Tracking Multiple Targets Using a Particle Filter Representation of the Joint Multitarget Probability Density

Chris Kreucher, Keith Kastella, Alfred Hero

This work was supported by United States Air Force contract #F33615-02-C-119, Air Force Research Laboratory contract #SPO900-96-D-0080 and by ARO-DARPA MURI Grant #DAAD19-02-1-0262.

Overview of Talk

- We present a method of tracking multiple targets based on recursive estimation of their Joint Multitarget Probability Density (JMPD).
- We give a grid-free implementation of the JMPD based on particle filtering techniques
  - We detail adaptive sampling schemes that exploit the multitarget nature of the problem.
  - We show the computational tractability PF provides
- We detail the inherent permutation symmetry associated with JMPD (related to measurement to track association) and show how this symmetry manifests itself in the particle filter implementation as partition swapping.

The Joint Multitarget Probability Density (JMPD)

- The state of an individual target is modeled by $\mathbf{x}$, e.g. $\mathbf{x} = [x\ \dot{x}\ y\ \dot{y}]'$
- We are interested in tracking multiple targets, so the state vector of the system (where perhaps the number of targets *T* is unknown) is defined as

$$\mathbf{X} = [\mathbf{x}_1 \quad \mathbf{x}_2 \quad \ldots \quad \mathbf{x}_{T-1} \quad \mathbf{x}_T]'$$

- The central element that summarizes our knowledge of the system at time *k* is the *joint multitarget probability density (JMPD),*

$$p\left(\mathbf{X}^k \mid \mathbf{Z}^k\right)$$

which is to be estimated based on a sequence of noisy measurements taken over *k* time steps,

$$\mathbf{Z}^k = \{\mathbf{z}^1 \cup \mathbf{z}^2 \ldots \cup \mathbf{z}^k\}$$

The Joint Multitarget Probability Density (JMPD)

- As examples, the sample space of $p(\mathbf{X}^k|\mathbf{Z}^k)$ contains

$p(\{\} \mid \mathbf{Z}^k)$ The posterior probability density for no targets in the surveillance region

$p(\mathbf{x}_1, \mathbf{x}_2 \mid \mathbf{Z}^k)$ The posterior probability density for two targets in states $\mathbf{x}_1$ and $\mathbf{x}_2$
Notice the permutation symmetry inherent in JMPD

- The target motion is modeled as Markov using a Kinematic prior

$$p(\mathbf{X}^k \mid \mathbf{X}^{k-1})$$

- The sensor output is modeled using

$$p(\mathbf{z}^k \mid \mathbf{X}^k)$$

- We allow for the target motion to be non-linear, the measurement to state coupling to be non-linear, and that posterior density to be non-Gaussian.

The Joint Multitarget Probability Density (JMPD), cont'd

- In principle, time evolution of the posterior can be computed via a two-step recursion, prediction and update:

Prediction (generating the Kinematic prior)

$$p\left(\mathbf{X}^k \mid \mathbf{Z}^{k-1}\right) = \int p\left(\mathbf{X}^k \mid \mathbf{X}^{k-1}\right) p\left(\mathbf{X}^{k-1} \mid \mathbf{Z}^{k-1}\right) d\mathbf{X}^{k-1}$$

Update (Bayes' rule to Incorporate Measurements)

$$p\left(\mathbf{X}^k \mid \mathbf{Z}^k\right) = \frac{p\left(\mathbf{z}^k \mid \mathbf{X}^k\right) p\left(\mathbf{X}^k \mid \mathbf{Z}^{k-1}\right)}{p\left(\mathbf{z}^k \mid \mathbf{Z}^{k-1}\right)}$$

$$\text{where} \quad p\left(\mathbf{z}^k \mid \mathbf{Z}^{k-1}\right) = \int p\left(\mathbf{z}^k \mid \mathbf{X}^k\right) p\left(\mathbf{X}^k \mid \mathbf{Z}^{k-1}\right) d\mathbf{X}^k$$

- In our general setting of non-linear target kinematics, non-linear measurements and non-Gaussian densities, an analytical solution for these recursions does not exist.

Particle Filter Implementation of JMPD

- One method of solving the prediction and update equations is to discretize the density on a fixed grid and solve using finite difference methods.
- A more natural solution strategy which eliminates the need for discretization and for a fixed grid is to use the Monte Carlo method known as particle filtering.
- Let the Joint Multitarget Probability Density (JMPD)

$$p(\mathbf{x}_1, \mathbf{x}_2, \ldots \mathbf{x}_{T-1}, \mathbf{x}_T \mid \mathbf{Z}) = p(\mathbf{X} \mid \mathbf{Z}), \qquad T = 1 \ldots \infty$$

be approximated by *N* weighted samples (particles) as

$$p(\mathbf{X} \mid \mathbf{Z}) \approx \sum_{p=1}^{N} w_p \delta(\mathbf{X} - \mathbf{X}_p)$$

where a particle is given by

$$\mathbf{X}_p = [\mathbf{X}_{p,1} \quad \mathbf{X}_{p,2} \quad \ldots \quad \mathbf{X}_{p,T-1} \quad \mathbf{X}_{p,T}]'$$

Particle Filter Implementation of JMPD

- Using the definition of a particle just given,

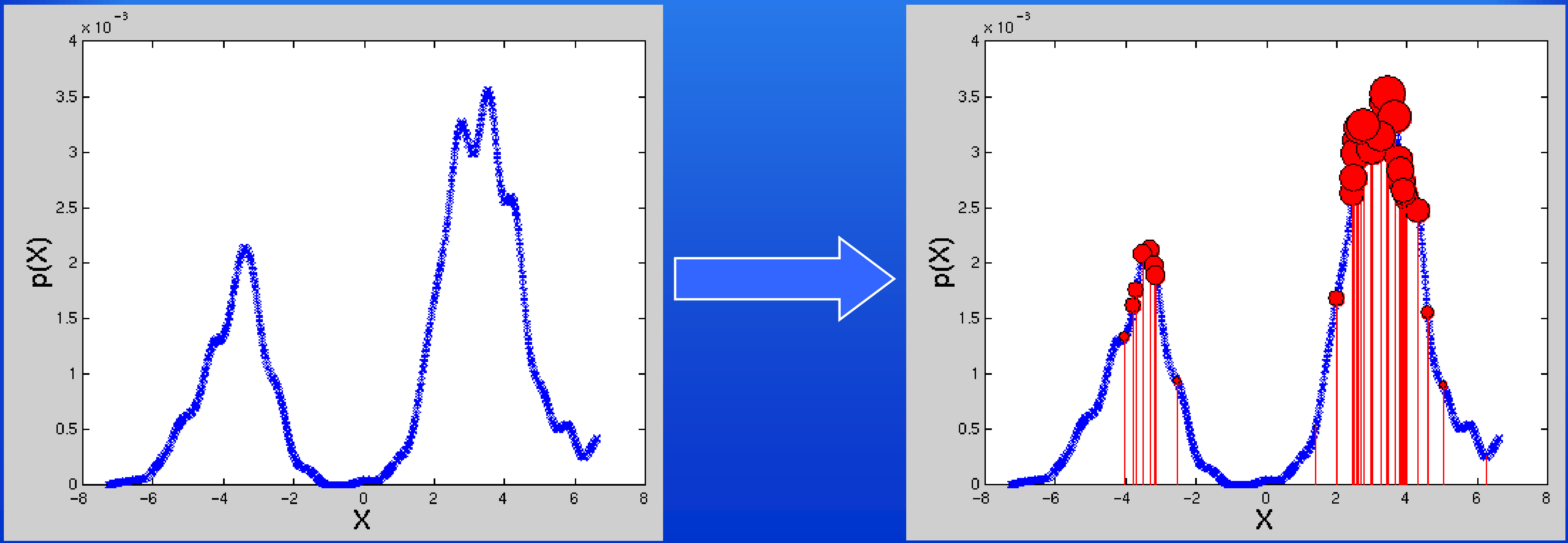
$$\mathbf{X}_p = [\mathbf{X}_{p,1} \quad \mathbf{X}_{p,2} \quad \ldots \quad \mathbf{X}_{p,T-1} \quad \mathbf{X}_{p,T}]'$$

- Each $\mathbf{X}_{p,i}$ in the particle $\mathbf{X}_p$ is the state vector of a particular target, and will be called a partition of the state vector.
- Each of the particles $\mathbf{X}_p$ is a sample drawn from the JMPD $p(\mathbf{X}^k|\mathbf{Z}^k)$
  - Therefore, a particle may have *0, 1, … ∞* partitions, each partition corresponding to a different target.
  - The number of partitions in a particle is that particles estimate of the number of targets in the surveillance region.

- We want to generate a set of samples (particles) that approximate the joint multitarget probability density $p(\mathbf{X}^k|\mathbf{Z}^k)$.

Particle Filtering 101

- A particle filter is a sequential Monte Carlo method used to solve the prediction integral and update equation.
- The key concept in a particle filter is that the posterior density function is represented by a set of random samples with associated weights.
- Particle Filtering can handle
  - Non-linear measurement to state coupling
  - Non-linear state evolution
  - Non-Gaussian densities



PF 101, cont'd

- We denote each sample (particle) *p* as $\mathbf{X}_p^k$ and its weight w_p^k
  We then approximate the density

$$p(\mathbf{X}^k \mid \mathbf{Z}^k) \approx \sum_p w_p^k \delta\left(\mathbf{X}^k - \mathbf{X}_p^k\right)$$

- Given this representation, evaluating the usual estimates is straightforward, e.g.

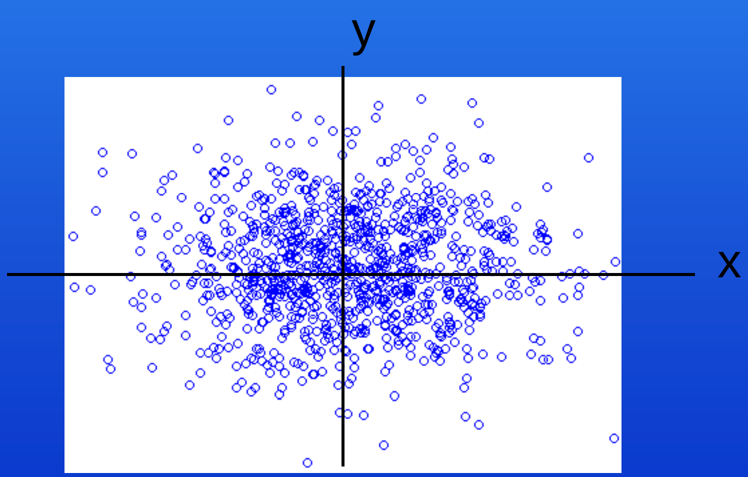
$$\begin{aligned} E\left\{\mathbf{X}^k \mid \mathbf{Z}^k\right\} &\equiv \int \mathbf{X}^k p\left(\mathbf{X}^k \mid \mathbf{Z}^k\right) d\mathbf{X}^k \\ &= \int \mathbf{X}^k \sum_p w_p^k \delta\left(\mathbf{X}^k - \mathbf{X}_p^k\right) d\mathbf{X}^k \\ &= \sum_p w_p^k \mathbf{X}_p^k \end{aligned}$$

PF 101 – Initialization

- PF is then a method of evaluating the prediction and update integrals numerically.
- To initialize, sample *N* particles from $p(\mathbf{X}^0|\mathbf{Z}^0)$:

$$\mathbf{X}_p^k,\ p = 1 \ldots N$$

$$\text{e.g. If we let } \mathbf{X} = [x\ \dot{x}\ y\ \dot{y}] \text{ and choose } p\left(\mathbf{X}^0\right) \Leftarrow N(\mathbf{0}, \mathbf{\Sigma})$$



PF

PF 101 – Prediction

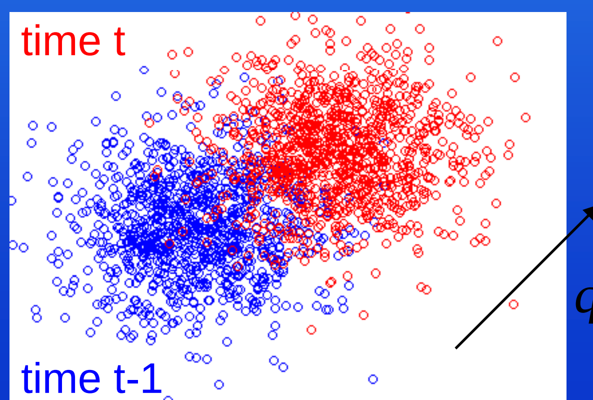
- For each particle that is used to approximate the density at *k-1*, $\mathbf{X}_p^{k-1}$, generate a sample

$$\mathbf{X}_p^k \Leftarrow p\left(\mathbf{X}^k \mid \mathbf{X}_p^{k-1}, \mathbf{z}^k\right)$$

- In general, it is very difficult to sample from this density, so we sample from an importance density

$$q\left(\mathbf{X}^k \mid \mathbf{X}_p^{k-1}, \mathbf{z}^k\right) \approx p\left(\mathbf{X}^k \mid \mathbf{X}_p^{k-1}, \mathbf{z}^k\right)$$

- Then the particle weights are given by $w_p^k \propto \dfrac{p(\mathbf{X}_p^k)}{q(\mathbf{X}_p^k)}$



$$q\left(\mathbf{X}^k \mid \mathbf{X}_p^{k-1}, \mathbf{z}^k\right)$$

PF 101 – Importance Density

- Choice of importance density (proposal density) is of critical importance as the performance of the PF can be dramatically effected by *q*.

- The weights can be rewritten

$$w_p^k \propto w_{k-1}^p \frac{p\left(\mathbf{z}^k \mid \mathbf{X}_p^k\right) p\left(\mathbf{X}_t^i \mid \mathbf{X}_p^{k-1}\right)}{q\left(\mathbf{X}_p^k \mid \mathbf{X}_p^{k-1}, \mathbf{z}^k\right)}$$

- And in the case where

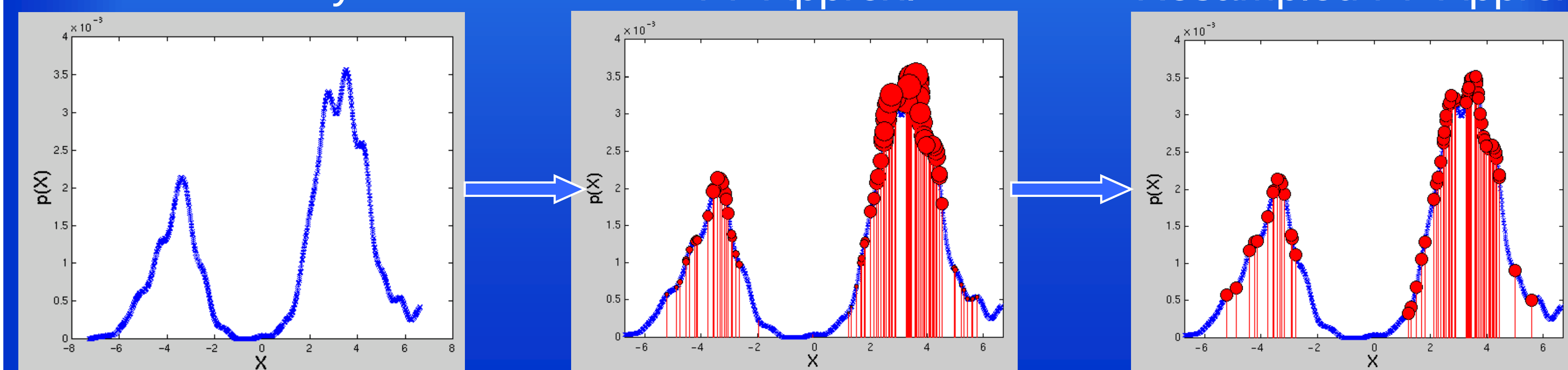
$$q\left(\mathbf{X}^k \mid \mathbf{X}_p^{k-1}, \mathbf{z}^k\right) \equiv p(\mathbf{X}^k \mid \mathbf{X}^{k-1})$$

$$w_p^k \propto w_{k-1}^p\, p\left(\mathbf{z}^k \mid \mathbf{X}_p^k\right)$$

PF 101 – Resampling

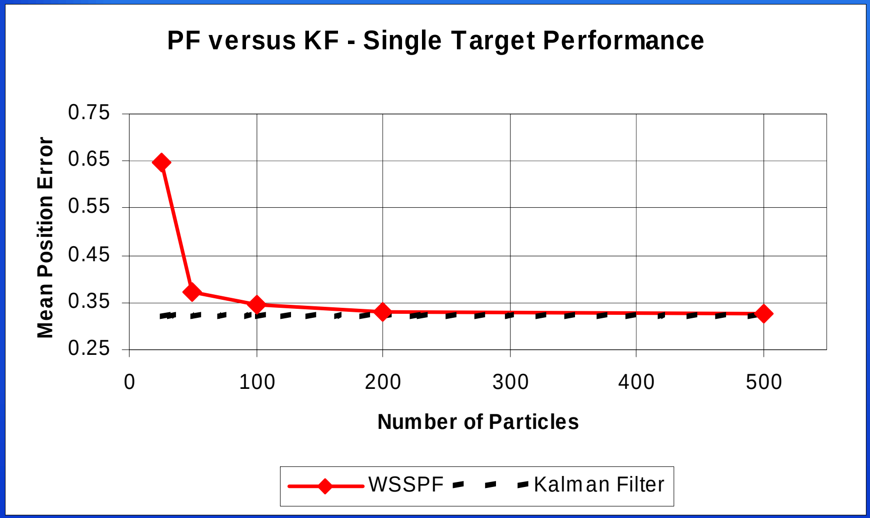
- With no adjustments, one finds that the variance of the w_p's can only increase (i.e. after a few iterations, all but 1 of the w_p's have near-zero weight).
- Therefore a resampling step is added
  - From the set of N particles, resample (with replacement) a new set of particles based on w_p. This way, only particles with high weights are retained.
- The PF with $q\left(\mathbf{X}^k \mid \mathbf{X}_p^{k-1}, \mathbf{z}^k\right) \equiv p(\mathbf{X}^k \mid \mathbf{X}^{k-1})$ and resampling at every time step is the 'standard' PF and called SIR in the literature.

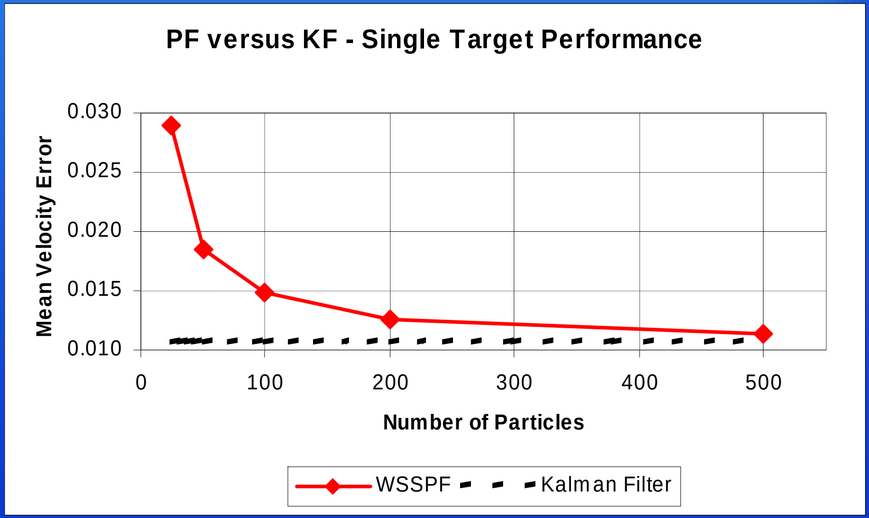
True Density | PF Approx. | Resampled PF Approx.



How well does it work? (Single Target Case)

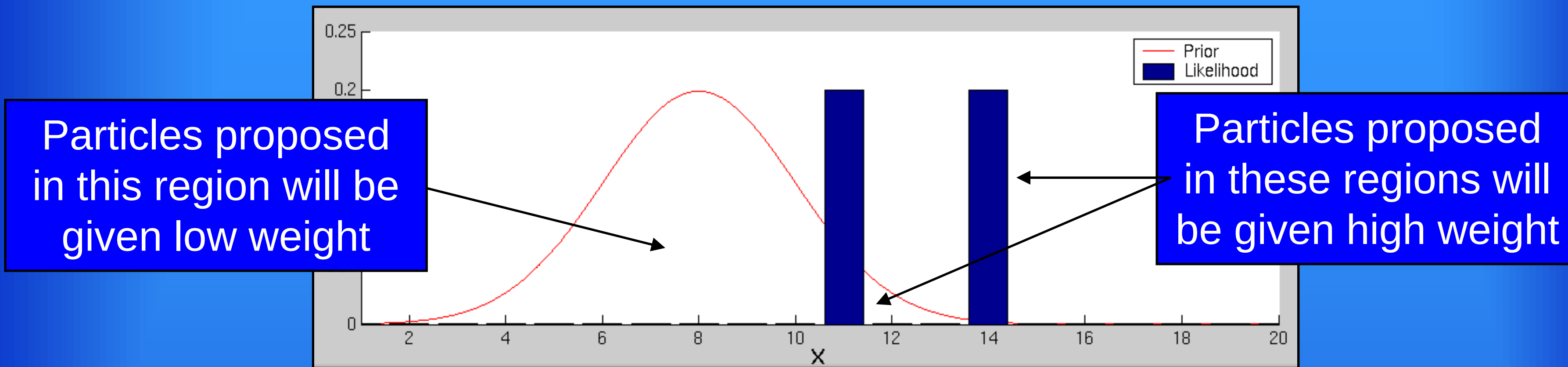
- Particle Filtering allows us to easily incorporate
  - Non-linear Measurement to State Coupling
  - Non-linear State Evolution (Target Motion)
  - Non-Gaussian Densities
- **Let's ignore all these benefits for a moment**
- How does it compare to a Kalman Filter in the regime where a Kalman Filter is applicable (and optimal)?
  - Simulation: Linear motion, linear measurements, Gaussian pdf. A single target with state vector $[x \ \dot{x} \ y \ \dot{y}]$





Flops Count : PF at 100 Particles = 10 MegaFlops, KF: 1 MegaFlop

Exploiting the Multitarget Nature of the Problem to Build Better Particle Proposals



- Propose particles in regions of high likelihood
  - Tailor proposal density so that only high-weight particles are proposed
  - Resampling becomes unnecessary if all particles are in high likelihood areas

- We focus here on exploiting the fact that this is a multi-target problem and that the partitions of a particle are tracking different targets

Multitarget Proposal Densities

- Recall that the posterior density is approximated by a set of N_{parts} particles

$$p(\mathbf{X} \mid \mathbf{Z}) \approx \sum_{p=1}^{N} w_p \delta(\mathbf{X} - \mathbf{X}_p)$$

- And each particle $\mathbf{X}_p$ is partitioned as

$$\mathbf{X}_p = \begin{bmatrix} \mathbf{X}_{p,1} \\ \vdots \\ \mathbf{X}_{p,T} \end{bmatrix} \qquad \text{e.g.} \qquad \mathbf{X}_p = \begin{bmatrix} 13.25 \\ -0.05 \\ 3.13 \\ 0.03 \\ 1.44 \\ 0.07 \\ 3.05 \\ 0.00 \end{bmatrix} \begin{matrix} \left.\vphantom{\begin{matrix}1\\1\\1\\1\end{matrix}}\right\} \mathbf{x}_{p,1} \\ \left.\vphantom{\begin{matrix}1\\1\\1\\1\end{matrix}}\right\} \mathbf{x}_{p,2} \end{matrix}$$

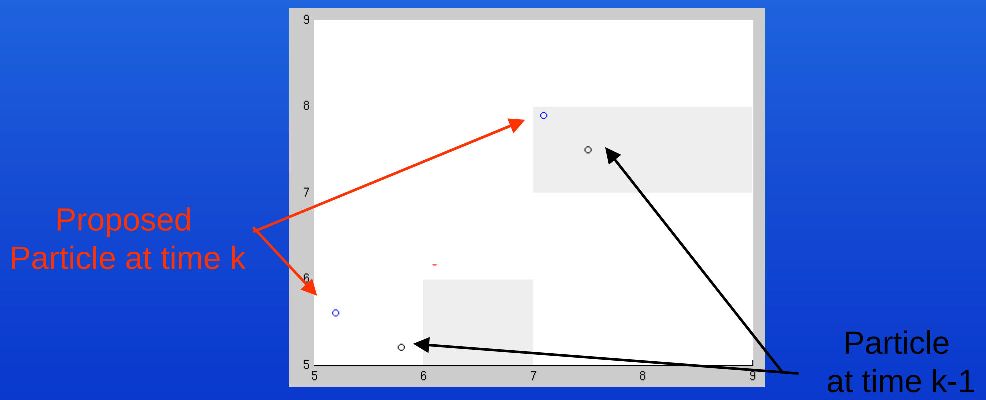
where each partition corresponds to a target $\mathbf{x}_{p,i} = [x_i \ \ \dot{x}_i \ \ y_i \ \ \dot{y}_i]^T$

Multitarget Proposal Densities

Kinematic

In this traditional method of proposing particles, each particle at time *k-1* generates a new particle at time *k* via the kinematic (motion) model $P(\mathbf{X}^k|\mathbf{X}^{k-1})$

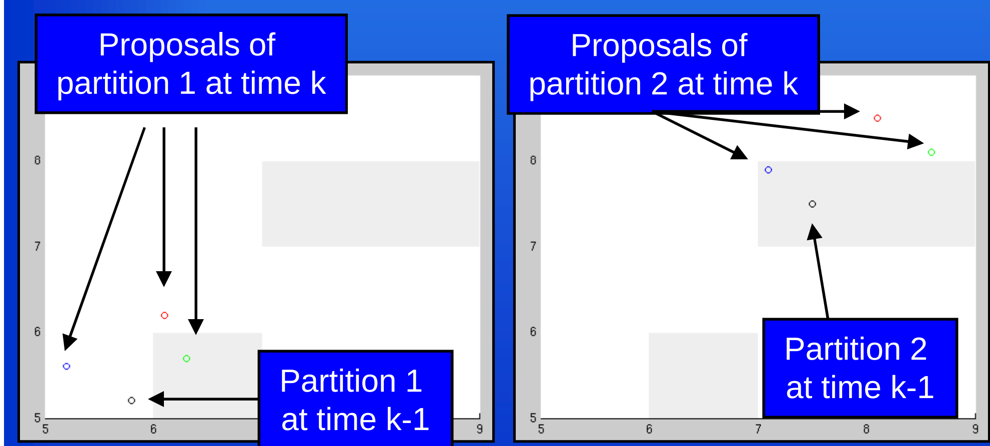
Measurements are not used when proposing particles



Multitarget Proposal Densities

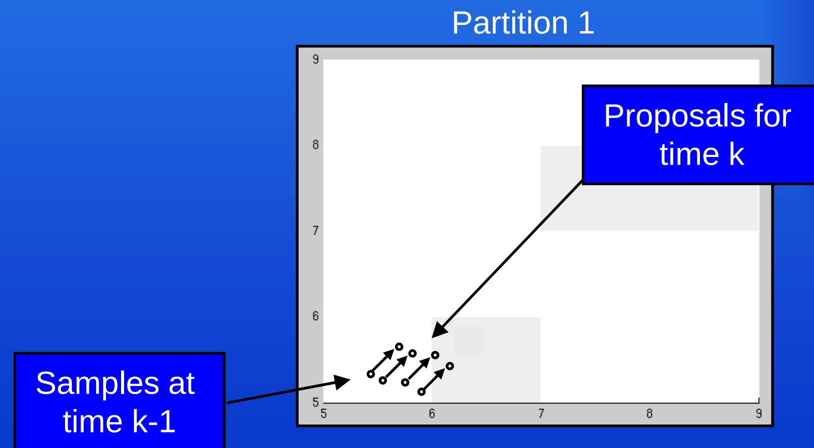
Coupled Partition

Particles at time *k* are built partition-by-partition. For each of the N_{parts} samples in a partition, we propose *M* possible samples via the Kinematic prior, weight each using the measurements, and select one.



Independent Partition

Particles at time *k* are built partition-by-partition. For each of the N_{parts} samples in a partition, we propose one new sample using the Kinematic prior and weight using the measurements. We then select with replacement N_{parts} samples from this group.



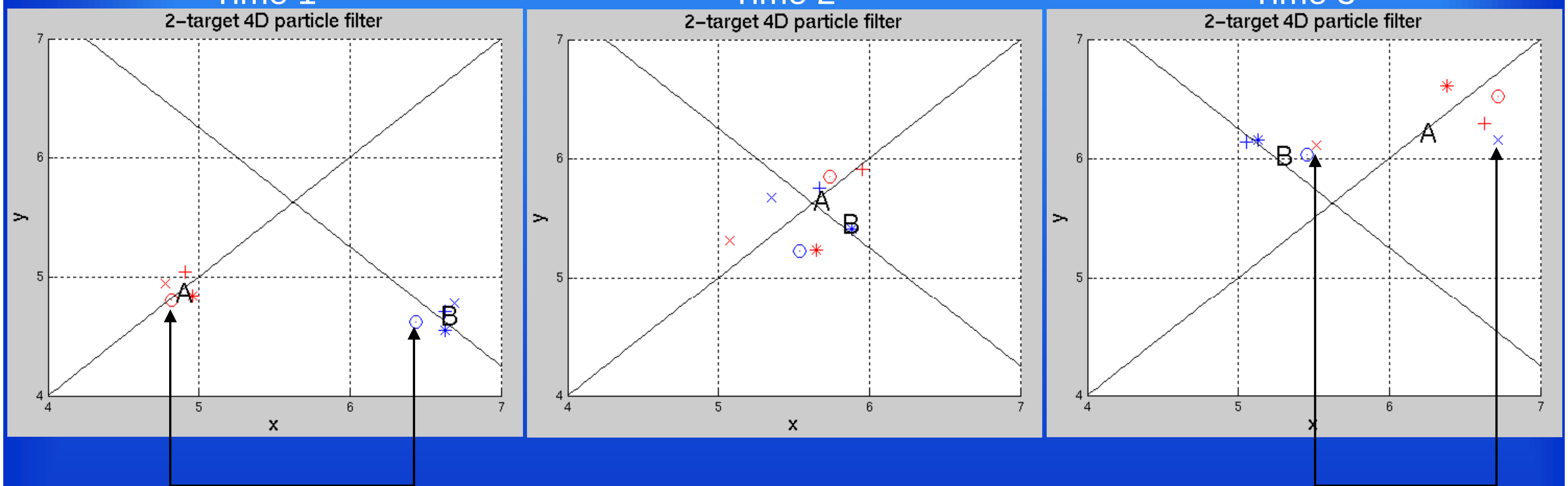
When is the IP method Applicable?

- The JMPD is permutation symmetric,
  - If $\mathbf{x}_1$ and $\mathbf{x}_2$ are the states of two targets, the multitarget states $\mathbf{X} = [\mathbf{x}_1,\mathbf{x}_2]$ and $\mathbf{X} = [\mathbf{x}_2,\mathbf{x}_1]$ refer to the same event.
  - The particle filter manifestation of this permutation symmetry is *partition swapping*.
  - This symmetry is directly related to the measurement-to-target association problem.
  - The particle filter implementation of JMPD must recognize this symmetry and account for it, particularly if sophisticated particle proposal schemes are utilized.

Partition Swapping

- Consider 4 particles (denoted by “o”,”x”,”+” and “*”) that are each tracking two targets (Target A and Target B)
- Each particle has two partitions – color coded blue and red
- When proposing according to the Kinematic prior, partition swapping may occur when targets cross – this is completely acceptable.

Time 1 Time 2 Time 3



Each particle has an estimate of both target A and target B.

When targets “cross” partition swapping is possible.

The ordering of target partitions in particle “x” is opposite of the others.

Partition Swapping

- A particle contains an estimate of both the number of targets and their states, e.g. when target state is modeled $[x_i \ \dot{x}_i \ y_i \ \dot{y}_i]^T$, 2-target particle may be

$$\mathbf{X}_p = \begin{bmatrix} 13.25 \\ -0.05 \\ 3.13 \\ 0.03 \\ 1.44 \\ 0.07 \\ 3.05 \\ 0.00 \end{bmatrix} \begin{matrix} \left.\begin{matrix} \\ \\ \\ \\ \end{matrix}\right\} \mathbf{x}_{p,1} \\ \left.\begin{matrix} \\ \\ \\ \\ \end{matrix}\right\} \mathbf{x}_{p,2} \end{matrix}$$

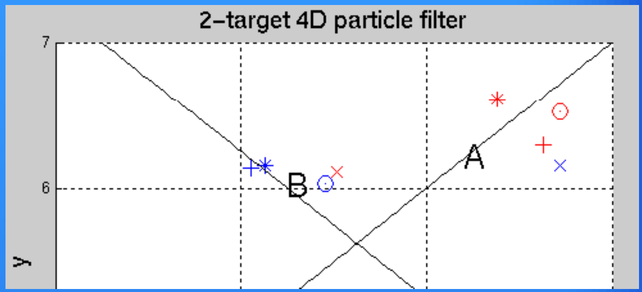
- This symmetry manifests itself directly in the particles used to approximate the density. The two particles $\mathbf{X}_1$ and $\mathbf{X}_2$ represent the same event:

$$\mathbf{X}_1 = \begin{bmatrix} 13.25 \\ -0.05 \\ 3.13 \\ 0.03 \\ 1.44 \\ 0.07 \\ 3.05 \\ 0.00 \end{bmatrix} \quad \mathbf{X}_2 = \begin{bmatrix} 1.44 \\ 0.07 \\ 3.05 \\ 0.00 \\ 13.25 \\ -0.05 \\ 3.13 \\ 0.03 \end{bmatrix} \longrightarrow \quad \overline{\mathbf{X}} = \begin{bmatrix} 7.35 \\ 0.01 \\ 3.09 \\ 0.02 \\ 7.35 \\ 0.01 \\ 3.09 \\ 0.02 \end{bmatrix}$$

Partition Swapping

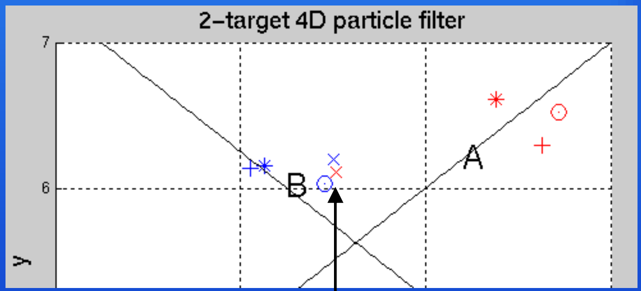
- Using the IP Method in scenarios where swapping has occurred is unacceptable
  - IP assumes that a particular partition is associated with one target
  - e.g. IP assumes all of the red partitions are tracking the same target.

Time 3



- Using IP at Time 3 leads to some particles that have both partitions associated with the same target
  - To build a new particle, IP proposes a new partition 1 by sampling from the set *, o, +, x and a new partition 2 by sampling from the set *, o, +, x
  - This may lead to a particle which is constructed using x and o

Time 4



This particle (x) now has both partitions tracking target B – i.e. it (incorrectly & artificially) contributes probability mass to the state “two targets at location B”

Partition Swapping, cont'd

- The CP Method does not mix particles – lineage is maintained.
  - New particles will be proposed with the same ordering as particles from the previous time step.
  - Permutation symmetry is respected and probability mass is not artificially transferred to incorrect states.
- CP applicable in all scenarios.
  - Significantly less efficient then IP method
  - When IP appropriate, it should be used.
- IP applicable when targets are 'well separated' (acting independently) and the partitions are ordered identically.

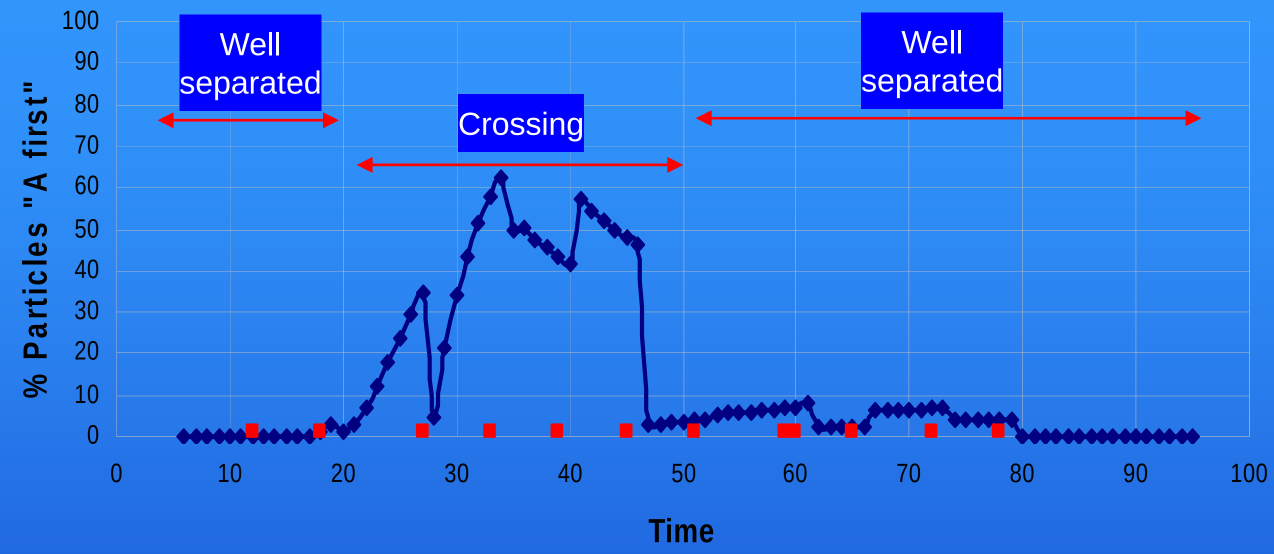
Reordering Partitions

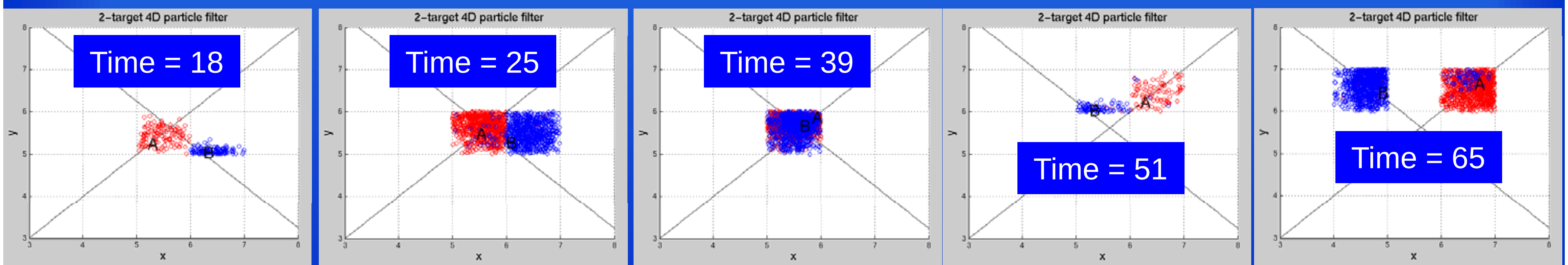
- Assume now that the actual targets are well separated, but different particles have different orderings

$\mathbf{X}_1 = \begin{matrix} A \\ B \end{matrix}$ $\quad \mathbf{X}_2 = \begin{matrix} B \\ A \end{matrix}$ $\quad \mathbf{X}_3 = \begin{matrix} A \\ B \end{matrix}$ $\quad \mathbf{X}_4 = \begin{matrix} B \\ A \end{matrix}$ $\quad \mathbf{X}_5 = \begin{matrix} A \\ B \end{matrix}$ $\quad \cdots$

- We call the [A B] particles "A-first" particles and the [B A] "B-first" particles.
- Resampling results in a new set of particles with different distribution of A first and B first particles.
  - The only stable state is for 100% to be A-first of 100% to be B-first.
  - In practice, resampling quickly moves the distribution to a stable state.

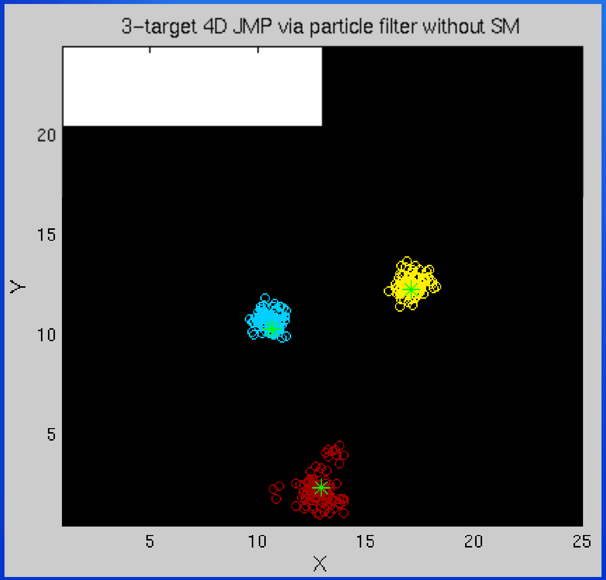
Reordering Partitions

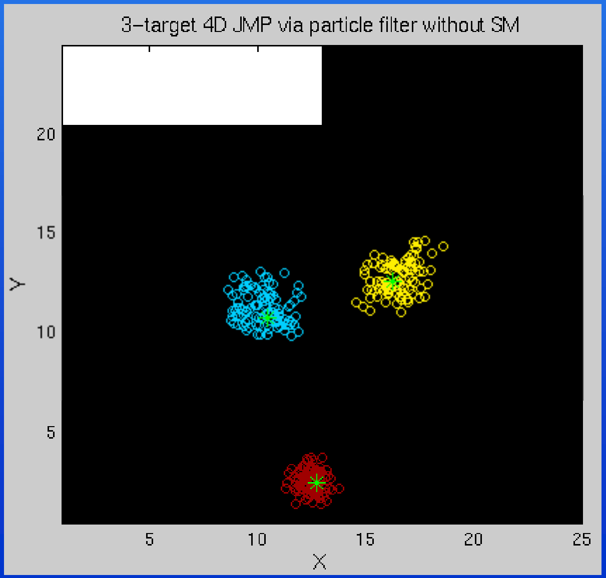


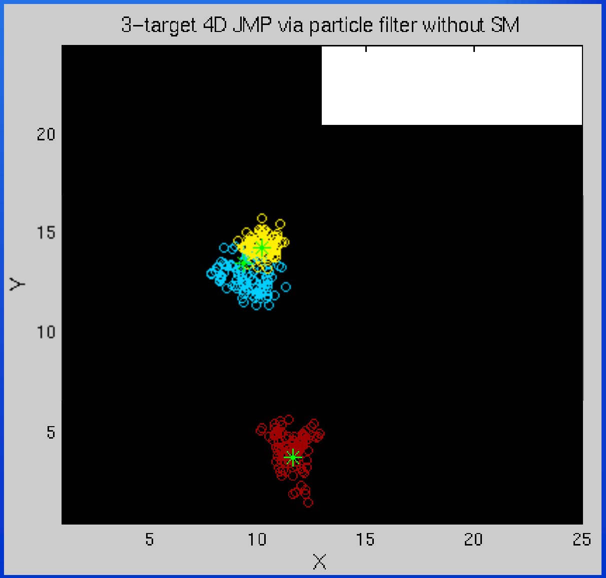


Multitarget Proposal Densities

- When targets are well separated (in the measurement space), each sample is associated with a particular target. IP is appropriate here.
- When targets become “close” samples commingle and measurements of one target may effect samples associated with other targets. IP is not appropriate.
- Use Independent Partitions (IP) when targets are well separated and Coupled Partitions (CP) when they are not.

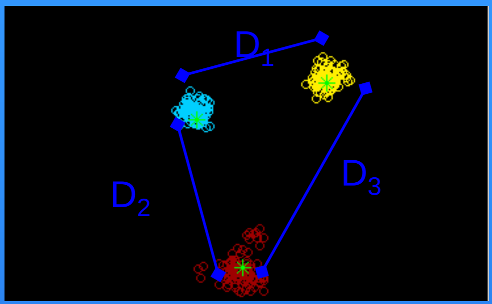




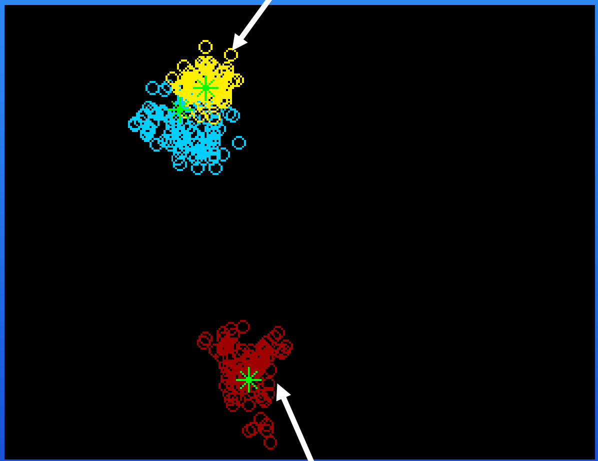


Adaptive Proposal Method Switching

When are partitions 'well separated'?



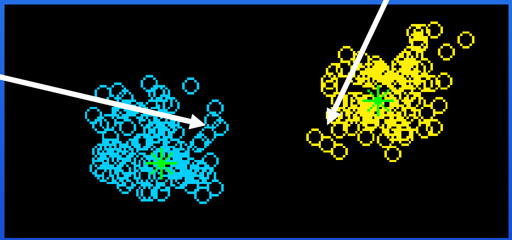
Use CP on these



Use IP on this

Sample from partition *i* closest to mean of partition *j*

Sample from partition *j* farthest from mean of partition *j*



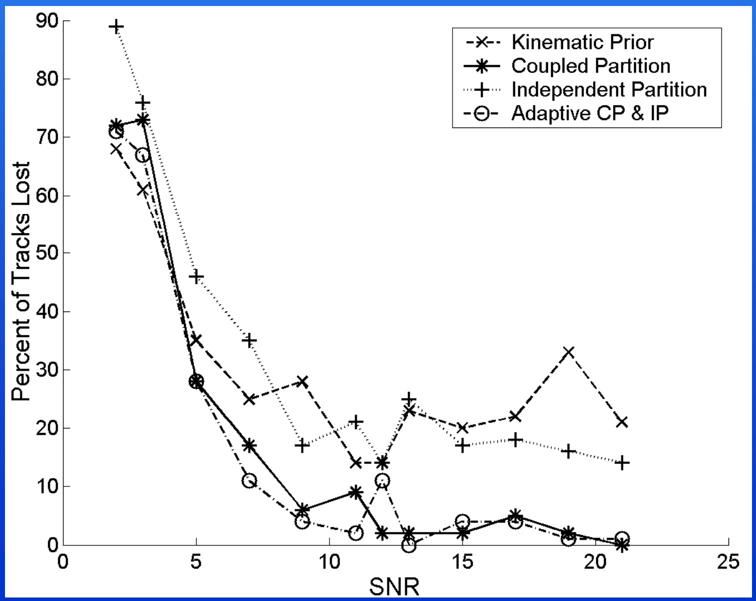
Mahalanobis Distance

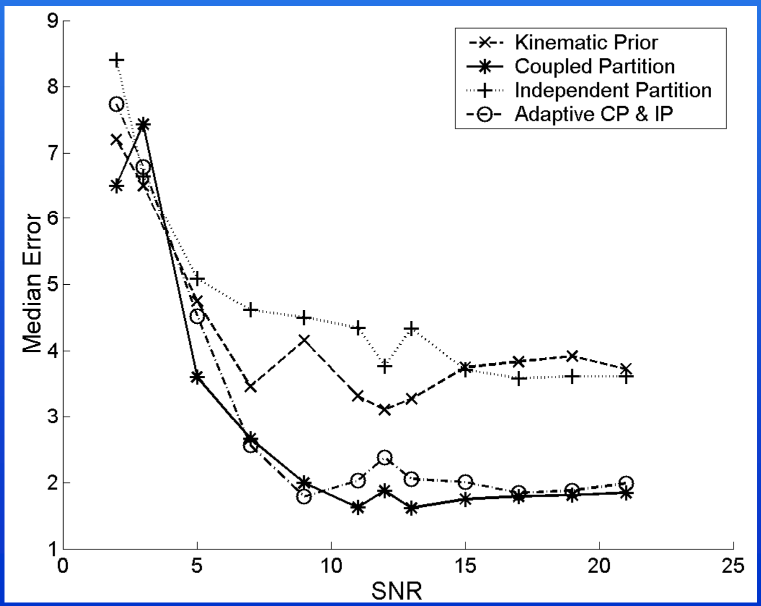
$$r_{i,j}^2 = (\mathbf{x}_i - \mathbf{m}_j)' \, \Sigma_j^{-1} (\mathbf{x}_i - \mathbf{m}_j)$$

Multitarget Proposal Densities

- Simulation: Three targets moving on a grid.
- Targets spend approximately 50% of the time ‘near’ each other (when only CP is appropriate) and 50% of the time well separated (where IP is appropriate)
- Adaptive method achieves similar performance as CP at half the FLOPS.

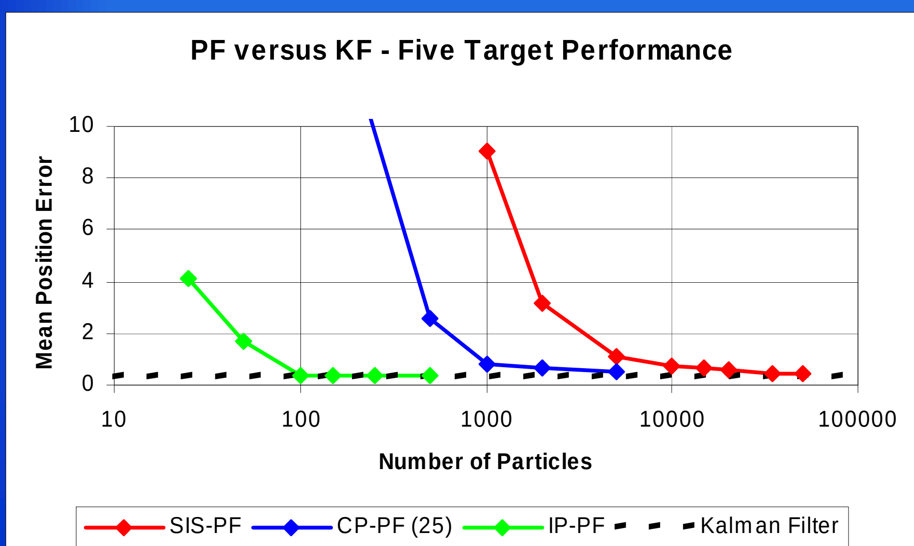
Method	Flops
Kinematic Prior	6.32E+06
Independent Partition	6.74E+06
Adapative CP/IP	5.48E+07
Coupled Partition	1.25E+08

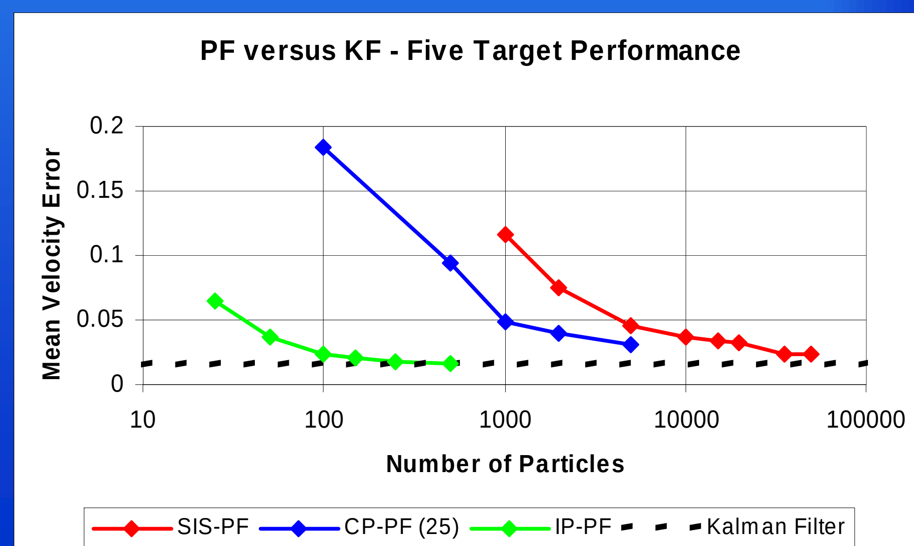




How much Effort does the adaptive strategy save?

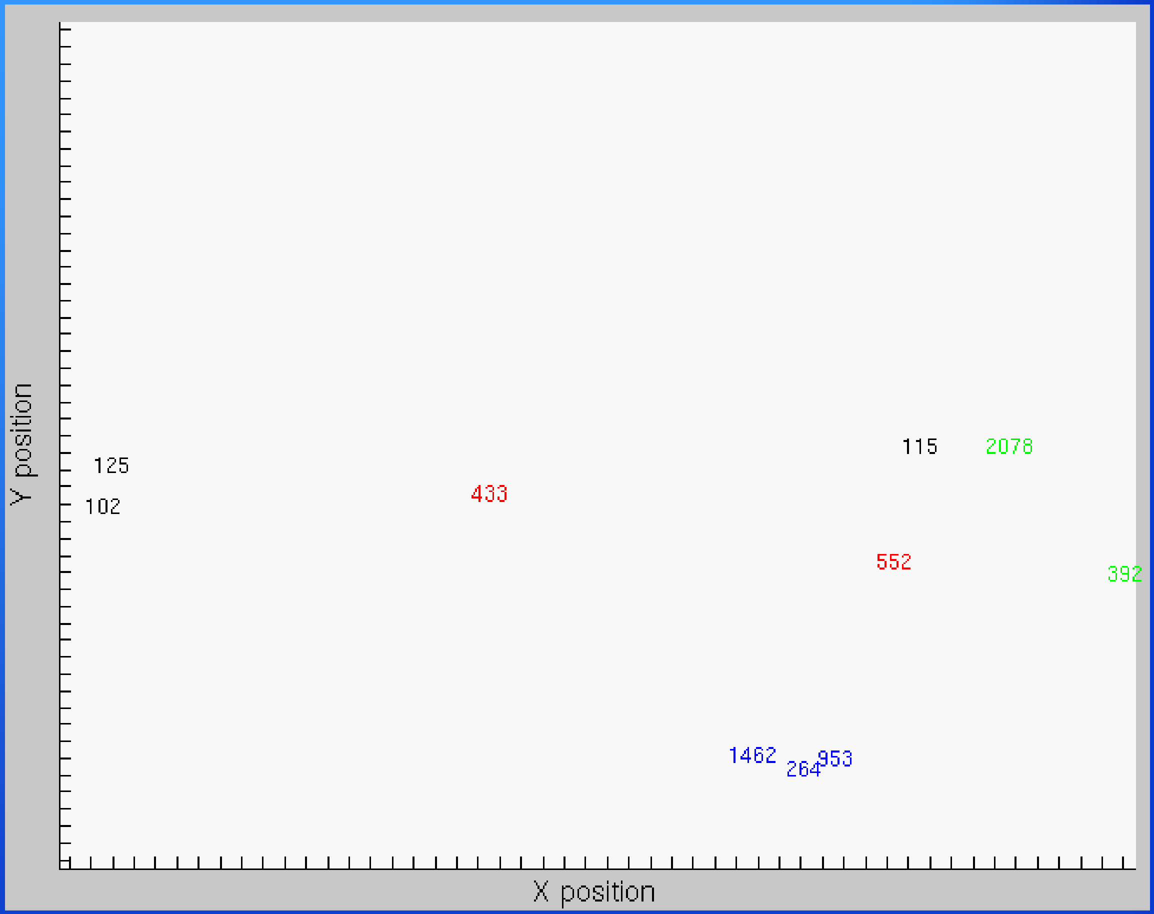
- We compare a PF using the Kinematic Prior with one using the adaptive strategy.
- Particle Filtering allows for
  - Non-linear Measurement to State Coupling
  - Non-linear State Evolution (Target Motion)
  - Non-Gaussian Densities
- **We ignore all these benefits for a moment**
- How well does the multi-target PF perform in comparison to a Kalman Filter in the regime where a Kalman Filter is applicable (and optimal)?
  - Simulation: Linear motion, linear measurements, Gaussian pdf.
  - Five (well separated) targets with state vectors $[x \ \dot{x} \ y \ \dot{y}]$





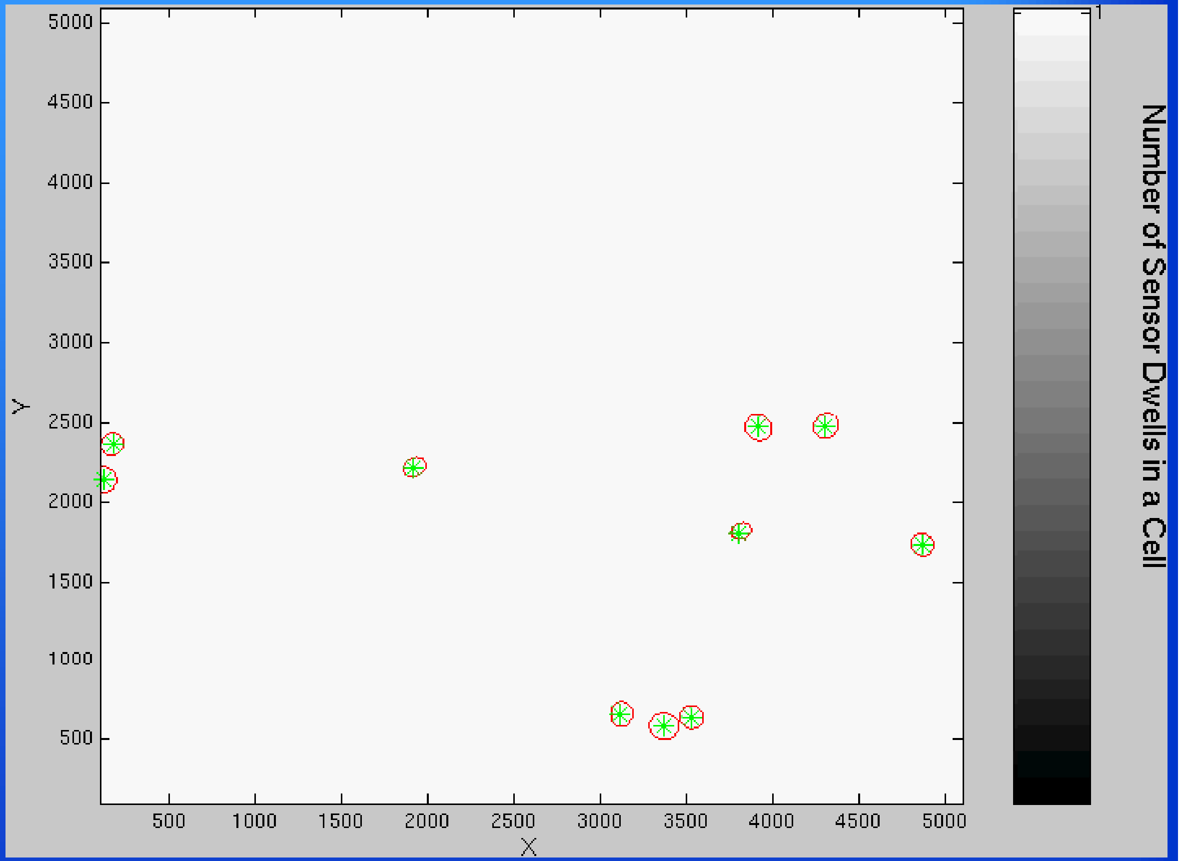
Is it Tractable? 10 ***Real*** Targets

- Vehicle Trajectories
  - 10 Real targets culled from the NTC Sensor Strike Track Files
    - #433, #552 Cross
    - #392, #2078 travel together sometimes
    - #264, #953, #1462 travel together a lot
    - #102, #115, #125 added to bring the total to 10
  - 1000 time steps, 1 second apart
  - Vehicles are time & space shifted to be in the same region at the same time



Is it Tractable? 10 ***Real*** Targets

- Sensor Simulation
  - The usual quasi-GMTI simulation where sensor measures 10x1 grid cell and gets 10 returns
  - The sensor grid is 50 cells x 50 cells. Each cell is 100m x 100m.
  - SNR = 12
- JMPD - Particle Filter
  - Nparts = 500
  - Fully adaptive switching between CP and IP based on sample distance



Runtime ~ 1 Hour on Off the shelf Linux Box
1/3 of “real time”

Conclusion

- We've presented a method of tracking multiple targets based on recursive estimation of their Joint Multitarget Probability Density (JMPD).
- Computational tractability is provided by Particle Filter-based implementation.
  - Adaptive sampling schemes exploit multitarget nature of the problem.
  - Permutation symmetry manifests itself as partition swapping
- Natural framework to do sensor management where the JMPD is used to compute the area of maximal expected information gain.