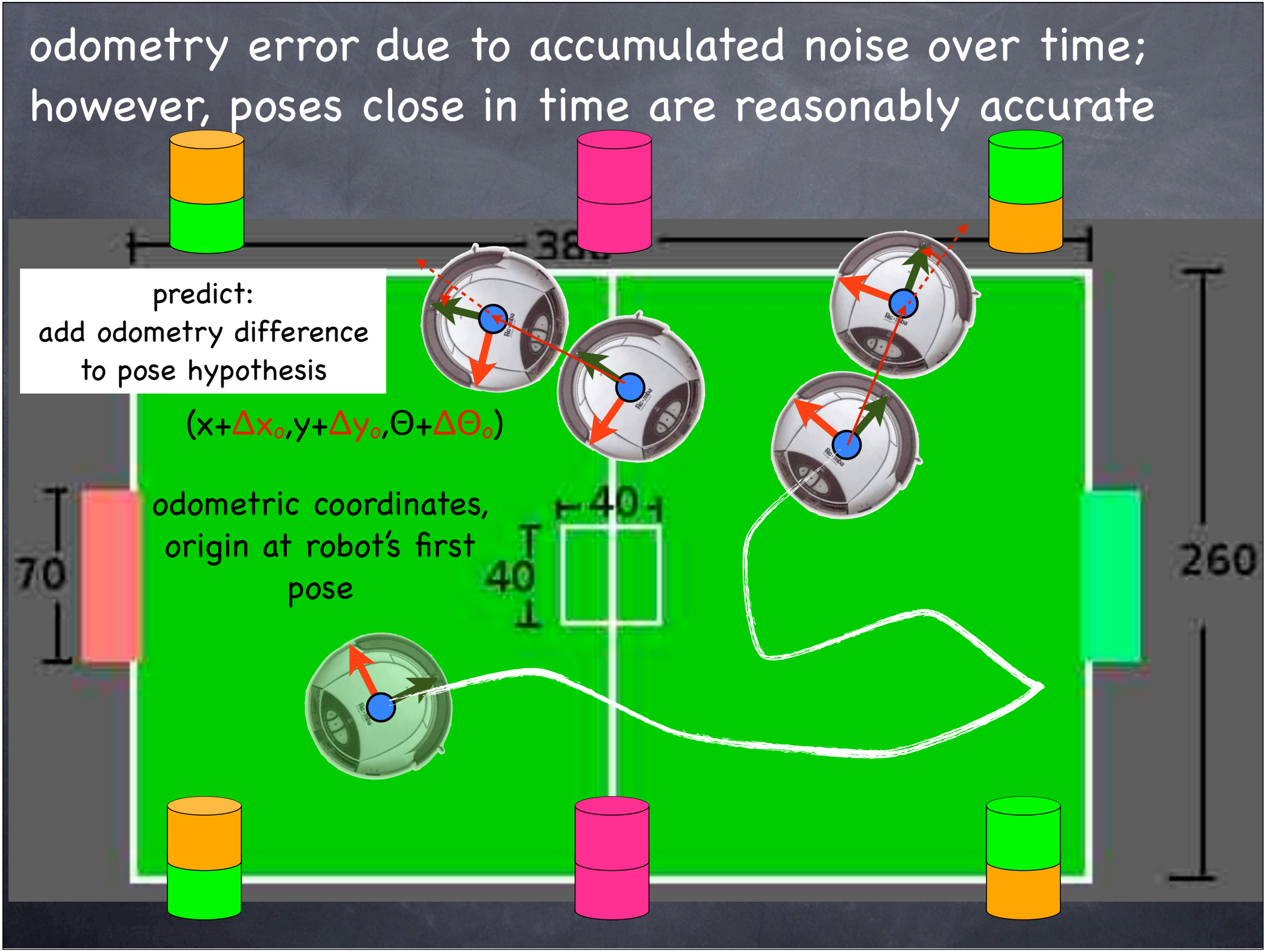
Player reports a pose estimate in odometric coordinates, we cannot trust this localization (eg, odometry lab)



odometry error due to accumulated noise over time;
however, poses close in time are reasonably accurate

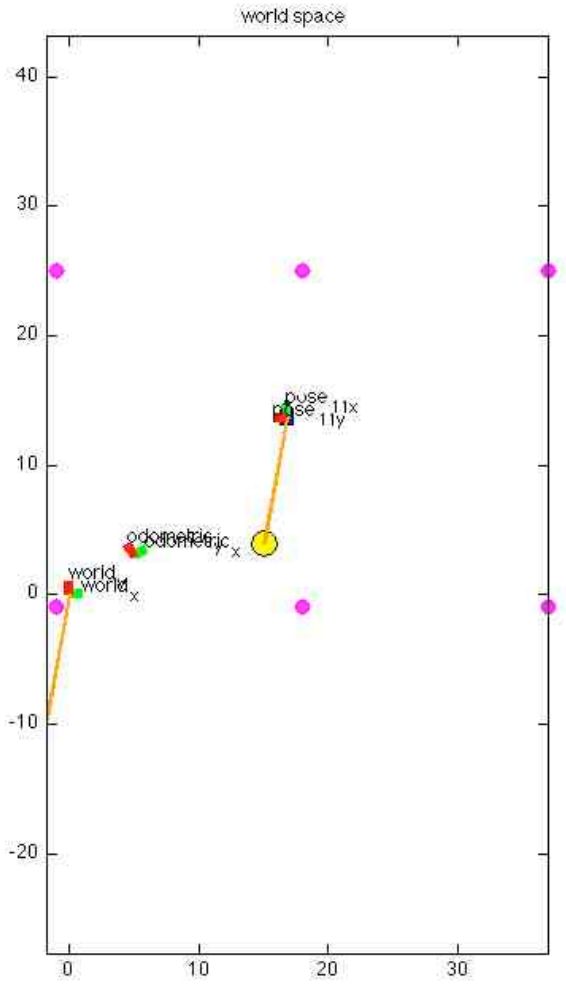


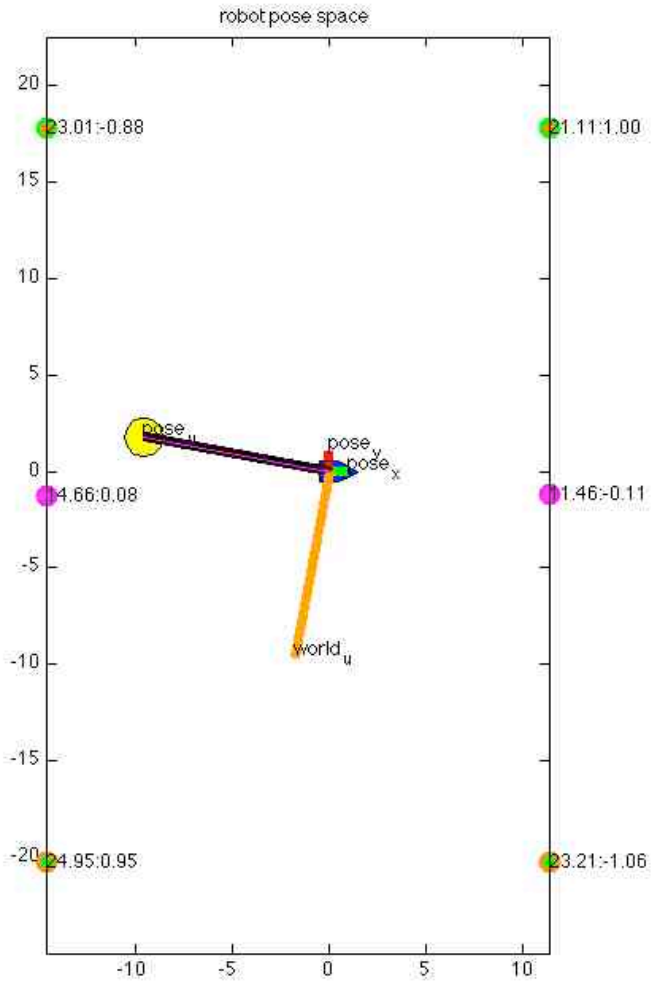
odometry error due to accumulated noise over time;
however, poses close in time are reasonably accurate



matlab example

- field_coordinate_transform.m





matlab example

- /course/cs148/pub/range_function_approximation.m