Decentralised Timescales for Large Scale Sensor Networks

Matthew Ridley, Sharon Ong and Salah Sukkarieh
ARC Centre of Excellence in Autonomous Systems (CAS)
The University of Sydney, Australia
m.ridley,s.ong,s.sukkarieh@cas.edu.au

Abstract

The availability of a reliable global time scale has typically been taken for granted in sensor network applications. Such a timescale is typically supplied by GPS receivers or network synchronisation. A truly decentralised system would operate without dependence on a global timescale. This paper proposes a method to extend decentralised data fusion algorithms to include multiple independent timescales at each node in a sensor network. Examples of remote clock observation, and information traversing the network are provided. A designer may then use the analysis procedure to design a sensor network to a specific scale, or to determine an existing sensor network's ultimate range of operation.

1 Introduction

Decentralised Data Fusion (DDF) is the process of estimating states of features by way of combining information from multiple sensors connected to processing nodes that are interconnected in such a way as to not depend on a single centralised resource. DDF offers advantages such as scalability, robustness and survivability in environments that cause high sensor node mortality. Figure 1 depicts the internal structure of a decentralised data fusion node.

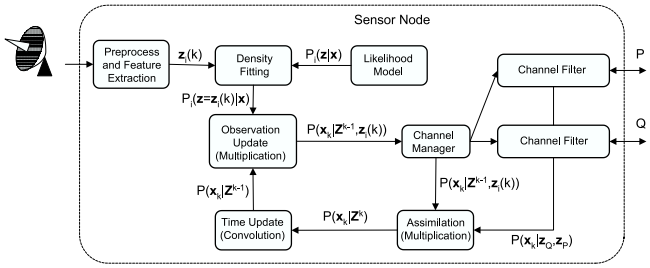

Figure 1: Decentralised Data Fusion Node

However, DDF algorithms still typically rely on a centralised resource: a single global timescale. Observations are registered against this timescale which is also used to predict the states being estimated. The aim of this paper is to demonstrate a means to remove this centralised assumption. This in turn will improve the robustness and versatility of decentralised systems.

Desirable properties of a sensor node would obviously be small size and low power consumption. The Intel Xscale architecture (otherwise known as the StrongARM) provides modest processing performance for low power consumption. It is also desirable that DDF sensor nodes have significant computational capacity as the overall goal is to remove any dependence on a centralised facility. Some sensor node hardware (such as the Berkley Mote [Hill and Culler, 2001]) have limited computing capacity and are not suited for use in DDF applications.

Figure 2: Typical sensor platforms. Stayton (left) and Stargate (right)

The Stayton board shown in Figure 2 was used for some of the experiments to be described later.

This paper first describes the problem as outlined above and describes other non centralised solutions for similar time synchronisation problems. A method for observing and modelling clocks of remote nodes is provided. This is followed by

an architecture which allows for the incorporation of temporal information into feature state estimators. Finally an analysis of these algorithms and a means for determining the boundaries of useful operation for such a network is provided.

2 Sources of Time Errors

In considering the variations of timescales on different sensor nodes, it is only prudent that other sources of timing errors of any observations such as those caused by the sensors themselves, are identified.

2.1 Timestamp Errors

Up till now, all operations with filtering and tracking applications gave little consideration to the accuracy of the timestamps assigned to sensor observations. Many typical "off the shelf" sensors operate over common peripheral interfaces that are likely to introduce unknown latencies into the system. Research into the effects of uncertainty in the time stamps of observations has been investigated by [Rubanenko, 2002],[Yaz and Ray, 1996] and [Basin *et al.*, 1999]. These approaches have typically centred on accounting for timestamp error rather than maintaining it as a state.

For example, consider a "smart" sensor utilising a serial interface to transmit data at modest rate. There is an unknown variable delay due to sensor processing, and a delay due to transmission through the serial link to the host. Serial transmission is likely to take hundreds of milliseconds to complete. The host must then assign a timestamp to the sensor reading which introduces further unknown delays, the duration of which will depend on the serial driver architecture of the system used.

A platform moving at over 100km/hr may move several metres during serial transmission alone, plus an additional unknown amount during sensor acquisition/processing and time stamping. Localisation with sub metre accuracy whilst using such hardware will be challenging indeed. Proper characterisation of the acquisition time and latency of all sensors seems to be mandatory for any real application.

2.2 Clock Phase Error

For a sensor network application, the sensor nodes are likely to be powered up at random. As a consequence, all their clocks will have different values if queried at any given time. This difference is also known as the phase error. Without any form of synchronisation operation, (as will be discussed later) the integration of these sensor nodes within the network will be difficult as there will be no way to determine the temporal location of events that may be observed by them.

2.3 Clock Frequency Error

At the heart of most clocks on computing equipment is a quartz crystal oscillator. While these are more stable than most other forms of oscillators, they do vary with temperature, pressure and other subtle influences. The precision of manufacture is also not without question. Each crystal oscillates at a slightly different frequency, causing the phase error of each individual clock to drift over time. Determining, and correcting for, the phase error alone, is therefore not sufficient.

3 Clocks and Clock Modelling

At the 13th General Conference of Weights and Measures [CGP, 1967] the SI definition of a second became: "The duration of 9,192,631,770 periods of the radiation corresponding to the transition between the two hyperfine levels of the ground state of the cesium atom 133". However, the use of caesium time sources is prohibitively expensive, as well as the physical dimensions of such units being unsuitable for compact, low power sensor nodes.

3.1 Typical Clock Hardware

Typically PC hardware contains a real time clock in the form of a low precision, but relatively high accuracy, battery backed device used to maintain time while the computer is unpowered. Also contained within a PC is standard counter-timer circuitry, which is typically used to provide regular timer tick interrupts for house keeping and process scheduling (typically 100Hz). In addition to this, some processors contain register access to a high precision counter with nanosecond resolution.

Low cost sensor network hardware which is suitable for large scale networks, does not typically contain such dedicated real time clock hardware. Such devices are often only able to make use of internal processor counters to provide a software clock started at power up. This is the case for the Stayton board.

3.2 Behaviour of Quartz Crystal Oscillators

A typical quartz crystal oscillator has an accuracy of 10^{-4} to 10^{-6} indicating a potential clock drift of between 0.1 to 10 seconds per day or 1 to 100 microseconds per second. However quartz crystals have a stability in the order of 10^{-9} to 10^{-11} indicating a drift of a few seconds per year. If it where possible to observe a crystal oscillator against a precision timesource for an adequate length of time, it is possible to characterise its behaviour. Clocks that have been calibrated in software using this method are known as "Disciplined" Clocks. However performing such calibration procedures impacts adversely the cost of producing low cost, large scale sensor networks.

3.3 Kalman Filter Based Models

Work by [Galleani and Tavella, 2003] provides a recent discussion on the suitability of using the Kalman filter to model a timesource. Other works by [Greenhall, 2001; 2003] provides further discussion on providing robust timescales using Kalman filters and multiple timesources. Current DDF algorithms typically model states with linear Gaussian models,

such as the Information Filter. This makes the task of incorporating temporal information into DDF less complicated.

4 Time Synchronisation

Given that this challenge may be viewed as a time synchronisation problem, a brief review of similar such concepts and their solutions will be provided.

4.1 Overview

The concept of temporal ordering events was first introduced by [Lamport, 1978]. Lamport's primary outcome was that it was possible to bound the time interval of common event occurring within a distributed system. This concept was then applied to time synchronisation in distributed systems in later work.

The most common approaches to time synchronisation in sensor networks employ a centralised master slave architecture, not too dissimilar to NTP developed by Mills[Mills, 1991; 1992; Mills *et al.*, 1997]. Some of these approaches, such as those described by other groups[S.Ganeriwal *et al.*, 2003; Dolev *et al.*, 1995], claim to be scalable or even decentralised. However, they all attempt to synchronise the clocks of all nodes to a single global timescale derived from a "master" node. This results in a system that is not truly decentralised and employ a variety of methods to improve robustness.

An exception to these centralised approaches is presented in work by Römer and Elson [Römer, 2001; Elson and Römer, 2003], where global timescales may be replaced by local timescales and maintenance of the difference between interconnected nodes. This approach is robust, and does not require elections of new "master" time sources. It also allows for dynamic changes in sensor networks, as complete subnetworks may join to the main network and share their information, without having to align previously stored information to the new timescale. However, this work has not been optimised for use in DDF applications.

4.2 Network Time Protocol

With the increased of use of networked computers in the late 1980's and early 1990's for file distribution and email, it became increasingly apparent that accurate clocks on all networked computers were required to ensure data integrity of services. In this application it was essential that all computers on a local network were aligned to a common timescale.

The Network Time Protocol (NTP) utilised existing networking protocols (TCP,UDP), thereby enabling entire clusters of computers to have their clocks rapidly synchronised and disciplined to a precision of a few tens of milliseconds. This approach generally relies on a single precision source such as a GPS receiver or cesium standard.

4.3 GPS

The Global Positioning System requires the use of atomic clocks onboard all orbiting satellites. As a consequence, GPS receivers provide a low cost time source with less than 20ns resolution. GPS receivers are often used as sources for NTP servers for this very reason.

In the context of decentralised data fusion, the ANSER project, described by [Sukkarieh *et al.*, 2001], uses GPS receivers to allow all platforms to obtain a single global timescale. In could be argued that the use of GPS provides a decentralised solution, as the time is derived from multiple satelites.

However GPS is not general enough to be considered for all applications. Outdoor operating environments with steep terrain or those underwater or underground render the entire system inoperable. These issues are the primary motivation for this research.

4.4 Other Schemes

Some other methods of providing clock synchronisation in sensor networks are:

- Broadcast - An approach by which a timestamp is broadcast by a beacon. No consideration is given to the transmission time (it is assumed to be constant). It is commonly used by cellular telephone technology.
- Master election - This is similar to NTP, however the master timesource is elected by a voting process. In the event of failure of the master timesource, a new master is elected.
- Time averaging - A group of sensors nodes average all their clocks to produce a suitable timescale with which to synchronise against. This approach has some merit as a the resultant global timescale is more a stable timescale than that which can be derived from a single oscillator.

All the above mentioned methods suffer from inadequate scalability or robustness. Isolated nodes suffer from additional problems, as information initially collected against their unsynchronised timescale must be converted to a new timescale when synchronisation commences. Any dynamic changes in network topology may also induce further variations in timescales, as nodes may vary their logical distance from the "master" time sources. In addition, communication bandwidth consumed by beacons and voting processes consume valuable bandwidth that may otherwise be utilised for other purposes.

In conclusion, time synchronisation by its very nature is not decentralised and a different approach is required to properly satisfy such a requirement.

5 Proposed solution

Goal: To provide a robust method of propagating information in a sensor network with multiple different timescales.

The approach described in this paper has some similarities to methods proposed by Elson and Römer and will be outlined as follows.

- Observe and model the phase and frequency difference of remote clocks.
- Account for the error in the temporal location of local observations.
- Account for temporal errors as information propagates throughout a sensor network.

All states, both feature and temporal, will be modelled as linear with Gaussian noise. As such, the discrete time model describing how the state $\mathbf{x}$ evolves from time $k-1$ to time k is:

$$\mathbf{x}(k) = \mathbf{F}(k)\mathbf{x}(k-1) + \mathbf{B}(k)\mathbf{u}(k) + \mathbf{G}(k)\mathbf{w}(k) \qquad (1)$$

where $\mathbf{F}(k)$, $\mathbf{B}(k)$, $\mathbf{G}(k)$ are the state, control and noise transition matrices, $\mathbf{u}(k)$ is the control input and $\mathbf{w}(k)$ is the noise input.

6 Observing Remote Clocks

In order to develop models of the clocks of nodes connected via wireless ethernet, a means for observing the remote clock via network packets needs to be employed.

6.1 Observation by Ping

It is possible to observe a remote clock by "pinging" or "bouncing" packets off remote nodes. Such an approach is employed by the NTP protocol [Mills, 1992]. A visual depiction of the observation of a remote clock by packets traversing two nodes is provided by Figure 3. Four timestamps from the local clocks of the interacting nodes are required. The transmit time (T0, T2) and receive time (T1,T3) at each node.

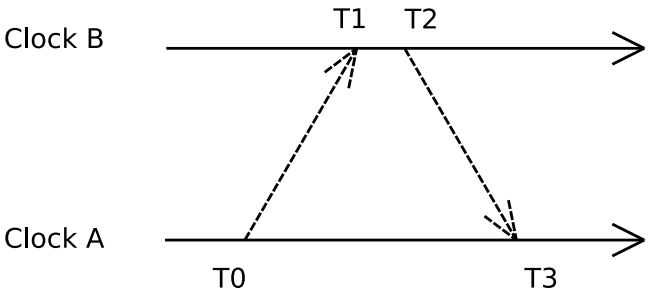

Figure 3: Ping packet traversal

The results from observing the remote clock of the Stayton Board are shown in terms of the round trip time of packets in Figure 4. This is typically characterised by a floor, indicating the minimum possible round trip time.

Figure 5 contains the error of all the observations made after correcting for the phase and frequency error of the remote clock.

The resulting PDF of the observations made is provided in Figure 6. Visual comparison to the mean and variance of the samples are provided by a Gaussian. It follows that modelling

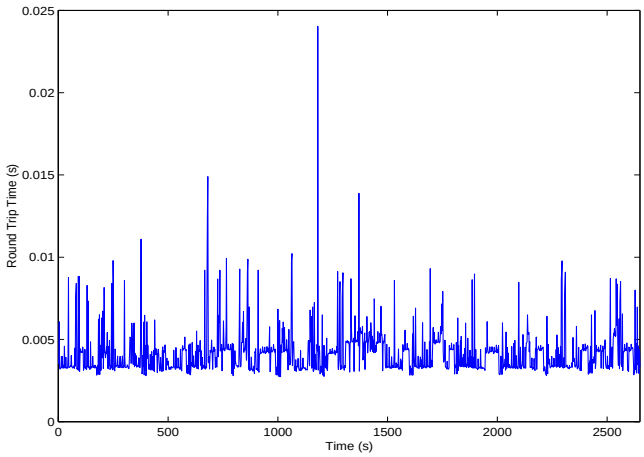

Figure 4: Round trip time of timing packets with the Stayton board

observed round trip time with Gaussian noise is a reasonable assumption.

By removing delays in the observed node, the effective round trip time becomes

$$T_{rtt} = T3 - T0 - (T2 - T1) \qquad (2)$$

Constructing an observation from this information is achieved by assuming that the remote time is T1 when half the round trip time has expired. The local time becomes:

$$T_l = T0 + T_{rtt}/2 \qquad (3)$$

The observed phase shift can now be awarded as:

$$\tau = T1 - T_l \qquad (4)$$

with a standard deviation of the same order as the round trip time.

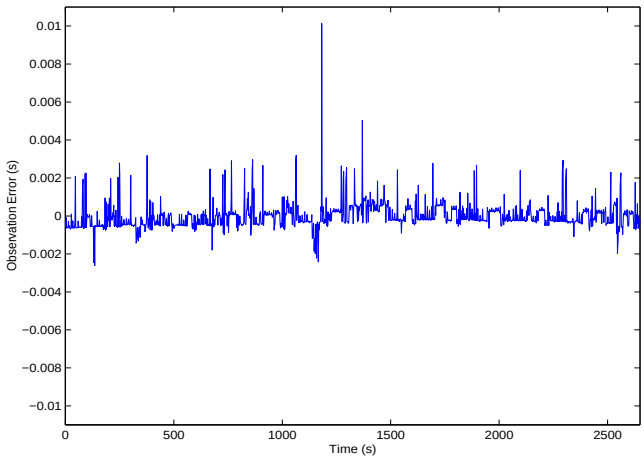

Figure 5: Error of observed time

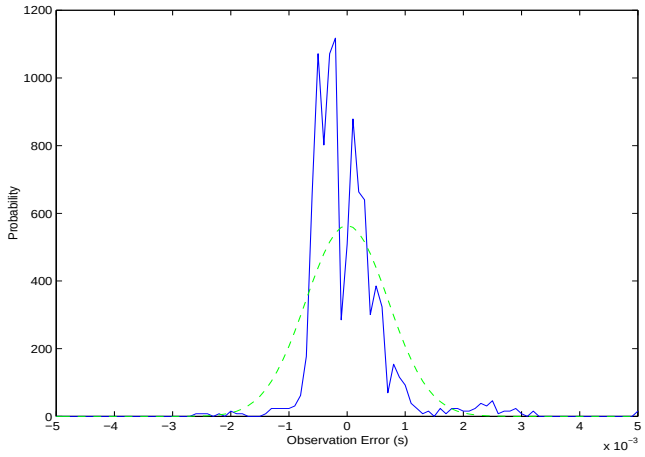

Figure 6: PDF of a timing observation

6.2 Remote Clock Model

Due to machine precision it is advantageous to model the phase and frequency differences instead of their absolute values. The frequency difference will be of the order 10^{-6}. Storing the frequency instead would mean storing $1 + \dot{\tau}$, which uses a large number of significant figures and may introduce machine precision errors.

The state representing the clock of each node shall be represented in the form:

$$\mathbf{x} = \begin{bmatrix} \tau \\ \dot{\tau} \end{bmatrix} \tag{5}$$

where τ is the difference between the remote clock and the local clock, and $\dot{\tau}$ is the rate of change in this difference. Clocks of remote nodes are therefore modelled by a constant velocity model. Clocks typically drift with temperature and other less influential effects. This requires that the time difference be regularly "observed". It should be noted that the model will be only locally valid and cannot be used for large predictions into the past or future.

$$\mathbf{H} = \begin{bmatrix} 1 & 0 \end{bmatrix} \tag{6}$$

With the corresponding observation $\mathbf{z}$

$$\mathbf{z} = \begin{bmatrix} \tau \end{bmatrix} \tag{7}$$

provided by:

$$\mathbf{z} = \mathbf{H}\mathbf{x} \tag{8}$$

The state transition matrix is:

$$\mathbf{F} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix} \tag{9}$$

The noise transition matrix is:

$$\mathbf{G} = \begin{bmatrix} \Delta t & \Delta t^2/2 \\ 0 & \Delta t \end{bmatrix} \tag{10}$$

with noise covariance $\mathbf{Q}$

$$\mathbf{Q} = \begin{bmatrix} \sigma_1^2 & 0 \\ 0 & \sigma_2^2 \end{bmatrix} \tag{11}$$

where σ_1^2 is the frequency noise variance and σ_2^2 is the random walk frequency noise variance. There is no control input for this model.

This provides a continuously updated model of another node's clock, containing the two states of interest. These are the position offset (phase difference) and velocity offset (frequency difference). These states will both gradually shift over time as this type of model will effectively only be able provide a window where the model is functional or useful. Excessive retrodiction or prediction is obviously not advisable.

7 Local Operations

Independent of remote clock observations, local states estimates must now contain the temporal uncertainty of feature information. The local feature estimates utilise an undisciplined model for time as there is now no possibility of comparing this clock with any global timescale.

7.1 Assumptions

It is never possible to determine the actual error of the local clock. Attempting to maintain the absolute error of the local clock, even if relative to the time the clock was started, is of little use as it is assumed that the global timescale will *never* be known nor will it ever be possible to observe it at a later time.

Feature state variables being estimated are considered to be independent of time, and as such will be uncorrelated. In practice, this would not be the case, as feature velocities will become correlated with time if they are not being directly observed. This is because the local clock is used to determine when filter predictions occur. The assumption is that the motion model error will dominate over any velocity-time correlation.

The temporal information of a feature will be treated just like any other state. Its error will grow without bound until it is observed again. The model used for temporal information is undisciplined and, as such, the error will grow at a high rate typical of the manufacturing tolerances of the oscillator. This error is relative to the unobservable global timescale.

Even though the temporal state is completely independent of the feature states and could operate in a separate filter, it is included with the other features to avoid the resulting architecture requiring duplicate channel filters. It also allows for a more exacting model including all correlations to be incorporated at a later date.

7.2 Local Models

The following example demonstrates the tracking of a particle in a single dimension with temporal information as a state. The particle is modelled with a constant velocity model.

The state vector will be:

$$\mathbf{x} = \begin{bmatrix} x \\ \dot{x} \\ t \end{bmatrix} \tag{12}$$

Both temporal and spatial information is now contained within a feature estimate along with its accompanying covariance matrix $\mathbf{P}$. Since the absolute error of a local clock of any node is not observable, there is no means to ascertain the frequency error in the local oscillator of a particular node. Subsequently the clock frequency is not included in the state vector of features.

For a sensor providing position information only, the observation matrix will be:

$$\mathbf{H} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{13}$$

With the corresponding observation

$$\mathbf{z} = \begin{bmatrix} x \\ t \end{bmatrix} \tag{14}$$

provided by:

$$\mathbf{z} = \mathbf{H}\mathbf{x} \tag{15}$$

The state transition matrix becomes:

$$\mathbf{F} = \begin{bmatrix} 1 & \Delta t & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{16}$$

control transition matrix is:

$$\mathbf{B} = \begin{bmatrix} 0 \\ 0 \\ \Delta t \end{bmatrix} \tag{17}$$

and the control input (oscillator frequency)

$$\mathbf{u} = \begin{bmatrix} f \end{bmatrix} \tag{18}$$

The clock frequency f has an expected value of 1.0 with a standard deviation σ_c of 10^{-4}, and is the sole control input to the system.

The noise transition matrix is:

$$\mathbf{G} = \begin{bmatrix} \Delta t^2/2 & 0 \\ \Delta t & 0 \\ 0 & \Delta t \end{bmatrix} \tag{19}$$

with noise covariance $\mathbf{Q}$

$$\mathbf{Q} = \begin{bmatrix} \sigma_p^2 & 0 \\ 0 & \sigma_c^2 \end{bmatrix} \tag{20}$$

where σ_p^2 is feature model noise variance and σ_c^2 is the clock frequency noise variance.

8 Channel Link Operations

For each link, a channel filter maintains information common to both nodes to avoid any information being counted twice "data incest". Since time is now one of the state variables, the operation of the channel filter becomes more complicated. Since each node operates on a different timescale, normal channel operations may not proceed until there has been a timescale conversion.

The timescale conversion simply involves subtracting the phase difference maintained by the clock model of a particular connected node from the incoming information. The noise of this model must also be included in the conversion process. Before this process is performed however, the feature information must be first predicted using the remote timescale to the time corresponding to the timestamp of the local estimate.

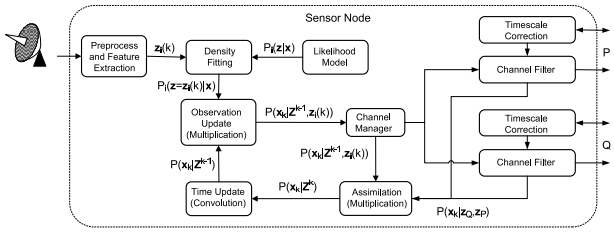

Figure 7: DDF Node with Decentralised Timescale

Figure 7 contains the additional functional block required to allow DDF to function with decentralised timescales. The timescale correction operation simply intercepts incoming estimates to correct temporal information. The same function block also performs the remote clock observations necessary to achieve this.

9 Evaluation

The behaviour of information traversing a sensor network is now subject to two sources of errors. The feature model causes the the error to grow during the time that elapses during communication and assimilation at a node. Upon receiving information a node performs the timescale conversion that results in the injection of more error in the temporal state. For the purposes of evaluation, each node is assumed to make sufficient observations of connected node clocks to reach a previously determined steady state error.

A decentralised data fusion system will typically employ a communication management system within the channel manager that makes decisions about what information is to be communicated. These decisions rely on an information measure of the features. In this case we use Entropy, defined for

a multivariate Gaussian on state vector $\mathbf{x}$ with covariance $\mathbf{P}$ to be:

$$H(\mathbf{x}) = \frac{1}{2} log\left[(2\pi e)^n |\mathbf{P}|\right] \quad (21)$$

The simulation is set up as follows. We have two variables of interest: the steady state error in node clock estimates, and the feature model noise. This information is communicated through the network until the Entropy reaches a threshold (set to zero). Figure 8 shows this result for a fixed communication latency of 0.1s and initial covariance matrix of:

$$\mathbf{P} = \begin{bmatrix} 5 & 0.2 & 0 \\ 0.2 & 5 & 0 \\ 0 & 0 & 10^{-8} \end{bmatrix} \quad (22)$$

The initial feature covariance is selected to be indicative of typical estimates a node may transmit. It is also assumed no further observations are made of this particular feature, as this is a test of the ultimate limit of feature propagation.

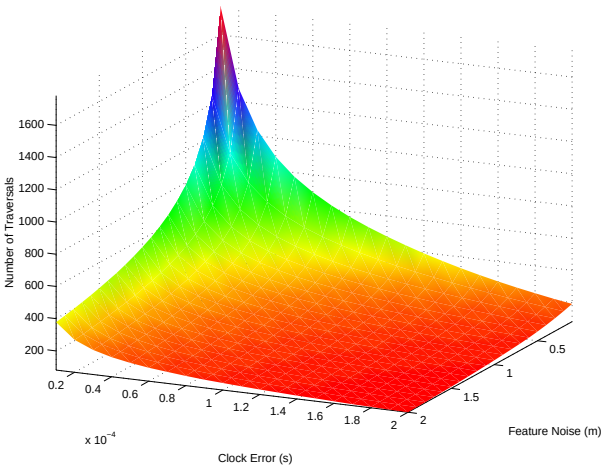

Figure 8: Traversal limits for different values of steady state error in clock estimates and feature model noise

10 Information Horizon

The traversal limit for a given decentralised data fusion network will be termed the "Information Horizon", as it is the farthest depth information may travel in a sensor network. While the information horizon may be improved by increased certainty of connected node clocks, this will increasingly consume communication bandwidth. Likewise, improved motion models, or reduced communication latency will also improve the information horizon, but again these improvements come at a cost of increased computing power or transmitter power.

A designer of a large scale decentralised sensor networks must perform a cost-benefit analysis in determining the ultimate size of a sensor network, or selection of subsystems given a fixed sized network.

11 Conclusion

Operating a decentralised sensor network with multiple independent timescales is possible provided one is willing to sacrifice all notion of a global timescale.

The factors determining the logical distance information may propagate within a decentralised sensor network with decentralised timescales are: clock model error and feature model noise. If communication latencies are large, then feature model errors will tend to dominate. If available bandwidth for remote clock estimation is limited then temporal errors will tend to dominate.

Given a particular DDF network size, it is possible to determine the required remote clock estimation accuracy, or required communication bandwidth. Conversely given previously selected hardware, the ultimate size of a sensor network may also be determined.

Acknowledgements

This work is supported in part by the ARC Centre of Excellence programme, funded by the Australian Research Council (ARC) and the New South Wales State Government.

References

[Basin *et al.*, 1999] Michael V. Basin, Mario A. Villanueva Llanes, and Irma R. Valadez Guzman. On filtering problems over observations with delays. In *IEEE Proceedings of the 38th Conference on Decision and Control*, pages 4572–4576, Dec 1999.

[CGP, 1967] *The 13th Conférence Générale des Poids et Mesures (CGPM)*, Geneva, Switzerland, October 1967.

[Dolev *et al.*, 1995] Danny Dolev, Rüdiger Reischuk, Ray Strong, and Ed Wimmers. A decentralized high performance time service architecture, 1995.

[Elson and Römer, 2003] Jeremy Elson and Kay Römer. Wireless sensor networks: a new regime for time synchronization. *ACM SIGCOMM Computer Communication Review*, 33(1):149–154, 2003.

[Galleani and Tavella, 2003] L Galleani and P Tavella. On the use of the kalman filter in timescales. *Metrologia*, 40(3):S326–S334, 2003.

[Greenhall, 2001] C. A. Greenhall. Kalman plus weights: A time scale algorithm. In *Proceedings of the 33rd PTTI Meeting*, pages 445–454, Long Beach, CA, 2001.

[Greenhall, 2003] C. A. Greenhall. Forming stable time scales from the jones-tryon kalman filter. *Metrologia*, 40:S335–S341, June 2003.

[Hill and Culler, 2001] J. Hill and D. Culler. A wireless embedded sensor architecture for system-level optimization. Technical report, 2001.

[Lamport, 1978] Leslie Lamport. Time, clocks, and the ordering of events in a distributed system. *Commun. ACM*, 21(7):558–565, 1978.

[Mills *et al.*, 1997] D.L. Mills, A. Thyagarajan, and B.C. Huffman. Internet timekeeping around the globe. In *Proc. Precision Time and Time Interval (PTTI) Applications and Planning Meeting (Long Beach CA)*, pages 365–371, December 1997.

[Mills, 1991] David L. Mills. Internet time synchronization: the network time protocol. In *IEEE Trans. Communications 39, 10*, pages 1482–1493, October 1991.

[Mills, 1992] David L. Mills. Modelling and analysis of computer network clocks. Technical Report 92-5-2, Electrical Engineering Department, University of Delaware, May 1992.

[Römer, 2001] Kay Römer. Time synchronization in ad hoc networks. In *Proceedings of the 2nd ACM international symposium on Mobile ad hoc networking and computing*, pages 173–182. ACM Press, 2001.

[Rubanenko, 2002] Ariel Rubanenko. State estimation using measurments with uncertain time-tag. Master's thesis, Israel Institute of Technology, March 2002.

[S.Ganeriwal *et al.*, 2003] S.Ganeriwal, Ram Kumar, S.Adlakha, and Mani Srivastava. Network-wide time synchronization in sensor networks(nesl technical report nesl-tr-jan- 2003). Technical report, UCLA, 2003.

[Sukkarieh *et al.*, 2001] S. Sukkarieh, B. Yelland, H. Durrant-Whyte, B. Belton, R. Dawkins, P.Riseborough, O. Stuart, J. Sutcliffe, J. Vethecan, S. Wishart, P.Gibbens, A. Goktogan, B. Grocholsky, R. Koch, E. Nettleton, J. Randle, K. Willis, and KC Wong. Decentralised data fusion using multiple uavs - the anser project. In *Proc. 3rd Int Conf. on Field Service Robotics (FSR 2001)*, pages 185–192, Helsinki, 2001.

[Yaz and Ray, 1996] Edwin Yaz and Asok Ray. Linear unbiased state estimation for random models with sensor delay. In *IEEE Proceedings of the 35th Conference on Decision and Control*, pages 47–52, Dec 1996.