Object Tracking with Particle Filtering

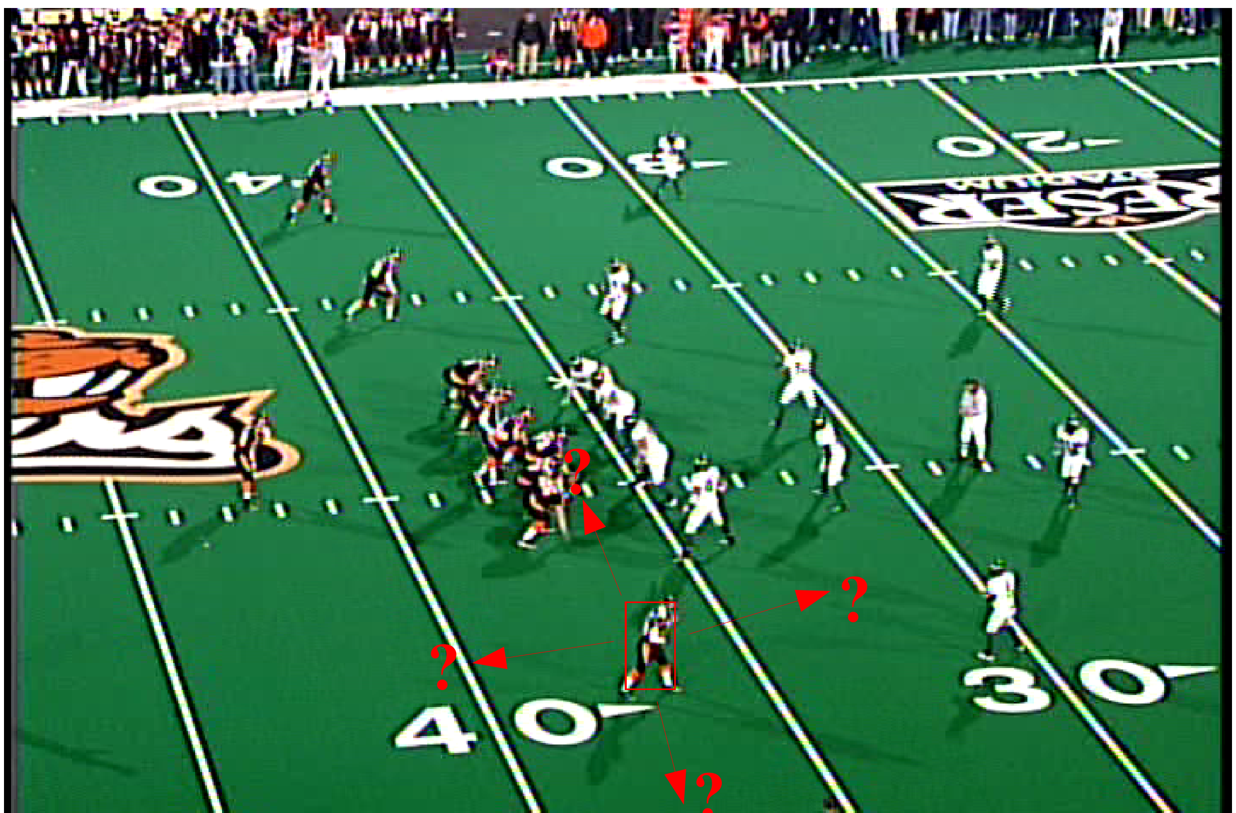
CS 556 Term Project

March 23, 2006

Rob Hess
School of EECS
Oregon State University

The Problem

- Track a specified object through a sequence of video



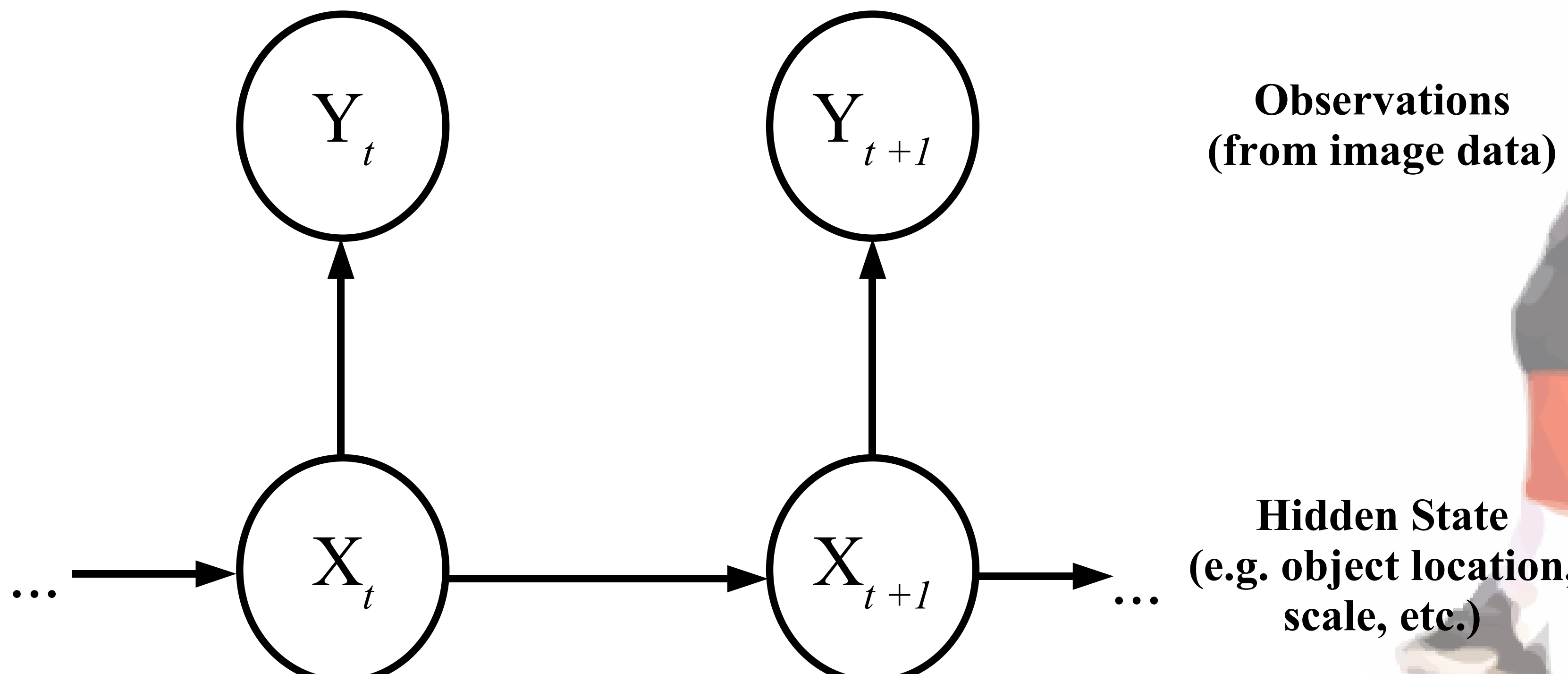
Difficulties

- Moving Camera
- Erratic object motion
- Cluttered Background
- Other Moving Objects

- Can't rely only on image processing techniques

The Solution

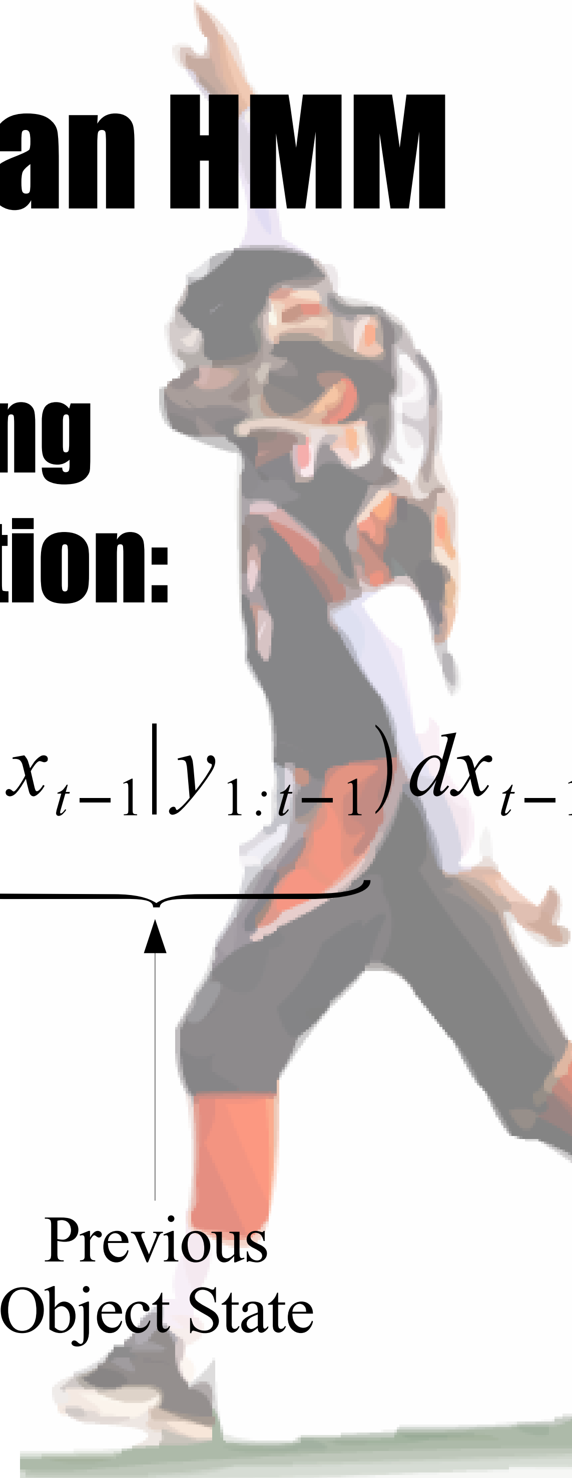
- **Use a probabilistic framework**
  - **Formulate tracking as inference in an HMM**



Bayesian Inference in an HMM

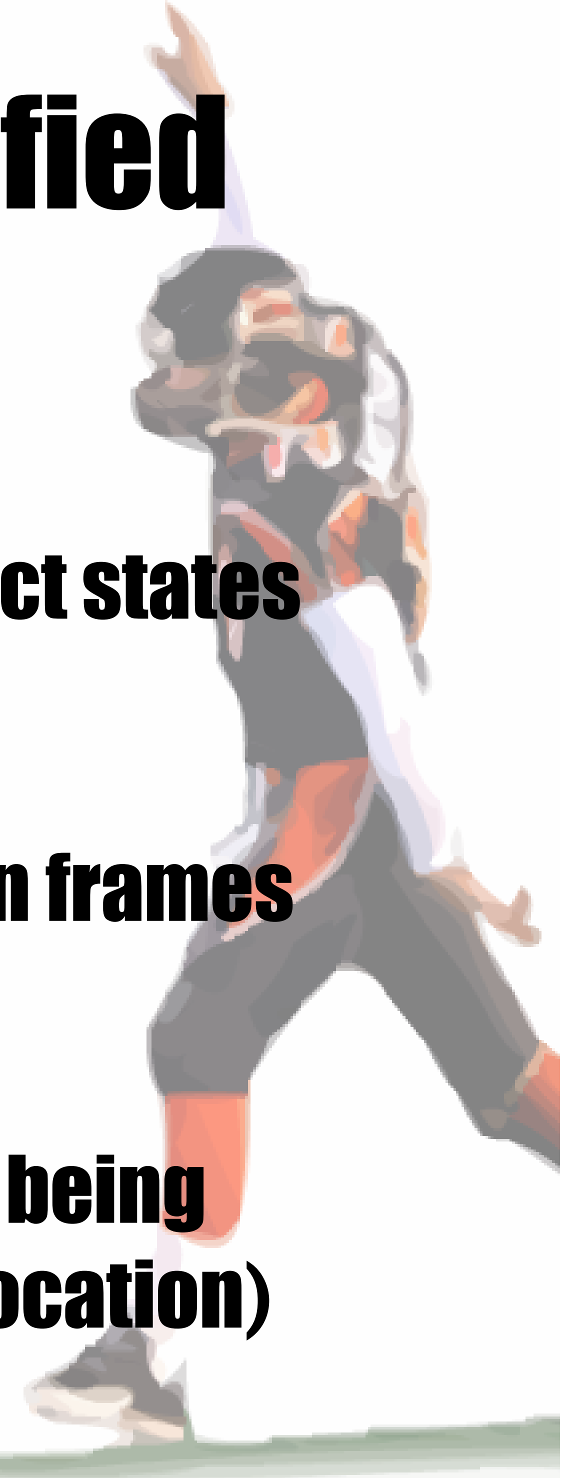
- **Perform forward inference using the Bayesian filtering distribution:**

$$\underbrace{p(x_t|y_{1:t})}_{\text{Current Object State}} = \alpha \underbrace{p(y_t|x_t)}_{\text{Observation Model}} \int_{x_{t-1}} \underbrace{p(x_t|x_{t-1})}_{\text{Transition Model}} \underbrace{p(x_{t-1}|y_{1:t-1})}_{\text{Previous Object State}} dx_{t-1}$$



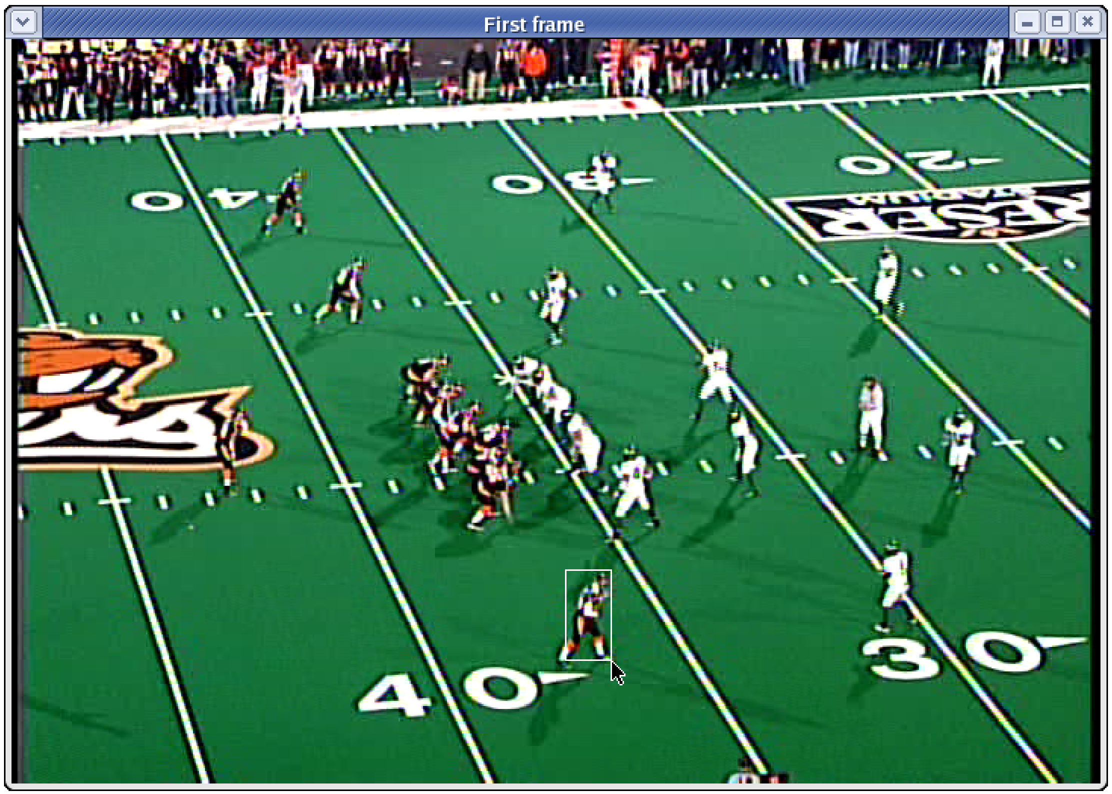
What must be specified

- **Prior Distribution:** $p(x_0)$
  - Describes initial distribution of object states
- **Transition Model:** $p(x_t | x_{t-1})$
  - Specifies how objects move between frames
- **Observation Model:** $p(y_t | x_t)$
  - Specifies the likelihood of an object being in a specific state (i.e. at a specific location)



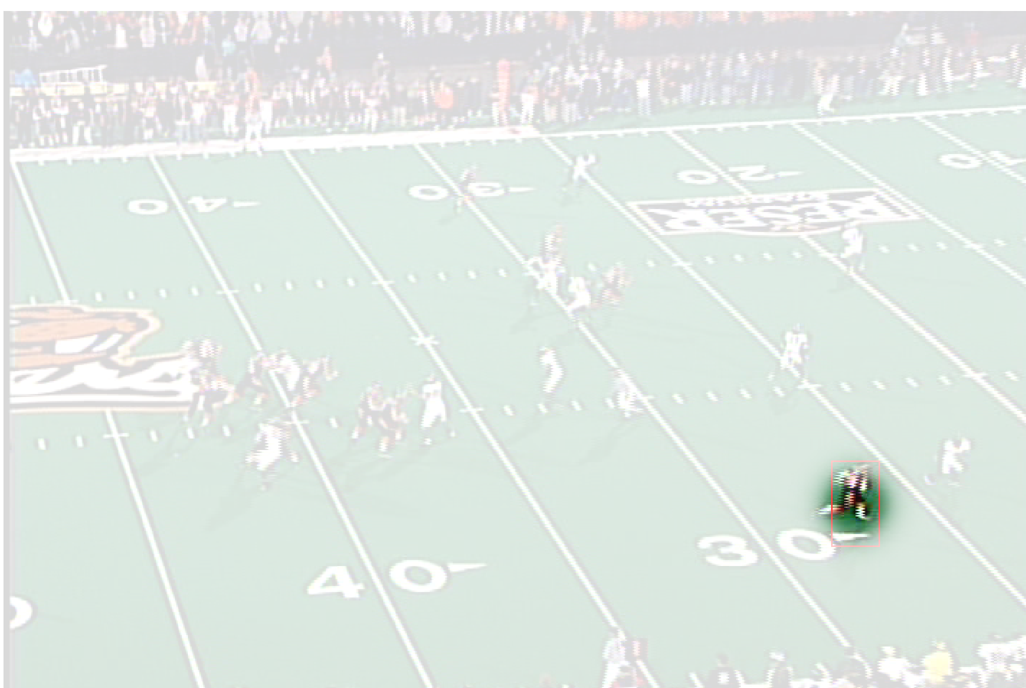
The Prior Distribution $p(x_0)$

- Could be based on an object detector
- Mine is an impulse based on user input



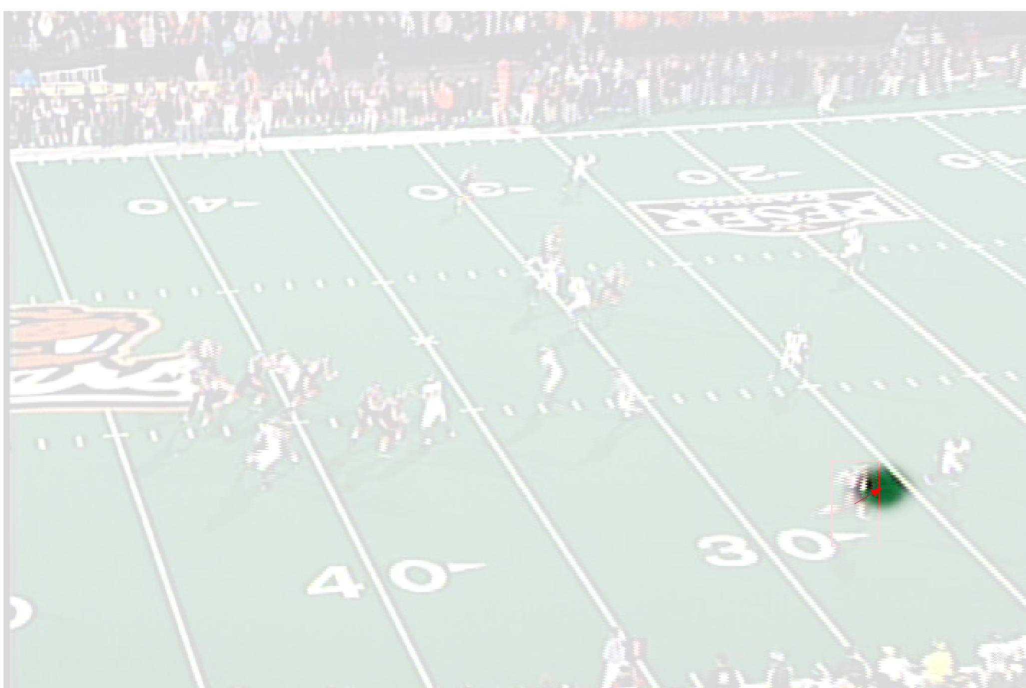
The Transition Model $p\left(x_t \middle| x_{t-1}\right)$

- A simple model: sample next state from a Gaussian window around current state



The Transition Model $p(x_t | x_{t-1})$

- A better model: take into account previous states for velocity and acceleration information

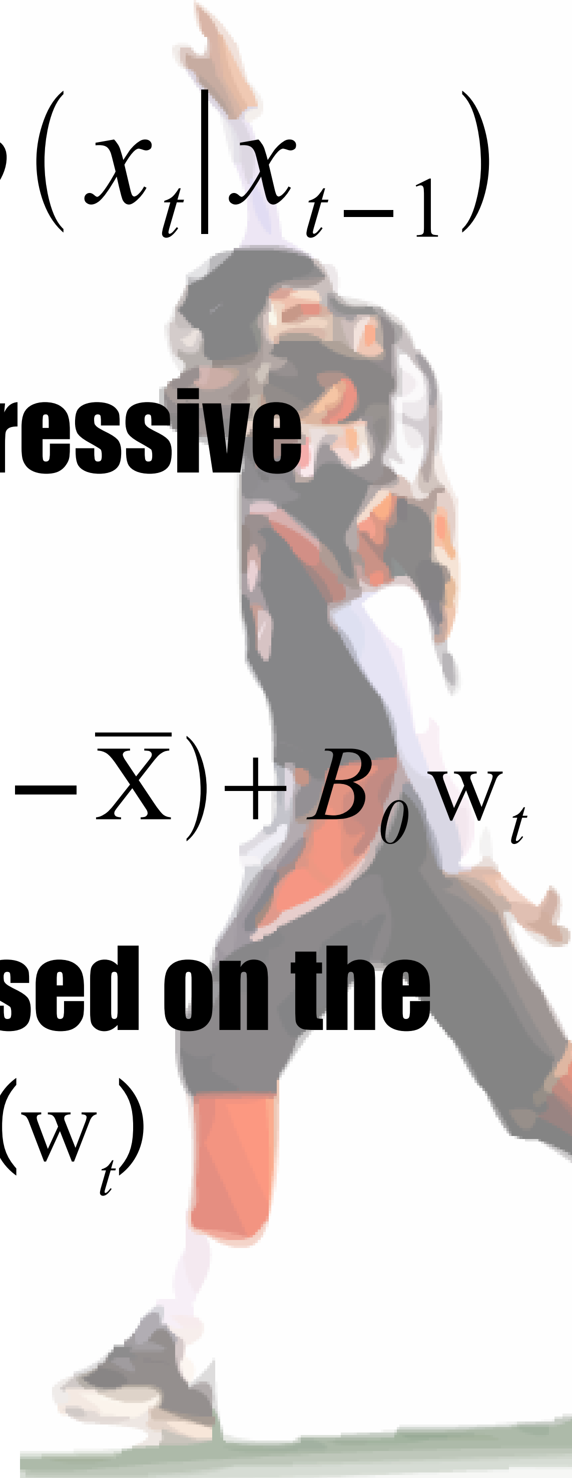


The Transition Model $p(x_t | x_{t-1})$

- I use a second-order, auto-regressive dynamical model:

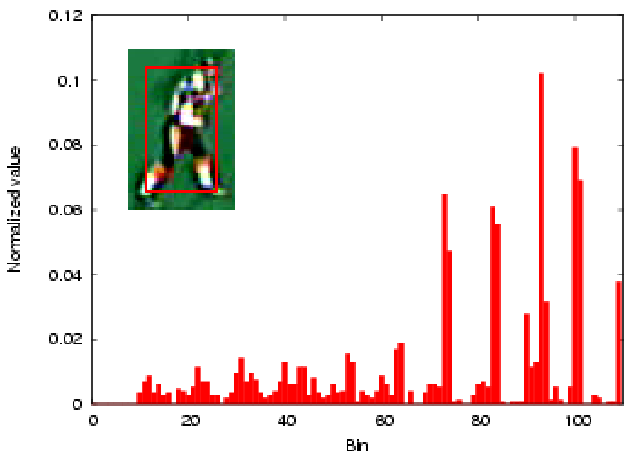
$$\mathrm{X}_t - \overline{\mathrm{X}} = A_2(\mathrm{X}_{t-2} - \overline{\mathrm{X}}) + A_1(\mathrm{X}_{t-1} - \overline{\mathrm{X}}) + B_0 \mathrm{w}_t$$

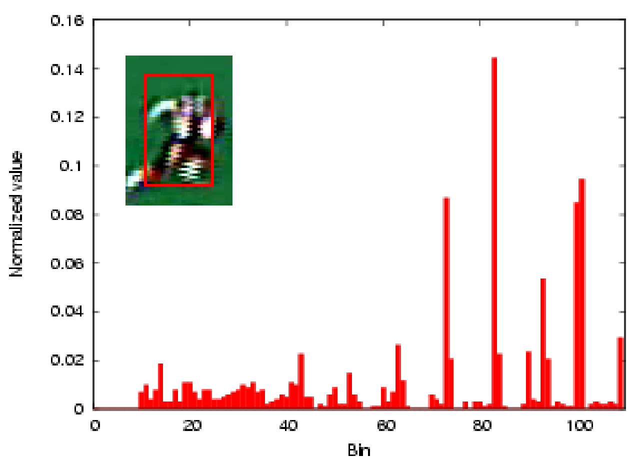
- This predicts the next state based on the previous two plus some noise (w_t)

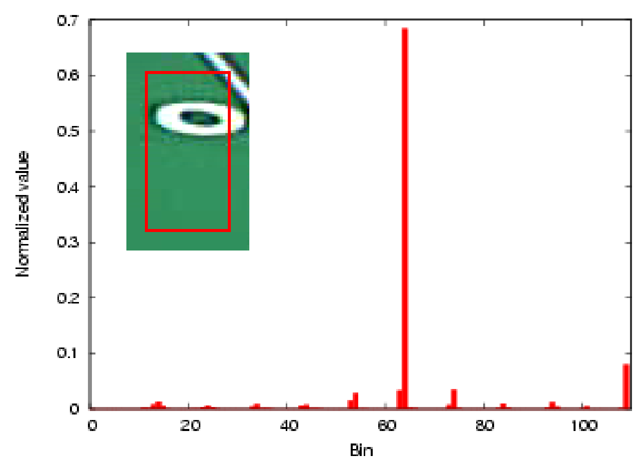


The Observation Model $p(y_t|x_t)$

- I use a simple HSV histogram-based model







The Observation Model $p(y_t|x_t)$

- **Likelihood is based on a distance metric D between histograms h_0 and $h(x_t)$:**

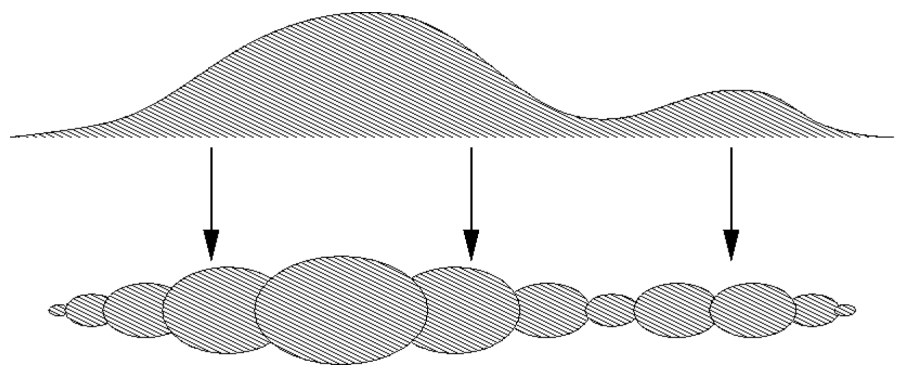
$$p(y_t|x_t) \propto e^{-\lambda D^2[h_0, h(x_t)]}$$

Particle Approximation

- Non-linear and non-Gaussian models yield an intractable filtering distribution
- Can approximate exact inference using Particle Filtering
  - Particle filtering = sequential Monte Carlo sampling

Particle Approximation

- **The idea: a set of weighted particles approximates the filtering distribution**

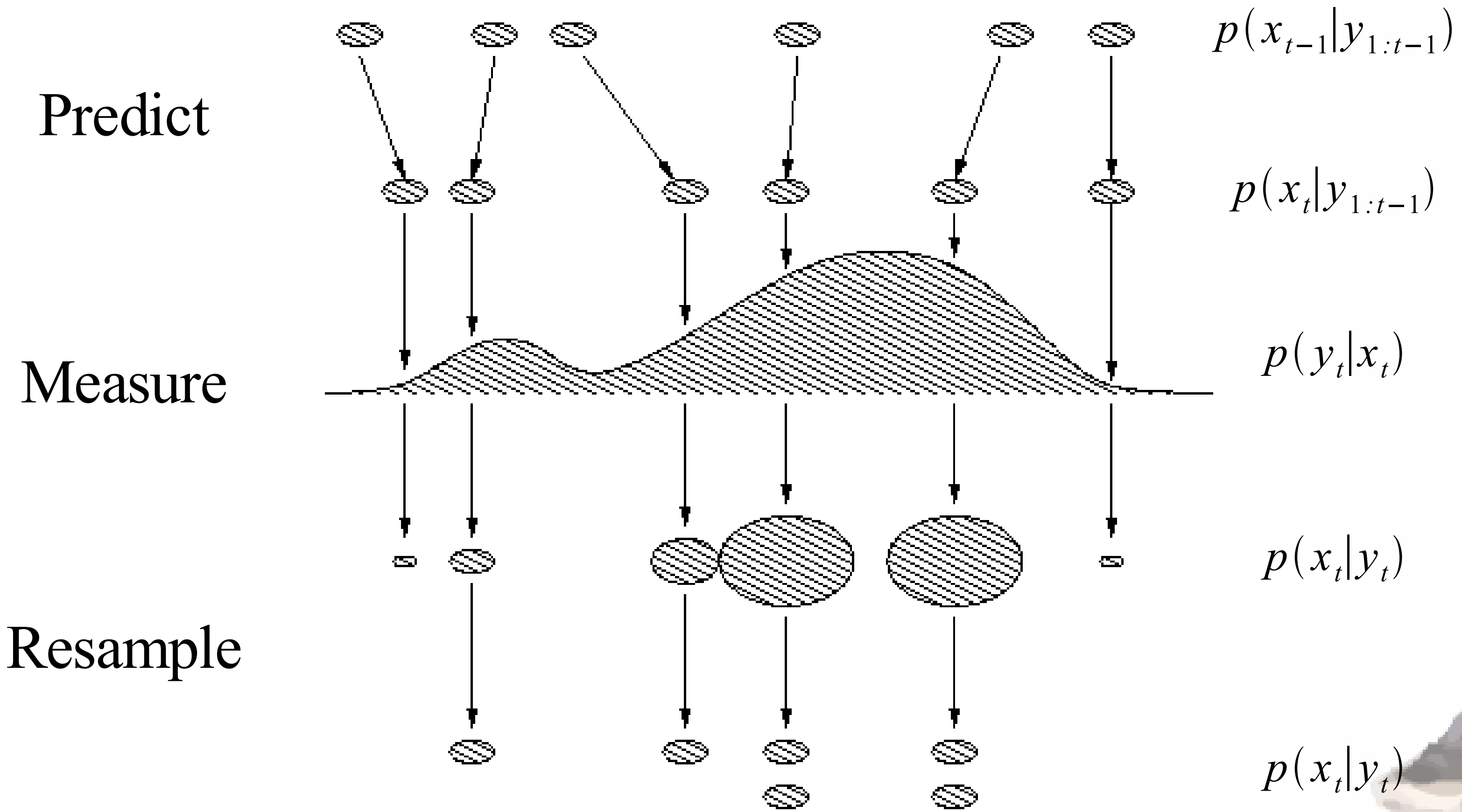


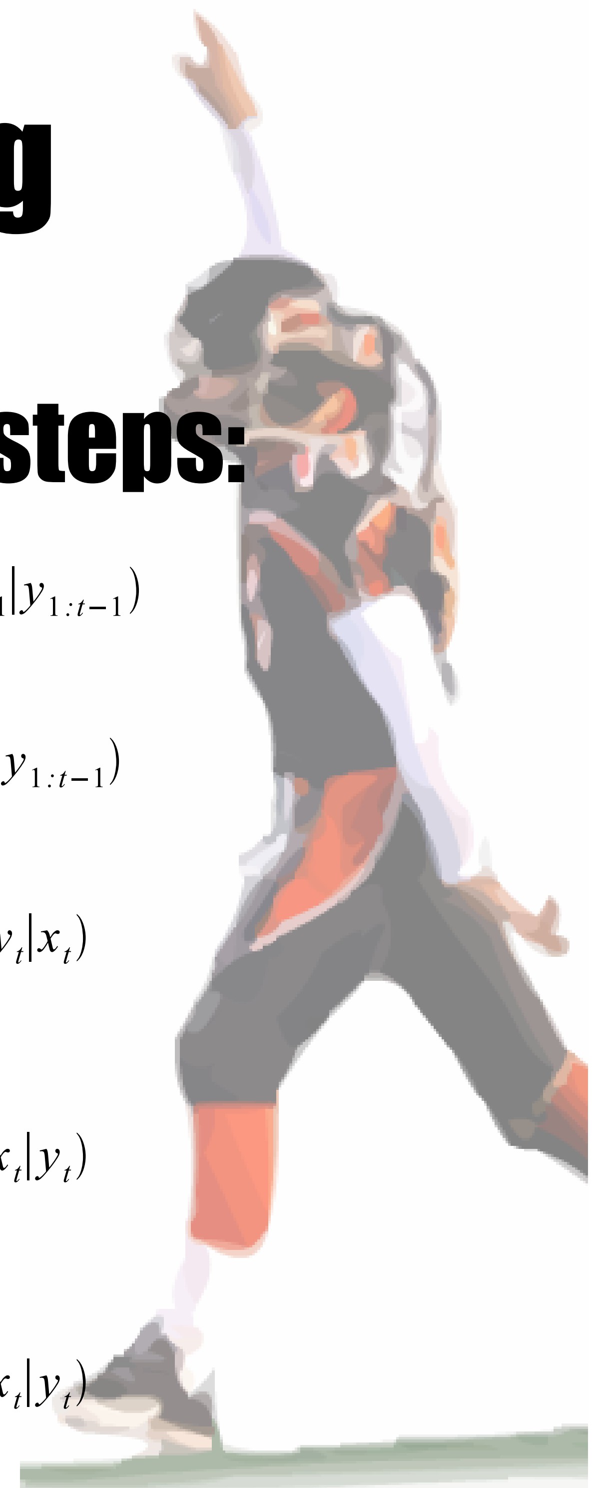
- **The filtering distribution becomes**

$$p(x_t | y_{1:t}) \approx \alpha\, p(y_t | x_t) \sum_i w_{t-1}^{(i)}\, p(x_t | x_{t-1}^{(i)})$$

Particle Filtering

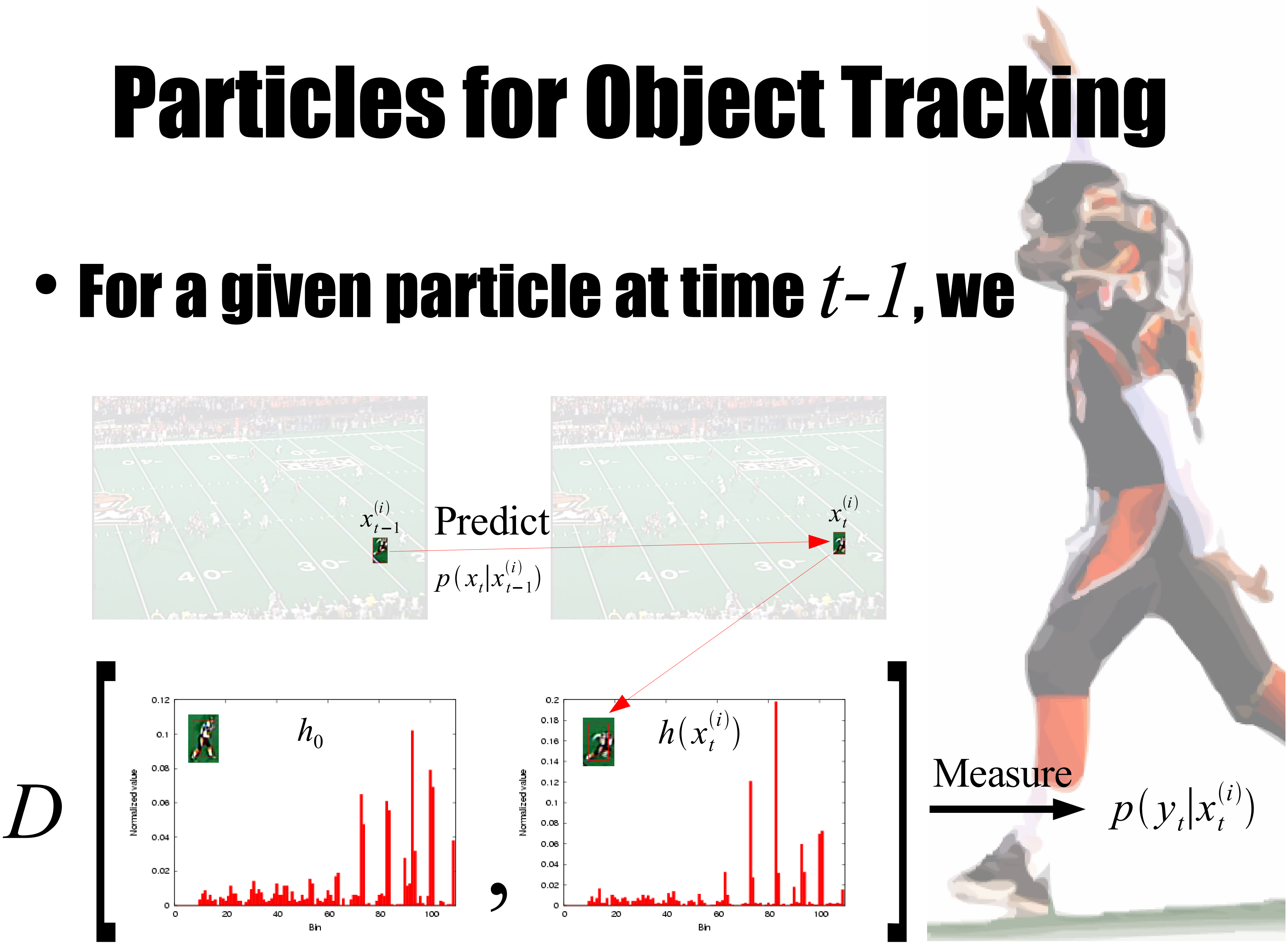
- **Particle filtering consists of 3 steps:**





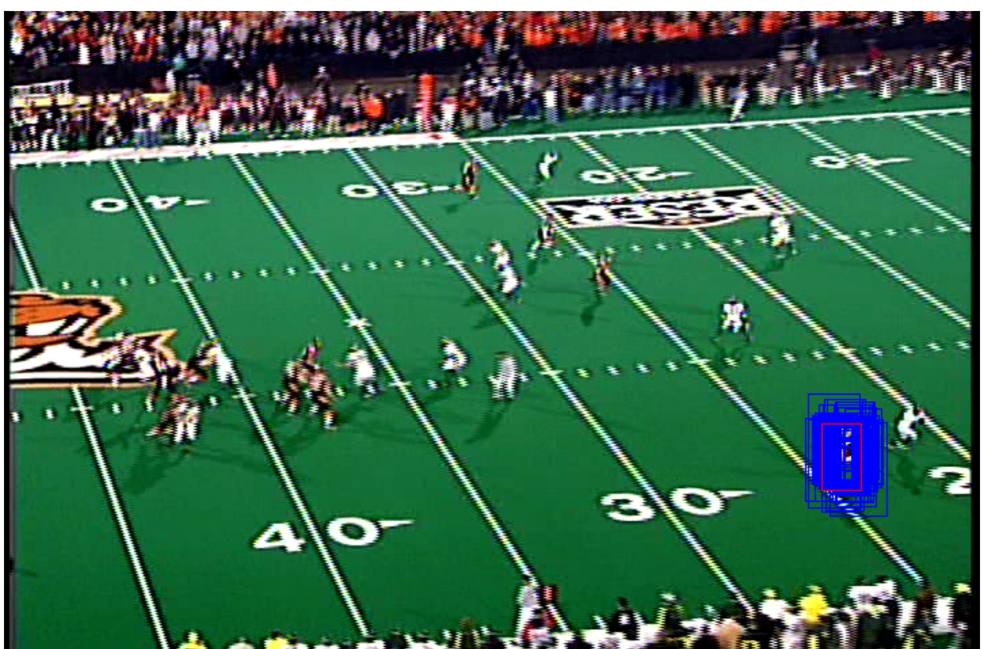
Particles for Object Tracking

- For a given particle at time t-1, we



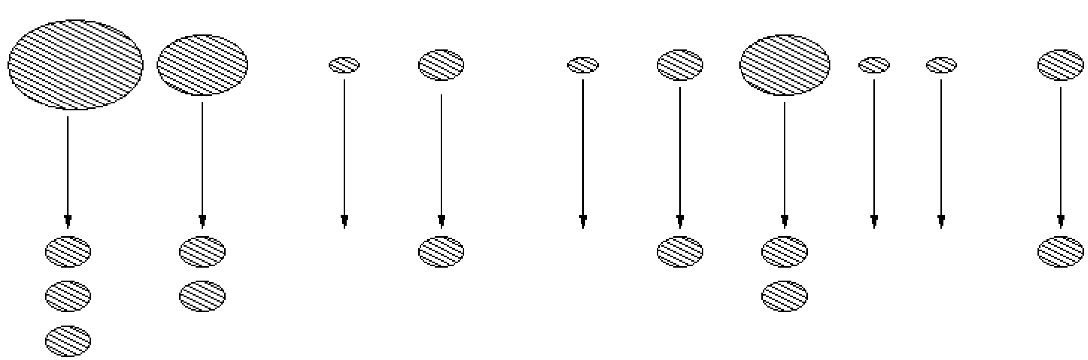
Particles for Object Tracking

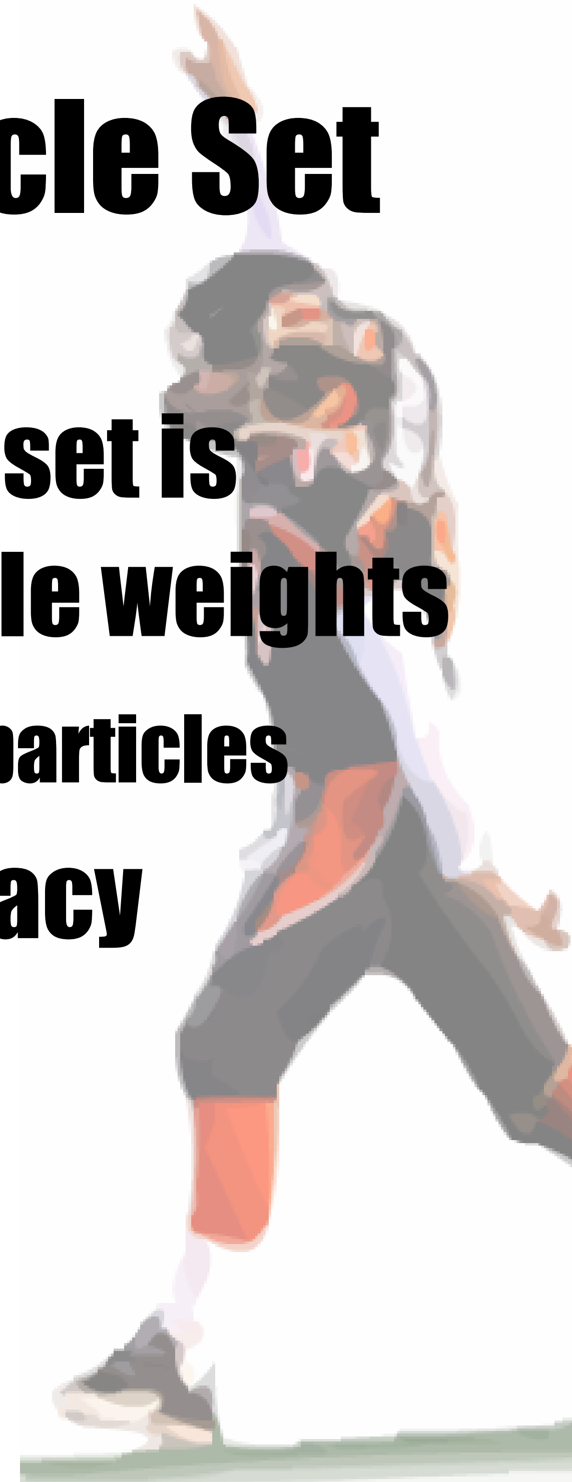
- Each particle is weighted by its likelihood
- Most likely particle represents the object state at time t



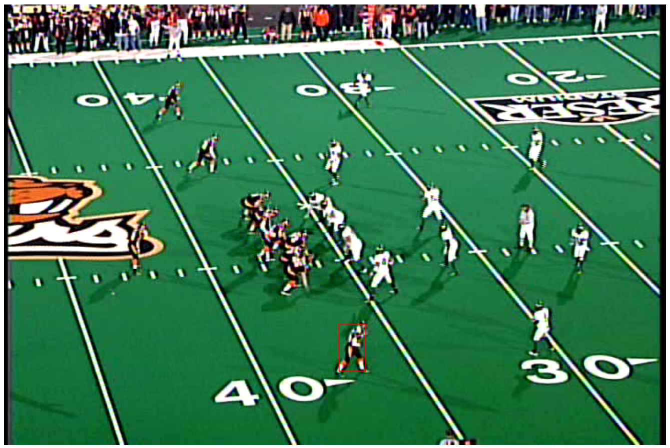
Resampling the Particle Set

- Before time step $t+1$, particle set is resampled according to particle weights
  - Produces a new set of unweighted particles
- Resampling prevents degeneracy of weights

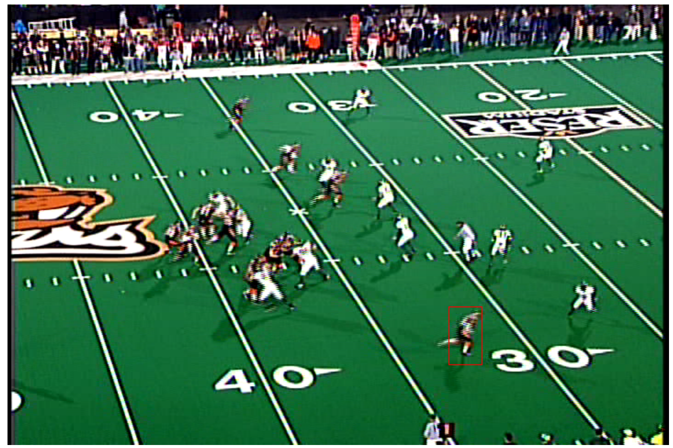




Good Results



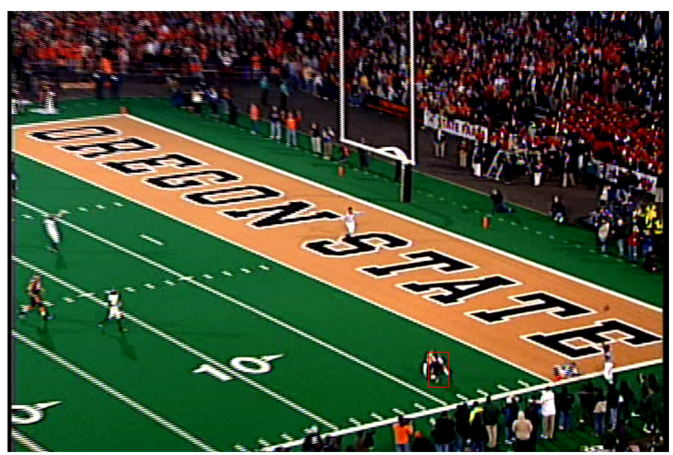
Frame 1



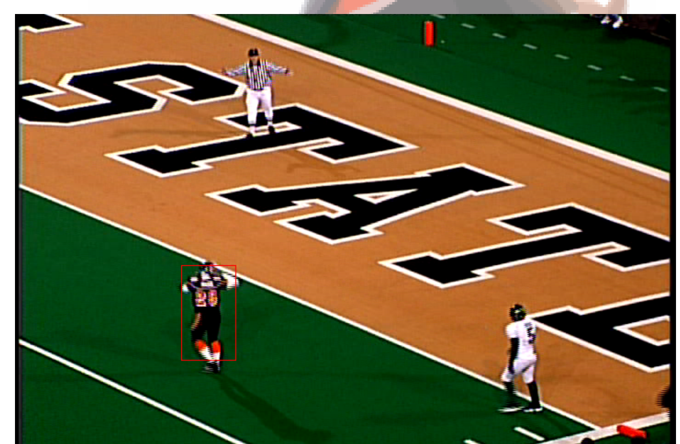
Frame 240

Frame 328

Frame 319



Frame 368



Frame 458

More Good Results

Frame 1

Frame 74

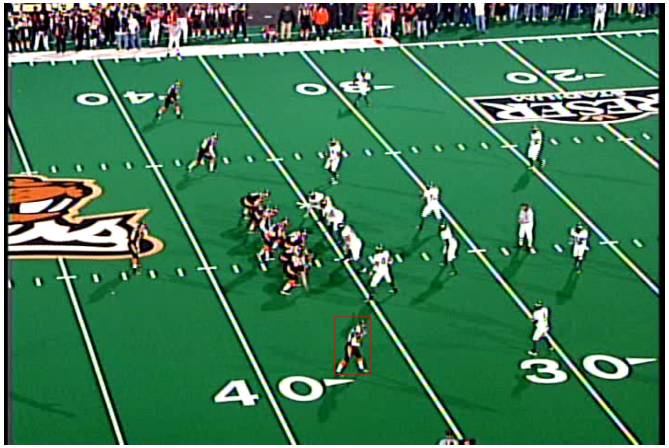
Frame 163

Frame 323

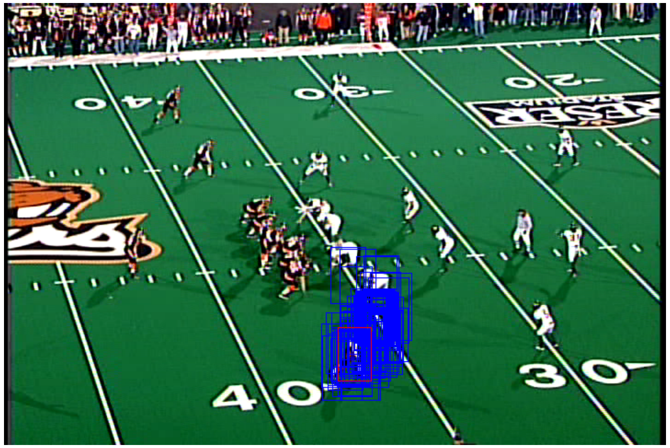
Frame 397

Frame 476

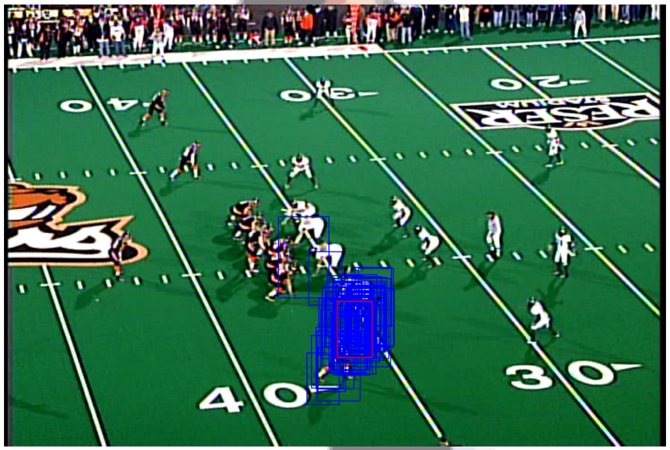
Bad results



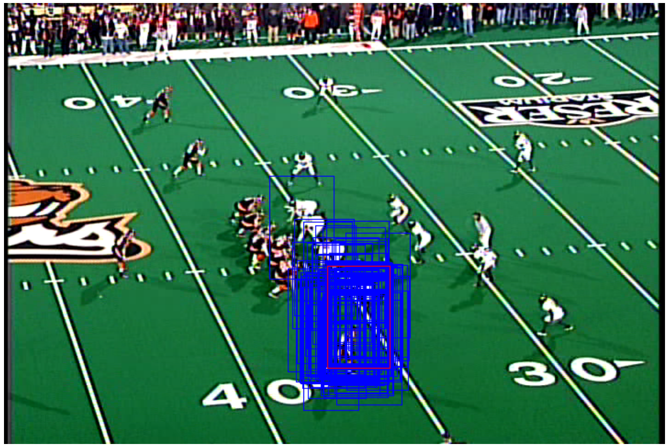
Frame 1



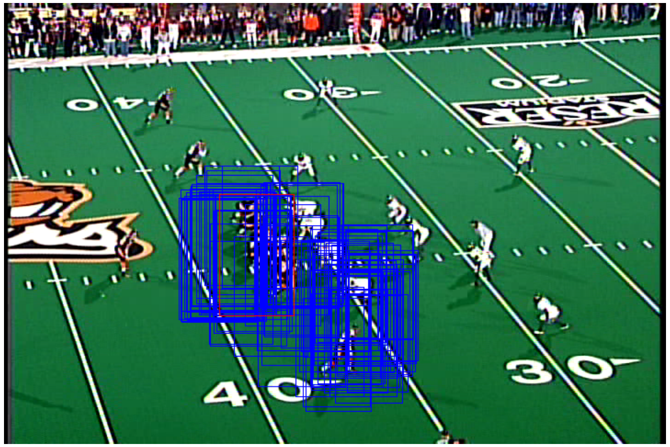
Frame 33



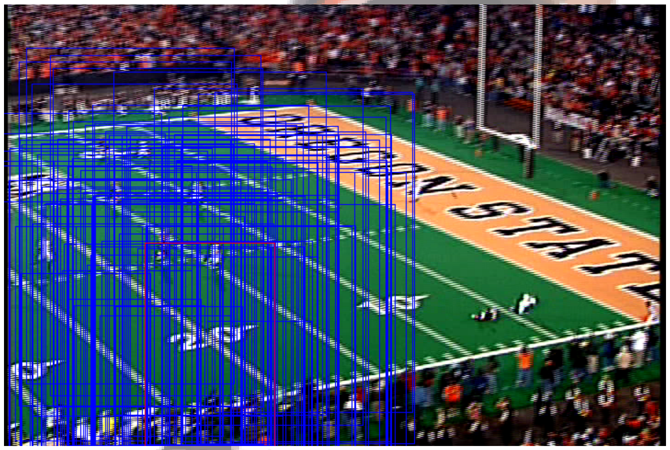
Frame 71



Frame 158



Frame 204



Frame 339