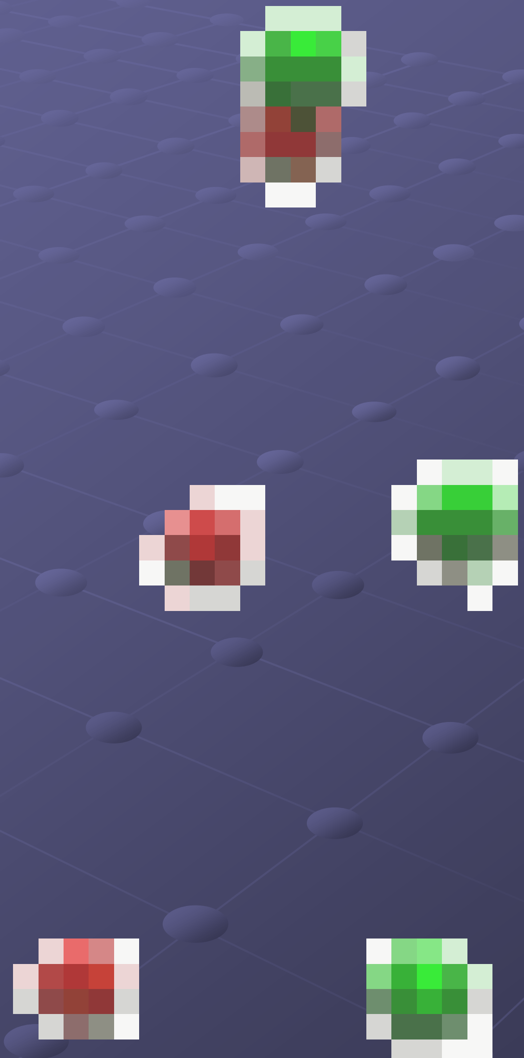
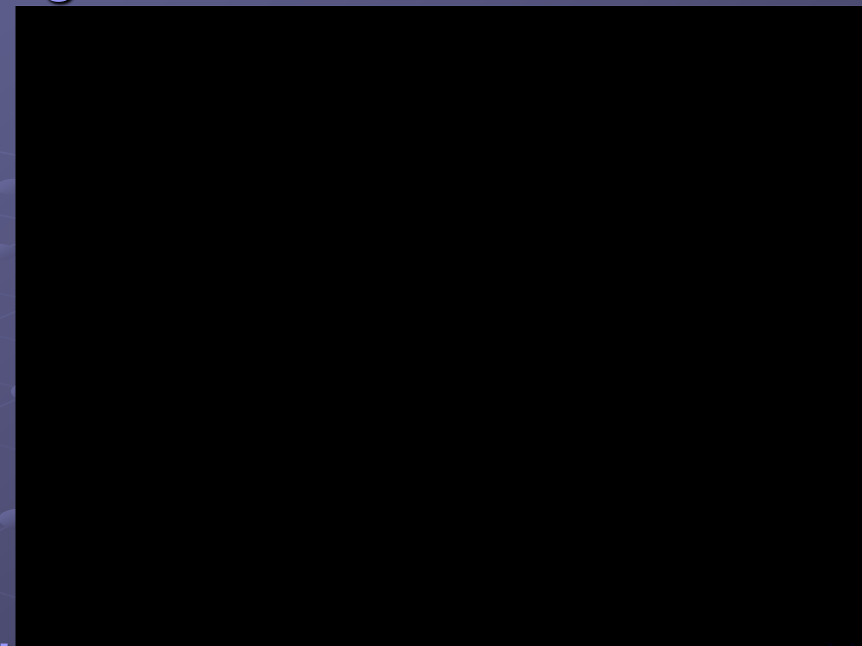
Tracking



TRACKING => Particle filtering => Overview

=> Stochastically predict joint angles for the next frame <=



KALMAN FILTER

- predict single position
- uni-modal & Gaussian

PARTICLE FILTER

- predict multiple positions
- multi-modal & non-Gaussian
- PROBLEM specification: 3 probability distributions
  1. Prior density $p(x_{t-1})$ for the state x ➔ joint angles in previous frame
  2. Process density $p(x_t|x_{t-1})$ ➔ kinematic and body models
  3. Observation density $p(z_{t-1}|x_{t-1})$ ➔ image in previous frame
- SOLUTION specification: 1 probability distribution
  1. State Density ≡ $(x_t|Z_t)$ ➔ joint angles in next frame

TRACKING => Particle filtering => Sample

SAMPLE $ś_t$ from the ***prior density*** $p(x_{t-1}|z_{t-1})$

x_{t-1}=joint angles in previous frame, z_{t-1}

SAMPLE SET ⇨ possible alternate values for parameters
When tracking through background clutter or occlusion, a joint angle may have *N* possible alternate values (samples) *s* with respective weights *w*

PRIOR DENSITY $p(x) \approx S_{t-1} = \{(s^{(n)},w^{(n)}), n=1..N\}$

Select samples for the next frame $ś_t = s_{t-1}$

by finding the smallest n for which $c^{(n)} \geq r$
Where $c^{(n)} = \sum w^{(n)}$ *(1..t), r is a random number [0,1]*

TRACKING => Particle filtering => Predict

PREDICT s_t from the *process density* $p(x_t|x_{t-1}=\acute{s}_t)$

Predict joint angles for the next frame considering *kinematic model, body model & error minimisation:*

Minimise edge errors E_e

$$E_e(S_t) = \frac{1}{2n_e v_e} \sum_{x,y} (|\nabla i_t(x,y)| - m_t(x,y,S_t))^2 + 0.5(S - S_t)^T C_t^{-1} (S - S_t) \rightarrow \min S_t$$

Minimise region errors E_r

$$E_r(S_t) = \frac{1}{2n_r v_r} \sum_{j=1}^{n_r} (i_t[p_j(S_t)] - i_{t-1}[p_j(S_{t-1})])^2 + E_e(S_t) \rightarrow \min S_t$$

Where i_t represents the image and m_t the model gradients at time t, n_e is the number of edge values summed, v_e is the edge variance, n_r is the number of region values summed, v_r is the region variance, p_j is the image pixel coordinate of the j^{th} surface point on a body part.

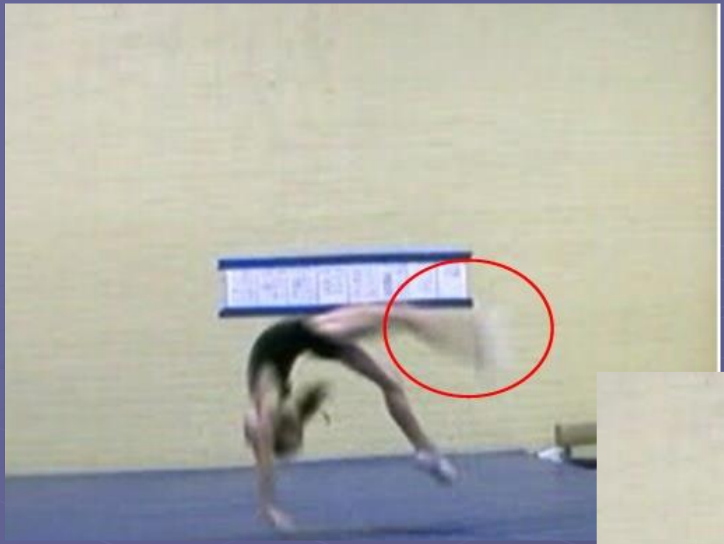
TRACKING => Particle filtering => Measure

MEASURE and weigh the new position in terms of the ***observation density***, $p(z_t|x_t)$

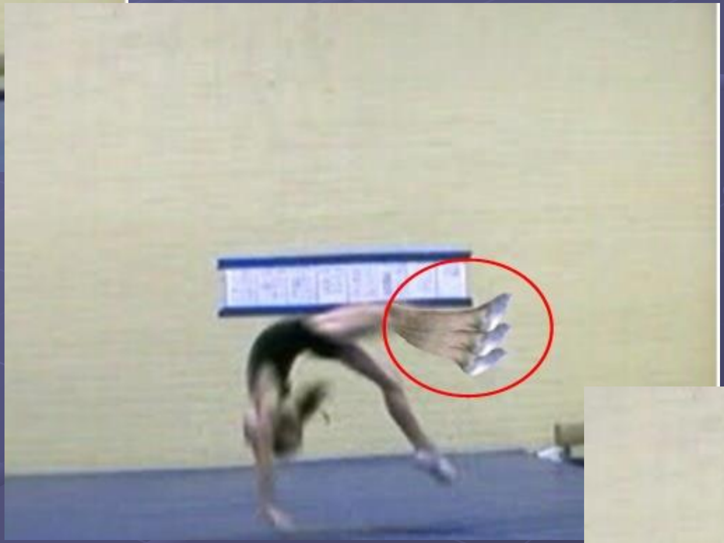
- Estimate weights $w_t = p(z_t|x_t = s_t)$
- Normalise weights $\sum_n w^{(n)} = 1$
- Smooth weights w_t over 1..t, for n trajectories
  - Replace each sample set with its *n* trajectories $\{(s_t,w_t)\}$ for 1..t
  - Re-weight all $w^{(n)}$ over 1..t
  - Trajectories tend to merge less than 10 frames ago
    => O(Nt) storage prunes down to O(N)

TRACKING => Particle filtering

tracking through motion blur of right calf and foot segments in a flic-flac (back-handspring)



alternative particles, $s_{1..n}$ (knee angles)



expected value of the distribution $\sum_n w_t s_t$

TRACKING => popular solutions

- **Kalman Filter**: For linear transitions and Gaussian distributions, exact solutions are obtained using this filter.
- **Particle Filter**: Also known as condensation algorithm, predicts multiple states/positions with non-Gaussian distributions.
- **Unscented Kalman Filter**: Improves the Kalman Filter approximation for non-linear systems, but still assumes Gaussian distributions.
- (Above, all are implemented in Python using filter.py by Roger Labbe https://github.com/rlabbe/Kalman-and-Bayesian-Filters-in-Python)