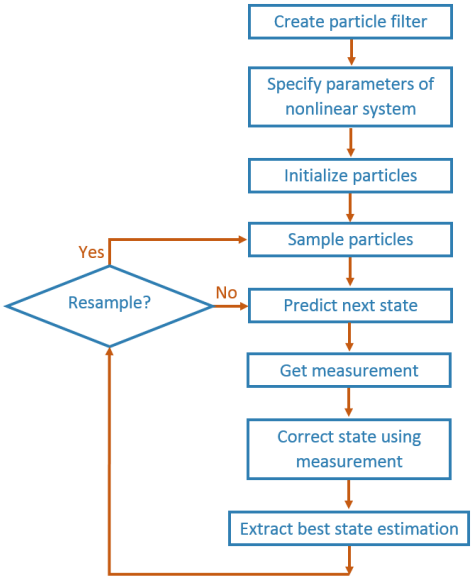
Particle Filtering

Functions: probability/statistics, weighted resampling
Inputs: noisy position/velocity measurements
Metrics: compute time, position/velocity mean-squared-error

1. Overview

Particle filtering is a method of efficiently estimating the state of an environment given noisy observations of prior states. The goal of this method is to make it more computationally feasible to evaluate complex state spaces by finding approximate, rather than exact, solutions. Mathematically, this involves treating data samples as "particles," weighting them according to the approximated world state at a prior epoch, and then resampling the distribution, progressively removing any particles with very low weights from consideration. For further reading, see https://www.cns.nyu.edu/~eorhan/notes/particle-filtering.pdf.



Sample particle filtering pipeline (source: https://www.mathworks.com/help/nav/ug/particle-filter-workflow.html)

MilSpec's particle filtering benchmark uses a custom implementation of a Gaussian particle filter[1] to approximate the true location of a moving object based on a sequence of noisy observations. The benchmark inputs a two-dimensional movement path, specified as a time-series set of noisy position and velocity measurements, performs Gaussian particle filtering, and evaluates the predicted path (both position and velocity) against ground-truth. Configuration options include the input movement path, location prediction mode, and the particle filter's resampling policy. One predefined testcase is supplied with the benchmark; others will be added in a future release.

[1] http://www.ee.sunysb.edu/~djuric/Publications_files/p-journal03b.pdf

2. System requirements

Platform: Ubuntu 18.04 LTS with g++ 7.4.0. Code may build and run successfully on other versions/platforms, but has not been tested with them.
Storage: ~4MB.
Dependencies: None.

3. Build and run

To benchmark:

- Download and extract the zipfile from www.adacenter.org/milspec
- From the `ParticleFilter/` directory, `make clean && make`
- Choose a testcase (example provided in `longpath.config`)
- From the `ParticleFilter/` directory, `./particlefilter [testcase]`
- Results are displayed in the terminal, as below:

```
[redacted]$ ./particlefilter longpath.c
onfig
------- Error Statistics -------
--- Before Filtering ---
x_pos MSE : 222.149
y_pos MSE : 223.823
x_vel MSE : 24.8528
y_vel MSE : 24.5345
--- After Filtering ---
x_pos MSE : 37.7457
y_pos MSE : 38.2122
x_vel MSE : 5.60809 (average magnitude of x_vel = 99.1285)
y_vel MSE : 5.97416 (average magnitude of y_vel = 125.424)
------- Time Statistics -------
Mean time per iteration: 239.764 usec
Maximum time over all iterations: 2148 usec
Minimum time over all iterations: 140 usec
Particle Filter operated in real-time (maximum time = 2148 usec < timestep = 250
000 usec)
------- End Statistics -------
```

4. Code structure

Coming soon!

5. MilSpec development notes, errata, changelog

v0.9:

- Built baseline Gaussian particle filtering implementation
- Added config file-based input options
- Added performance instrumentation
- Added input generation tool
- Added result verification/sanity-check tool

6. Acknowledgements

MilSpec is a project under development at the University of Michigan – Ann Arbor by Pete Ehrett, Bing Schaefer, Adrian Berding, Nathan Block, John Paul Koenig, Pranav Srinivasan, Todd Austin, and Valeria Bertacco. This project is supported by the Applications Driving Architectures Center, one of six centers of JUMP, a Semiconductor Research Corporation program co-sponsored by DARPA.