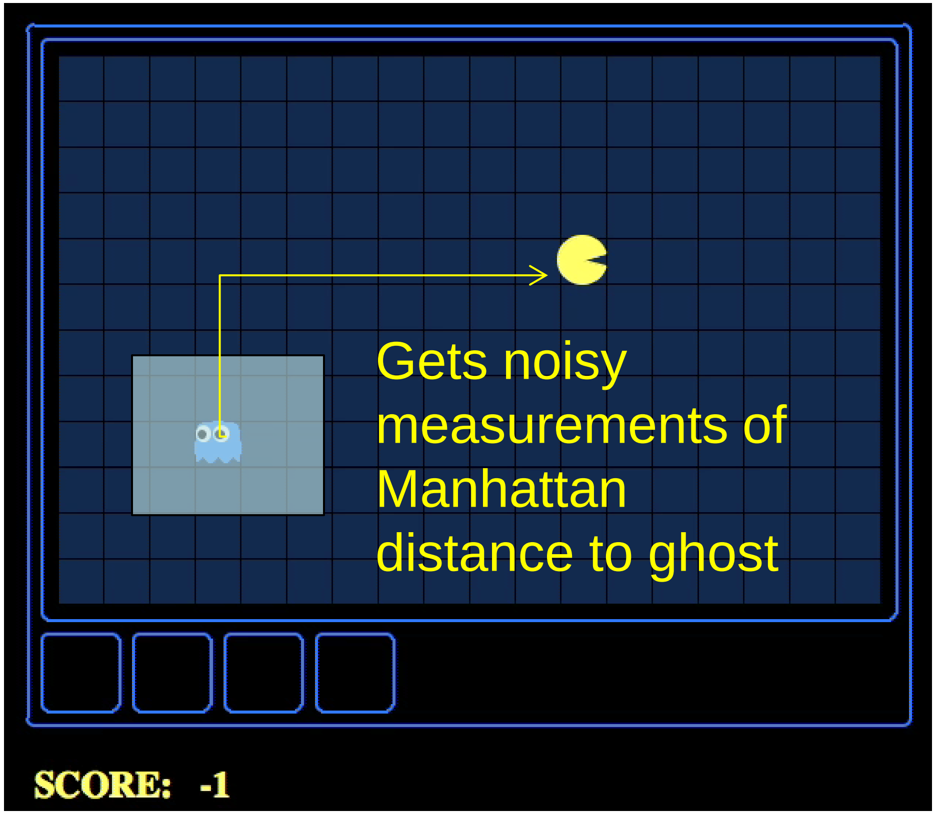
CSE 473

Lecture 23
(Chapters 15 & 18)

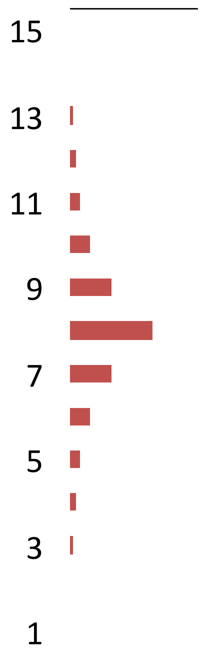
Hidden Markov Models (HMMs) and Particle Filters

Pac-Man goes Ghost Hunting

Pac-Man does not know true position of moving ghost

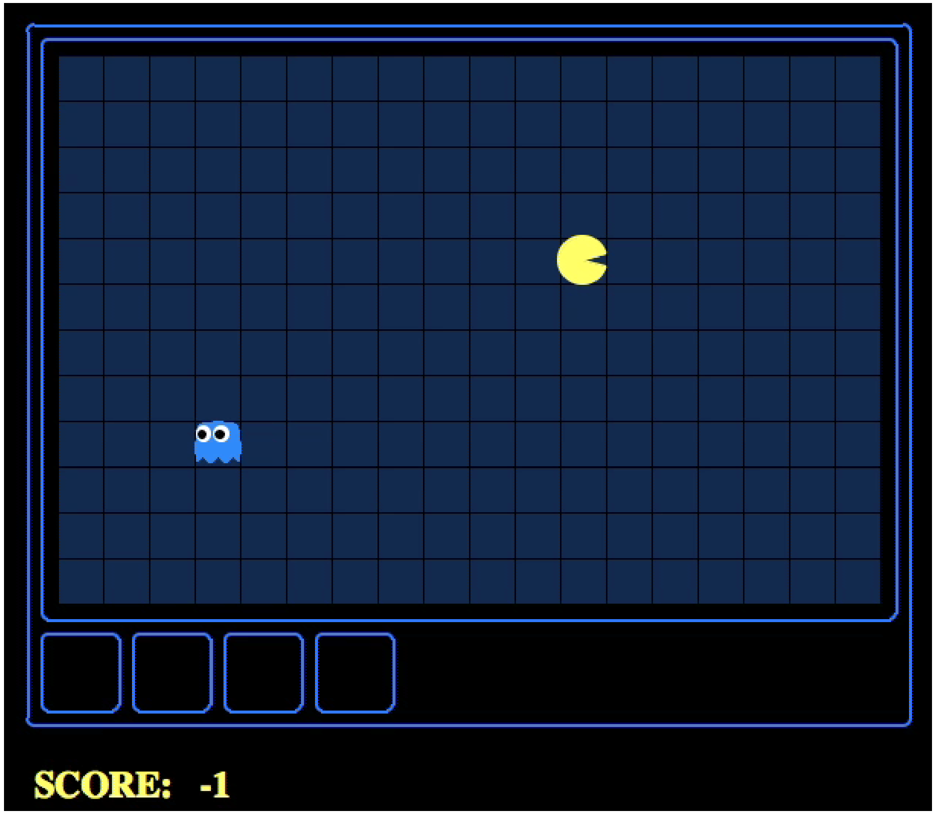


Noisy distance prob
(if true distance = 8)

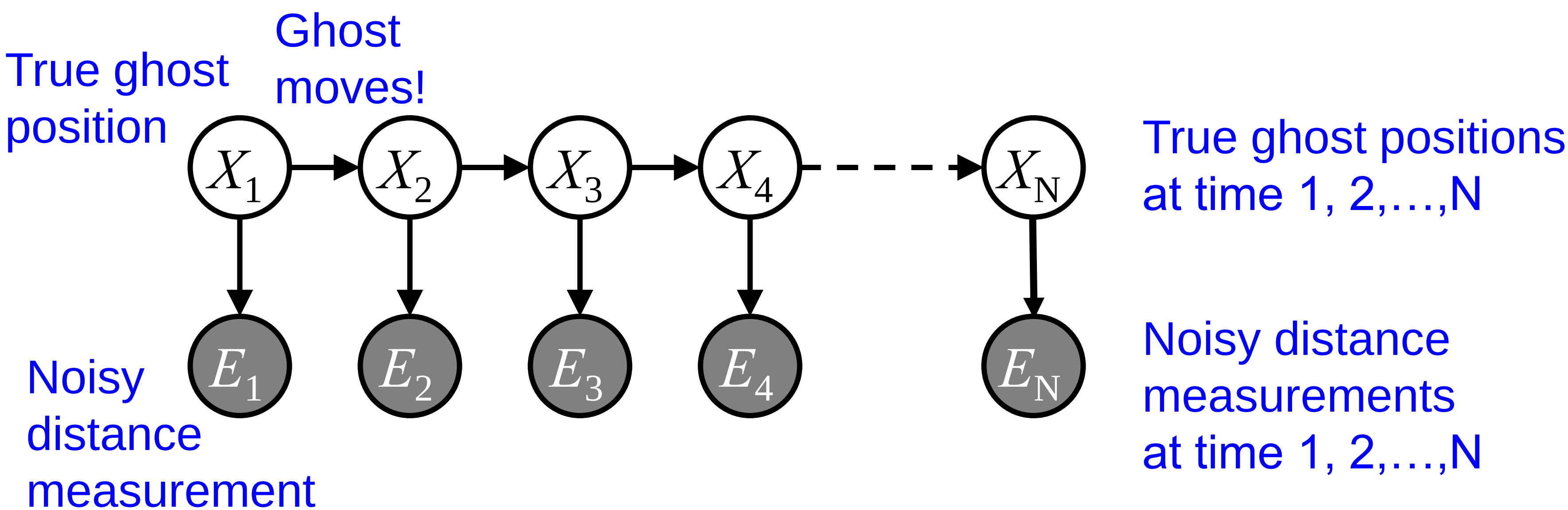


Must infer probability distribution over true ghost position

Example of Ghost Tracking (movie)



Bayesian Network for Tracking

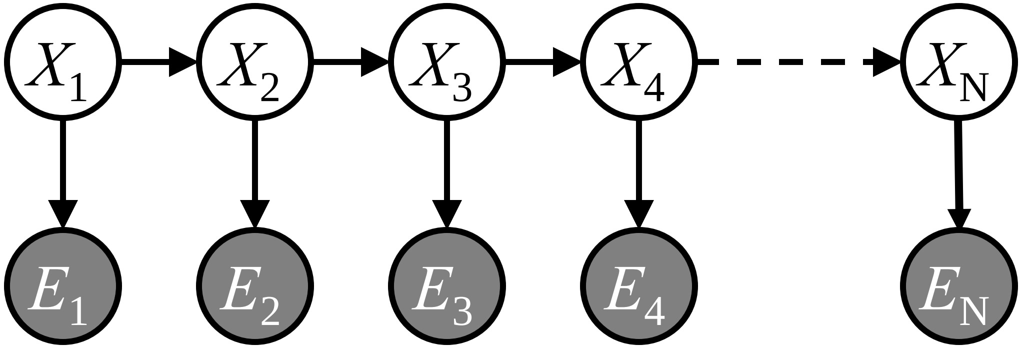


This "Dynamic" Bayesian network is also called a Hidden Markov Model (HMM)

- Dynamic = time-dependent
- Hidden = state (ghost position) is hidden
- Markov = current state only depends on previous state

Similar to MDP (Markov decision process) but no actions

Hidden Markov Model (HMM)



Hidden State at time t = 1, 2,…,N

Emissions (measurements) at time t = 1, 2,…,N

HMM is defined by 2 conditional probabilities:

$P(X_t \mid X_{t-1})$ Transition model $= P(X' \mid X)$

$P(E_t \mid X_t)$ Emission model $= P(E \mid X)$

(aka measurement/observation model)

plus initial state distribution $P(X_1)$

Project 4: Ghostbusters

- **Plot:** Pacman's grandfather, Grandpac, learned to hunt ghosts for sport
  - Was blinded by his power, but can hear the ghosts' banging and clanging sounds.
- **Transition Model:** Ghosts move randomly, but are sometimes biased
- **Emission Model:** Pacman gets a "noisy" distance to each ghost

Ghostbusters HMM

- $P(X_1)$ = uniform

$P(X_1)$

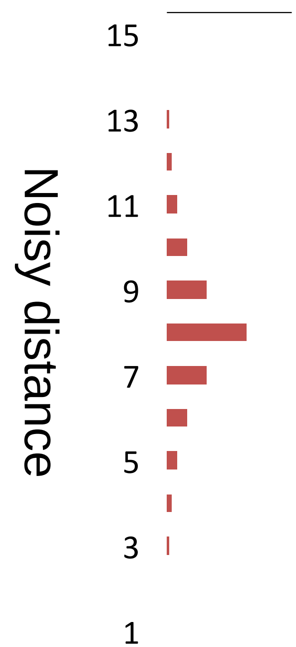
1/9	1/9	1/9
1/9	1/9	1/9
1/9	1/9	1/9

- $P(X'|X)$ = ghost usually moves *clockwise*, but sometimes moves in a random direction or stays in place

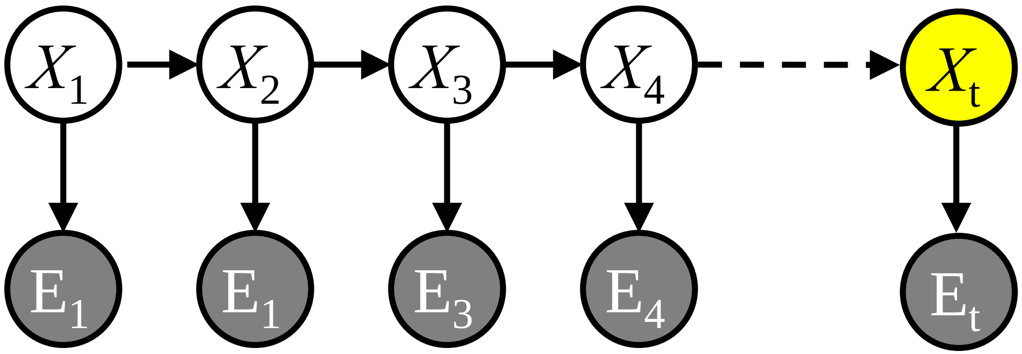
$P(X'|X=<1,2>)$

1/6	1/6	1/2
0	1/6	0
0	0	0

- $P(E|X)$ = compute Manhattan distance to ghost from Pac-Man and emit a *noisy distance* given this true distance (see example for true distance = 8)



HMM Inference Problem



Where is the ghost now?

Compute posterior probability over X_t

- Given evidence $\boldsymbol{E_1, \ldots, E_t = E_{1:t} = e_{1:t}}$
- Inference problem (aka *Filtering* or *Tracking*):

Find posterior $P(X_t|e_{1:t})$ for current t

The "Forward" Algorithm for Filtering

- Want to compute the "belief" $B_t(X) = P(X_t|e_{1:t})$
- Derive belief update rule from probability definitions, Bayes' rule and Markov assumption:

$$P(X_t \mid e_{1:t}) = \alpha\, P(e_t \mid X_t, e_{1:t-1})\, P(X_t \mid e_{1:t-1})$$

Bayes

$$= \alpha\, P(e_t \mid X_t) \sum_{X_{t-1}} P(X_t, X_{t-1} \mid e_{1:t-1})$$

Marginalize

$$= \alpha\, P(e_t \mid X_t) \sum_{X_{t-1}} P(X_t \mid X_{t-1}, e_{1:t-1}) P(X_{t-1} \mid e_{1:t-1})$$

Markov

$$= \alpha\, P(e_t \mid X_t) \sum_{X_{t-1}} P(X_t \mid X_{t-1}) P(X_{t-1} \mid e_{1:t-1})$$

Previous estimate

New estimate

"Forward" Algorithm: Summary

$$P(X_t \mid e_1,\ldots,e_t) = \alpha\, P(e_t \mid X_t) \sum_{X_{t-1}} P(X_t \mid X_{t-1}) P(X_{t-1} \mid e_1,\ldots,e_{t-1})$$

Normalization constant

Emission model

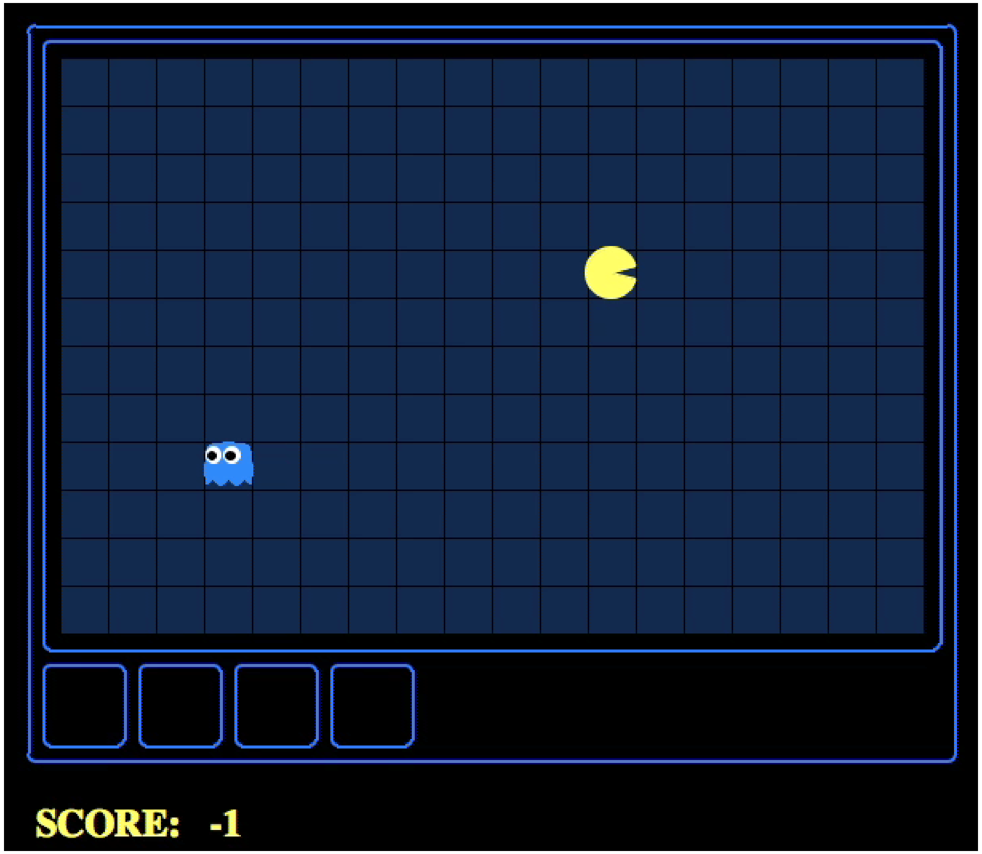
Transition model

Previous estimate

New estimate

At each time step *t*, compute and maintain a table of *P* values over all possible values of *X*

Filtering using the Forward Algorithm

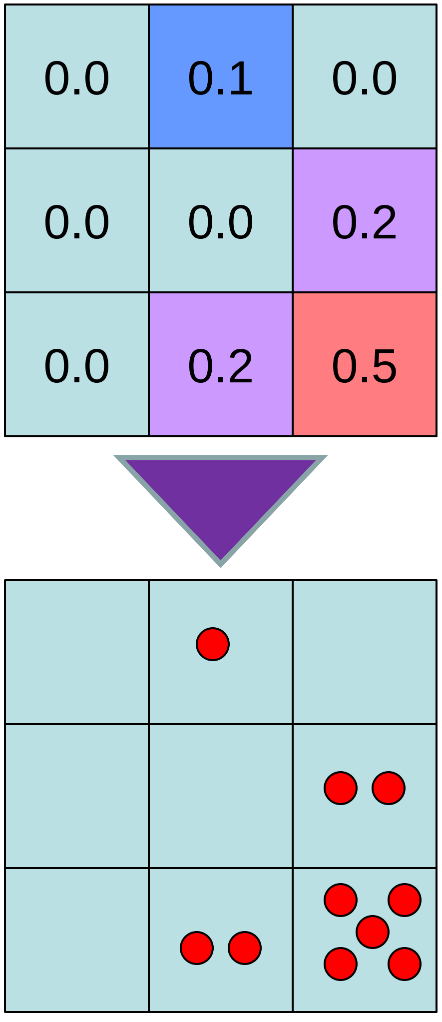


$P(X_t \mid e_1,\ldots,e_t)$ is an array of $12 \times 18 = 216$ values

(one for each location)

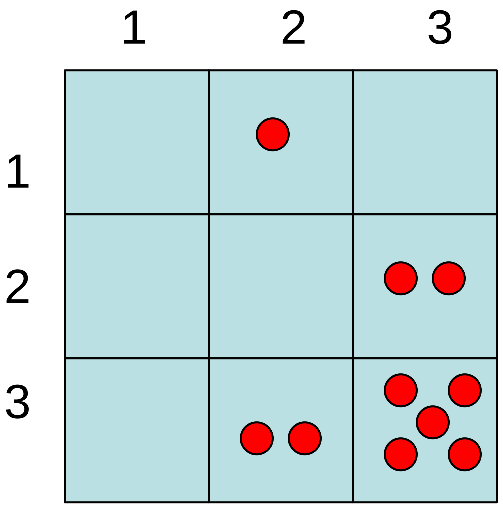
Particle Filtering

- Sometimes |X| is too big for exact inference
  - |X| may be too big to even store $P(X_t \mid e_{1:t})$
    E.g. when X is continuous
- Solution: Approximate inference
  - Track a set of *samples* of X
  - Samples are called ***particles***
  - Number of samples for X=x is proportional to probability of x



Representation: Particles

- Our representation of P(X) is now a list of N particles (samples)
  - ***Generally, N << |X|***
- P(x) approximated by number of particles with value x
  - Note: Many x will have P(x) = 0!
  - More particles, more accuracy



Particles:

(1,2)	(3,3)
(2,3)	(3,3)
(2,3)	(3,3)
(3,2)	(3,3)
(3,2)	(3,3)

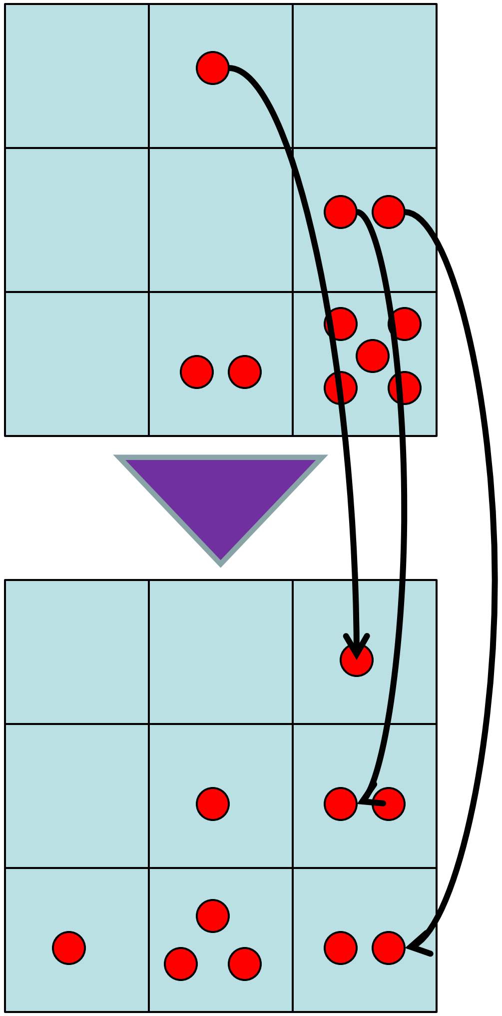
Particle Filtering
Step 1: Elapse Time

- Each particle x is moved by sampling its next position using the transition model

$$x' = \text{sample}(P(X'|x))$$

  - Samples' frequencies reflect the transition probabilities
  - In example, most samples move clockwise, but some move in another direction or stay in place

- This step captures passage of time



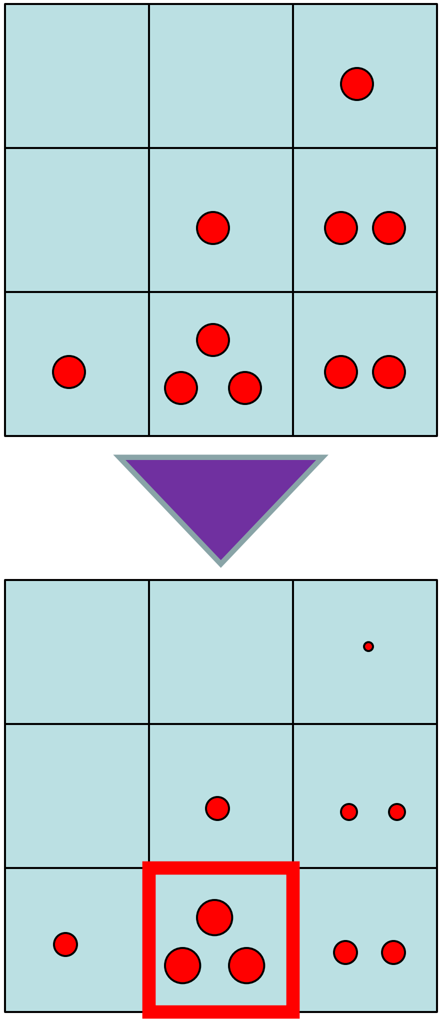
Particle Filtering
Step 2: Observe

Weight particles according to evidence

- ***Assign weights w to samples*** based on the new observed evidence e

$$w(x) = P(e|x)$$

- In example, true ghost position is shown in red outline; samples closer to ghost get higher weight (bigger size of circles) based on noisy distance emission model

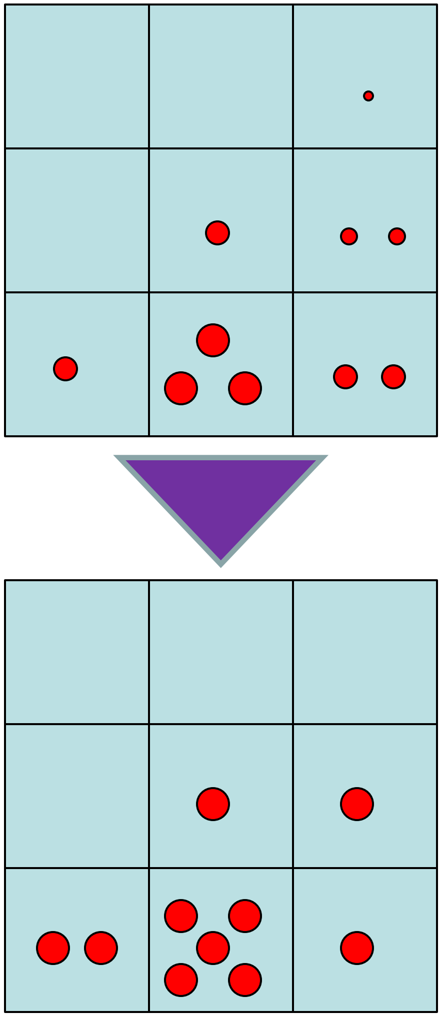


Particle Filtering
Step 3: Resample

- N times, we choose from our weighted sample distribution (i.e. randomly select with replacement)
  - Each sample selected with probability proportional to its weight
- Now the update is complete for this time step, continue with the next one

Old Particles:
(1,3) w=0.1
(3,2) w=0.9
(3,2) w=0.9
(3,1) w=0.4
(2,3) w=0.3
(2,2) w=0.4
(3,3) w=0.4
(3,3) w=0.4
(3,2) w=0.9
(2,3) w=0.3

New Particles:
(3,2) w=1
(3,2) w=1
(3,2) w=1
(2,3) w=1
(2,2) w=1
(3,2) w=1
(3,1) w=1
(3,3) w=1
(3,2) w=1
(3,1) w=1



Next Time

- More on Particle Filtering
- Supervised Learning
- Learning Decision Trees from data
- To Do:
  - Project 4
  - Read Chapter 18