Tracking and Localization
Kalman Filters and HMMs

CS 4758

Ashutosh Saxena

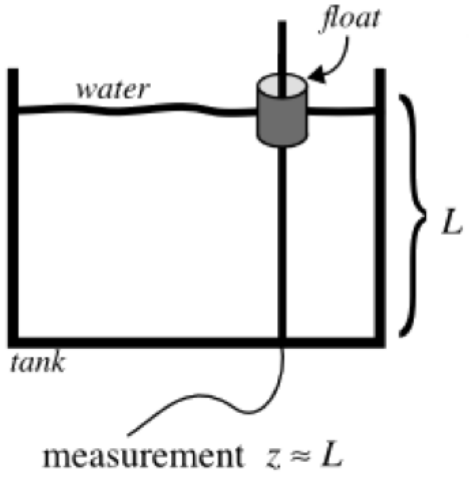
Cornell University

Application Examples.

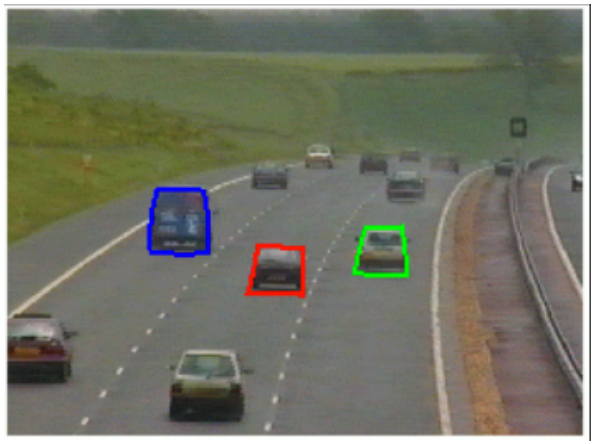
- Tracking in 1D.
- Tracking in 2D (visual tracking).

- Localization in 2D.

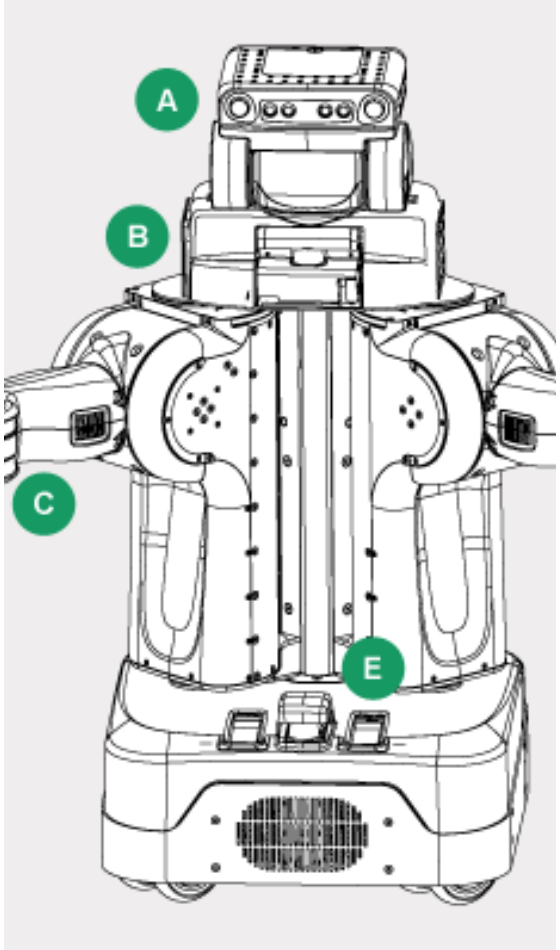
Tracking

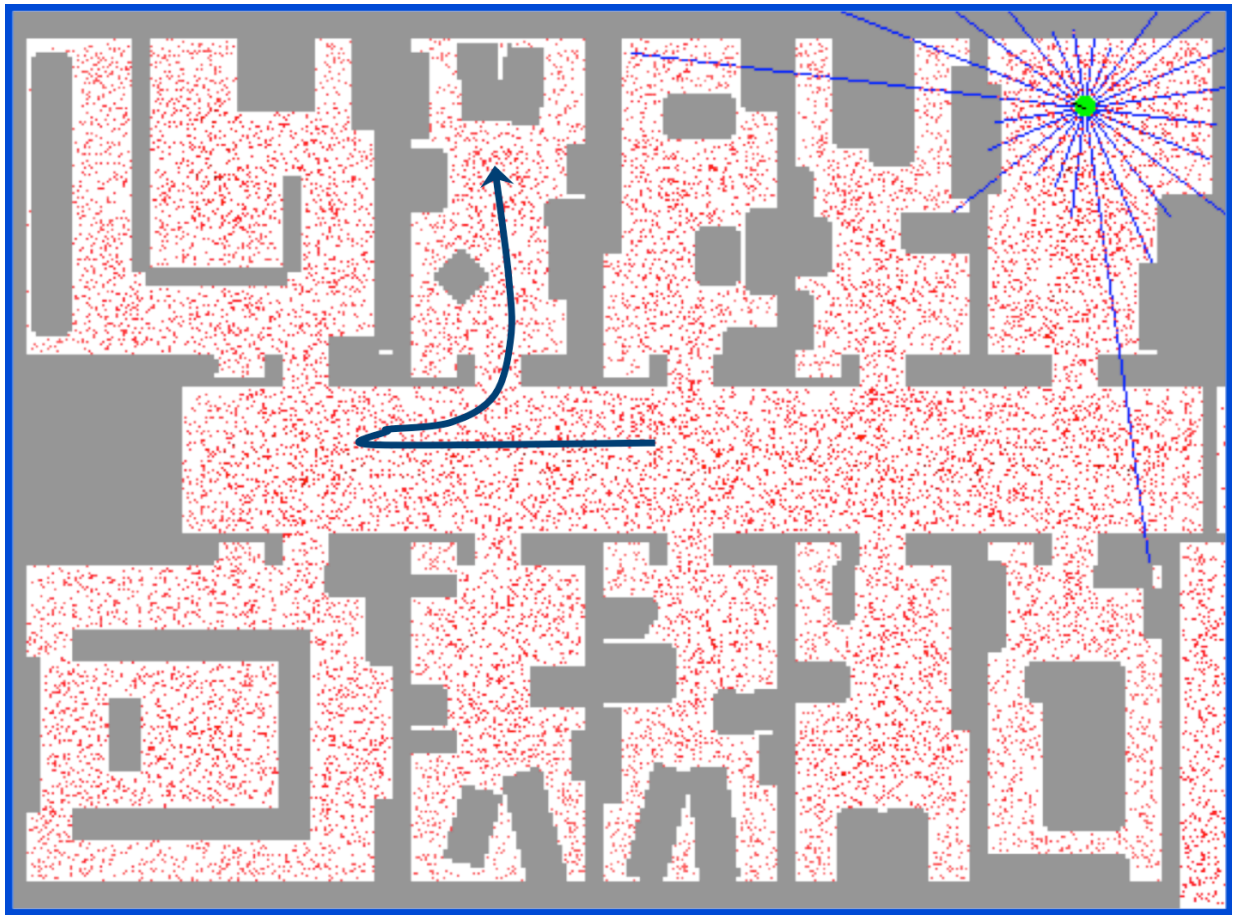


Visual Tracking



Localization in 2D





Learning Algorithms.

- **R**epresentation.
- **M**odel.
- **I**nference.
- **L**earning.

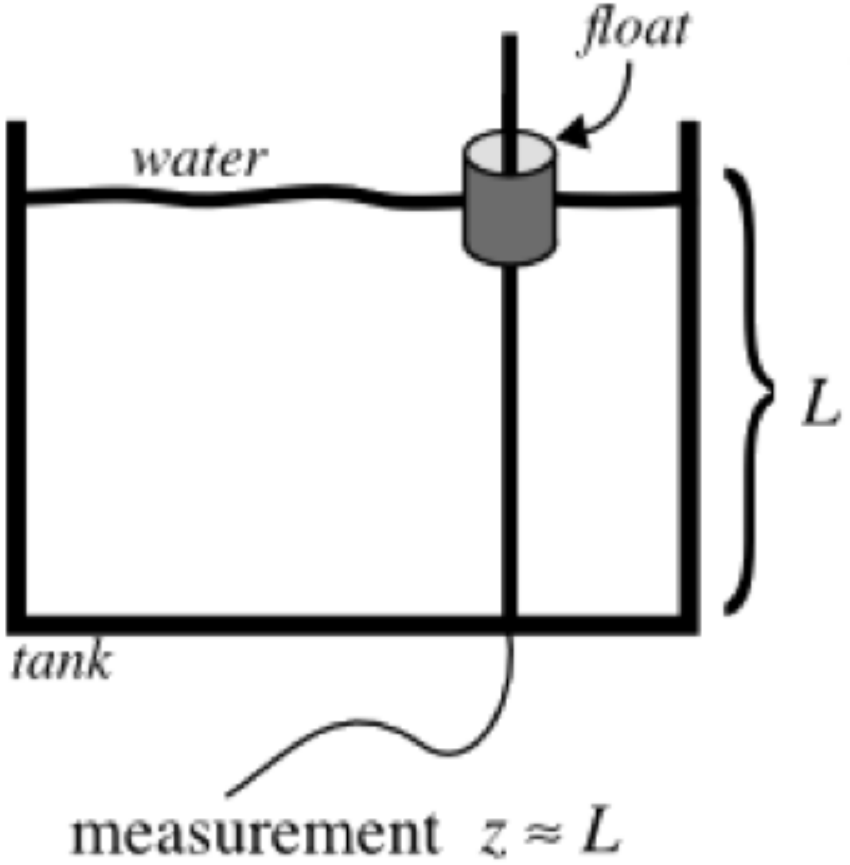
Others:

- Dataset.
- Details: Features, etc.

TRACKING IN 1D

Tracking in 1D: Representation

- Problem specification: Static boat, only height varies.
- The boat has a noisy altitude sensor.



Tracking in 1D: Model

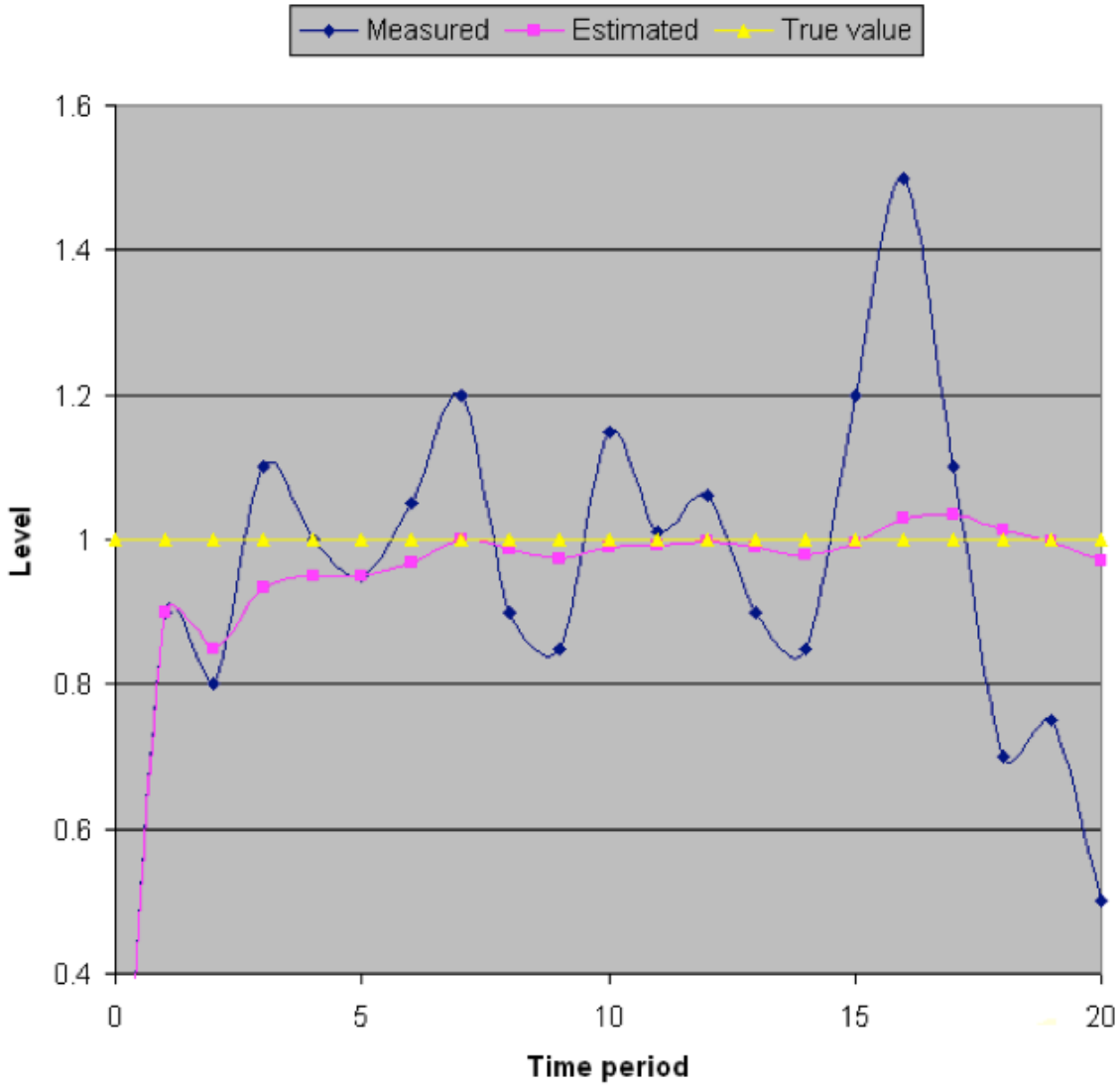
What reasonable assumptions we can make about the environment (i.e., boat and the water)?

Option 1: static model

q = .0001

	Predict		Update			
t	$x_{t\|t-1}$	$p_{t\|t-1}$	y_t	K_t	$x_{t\|t}$	$p_{t\|t}$
3	0·8499	0·0501	1·1	0·3339	0·9334	0·0334
4	0·9334	0·0335	1	0·2509	0·9501	0·0251
5	0·9501	0·0252	0·95	0·2012	0·9501	0·0201
6	0·9501	0·0202	1·05	0·1682	0·9669	0·0168
7	0·9669	0·0169	1·2	0·1447	1·0006	0·0145
8	1·0006	0·0146	0·9	0·1272	0·9878	0·0127
9	0·9878	0·0128	0·85	0·1136	0·9722	0·0114
10	0·9722	0·0115	1·15	0·1028	0·9905	0·0103

Option 1: static model results



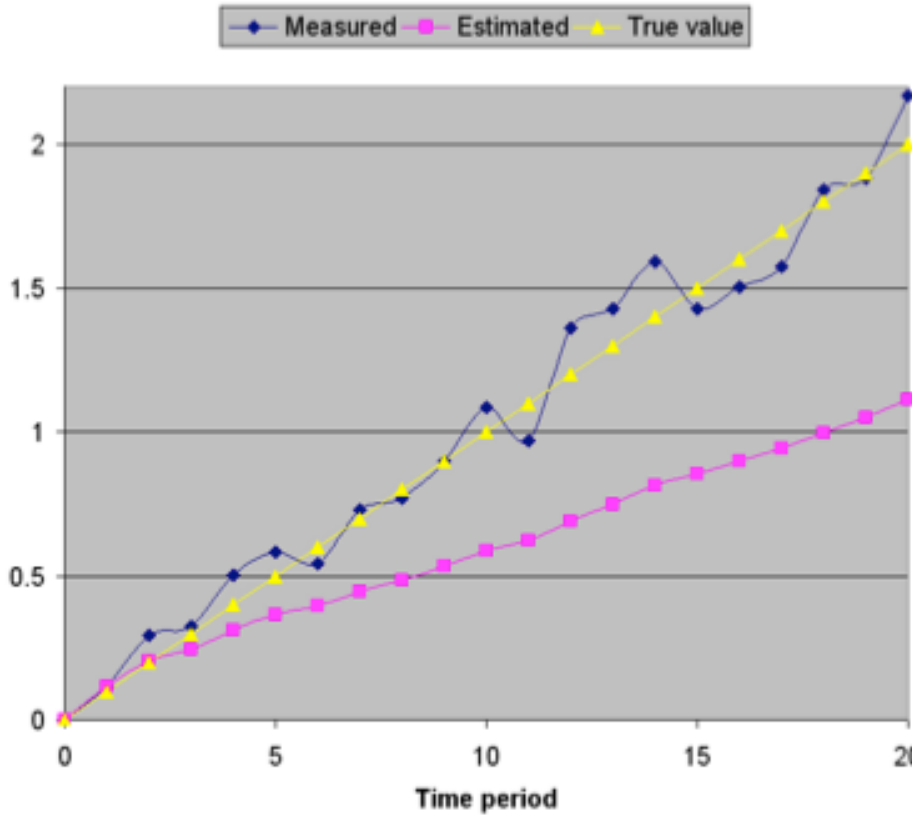
- Even even measurements are noisy (i.e., 20% error), the Kalman filter produced estimates close to true value (e.g., 5%).

- Good Model -> Good Estimates.

Wrong model?

- Say the dam in the lake was filling up at a constant rate.

- And we use the same model?



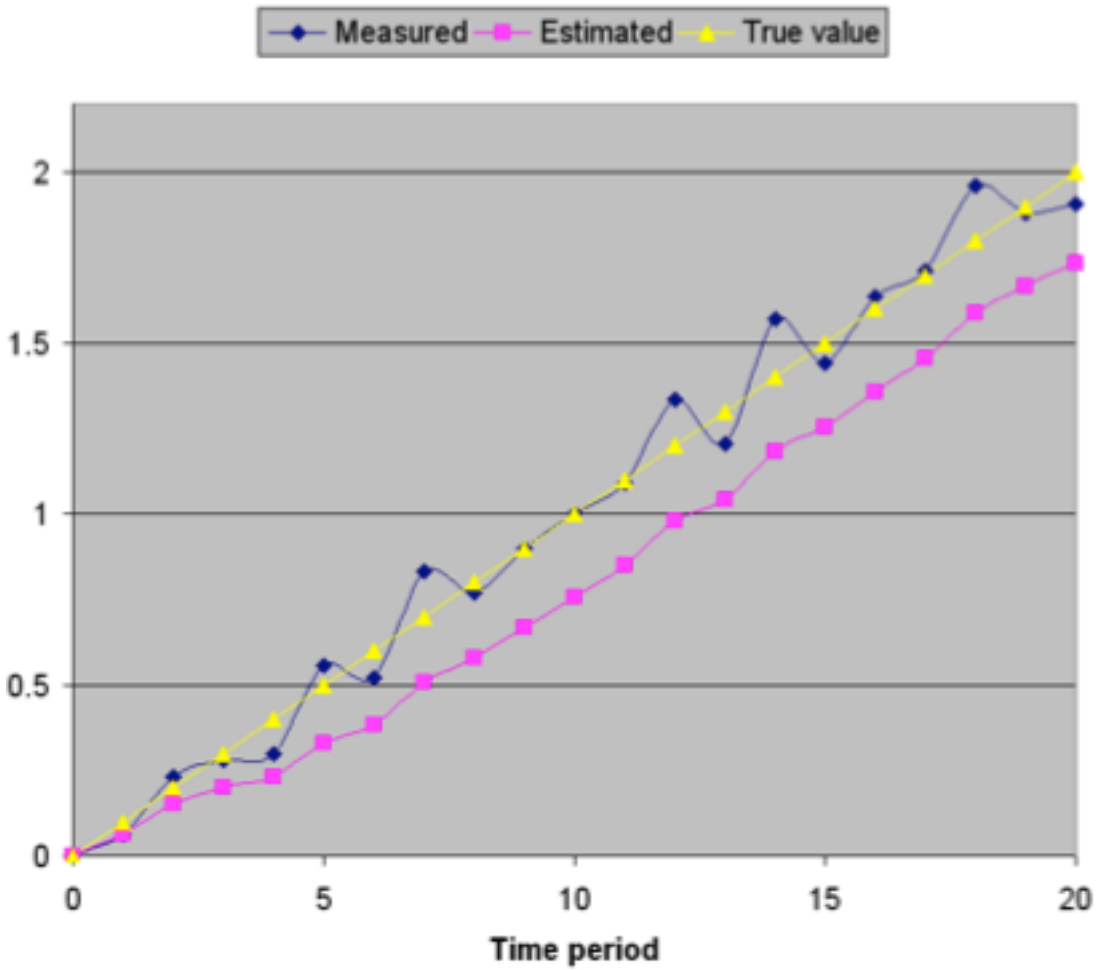
	Predict		Measurement and Update				Truth
t	$x_{t\|t-1}$	$p_{t\|t-1}$	y_t	K_t	$x_{t\|t}$	$p_{t\|t}$	L
0	—	—	—	—	0	1000	0
1	0·0000	1000·0001	0·11	0·9999	0·1175	0·1000	0·1
2	0·1175	0·1001	0·29	0·5002	0·2048	0·0500	0·2
3	0·2048	0·0501	0·32	0·3339	0·2452	0·0334	0·3
4	0·2452	0·0335	0·50	0·2509	0·3096	0·0251	0·4
5	0·3096	0·0252	0·58	0·2012	0·3642	0·0201	0·5
6	0·3642	0·0202	0·54	0·1682	0·3945	0·0168	0·6

How to fix this?

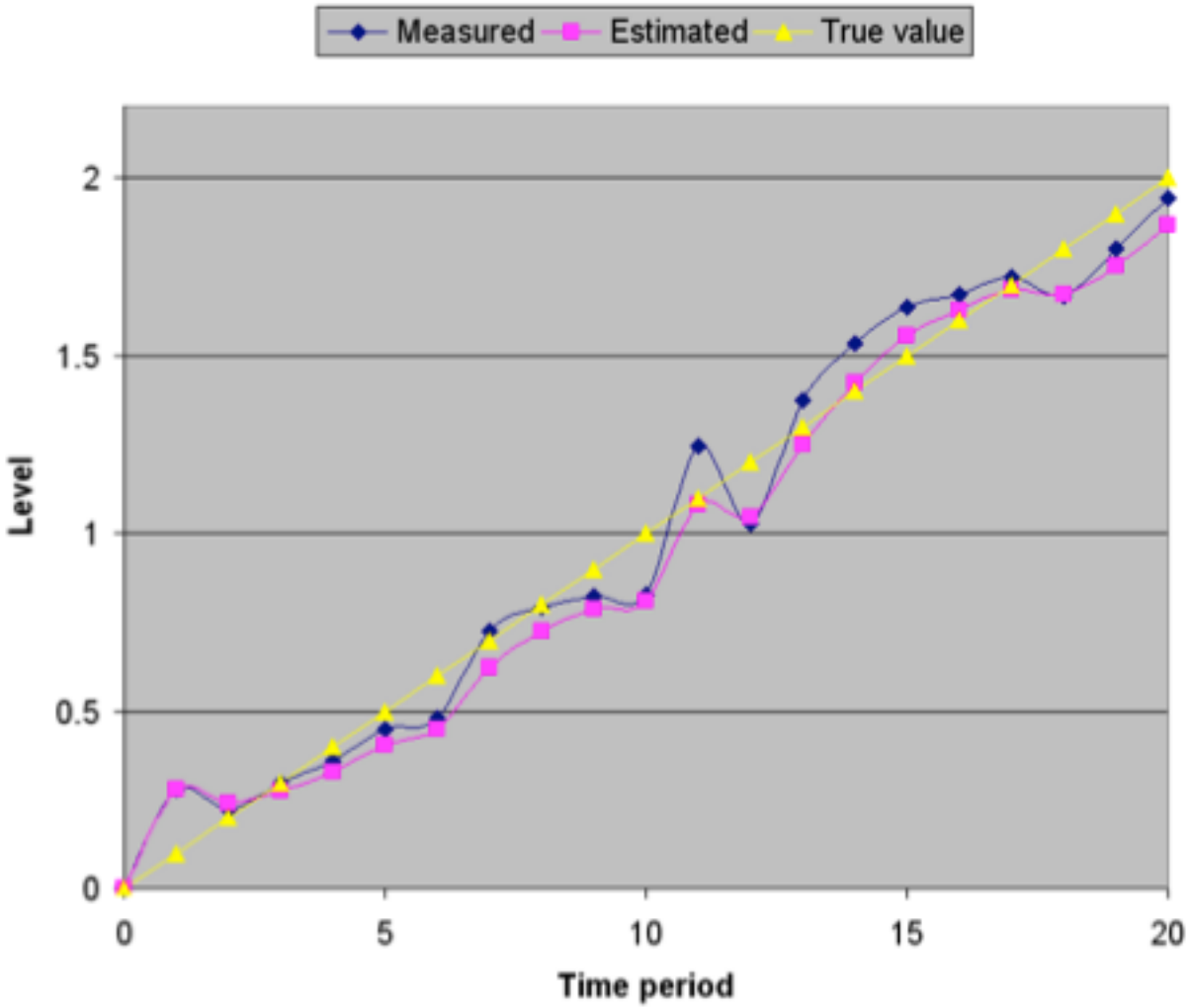
The following two assumptions could be causing this:

- The model we have chosen.
- The parameters of our model.

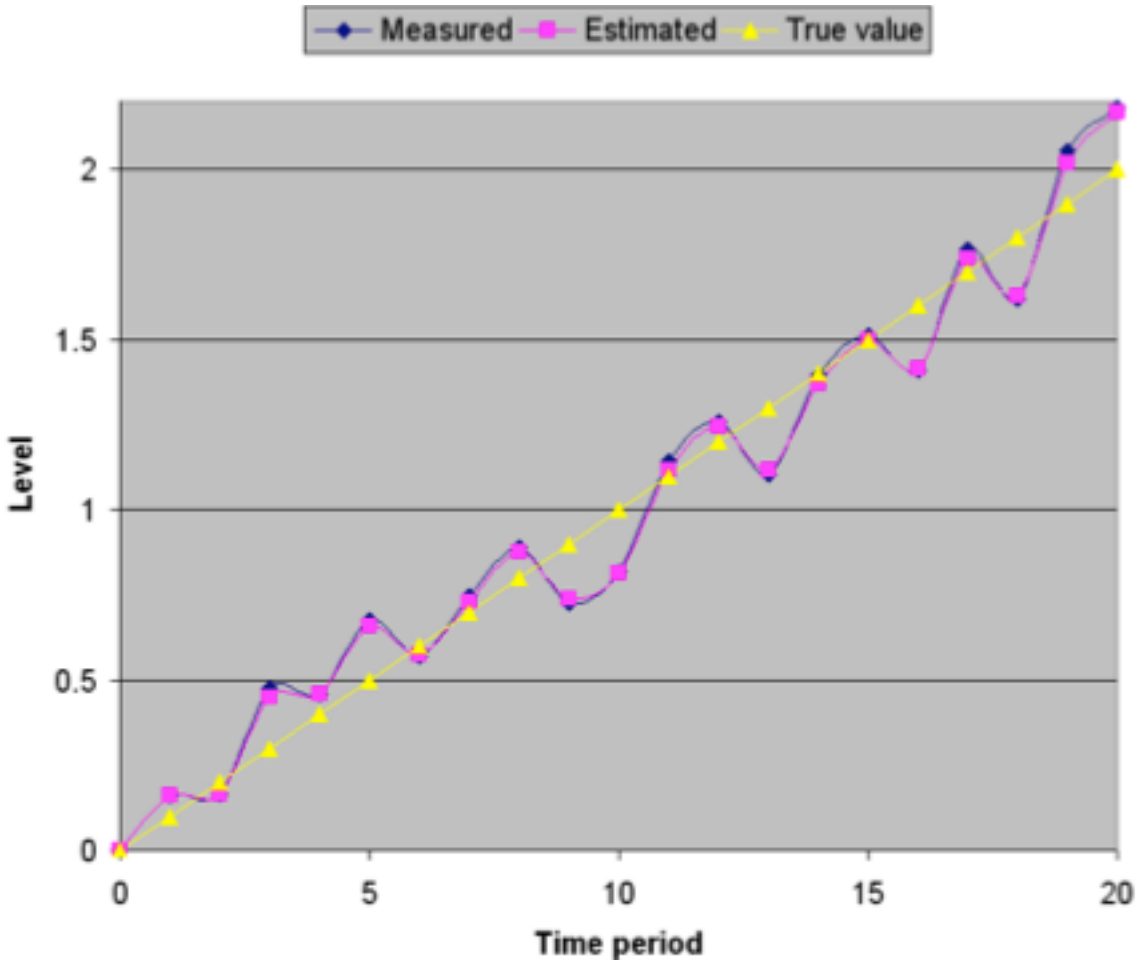
q = .01


Measured
Estimated
True value
2
1.5
1
0.5
0
0
5
10
15
20
Time period

q = 0.1


Measured
Estimated
True value
Level
2
1.5
1
0.5
0
0
5
10
15
20
Time period

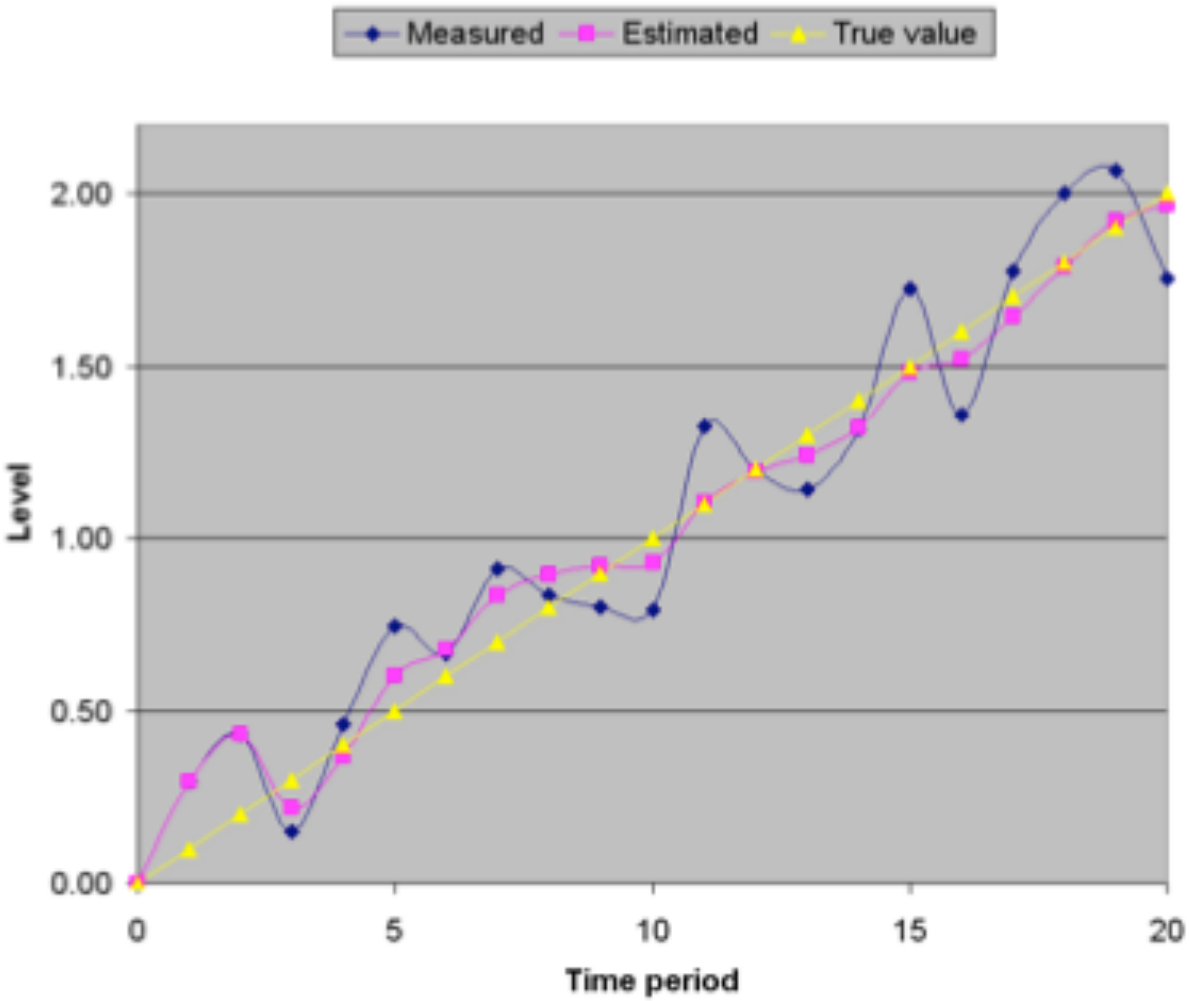
q = 1


Measured
Estimated
True value
Level
0
0.5
1
1.5
2
0
5
10
15
20
Time period

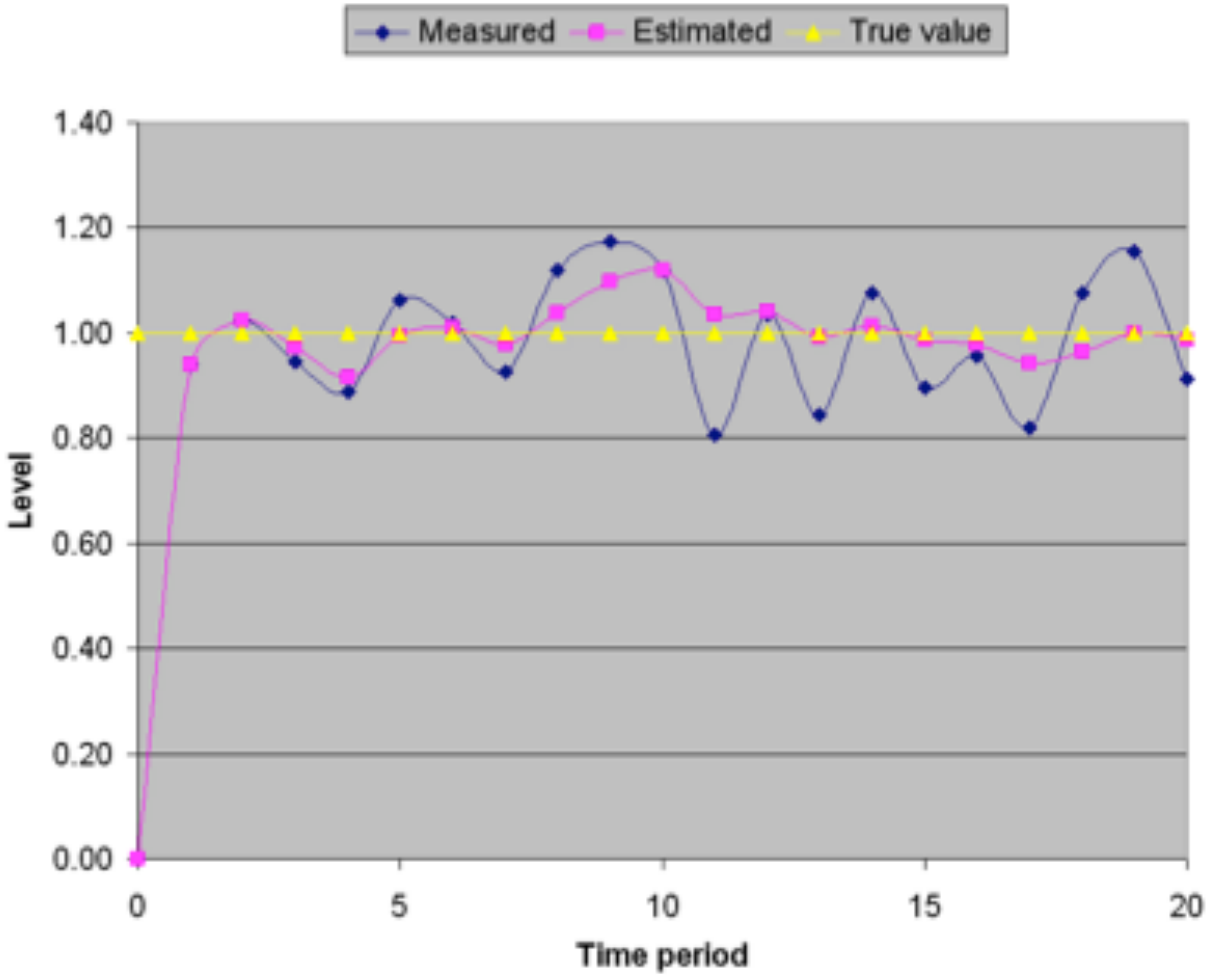
Second Option: Filling Model

- Explicitly model slowly changing water level.

Filling Model.



Filling Model: results on static case



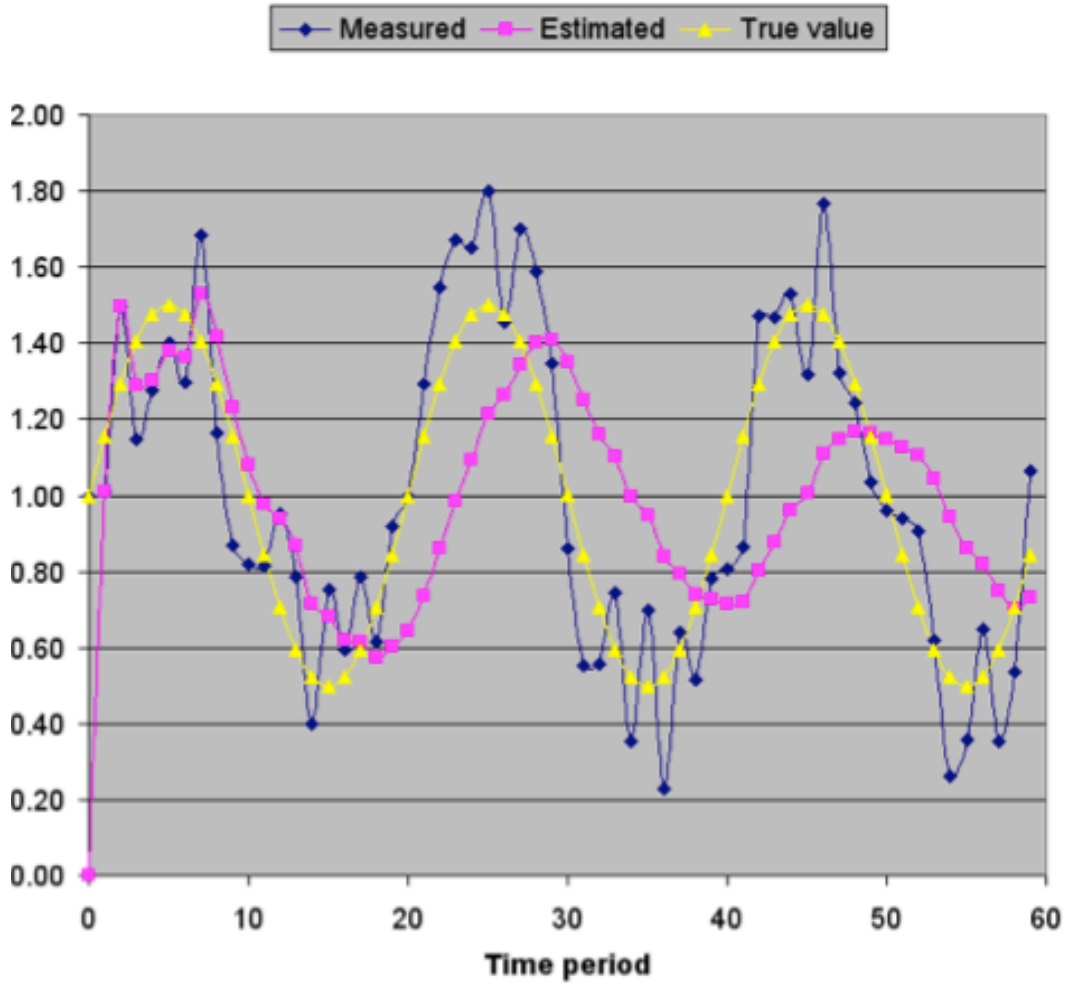
Waves in the Lake!

$$L = c\ \sin(2\pi\ r\ \Delta t) + l$$

Use Filling Model

- Model is smoother, but there is **lag.**
- Amplitude of the model grows smaller over time.

How to fix this?

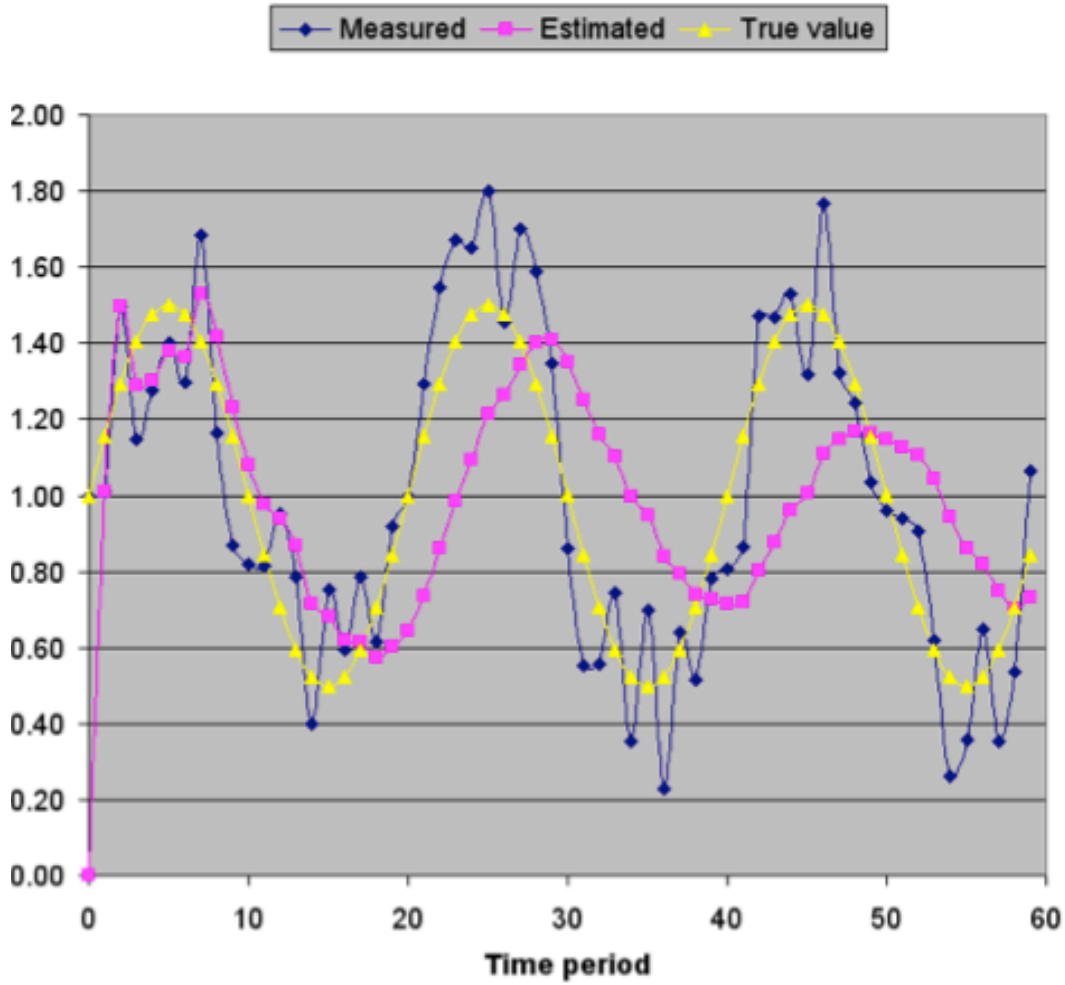


Use Filling Model

- Model is smoother, but there is **lag.**
- Amplitude of the model grows smaller over time.

How to fix this?

Extended Kalman Filter.



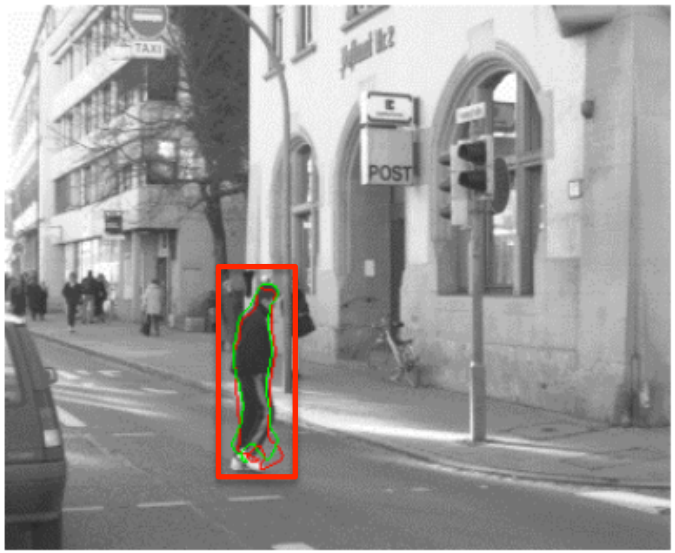
TRACKING IN 2D

Visual Tracking in 2D

- Problem: Find out a given object in every frame.

Visual Tracking in 2D: Representation

- Problem: Find out the (x,y) and (len,width) in each frame.

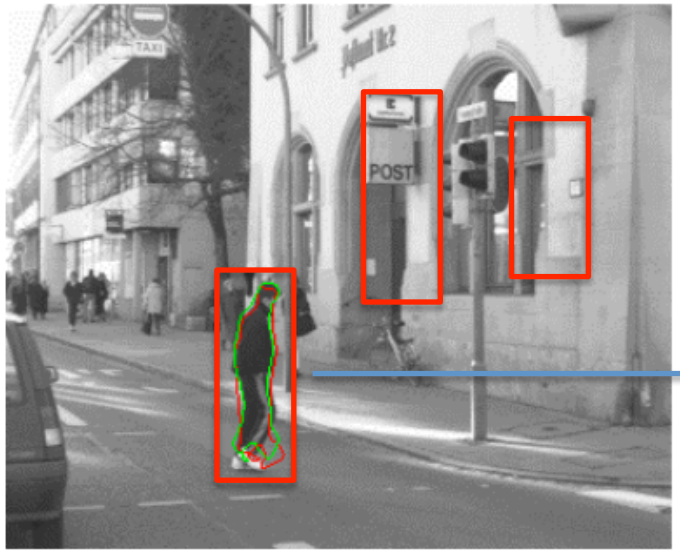


Visual Tracking in 2D: Model

Two parts

- Per frame detection.
- Tracking in time.

Visual Tracking in 2D: Detector



Features → Detector (logistic classifier, SVM) → Probability of being person (0,1).

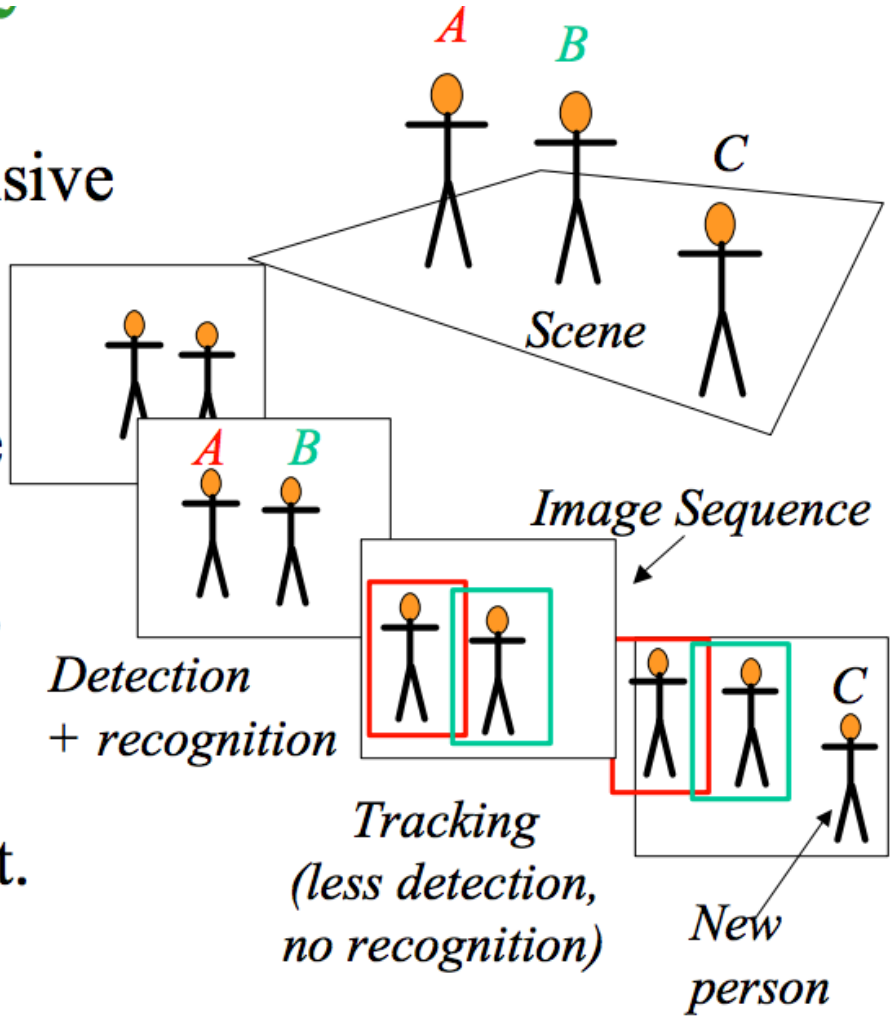
Representation: y={0,1}
Does the rectangle at (x,y,len,width) contains a person.

Model: Use logistic classifier or SVM, which has parameters **w.** (Cost function?)
Learning algorithm: Gradient descent. (Or use existing code.)
Inference algorithm: $\mathbf{w^T \Phi(x) > 0}$ Which side of separating hyperplane does it lie.

Other details: Which features to use, dataset, etc.

Why Tracking?

- Detection and recognition are expensive
- If we get an idea of where an object is in the image because we have an idea of the motion from previous images, we need less work detecting or recognizing the object.



Visual Tracking in 2D: Tracking

Representation: ?

Model: ?

(a) Frame # 1

(b) Frame # 9

(c) Frame # 17

(d) Frame # 18

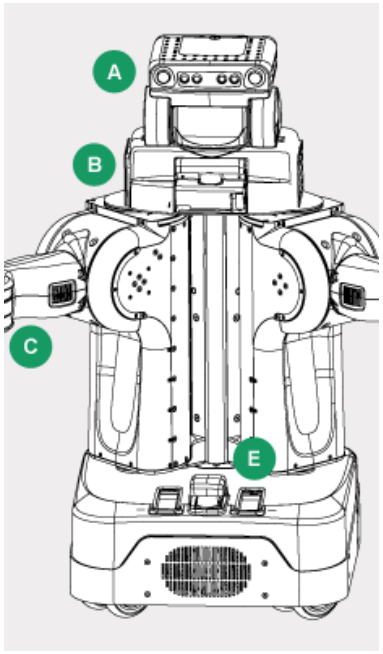
(e) Frame # 25

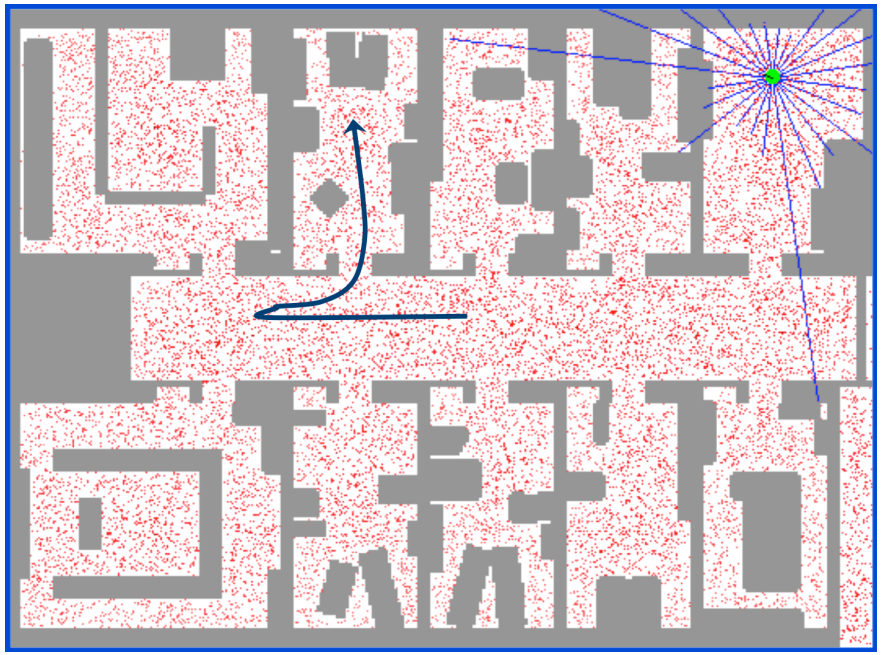
(f) Frame # 29

Tracking in 3D using Kinect

- Representation: ?

- Model: ?

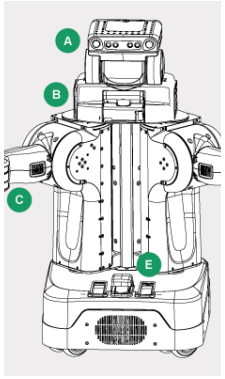


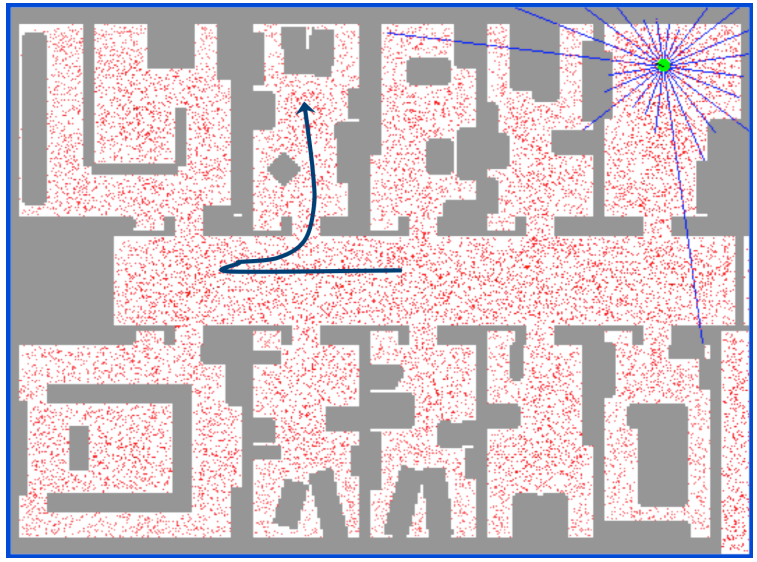


ROBOT LOCALIZATION

Problem

- Sensor: laser scans.
- Given a 2D map.

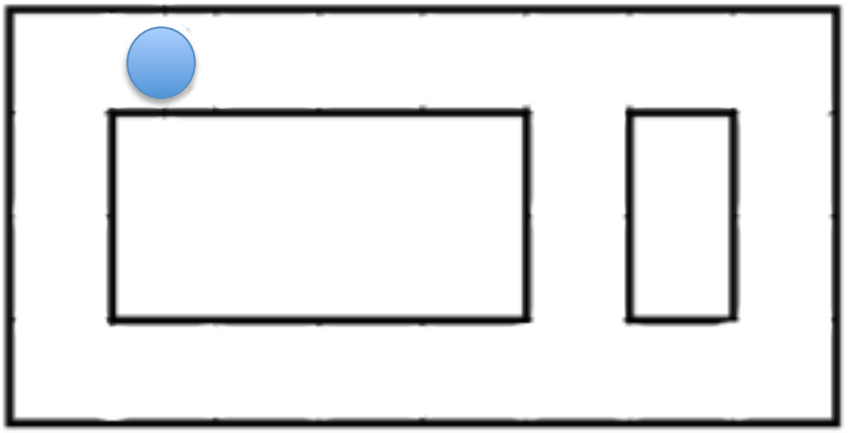




- Figure out where in the 2D map the robot is, i.e., (x,y) location of the robot.

Simpler Problem

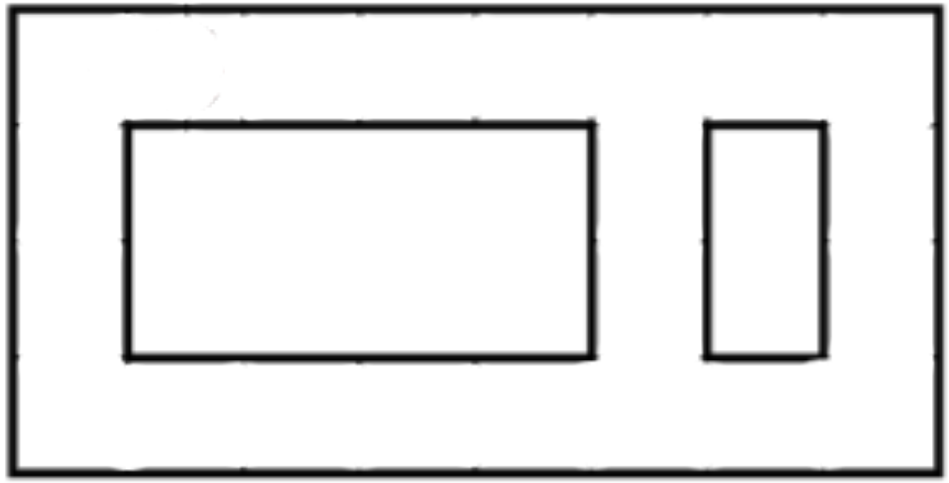
- Sensor: presence/absence of well in each direction.
- Given a 2D map.



- Figure out where in the 2D map the robot is, i.e., (x,y) location of the robot.

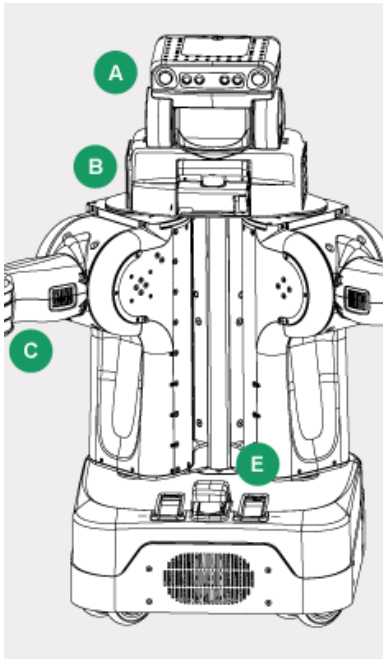
Representation

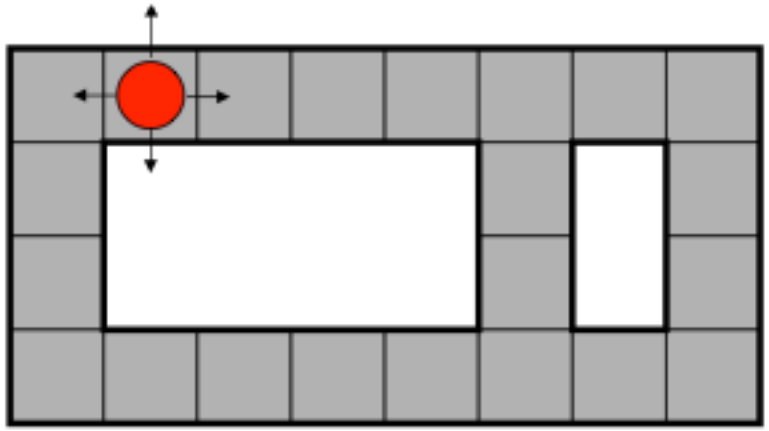
- Kalman Filters?
  - How to interpret map boundaries?
- HMM?



Localization in 2D: HMM representation

- Representation:
  - State is a cell in the discretized space.
  - Observation is {0,1} for each sensor direction.





2D Localization: Model

$\{N, M, a_{ij}, b_i(j), \pi_i\}$

$N =$

$M =$

No knowledge about robot state in the beginning:

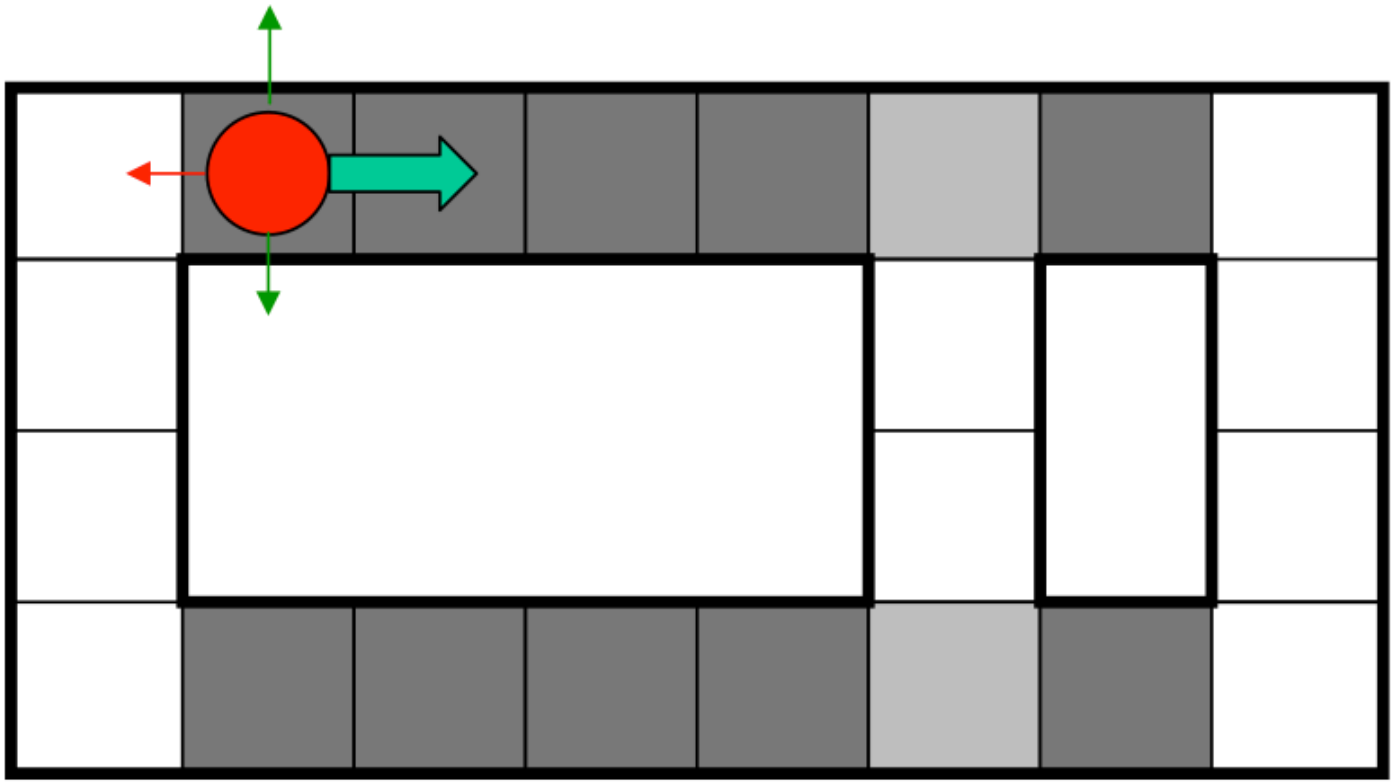
$\pi_i =$

Sensors make never more than one mistake*: $b_i(j) =$*

25% robot moves in each direction:

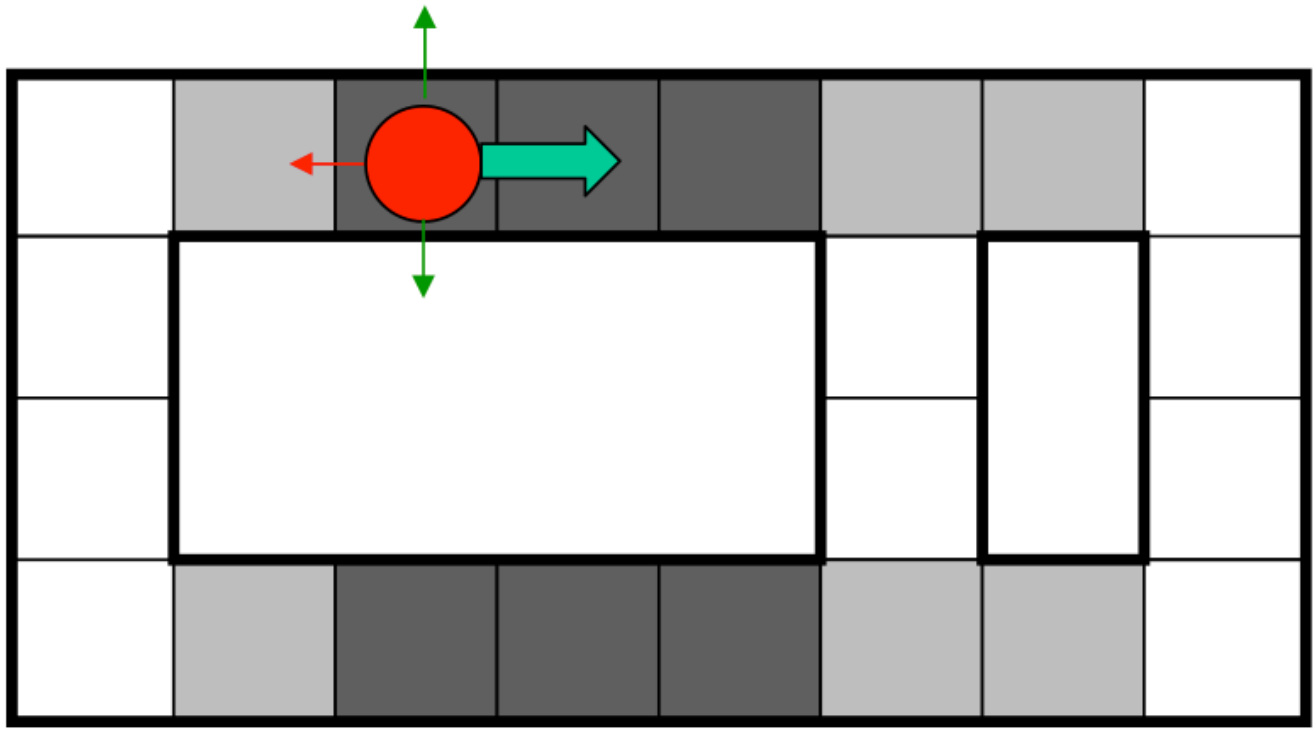
$a_{ij} =$

t=1



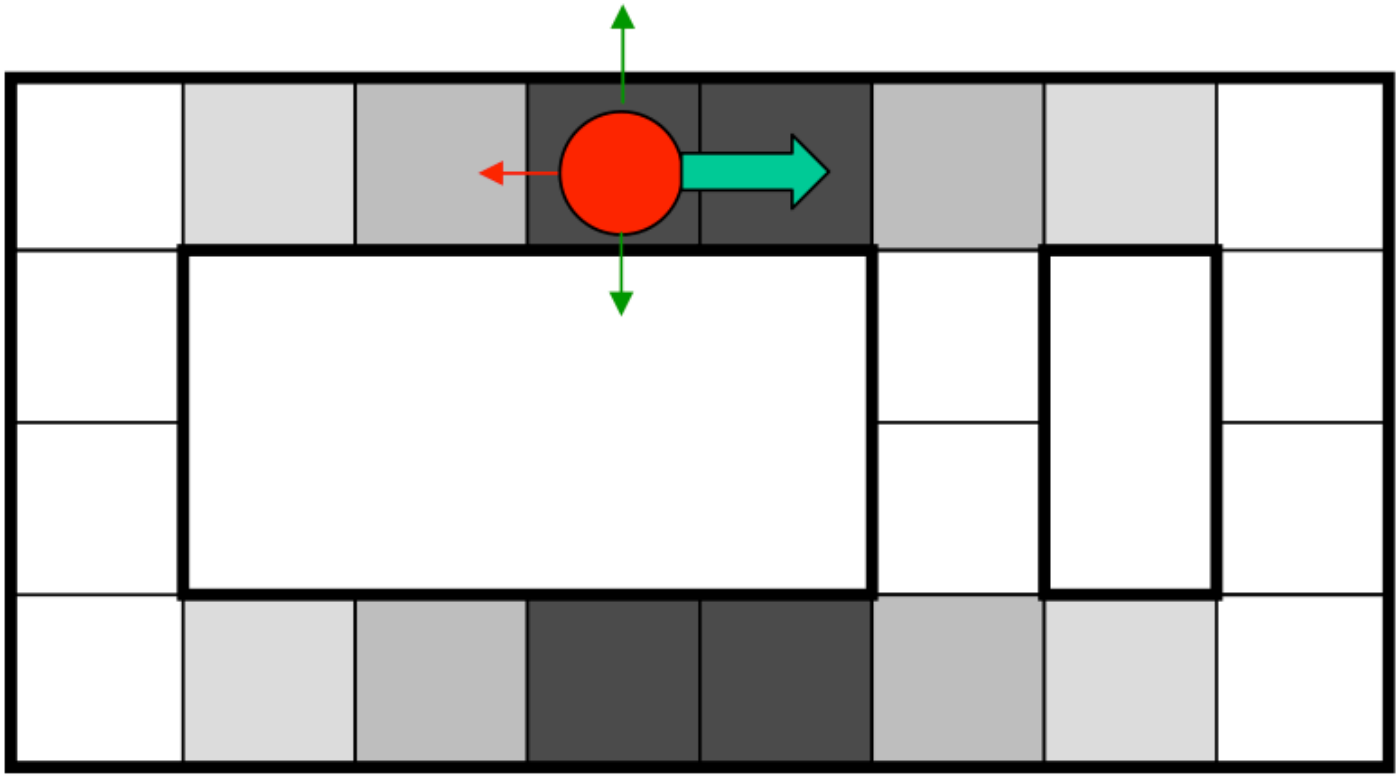
Prob 0 1

t = 2



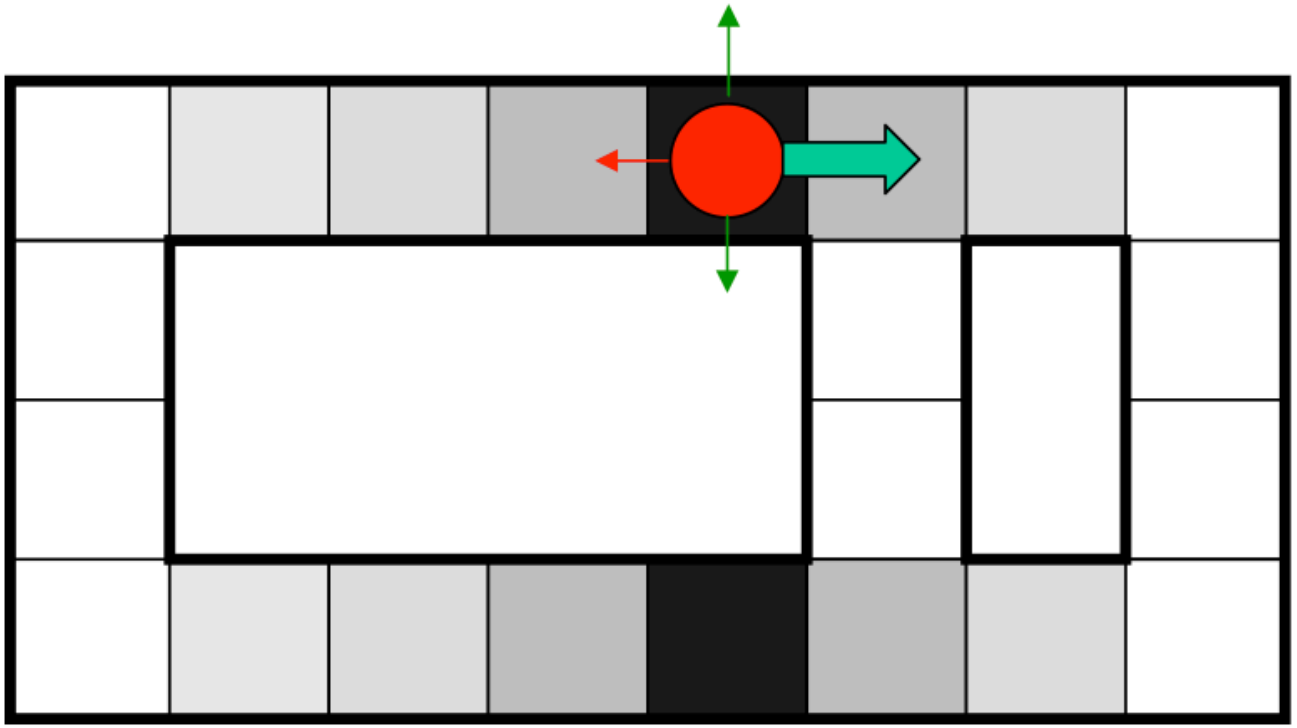
Prob 0 1

t = 3



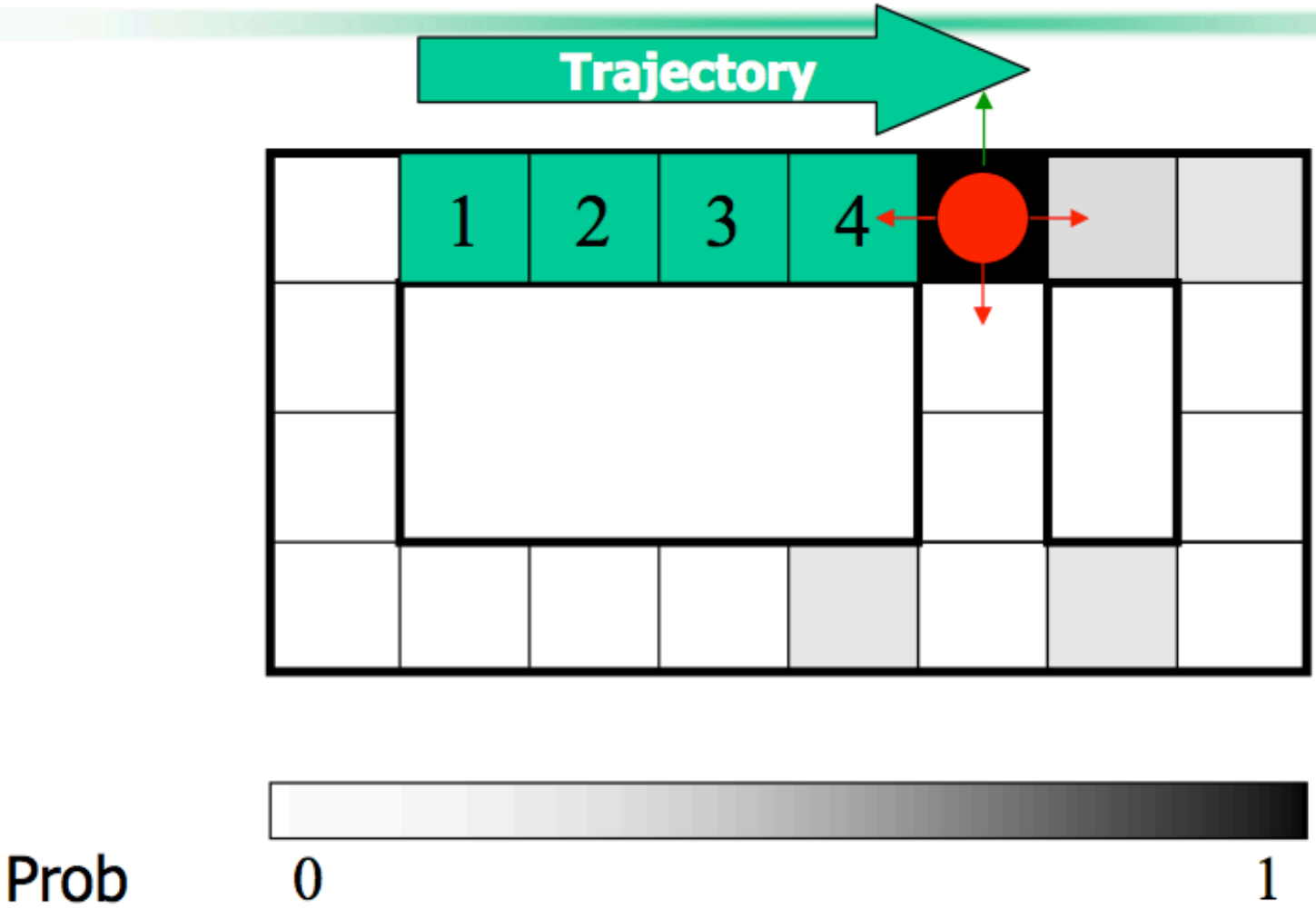
Prob 0 1

t = 4

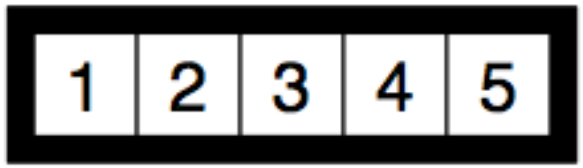


Prob 0 1

t = 5



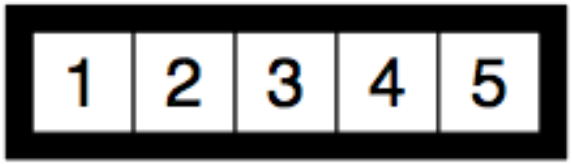
Even Simpler Example



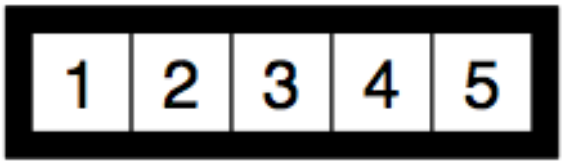
- Consider the 5-state hallway shown above
- The start state is always state 3
- The observation is the number of walls surrounding the state (2 or 3)
- There is a 0.5 probability of staying in the same state, and 0.25 probability of moving left or right; if the movement would lead to a wall, the state is unchanged.

	start	to state					see walls				
state		1	2	3	4	5	0	1	2	3	4
1	0.00	0.75	0.25	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00
2	0.00	0.25	0.50	0.25	0.00	0.00	0.00	0.00	1.00	0.00	0.00
3	1.00	0.00	0.25	0.50	0.25	0.00	0.00	0.00	1.00	0.00	0.00
4	0.00	0.00	0.00	0.25	0.50	0.25	0.00	0.00	1.00	0.00	0.00
5	0.00	0.00	0.00	0.00	0.25	0.75	0.00	0.00	0.00	1.00	0.00

Inference: HMM



Time t	1
Obs	2
$\alpha_t(1)$	0.00000
$\alpha_t(2)$	0.00000
$\alpha_t(3)$	1.00000
$\alpha_t(4)$	0.00000
$\alpha_t(5)$	0.00000



Time t	1	2	3
Obs	2	2	2
$\alpha_t(1)$	0.00000	0.00000	0.00000
$\alpha_t(2)$	0.00000	0.25000	0.25000
$\alpha_t(3)$	1.00000	0.50000	0.37500
$\alpha_t(4)$	0.00000	0.25000	0.25000
$\alpha_t(5)$	0.00000	0.00000	0.00000

Time t	1	2	3
Obs	2	2	3
$\alpha_t(1)$	0.00000	0.00000	0.06250
$\alpha_t(2)$	0.00000	0.25000	0.00000
$\alpha_t(3)$	1.00000	0.50000	0.00000
$\alpha_t(4)$	0.00000	0.25000	0.00000
$\alpha_t(5)$	0.00000	0.00000	0.06250

HMM

- Representation: state, observation.
- Model
- Inference

- Learning: ?
  - EM algorithm (use existing HMM toolboxes.)

PARTICLE FILTER

Localization: large maps, particle filters.

- Future lectures.