

Minimising loss-induced errors in real time wireless sensing by avoiding data dependency

A. D. Young and M. J. Ling
Institute for Computing Systems Architecture
School of Informatics, University of Edinburgh
10 Crichton Street Edinburgh EH8 9AB, United Kingdom
Email: ayoun9@inf.ed.ac.uk
m.j.ling@ed.ac.uk

Abstract—The use of local processing to reduce data transmission rates, and thereby power and bandwidth requirements, is common in wireless sensor networks. Achieving the minimum possible data rate, however, is not always the optimal choice when the effects of packet loss on overall measurement error are considered. This paper presents a case study from the area of wireless inertial motion capture, in which the best distributed processing strategy is shown to be that which minimises inter-packet data dependency, rather than overall data rate. Drawing on this result and further analysis, we identify questions that should be raised during the design process in order to understand the effects of packet loss on distributed signal processing tasks, decide whether the errors are acceptable for an application, and choose appropriate techniques to mitigate them if necessary.

I. INTRODUCTION

A common theme in sensor network research is the use of local processing within the network to reduce data transmission requirements. This is motivated by the cost of data transmission relative to processing [1], in terms of both power and bandwidth. This data reduction can take many forms including straightforward compression, data aggregation, event detection, or various domain-specific techniques.

When data is transmitted over radio networks, some packet loss is practically inevitable. Research into tackling this problem generally deals with MAC and transport layer protocols which retransmit missing packets. This work is largely independent of the data being transmitted. Meanwhile, other researchers tend to focus on the specific processing requirements of their applications, stating or assuming that existing work can be employed for their communications requirements.

In realtime systems employing a transport layer may be unfeasible due to the delay and overhead incurred by retransmission protocols. Yet in body sensor networks, applications are generally demonstrated using PC based implementations of algorithms [2]–[6] without consideration of packet loss. Although the concept of moving this processing to the device is often mentioned, analyses of the benefits or implications of this approach are rarely presented.

This paper presents a case study investigating the effect of unreliable data links on the *Orient* inertial tracking system developed in our previous work [7]. We aim to show that although local processing can reduce the required data rate, maximising this reduction is not the only consideration in the

design of such an application, and that the effects of packet loss must be considered.

Section II introduces the application and processing techniques involved. Three different strategies for distributed processing are then proposed, and their bandwidth and error performance investigated through simulation. Section III then discusses these results and presents some further analysis of the issues raised, and Section IV summarises our conclusions.

II. CASE STUDY APPLICATION

The motivating application used in this study is that of tracking the posture of a human subject in real time. Traditionally, this problem has been tackled using optical triangulation to track the position of reflective markers attached to the subject. The limitations of this approach, in particular its limited tracking volume and problems with marker occlusion, have motivated the development of inertial tracking systems using sensors worn on the body [6], [8], [9]. In such systems, sensor data from triads of accelerometers, magnetometers and rate gyroscopes is used to estimate the orientation of each sensor device and thereby the complete posture of the subject.

Estimating the orientation of a device presents a significant data processing problem. The estimation process used for such devices combines two techniques. The first is based on the integration of rate gyroscope data to calculate rotation relative to an initial orientation. The second works by observing the direction of two vectors, the local acceleration due to gravity and magnetic north, and comparing them with their known directions in a global co-ordinate frame. Results from these techniques are combined using data fusion algorithms such as complementary filters [7], [10], or Kalman filters [2], [3].

For the purposes of this paper we will consider the complementary filter shown in Fig. 1, as used in the *Orient* system. The filter combines rate gyroscope integration with a low-pass filtered error correction derived from a vector observation process. The aim is to combine the best features of both estimates: gyroscope integration has a fast response but suffers from drift, whilst vector observation provides an absolute reference but must be filtered to remove noise from linear accelerations.

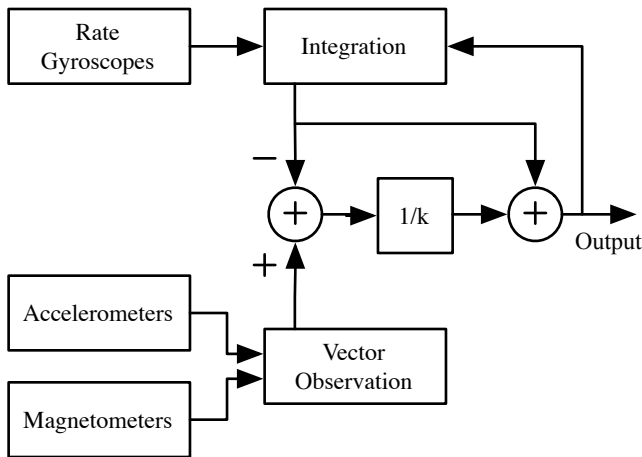


Fig. 1. Orientation Filter Block Diagram

In order to reconstruct the full posture of the tracked subject, including arms, hands, legs, feet, spine and head, fifteen sensors are required [11]. Data must be gathered from all sensors and relayed to a central point, where the posture is reconstructed by mapping the orientation of the sensors to the rotations of rigid rods representing the subject's skeleton.

To make best use of the system for interactive applications it is desirable to minimise the latency between acquiring sensor data and updating the body model. Reducing latency allows for more accurate interaction [12] and greater immersion [13] with virtual environments. Latency requirements vary depending on the exact application but are typically of the order of tens of milliseconds [14], [15].

Our experience of operating the *Orient* system indicates that packet loss is not uncommon. Typical system use involves transmitting data from a mobile user to a fixed basestation for body modelling and visualisation. Packet loss can occur due to limitations of radio transmission range, multipath fading, or interference. Often we have observed link loss for significant fractions of a second as opposed to single packet losses.

A. Data rate reduction

Most existing inertial tracking systems transmit raw sensor data to a host PC for processing [2], [3], [6], [16]. To achieve reasonable tracking, rate gyroscope data must be integrated at relatively high sample rates, typically 120-180Hz. Therefore, assuming 12-bit samples from each of the nine sensors, this method implies a data rate of $120 \times 9 \times 12 = 12960\text{bps}$.

An alternative approach is to perform orientation estimation on the individual devices, as first demonstrated by the *Orient* system. In this case, a high sample rate of 256Hz is used on the device to update the filter, but the computed orientation can be transmitted at a lower rate since human motion is typically low in frequency. For example, human gait is in the range of 4-6Hz [17]. For interactive purposes, a subsampled rate of 64Hz is preferable as this limits update latency to approximately 15ms,

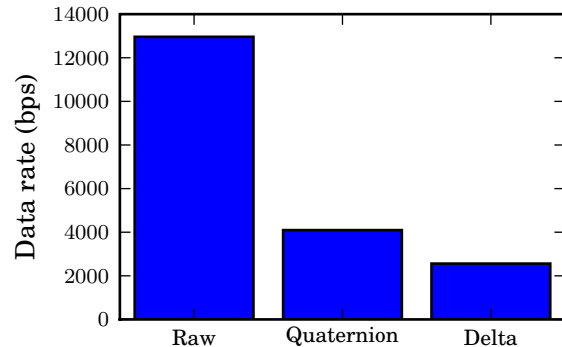


Fig. 2. Data Rate Comparison

which has been identified as a key perceptual requirement for virtual reality applications [14]. When using quaternions comprised of four 16-bit values to represent orientation this leads to a data rate of $64 \times 4 \times 16 = 4096\text{bps}$.

A greater reduction in data rate could also be achieved by compressing the resulting quaternion stream. For example, since the maximum change per time step in each quaternion component is limited, difference coding could be used. If the maximum expected rotation rate is not greater than 1200 degrees/second, the maximum rate supported by the *Orient* device's gyroscopes, only a 10-bit delta of each quaternion component need be sent, reducing the data rate to $64 \times 4 \times 10 = 2560\text{bps}$.

These reductions in transmission requirements provide a double benefit. Firstly, they reduce the energy expended by each node on data transmission. Secondly, they reduce the bandwidth required per device, and thereby increase the number of devices that can be tracked simultaneously using a given channel capacity. The *Orient* system operates in the 433MHz band, in which the bandwidth available limits the data rate 250kbps [18]. To support the 15 devices required for full-body tracking at 64Hz, local processing is therefore essential. Further reductions by compression would increase the number of devices that could be simultaneously tracked.

Fig. 2 compares the bandwidth requirements of each of the above approaches: raw data transmission, quaternion transmission, and quaternion delta transmission.

B. Effect of packet loss

Now let us consider the effects of packet loss in each of the three approaches suggested in Section II-A, with the help of a simulation. Sensor measurements were generated for a simulated sensor package mounted on the end of a rotating arm. The arm was set to rotate at $90^\circ/\text{s}$ around one axis. The orientation of the sensor was then estimated using the *Orient* complementary filter algorithm. Three implementations were simulated:

- filtering remotely on a host system with raw data transmitted by the device.

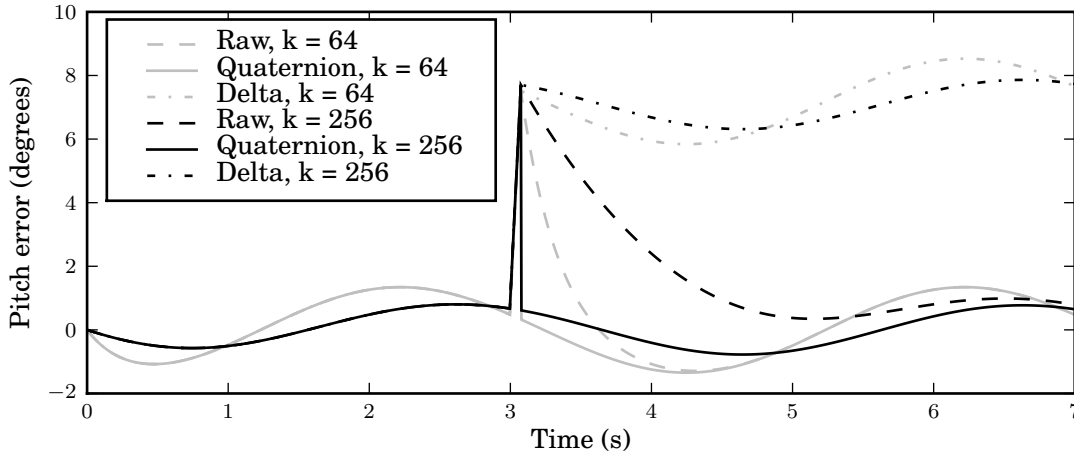


Fig. 3. Error Angle Caused by Packet Loss

- filtering locally on the sensor device with a quaternion then transmitted.
- filtering locally on the sensor device with a quaternion delta then transmitted and the deltas integrated at the receiver.

Additionally, two alternative values of the filter drift correction factor k were simulated. The simulation was run for 7 seconds. At 3 seconds into the simulation, a 100ms link failure occurs.

Fig. 3 plots the error over time of the resulting orientation estimate at the receiver. Prior to the packet loss the results are identical for each approach. The errors present at this stage are due to linear accelerations of the sensor that have survived the low-pass filtering process. The error in the filters with $k = 256$ is less than that for $k = 64$, since the filter is more aggressive with a lower effective cutoff frequency.

During the link failure, the error increases as the true rotation of the arm diverges from the last successful update of the estimate. The resulting initial peak error is the same for all three approaches.

When communication resumes, the three approaches respond very differently. The locally produced estimate, transmitting complete orientations in each packet, immediately informs the receiver of its best estimate of the true orientation. In contrast, the version run at the receiver takes a significant time to correct the error resulting from the missed gyroscope samples. The time taken to correct the error is directly proportional to the value of the filter co-efficient controlling drift correction. The $k = 256$ filter, which achieved smaller errors before the packet loss, now gives larger errors since the correction takes longer. Finally, the delta coded data maintains a constant offset from the true value, as it provides no way to detect the error.

Table I shows the peak and RMS error figures for the different implementations, over the duration of the simulation.

TABLE I
COMPARISON OF PEAK AND RMS ERROR

Data format	k	Peak Error ($^{\circ}$)	RMS Error ($^{\circ}$)
Raw	64	7.51	1.05
Quaternion	64	7.51	0.89
Delta	64	8.53	4.43
Raw	256	7.70	1.23
Quaternion	256	7.70	0.52
Delta	256	7.86	4.22

III. DISCUSSION

The case study presented in Section II demonstrates how changing where data processing occurs has an effect both on the transmitted data rate and on the sensitivity to packet loss. Reducing the data rate, however, does not correlate with the change in error. Rather, we see that the key factor in determining the response to packet loss is the data dependency between packets. As the full quaternion data has no inter-packet dependency it immediately returns to an accurate estimate after a loss, producing the smallest RMS error. The raw sensor data has limited dependency, resulting in an eventual return. The delta coded data has complete data dependency, and therefore never returns.

In this section, we analyse these issues in more detail and identify relevant questions to be considered during the design of a network application that employs signal processing.

A. Data dependency in filters

Consider the general problem of running a filter f over a time series data stream x , such that the output y at time n is given by:

$$y(n) = f(y(n-1), \dots, y(n-M), x(n), \dots, x(n-N)) \quad (1)$$

If the input stream is being transmitted over an unreliable link and some samples of x are unavailable, the output of the filter is no longer clearly defined. In this case there are three

potential choices: compute the output using the last n received samples instead, attempt to estimate replacement samples, or make further attempts to obtain the missing samples, incurring some cost and delay.

The error resulting from proceeding with incorrect values will depend on the nature of the filter, and in particular on the dependency of the output on the input history. For some classes of filters, this dependency may be shown to be time limited. Consider the general finite impulse response filter given by:

$$y(n) = \sum_{i=0}^N b_i x(n-i) \quad (2)$$

Here the output is dependent only on the last N samples, and therefore the duration of the error following a missing or incorrectly estimated value is limited to N time steps. This is true for all non-recursive functions with a limited input history. Another common example of this type of function would be the discrete Fourier transform.

For other classes of filters, the dependency is theoretically infinite in duration but may be shown to tend to zero over time. Consider the infinite impulse response filter given by:

$$y(n) = \sum_{i=0}^N b_i x(n-i) + \sum_{i=1}^M a_i y(n-i) \quad (3)$$

If the coefficients a_i are set such that the effects of the previous outputs decay with time, the dependency on each input value also tends to zero over time. The orientation filter discussed in Section II can also be shown to be of this class, hence its ability to correct for an error after a period of time.

Some filters maintain a complete dependency on previous values, such as the discrete integration function, given by:

$$y(n) = y(n-1) + x(n) \quad (4)$$

For such filters, any error in input data persists permanently.

B. Distribution of processing

Given the necessity to transmit data across an unreliable link, there is a choice of how to distribute filtering operations onto either side of the link. Formally, we can consider the design of two filters: t , running at the transmitter, and r , running at the receiver, with the overall system output given by:

$$y(n) = r(t(n)) \quad (5)$$

The dependency of the system output on errors due to packet loss is then determined by the dependency of r on its input from t . Now we consider again, in these terms, the three approaches to the case study application investigated in Section II-B.

When raw samples are transmitted, t is the identity function. The filter at r is the orientation estimation filter. The error sensitivity of the whole system is thus given by the time-decaying input dependency of that filter. As a result, the system can gradually recover from a packet loss error.

In contrast, when the orientation is estimated on the device, these roles are reversed. As the identity function has no

dependence on prior inputs, nor does the resulting system output. The system recovers immediately after a packet loss.

When delta coding is applied to the quaternions to reduce the data rate, t is the combination of the orientation estimation filter and a differentiator. r is then the integration function (4), with complete dependence on the input values. Hence, the system has no ability to recover after a packet loss.

Applying a similar analysis to any proposed distributed processing strategy allows the designer to predict the system's ability to recover from errors caused by packet loss.

C. Reduction of data rate

Processing of sampled data can often lead to a reduction in transmitted data. This reduction can come about through a change of data representation or the ability to subsample data. These techniques reduce the amount of data that needs to be transmitted by removing extraneous information from the sensed signals.

Subsampling is possible when the frequency content of the signal of interest is limited. Often high sample rates are required to capture a signal without aliasing noise due to other signals in the data. By filtering out signals that are not required for an application the output sample rate can be reduced without loss of information.

Other processes, such as the orientation estimation algorithm discussed, both remove extraneous information and change the fundamental representation of the data. These methods are application specific and must be identified by the application designer.

Both these methods of data reduction differ from many standard compression techniques, such as delta coding, in that they do not create a dependency between output values. In contrast, typical stream compression techniques rely on reference to previous data in order to reduce the subsequently transmitted information, therefore increasing data dependency and the sensitivity to packet loss.

D. Transport and MAC layer techniques

The conventional approach to handling the issue of lost data is to employ some form of transport layer or retransmitting MAC protocol to retransmit the lost values. This approach inherently introduces some delay, as detecting the loss and retransmitting the data must take time to complete. Therefore, this approach may be unsuitable for real time applications where either the absolute minimum latency is sought, or the time required for retransmission exceeds the allowable delay. Even if this is not the case, it must be considered whether the additional power and bandwidth costs of the retransmission are preferable to the error induced by simply ignoring the loss, especially given that the number of retransmissions required is theoretically unbounded. This decision can again be guided by an understanding of the dependencies in the data stream.

E. Estimating missing samples

Missing samples are an inevitable problem of unreliable network links. As we have shown, reducing the dependency

between packets minimises the effect of packet loss after the link is re-established. Nothing so far has reduced the peak error during the actual link loss. Two approaches can be used to minimise the peak error: interpolation and extrapolation.

Interpolating filters, such as the Woltring filter [19], can be designed that use spline fitting, or other interpolation functions, to fill in for missing samples. Such filters are commonly used for post-processing of optical motion capture data where missing samples occur due to occlusion. Interpolation is limited to applications where strict realtime response is not required as interpolation requires knowledge of the characteristics of the data before and after the missing samples.

Extrapolation of missing signal samples presents the advantage of not requiring knowledge of future values, but a good understanding of the process dynamics governing the signal of interest is required. For highly periodic signals, such as those used in the case study simulations, or found in human gait, extrapolation could result in significant reduction in error during link failure. For signals that are more random in nature, such as the movement of the arms while gesturing, extrapolation is less likely to succeed, due to the difficulty in predicting future states.

Both techniques can reduce the error accumulated during a link failure. When the link is re-established, the error response will decay from the remaining error in the same way as shown before: immediately, gradually or never, depending on the dependency on the estimated samples.

F. Questions to consider during design

The discussion in this section has identified a range of inter-related issues to be considered when designing the processing and communication architecture of a signal processing sensor network application. To help determine the best approach for a given application, we suggest that the following questions should be raised during the design process:

- *What input sample rate is necessary to capture the signal of interest, and what output sample rate is necessary to represent it after extraction?*
The input sampling rate is determined by the Nyquist criterion for the signal combined with all other overlaid signals and noise, but the minimum sample rate required to represent it after extraction may be far lower.
- *How is the output signal to be represented, and what is the data dependency of that representation?*
Alternate representations may require lower data rates to transmit, but affect the dependency between packets. This will in turn determine the duration of errors introduced by packet loss, as demonstrated in the case study example.
- *What is the nature of the packet loss likely to be?*
Any loss of data will introduce error into the system output, but the frequency and duration of packet losses will greatly affect the error introduced. Prolonged link outages will have a very different effect to randomised single packet losses. Experiments may be required to identify the patterns of losses likely to occur in practice.
- *Are the expected errors due to packet losses acceptable?*

Based on knowledge of expected packet losses and the data dependencies in the system, the errors that will be introduced by packet losses can be estimated or simulated. If the error is unacceptable, it will be necessary to employ some technique to mitigate the effects of packet loss; either retransmission, alternative partitioning, extrapolation, or a change in data representation. Alternatively if the error is found to be acceptable, it may be possible to employ a simpler design than originally expected, omitting retransmission protocols.

- *Are the costs of retransmission acceptable?*
Any attempt to retransmit lost packets will incur some overhead in the form of energy expended, channel capacity used and latency incurred. These costs are a concern to any application, but particularly for realtime streaming applications where minimising latency can be a critical requirement; data retransmitted later may be of no use.
- *Can the processing be repartitioned or the representation altered to reduce dependency?*
If the initial designs fail to perform in the face of packet loss, and the costs associated with retransmission are too high, then it may be necessary to re-evaluate the system partitioning. Although sensor nodes have limited processing resources, and may be unable to implement complex algorithms, the benefits of local processing may outweigh the benefits of more demanding filters, when the overall system error including packet losses is considered.
- *Can losses be covered by extrapolation or interpolation?*
In applications where the signal of interest is relatively easy to predict, extrapolation or interpolation may be used to fill in for missing data. Care should be taken not rely too greatly on these methods as they may increase the overall error by misprediction.

IV. CONCLUSION

Local processing of data is commonly cited as a way to reduce data transmission and thereby reduce power consumption of wireless sensor network devices. These benefits are confirmed by the case study presented. Yet consideration is rarely given to the effects of distributing processing functions throughout the network on the overall accuracy of the resulting system. We have demonstrated that local processing can result in either improved or worsened levels of error in the face of packet losses, due to data dependencies in the resulting packet streams.

By repartitioning processing or changing data representations, dependency between data packets can be reduced. Errors can thus be reduced without resort to transport layer retransmission techniques which would introduce a cost in latency and overhead. Even if a transport layer is employed to reduce data loss, no radio network can guarantee delivery and so these techniques are still relevant. The potential improvements in error response also mean that local processing can be useful even when it does not reduce data rate.

We therefore recommend that during the design of future wireless sensor systems employing signal processing tech-

niques, more care should be taken to evaluate the effects of packet loss on the final output error. The questions in Section III-F may assist with this process.

ACKNOWLEDGEMENTS

This research was supported by grants from the Scottish Funding Council (R37329) and the UK Science and Engineering Research Council (C523881) for the Research Consortium in Speckled Computing.

REFERENCES

- [1] D. Estrin, L. Girod, G. Pottie, and M. Srivastava, "Instrumenting the world with wireless sensor networks," *Acoustics, Speech, and Signal Processing, Proceedings of the IEEE International Conference on*, vol. 4, pp. 2033–2036, 2001.
- [2] J. Torres, B. O'Flynn, P. Angove, F. Murphy, and C. O. Mathuna, "Motion tracking algorithms for inertial measurement," in *Body area networks, Proceedings of the ICST 2nd international conference on*. ICST, Brussels, Belgium: ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2007, pp. 1–8.
- [3] X. Yun and E. R. Bachmann, "Design, implementation, and experimental results of a quaternion-based Kalman filter for human body motion tracking," *Robotics, IEEE Transactions on*, vol. 22, no. 6, pp. 1216–1227, 2006.
- [4] B. de Silva, A. Natarajan, M. Motani, and K.-C. Chua, "A real-time exercise feedback utility with body sensor networks," *Medical Devices and Biosensors, 5th International Summer School and Symposium on*, pp. 49–52, June 2008.
- [5] W.-S. Yeoh, J.-K. Wu, I. Pek, Y.-H. Yong, X. Chen, and A. B. Waluyo, "Real-time tracking of flexion angle by using wearable accelerometer sensors," *Medical Devices and Biosensors, 5th International Summer School and Symposium on*, pp. 125–128, June 2008.
- [6] V. van Acht, E. Bongers, N. Lambert, and R. Verberne, "Miniature wireless inertial sensor for measuring human motions," in *Engineering in Medicine and Biology Society, Proceedings of the 29th Annual International Conference of the IEEE on*, Aug. 2007, pp. 6278–6281.
- [7] A. Young, M. Ling, and D. K. Arvind, "Orient-2: A realtime wireless posture tracking system using local orientation estimation," in *Embedded networked sensors, Proceedings of the 4th workshop on*. New York, NY, USA: ACM, 2007, pp. 53–57.
- [8] A. Lynch, B. Majeed, J. Barton, F. Murphy, K. Delaney, and S. O'Mathuna, "A wireless inertial measurement system (WIMS) for an interactive dance environment," *Journal of Physics: Conference Series*, vol. 15, pp. 95–100, 2005.
- [9] C. Park, J. Liu, and P. Chou, "Eco: an ultra-compact low-power wireless sensor node for real-time motion monitoring," *Information Processing in Sensor Networks, Fourth International Symposium on*, pp. 398–403, 2005.
- [10] E. R. Bachmann, "Inertial and magnetic tracking of limb segment orientation for inserting humans into synthetic environments," Ph.D. dissertation, Naval Postgraduate School, Monterey, California, 2000.
- [11] R. Zhu and Z. Zhou, "A real-time articulated human motion tracking using tri-axis inertial/magnetic sensors package," in *Neural Systems and Rehabilitation Engineering, IEEE Transactions on*, vol. 12, no. 2, June 2004, pp. 295–302.
- [12] C. Ware and R. Balakrishnan, "Reaching for objects in vr displays: lag and frame rate," *Computer-Human Interaction, ACM Transactions on*, vol. 1, no. 4, pp. 331–356, 1994.
- [13] M. Meehan, S. Razzaque, M. Whitton, and J. Brooks, F.P., "Effect of latency on presence in stressful virtual environments," *Virtual Reality, 2003. Proceedings. IEEE*, pp. 141–148, March 2003.
- [14] K. Mania, B. Adelstein, S. Ellis, and M. Hill, "Perceptual sensitivity to head tracking latency in virtual environments with varying degrees of scene complexity," in *Applied Perception in Graphics and Visualization, Proceedings of the 1st Symposium on*. ACM New York, NY, USA, 2004, pp. 39–47.
- [15] D. Wessel and M. Wright, "Problems and prospects for intimate musical control of computers," *Computer Music Journal*, vol. 26, no. 3, pp. 11–22, 2002.
- [16] *InertiaCube3 Datasheet*, InterSense, Inc. [Online]. Available: <http://www.intersense.com>
- [17] C. Angeloni, P. Riley, and D. Krebs, "Frequency content of whole body gait kinematic data," *Rehabilitation Engineering, IEEE Transactions on*, vol. 2, no. 1, pp. 40–46, Mar 1994.
- [18] *Application Note AN039 - Using CC1100/CC1150 in European 433/868 MHz bands*, Chipcon.
- [19] H. Woltring, "A FORTRAN package for generalized, cross-validatorspline smoothing and differentiation," *Advanced Engineering Software*, vol. 8, no. 2, pp. 104–107, 1986.