

Adapting proposal distributions for accurate, efficient mobile robot localization

Patrick Beeson
Aniket Murarka
Benjamin Kuipers

Department of Computer Sciences
The University of Texas at Austin

2D particle filter example



2D particle filter example

- State estimated by a set of **particles**.



2D particle filter example

- After an action



2D particle filter example

- The **action model** moves these particles into a **proposal distribution**.



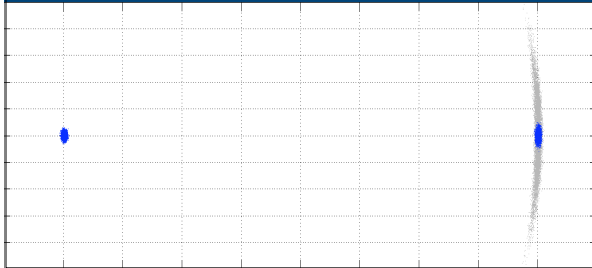
2D particle filter example

- After an observation



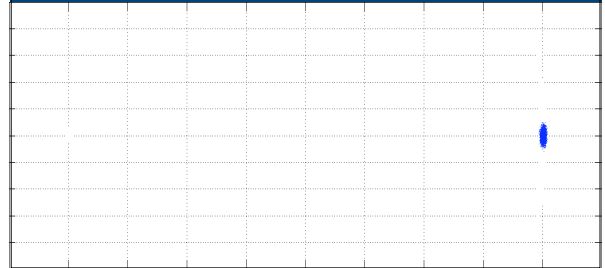
2D particle filter example

- The **observation model** weights particles by their **likelihood**.



2D particle filter example

- The weighted particles represent the new state estimate, the **posterior**.



Localization failure example

- **Wheels slip** → proposal is incorrect → particles have low (or zero) likelihood.



Observations

- **Odometry** is prone to **errors**
 - proposals may be in wrong location, leading to localization failures
 - changing surfaces (incorrect variance)
 - wheels slip (incorrect mean)
- **Laser range finders** are extremely **accurate and precise**
 - likelihoods are often highly peaked

Our work

- Claim:
 - The robot should **believe laser observations more than odometry**.
 - proposal simply defines the “search space” of the particle filter
 - **likelihood approximates posterior estimate**
- Hypothesis:
 - Ensuring that proposals approximate likelihoods improves localization.**

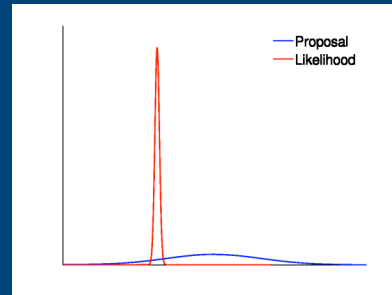
Improving proposals

1. Introduce **new action model** for differential drive robots.
2. Evaluate **changing each proposal at every time-step**.
3. Evaluate **online action model tuning to change future proposals**.

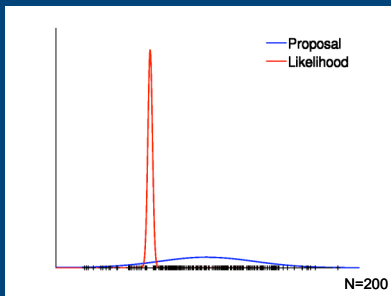
Improving proposals

1. Introduce new action model for differential drive robots.
2. Evaluate **changing each proposal at every time-step**.
 - Data driven proposals
3. Evaluate online action model tuning to change future proposals.

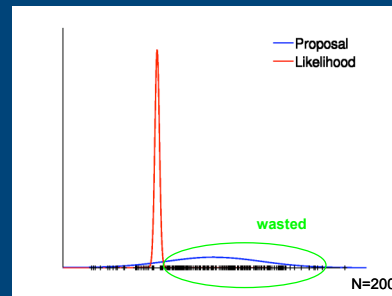
Overestimating example



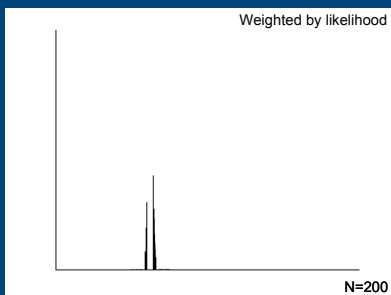
Regular PF example



Regular PF example



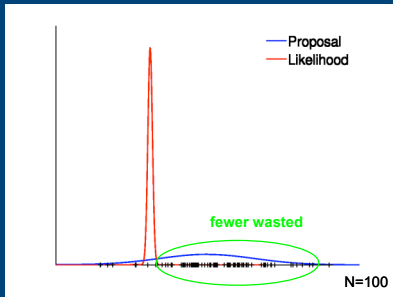
Regular PF example



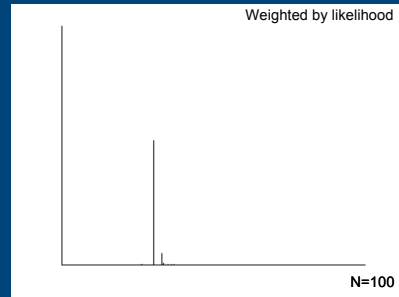
Shrink algorithm

- Try to improve localization when proposal **overestimates** likelihood.
- Use first proposal to estimate a new, second proposal.
- Related work:
 - Grisetti, Stachniss, Burgard, ICRA05

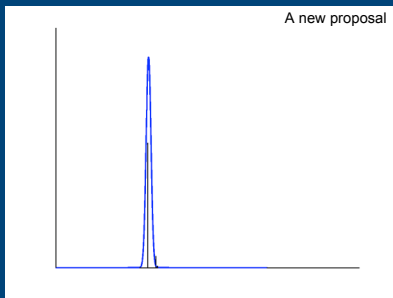
Shrink algorithm example



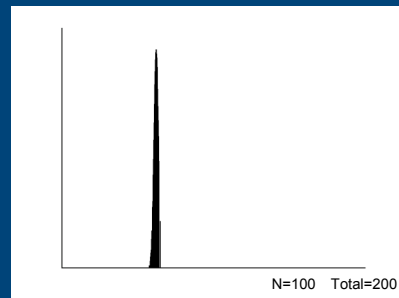
Shrink algorithm example



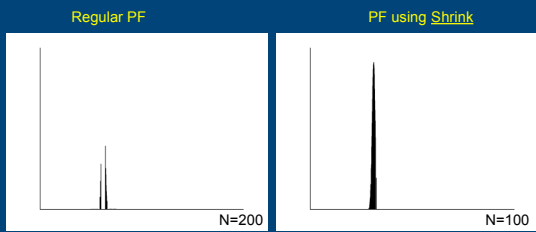
Shrink algorithm example



Shrink algorithm example

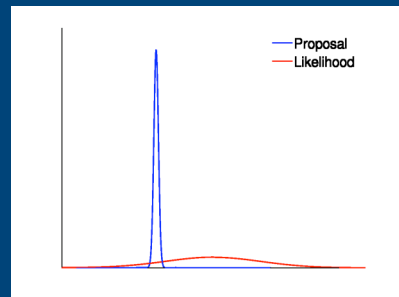


Comparing posterior estimates

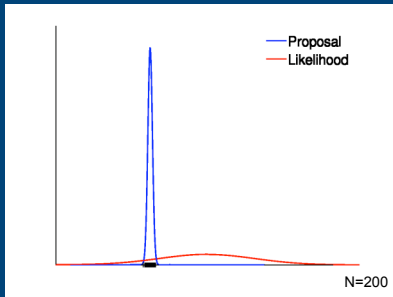


200 total particles

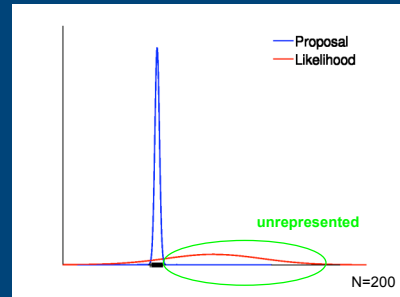
Underestimating example



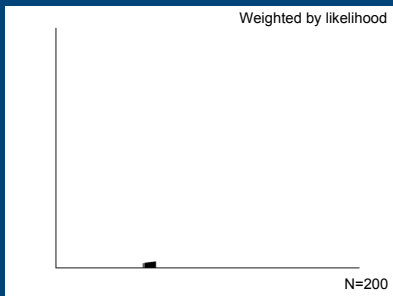
Regular PF example



Regular PF example



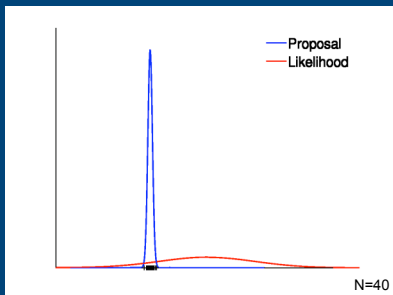
Regular PF example



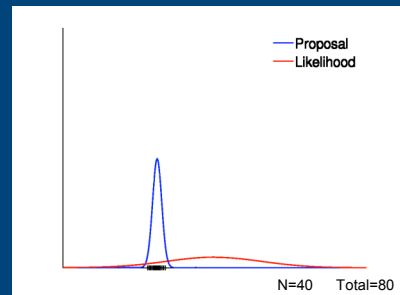
Grow algorithm

- Try to improve localization if proposal **underestimates** likelihood.
- While max. likelihood particle < threshold, double standard deviation

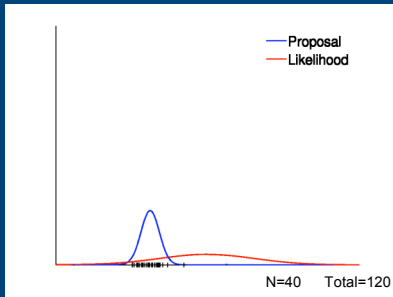
Grow algorithm example



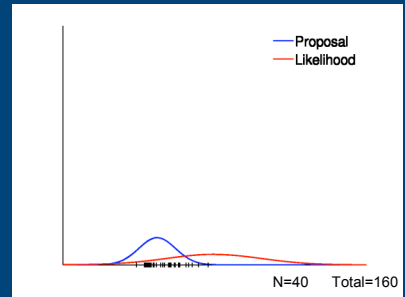
Grow algorithm example



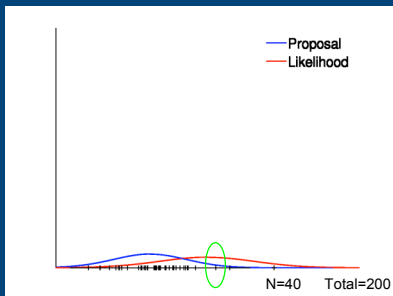
Grow algorithm example



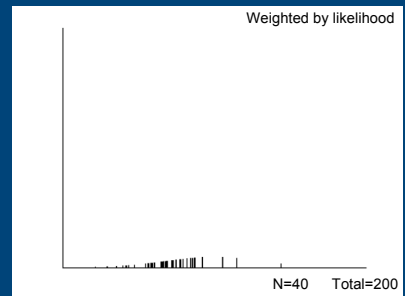
Grow algorithm example



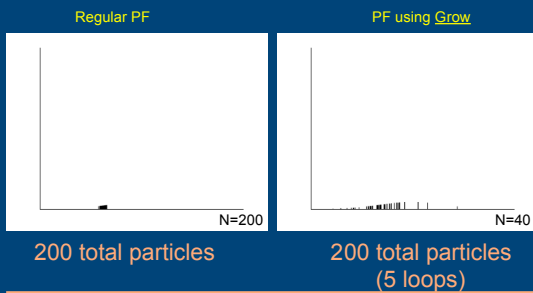
Grow algorithm example



Grow algorithm example



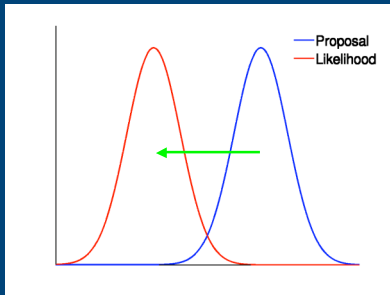
Comparing posterior estimates



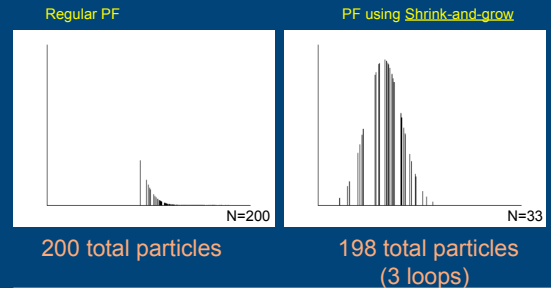
Shrink-and-grow algorithm

- Shrink should enhance PF for overestimating proposals.
- Grow should handle underestimating proposals.
- We examine the combination of the two.
 - Shrink, then if necessary, Grow (then loop)
 - bounded # of loops

Wheel-slippage example



Comparing posterior estimates



Experiment Details

- 4 particle filter algorithms: regular PF, Shrink, Grow, Shrink-and-grow
- Tested each PF algorithm on different action models
 - Large, constant-sized proposals (overestimating)
 - Small, constant-sized proposals (underestimating)
 - Tuned offline to fit experimental trace
- Results highly significant: $p \leq 0.01$ for paired t-test

A few results

- Both Shrink and Grow
 - improves accuracy over regular PF
 - for all 3 action model types (under, over, tuned)
- Shrink
 - improves efficiency
 - fewer wasted particles
- Shrink-and-grow
 - more accurate than all 3 other PF algorithms
 - for all 3 action model types
 - improves efficiency over regular PF

Improving proposals

1. Introduce new action model for differential drive robots.
2. Evaluate changing each proposal at every time-step.
3. Evaluate online action model tuning to change future proposals.
 - Details in paper
 - Related: Eliazar and Parr, ICML04

Online tuning results

- Overestimating action models “shrink”
 - become more efficient
 - higher mean likelihood; fewer wasted particles
- Underestimating action models “grow”
 - become more accurate
 - higher max. likelihood
- Online tuning enhances performance of all 4 PF algorithms, especially when starting with poorly tuned action models.

Conclusions

- With lasers, proposals that approximate likelihoods improve localization.
 - likelihoods are good estimates of the posteriors
- **Shrink-and-grow** is a combination of techniques to **improve proposals at each time step**.
 - improved accuracy (max. likelihood)
 - improved efficiency (mean likelihood)
- **Online parameter tuning improves future proposals**.
 - trade-off between accuracy and efficiency
 - further improves localization performance

Additional points

- Anecdotal evidence from **long-term use**:
 - **Shrink-and-grow** with online parameter tuning has also **eliminated** localization failures during **SLAM**.
- Future work:
 - Demonstrate the benefits of online tuning when surfaces change
 - e.g. shag carpet vs. linoleum
 - or when the physical robot changes.
 - e.g. transporting a heavy load; changing tire pressure

Questions

<http://www.cs.utexas.edu/~qr/robotics>