Real-Time Rodent Tracking

Eric Jonas

October 23, 2012

Abstract

Here I propose a short-duration project (four to six weeks) with two goals: advance the state-of-the-art in rodent behavioral tracking for awake behaving electrophysiology and to expand my understanding of sequential Monte Carlo methods.

1 Motivation

Sequential Monte Carlo (SMC) methods differ tremendously in both mathematical foundation and practical application from the Markov-Chain Monte Carlo methods that I have been using successfully for the past five years. They excel in time-series models with ever-expanding state-spaces and long-running correlations, but at the same time provide weaker asymptotic guarantees than MCMC. Part of my research/academic interest has been determining which inference methods let me focus on building more complex models without worry about inference details. I believe that some recent advances in SMC methods might be getting us close to that point, but this is based on a cursory review of the literature and a very limited background.

Simultaneously, it has been a very long time since I've looked at non-econometric time series, and I've never tackled a tracking problem before. I miss working with real data in a neural context, and if I'm going to attempt to contribute going forward to the field of neural data analysis, I'm going to need more experience. And I'm on vacation, and this is a lot more fun than going to the beach.

2 Proposal

Currently animals are tracked using software that implements various heuristics and, in regions of uncertainty, asks for human intervention. The utility of this approach is the resulting tracking data is very "clean" for downstream analysis, with the downside that in complex environments a great deal of human annotation is necessary.

I propose developing a state-space model that explicitly models the animal's location, orientation, and other kinematic properties to enable high-fidelity reconstruction of the animal's position. I'm breaking this down into three components:

- **Phase 1** : sequential state estimation – develop various models for estimating the animal's position x_t at time t (the "filtering" problem). Implement and evaluate those models across existing data sets.
- **Phase 2** : smoothing – higher quality results should be possible exploiting data from $1..T$ to estimate x_t. Smoothing algorithms are more suited for offline analysis, but are generally more complex and have higher memory requirements.
- **Phase 3** : online / real-time tracking – implement the filtering algorithm with constant-memory and constant-runtime performance in a manner suitable for use in real-time applications, and optionally extend with more real-time sensors (such as accelerometers and gyroscopes) for final temporal accuracy.

The existing diode-tracking system has several challenges which we will account for, and will produce an exciting model:

- **Occlusion** In many frames, it is hard to see both LEDs. They are differentiated by size, and when one is occluded it can be confusing to determine whether you are tracking the large or small one.
- **Complex behavior** The animal will rear in various contexts, which means that the LED's motion is not simply in the $x - y$ plane.
- **Reflection** The LEDs are quite bright relative to the background, and will sometimes reflect off nearby surfaces
- **Correlations** A great deal of short-term occlusion comes from the wire bundle emerging from the head stage. Due to the geometry of the setup, then, occlusion will be strongly correlated with position.
- **Diversity of experimental environment** Rodents are run in many different environments, with different geometries and different occupancy profiles.
- **Long-running** Experiments can be many hours long, and at a framerate of 60 Hz that means over $200k$ points per hour. In these regimes, SMC methods can suffer from pathologies like particle depletion.
- **Complex environmental motion** Often, other things in the scene can be moving, ranging from parts of the experiment (for reward delivery or track reconfiguration) to simply the experimenter.

3 Methodology

Sequential Monte Carlo methods, beginning with the rise of Sequential-Importance-Resampling in 1993 [7], have become the go-to method for nonlinear time series state estimation problems. They have seen a great deal of success in problems around tracking and navigation, especially in the areas of simultaneous localization and mapping [11], where the goal of a robot is to explore a novel space while simultaneously building up a map of that space.

Recent advances hint at the possibility of these methods becoming substantially more powerful for the average experimenter. This includes the incorporation of various types of Markov-chain Monte Carlo into SMC [5] and vice-versa [2], automatic decomposition of state spaces for efficient inference [1], and the development of exact approximations for previously-analytic operations like Rao-Blackwellization [9]. Over the course of this project I want to become an expert in these contemporary methods, reimplementing many of them from recent papers to evaluate their suitability in real-world tracking problems.

My other love, nonparametric Bayesian methods, have seen increasing use in time series analysis [4]. While outside of the scope of this project, the hope is that the generative state-space view encouraged by SMC will serve as a natural bridge to the more familiar ground of np-Bayes.

4 Data

The input and evaluation data will be existing tracking datasets provided by the Frank Lab. These consist of a video of the animal's behavior along with the human-annotated tracking data. The more of these I can obtain, the better, as they will give a better indication of the variety of animal behaviors that can be expected. I will curate these data into a collection of training sets for algorithmic evaluation.

5 Tools

I plan to experiment, write, and deliver everything in Python. Python is an easy-to-use interpreted programming language that has seen widespread adoption by the same scientific community that has traditionally relied on MATLAB for experimental data analysis. The addition of various excellent libraries (Including numpy [13] for matrix-vector operations, SciPy [10] for scientific functions, and Matplotlib [8] for plotting) have made it an excellent matlab replacement. When Python's interpreted speed becomes a bottleneck, it is now trivial to implement c-level functions using Cython[3].

The rise of ubiquitous cloud computing infrastructure has made it possible for researchers to easily exploit computing resources across thousands of machines. The per-cpu-core-hour pricing model results in 100 hours of computing time costing several dollars, and the parallelization and elasticity afforded allow for those 100 hours to be completed in one hour of wall-clock time. I plan on using Pi-cloud's excellent services [12] to test algorithm changes across multiple datasets simultaneously.

6 Deliverables

All source code will be delivered in a git repository served on Github [6], as well as documentation, pseudocode, and results. I will write up the results in journal format and would like to ultimately submit it to one of the IEEE journals on tracking or time-series analytics.

The tracking code program will work similar to the existing program, where an input mpeg video along with timing synchronization information will yield a file consisting of position data. It's likely that the output file will condition additional information not provided by the current tracking software, including head direction and a confidence bound around the tracking estimate.

7 Conclusion

The expected duration of the project is between four and six weeks, with weekly meetings at the Frank Lab to review progress. Total expenses for computing resources are estimated at less than $1000. Should this project succeed, it will hopefully save researchers many hours of hand-labeling video and provide higher-quality spatial data for subsequent analysis. The realtime potential will hopefully contribute to current experiments at online experiment design.

References

[1] Mohamed Osama Ahmed et al. "Decentralized, Adaptive, Look-Ahead Particle Filtering". In: i (Mar. 2012), p. 16. arXiv:1203.2394. URL: `http://adsabs.harvard.edu/abs/2012arXiv1203.2394Ohttp://arxiv.org/abs/1203.2394`.

[2] Christophe Andrieu, Arnaud Doucet, and Roman Holenstein. "Particle Markov chain Monte Carlo methods". In: *Journal of the Royal Statistical Society: Series B (Statistical Methodology)* 72.3 (June 2010), pp. 269–342. ISSN: 13697412. DOI: `10.1111/j.1467-9868.2009.00736.x`. URL: `http://doi.wiley.com/10.1111/j.1467-9868.2009.00736.x`.

[3] Stefan Behnel et al. "Cython: The Best of Both Worlds". In: *Computing in Science & Engineering* 13.2 (Mar. 2011), pp. 31–39. ISSN: 1521-9615. DOI: `10.1109/MCSE.2010.118`. URL: `http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=5582062`.

[4] Emily Fox. "Bayesian Nonparametric Learning of Complex Dynamical Phenomena". PhD. Massachusetts Institute of Technology, 2009. URL: `http://people.csail.mit.edu/fisher/publications/theses/Fox\_PhDThesis09.pdf`.

[5] WR Gilks and C Berzuini. "Following a moving target - Monte Carlo inference for dynamic Bayesian models". In: *Journal of the Royal Statistical Society: ...* 63 (2001), pp. 127–146. URL: `http://onlinelibrary.wiley.com/doi/10.1111/1467-9868.00280/abstract`.

[6] *Github*. URL: `http://www.github.com`.

[7] NJ Gordon, DJ Salmond, and AFM Smith. "Novel approach to nonlinear/non-Gaussian Bayesian state estimation". In: *IEE Proceedings F: Radar and Signal Processing* 140.2 (1993), pp. 107–113. URL: `http://ieeexplore.ieee.org/xpls/abs\_all.jsp?arnumber=210672`.

[8] John D. Hunter. "Matplotlib: A 2D Graphics Environment". In: *Computing in Science & Engineering* 9.3 (2007), pp. 90–95. ISSN: 1521-9615. DOI: `10.1109/MCSE.2007.55`. URL: `http://www.matplotlib.org`.

[9] Adam M Johansen, Nick Whiteley, and Arnaud Doucet. "Exact Approximation of Rao-Blackwellised Particle Filters". In: *Proceedings of 16th IFAC Symposium on Systems Identification*. Brussels, 2012.

[10] Eric Jones et al. *SciPy: Open Source Scientific Tools for Python*. URL: `http://www.scipy.org`.

[11] Michael Montemerlo et al. "FastSLAM: A Factored Solution to the Simultaneous Localization and Mapping Problem". In: *Proceedings of the AAAI National Conference on Artificial Intelligence*. AAAI, 2002, pp. 593–598. ISBN: 978-3540463993.

[12] *PiCloud : Cloud Computing Simplified*. URL: `http://www.picloud.com`.

[13] Stefan van der Walt, S Chris Colbert, and Gael Varoquaux. "The NumPy Array: A Structure for Efficient Numerical Computation". In: *Computing in Science & Engineering* 13.2 (Mar. 2011), pp. 22–30. ISSN: 1521-9615. DOI: `10.1109/MCSE.2011.37`. URL: `http://www.numpy.org`.