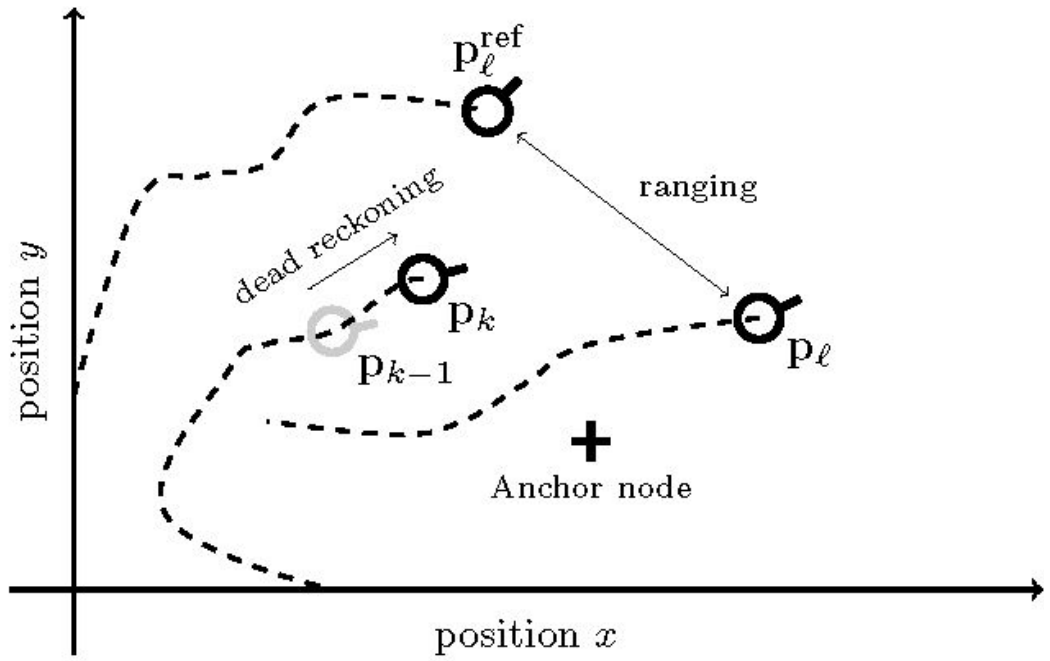

Understanding animal behaviour and location in the landscape using animal-borne technologies

16.03.2016

David McCormick & Thomas Roberts
COSC591 / COSC320
University New England
Armidale, Australia

Research Question / Problem

Currently, research staff utilize a command line library (animalTrack) to help interpret sensor data, however the workflow is cumbersome and requires in depth statistical knowledge to be able to leverage the library.

As a response to this, development of a user interface to upload data from sensors to process data for further analysis is required. Part of this task will be to develop a path integration algorithm for tracking animal trajectories based on accelerometer and magnetometer signals.

To assist with this study, a system must be designed to simplify the current workflow of processing data, as well as to provide some new predictive features, based upon dead reckoning, and markov models.

The purpose of the study is to come up with an algorithm that could be used within the sensor or as part of an online system that maps the path an animal and based on the variables at that point in time, predict where it will move to next.

Motivation

The overall motivation for this system is to provide the non-technical end users with an integrated, easy to use software package that will allow them to conduct research upon large amounts of data obtained by tracking sensors placed upon livestock.

The data obtained by the sensors is not usable in its raw form, as it requires cleaning, validation, calibration and transformation into a format that can be statistically analysed. Most end users for this information are not familiar with R, or the other related packages that are used to transform the data, and so there is motivation to simplify their workflow using an all in one package.

There is also motivation to enhance the researcher's ability to interpret the data, by developing and implementing a dead reckoning algorithm, which calculates past, present or future positioning, based upon a set of sensor data. Time permitting, this could also be augmented with a Hidden Markov Model implementation which would provide more predictive capabilities, to analyse an animal's state (Eating, sleeping) while static.

Goals

1. Develop UI to streamline user workflow
2. Implement High Pass and Low Pass Filter to reduce noise from signal.
3. Develop dead reckoning algorithm
4. Develop hidden markov model

Defining Success

Successful implementation of the system will be assessed by the user's experience while operating the GUI, whether all of the expected export functionality is present, and if the data is both transformed and filtered correctly during usage. The system should also correctly output new data files to the user's specification, for further analysis.

Proposal

We propose that to solve this problem we build a native application leveraging the Python programming language. The scope of this project will include building a bespoke Graphical User Interface (GUI) as well as a back-end that allows the user to complete the workflow seen in Figure 1.1.

Figure 1.1

Python has been chosen as the development language as it provides the best combination of R integration and machine portability. R integration is a key goal, as it allows us to leverage existing codebases familiar to research staff, to do tasks such as data filtering and calibration. This allows us to reduce repeated work, and focus upon the larger research problem, it also allows the software to be extensible after the project team has completed its work; if research staff want to improve or refine the underlying R

algorithms, provided they do not change the interface, the software will continue to operate as expected.

By utilising Python, we can simplify the deployment process, removing complexities related to operating system portability, while still retaining a fast and powerful development environment. This ties into our choice of user interface (UI) framework, QT, and its Python binding, PyQT, which will allow us to develop a scalable and universal UI.

Stretch Goals

1. Hidden Markov Model to predict state of animal whilst in stay area
   A Hidden Markov Model (HMM) could enhance researchers understanding of animal state. HMM can assign a probability to a series of state changes, or more simply predict the next state given the current state.
2. HTML Front end and associated web server
   A web interface could further simplify the usage of the software, allowing for a single installation of the system, including dependencies, to be used in a distributed manner. This would reduce setup overhead for all new users, as well as providing an easy update mechanism, allowing for future modifications.

Requirements

1. UI can ingest files (Note: Initially, CSV format, however, if resources allow, the team will endeavour to build a plugin architecture to allow it to be extensible.)
2. Backend can call through to R to allow for extensible work and update of research staff's underlying R source code.
3. Portable architecture so as to not require recompilation for different operating systems so as to future proof the software.
4. UI is clear and obvious to non-technical users.

Milestones

I. Initial research complete

15/3/16 - After some initial discussion between David, Gregory, Robin and Thomas the group was able to identify a project, the specific requirements of it, and the knowledge required to complete the project (understanding of the various technical stacks, and statistical methods).

II. Proposal

30/3/16 - This proposal will outline the specifications for the project and its deliverables. It is a prerequisite that we complete the initial research before undertaking the proposal milestone.

III. Backend

10/5/16 - The dead reckoning algorithm, in addition to the high and low pass filters. Integration with the existing animalTrack library, and the ability to plot graphs that the UI can use. The backend should be encapsulated in an interface that the UI can call.

IV. Frontend

10/5/16 - A simple, self explanatory user interface that allows the user to interface with the backend API. It should be platform independent, and have room for potential future expansions.

V. Presentation

20/5/16 - Video Recording and walk through of completed software and documentation.

VI. Documentation

20/5/16 - Documentation - there are two components of the documentation, the first is the API documentation, this has been achieved via sphinx, leveraging Python's docstrings. The team will also write a short user manual for the UI.

VII. Stretch Goal

20/5/16 - Both team members have assigned themselves stretch goals. Davids is to implement a Hidden Markov Model and Thomas' is to implement a web front end and web service.

Initial Research

David's initial research consisted of looking into the statistical methods required to perform Dead Reckoning and also Hidden Markov Models. David also familiarized himself with R so as he could effectively utilize Robins provided animalTrack R library.

David undertook some follow up research to understand which was the most appropriate technology stack given the time constraints (roughly 100 hours per student team member, over the course of semester) and other project requirements. David also began reading “Learning statistics through R” - http://health.adelaide.edu.au/psychology/ccs/teaching/lsr/ to improve his knowledge of statistics specifically utilizing R. He also undertook some Machine Learning from Coursera classes to improve his knowledge of machine learning that will be used for the Hidden markov model component of the project.

Thomas’ initial research mostly concerned familiarisation with the chosen technology stack, with the most focus on the chosen UI framework, PyQT. Thomas also gained a shallow understanding of the statistical methods likely to be required for the project, as to gain a familiarity with the kinds of information the frontend is required to display.

Design Choices - UI

Tech Stack

Python (3.4.*) using PyQT

Design Choices - Backend

Tech Stack

Python (3.4.*) using scipi, rpy2.

R

Experiments

David wanted to find the most appropriate tech stack to allow the team to produce the greatest amount of value. David was most familiar with Python, but also wanted to look into using Java for its higher processing power, and the other option was to try and use purely R, which neither David or Thomas had much experience with.

Given Robin already having an extensive R library, and given that due to this he was familiar with how R functioned, the team felt that it was important to continue using R, despite the team not having a large amount of experience with it. David undertook two sets of tutorials to improve his understanding of R - some generic youtube tutorials, and also the Datacamp Titanic Kaggle (https://www.kaggle.com/c/titanic/details/new-getting-started-with-r) competition

tutorial. With this set of tutorials complete, David felt he had enough of an understand of R's syntax to leverage Robins existing library.

Implementation

https://github.com/davemccormick/pyAnimalTrack/wiki

http://davemccormick.github.io/pyAnimalTrack/

https://github.com/davemccormick/pyAnimalTrack

References

https://www.kaggle.com/c/titanic/details/new-getting-started-with-r

http://health.adelaide.edu.au/psychology/ccs/teaching/lsr/

https://github.com/robbykraft/AnimalTrack

https://wiki.python.org/moin/PyQt

https://pypi.python.org/pypi/rpy2