



THE UNIVERSITY OF
NEWCASTLE
AUSTRALIA

DOCTORAL THESIS

Ambient Sensor Fusion for Virtual Reality Systems

Author:

[Jake Alexander FOUNTAIN](#)

B. Mathematics / B. Science (Physics) / B. Computer Science (Hons),
The University of Newcastle

A thesis submitted in fulfillment of the requirements

for the degree of Doctor of Philosophy in Computer Science

in the

[School of Electrical Engineering and Computing](#)

[The University of Newcastle, Australia](#)

Submitted October 2018

*This research was supported by an Australian Government Research Training Program
(RTP) Scholarship*

Declaration of Authorship

I hereby certify that the work embodied in the thesis is my own work, conducted under normal supervision. The thesis contains no material which has been accepted, or is being examined, for the award of any other degree or diploma in any university or other tertiary institution and, to the best of my knowledge and belief, contains no material previously published or written by another person, except where due reference has been made. I give consent to the final version of my thesis being made available worldwide when deposited in the University's Digital Repository, subject to the provisions of the Copyright Act 1968 and any approved embargo.

Signed:

Date:

Jake Fountain

31/10/2018

Contribution to joint publications

I hereby certify that the work embodied in this thesis contains published papers of which I am a joint author. I have included as part of the thesis a written declaration endorsed in writing by my supervisor, attesting to my contribution to the joint publication.

By signing below I confirm that Jake Fountain contributed the majority of the work for the following publications. This included motivating the work, developing the algorithms and performing the research and writing the papers. I, Shamus P. Smith, provided academic advice and writing assistance.

Automatic Identification of Rigidly Linked 6DoF Sensors J. Fountain and S. P. Smith (Mar. 2016). ‘Automatic Identification of Rigidly Linked 6DoF Sensors’. In: *IEEE Virtual Reality 2016*, pp. 175–176. DOI: [10.1109/VR.2016.7504710](https://doi.org/10.1109/VR.2016.7504710)

Real-time Ambient Fusion of Commodity Tracking Systems for Virtual Reality J. Fountain and S. P. Smith (2017). ‘Real-Time Ambient Fusion of Commodity Tracking Systems for Virtual Reality’. In: *ICAT-EGVE 2017 - International Conference on Artificial Reality and Telexistence and Eurographics Symposium on Virtual Environments*. Ed. by R. W. Lindeman et al. The Eurographics Association. DOI: [10.2312/egve.20171331](https://doi.org/10.2312/egve.20171331)

Detecting Rigid Links Between Sensors for Automatic Sensor Space Alignment J. Fountain and S. P. Smith (Mar. 2018). ‘Detecting Rigid Links between Sensors for Automatic Sensor Space Alignment in Virtual Environments’. en. In: *Virtual Reality*, pp. 1–14. DOI: [10.1007/s10055-018-0341-8](https://doi.org/10.1007/s10055-018-0341-8)

Signed:

Date:

Shamus P. Smith

24/9/2018

Acknowledgements

Firstly, I want to acknowledge the support of some people in my personal life. To Melinda Duncan, thank you for being my best friend for nearly a decade. You have supported me enormously during the process of realising this work. To Anneliese Wild, thank you for making it easier to keep working even when things were at their hardest. And to my Father, Ian Fountain, thank you for always pushing me out of my comfort zone and teaching me the value of adventure.

Secondly, I would like to acknowledge the support in the form of equipment, office space and technical advice provided by the Newcastle Robotics Laboratory and the NUBots RoboCup team. In particular, I would like to acknowledge Trent Houliston for the numerous enjoyable and insightful conversations which informed my work.

I am also grateful to the University of Newcastle's academic staff, particularly my supervisors Dr Shamus P. Smith and Associate Professor Stephan Chalup for their guidance and assistance over the course of my time at The University of Newcastle.

Lastly, this PhD research would not have been possible without the financial support I have received through the Australian Government Research Training Program Scholarship (RTP) and the University of Newcastle's research allowances.

I am grateful for these opportunities.

*This thesis is dedicated to my mother, Jennifer Fountain,
who gave me the confidence and tools to carve my own
path.*

The University of Newcastle, Australia

Abstract

Faculty of Engineering and Built Environment
School of Electrical Engineering and Computing

Doctor of Philosophy

Ambient Sensor Fusion for Virtual Reality Systems

by **Jake Alexander FOUNTAIN**

Modern Virtual Reality (VR) systems rely on precise measurements of the real world to realistically present a virtual environment. Range, capability and accuracy are key tracking properties which determine the realism and applicability of a VR system. Integration of two or more sensor systems can enhance these properties compared to each individual component system. To support a rich ecosystem of diverse tracking devices for all levels of user competency, algorithms for *ambient sensor fusion* are required - algorithms which do not require user intervention or knowledge. This thesis investigates each of the following three steps required for ambient sensor fusion. First, *Correlation* involves identification of sensor dependencies and temporal relationships. Second, *Calibration* aligns sensor measurement domains. Finally, *Fusion* combines sensor data to extract an underlying statistical model.

To address the correlation step, an algorithm for identifying rigid links between position and rotation sensors *from different systems* was developed and tested with several real world systems. Additionally, a novel *model-less* approach to determining latency between two sensor systems has shown promising results for use in comparison of arbitrary dependent signals.

To address the second sensor fusion step, an ambient calibration technique has been developed which determines the relationship between two systems solely from self-directed user movement. The algorithm was applied to the Microsoft Kinect v2 and popular VR systems to enable body tracking in VR with commodity devices and minimal user setup.

To address the final sensor fusion step, a modular multi-modal skeleton fusion algorithm was developed. The algorithm employs a novel constrained articulated Kalman filter to combine skeletal tracking results with high modularity in real time. To test the fusion system, the optical Leap Motion hand tracking system was fused with the inertial Perception Neuron hand tracking system. A user study was performed (n=18) and results suggested that the proposed system succeeds in generalising the component tracking systems to perform well in a wider variety of scenarios.

Finally, the research software has been made available in an open source C++ plugin called ‘Spooky’. Spooky currently supports Unreal Engine 4. Future work will be focused on improving the reliability and usability of the Spooky framework, while extending the techniques developed for each of the fusion steps to broader applications.

Contents

Declaration of Authorship	i
Acknowledgements	iii
Abstract	v
Contents	vii
List of Figures	xi
List of Tables	xix
Thesis Highlights	xxi
1 Introduction and Motivation	1
1.1 Virtual Reality and Tracking Systems	4
1.2 Device Fragmentation and the Fusion Middleware	6
1.3 Fusion Middleware Features	10
1.3.1 Hardware-Software Abstraction	10
1.3.2 Modularity and Sensor Fusion	12
1.3.3 User Configuration and Representation	14
1.4 Research Question and Contributions	16
2 Literature Review	21
2.1 Sensor Calibration and the Need for Sensor Correlation	22
2.1.1 Automatic and Manual Calibration	22
2.1.2 Representing 3D Rigid Bodies	24
2.1.3 6DoF Sensor Calibration	32
2.1.4 Sensor Identification	34
2.2 Sensor Fusion	37
2.2.1 Formal Definition of Sensor Fusion	37

2.2.2	Bayesian Filtering	39
2.2.3	Skeleton Fusion	42
2.3	Existing VR Middleware Software	45
3	Sensor Identification	50
3.1	Matching of Sensors	52
3.1.1	Invariant Functional (IF) Method	52
3.1.2	Calibration Error (CE) Method	55
3.1.3	Match Selection and Score Filtering	56
3.2	Simulation Analysis	57
3.2.1	Notation for Randomly Generated Transforms	57
3.2.2	Procedure	58
3.2.3	Results	59
3.3	Case Study: Offline Experiment With Real Sensors	62
3.4	Case Study: Interactive Demonstration with Real Sensors	66
3.5	Reflection and Future Work	70
4	Latency Calibration	73
4.1	Related Work	74
4.2	Model-less Latency Between Two Multi Dimensional Signals	75
4.3	Application to Simulation	78
4.3.1	Generating Simulation Data	78
4.3.2	Simulation Results	80
4.4	Application to Video Data	84
4.5	Reflection and Future Work	89
5	Ambient Sensor Calibration	94
5.1	Related Work	95
5.2	Implementation of Ambient Calibration	98
5.2.1	The Calibrator	99
5.2.2	Calibration State Machine	101
5.2.3	Fault Detection	104
5.3	Evaluation	105
5.3.1	Method	105
5.3.2	Results	108
5.3.3	Discussion	109
5.4	Reflection and Future Work	113
6	Skeleton Sensor Fusion	115
6.1	The Fusion Graph	116
6.2	Fusion Algorithm	120
6.2.1	Recursive Fusion	121

6.2.2	Prediction Update	125
6.2.3	Constrained Measurement Update	127
6.2.4	Computing the Jacobian	131
6.3	Evaluation	133
6.4	Reflection and Future Work	134
6.4.1	Confidence Modelling	134
6.4.2	Advanced Fusion Chain Selection	135
6.4.3	Managing Off-Diagonal Covariance	136
7	Spooky: A Sensor Fusion Plugin	137
7.1	Software Implementation	137
7.2	Spooky UE4 Features	139
7.2.1	Spooky Fusion Plant	140
7.2.2	Measurements	141
7.2.3	Spooky Skeletons	141
7.2.4	Spooky Spirits	143
7.2.5	Spooky Graveyards	144
7.3	Reflection and Future Work	145
8	User Study	146
8.1	Method	147
8.1.1	Tracking Setup	148
8.1.2	Grip Detection	150
8.1.3	Leap Motion Confidence Model	151
8.1.4	Tasks	156
8.1.5	Experiment	159
8.2	Results	164
8.2.1	Objective Performance Metrics	166
8.2.2	Throwing Task	169
8.2.3	Subjective Performance Metrics	172
8.2.4	Verbal Responses	174
8.2.5	Summary of Results	175
8.3	Discussion and Conclusion	177
8.3.1	Explanation of Participant Performance	179
8.3.2	Ground Truth Accuracy	181
8.3.3	Reflection on Proposed Fusion Algorithm	186
8.3.4	Reflection and Future Work	187
9	Conclusion	189
9.1	Thesis Summary	189
9.2	Future Work	191

A User Study Documents	194
Bibliography	206

List of Figures

1.1	An example software interface model for development of a virtual reality application. A developer must target each possible input and output device software development kit (SDK). Additional work is required to support alternative input systems; here Input Device 3 and Output Device 3 cannot be used because the developer did not include support. Input/Output Device 2 is compatible with the system, but not compatible with Input/Output Device 1 simultaneously. The majority of included devices are not used by a given user.	7
1.2	The software structure of a <i>standard middleware</i> . Input and output devices are abstracted from the developer of the virtual environment. This allows support for a wide variety of input and output devices. However, many devices will not be compatible due to overlapping device domains, unless the developer explicitly handles each combination. For example, here the user can choose between Input 1 or Input 2, but not both.	8
1.3	Sensor matching and alignment concept.	9
1.4	The proposed sensor fusion middleware acts like a standard middleware, but also allows for the user to combine systems with overlapping domains. Overlapping sensors are fused, reducing noise and increasing tracking range.	10
1.5	The ambient sensor fusion pipeline visualised with a 2D toy example. Each pair of arrows represents a sensor coordinate frame. The grey trails represent recent data from the sensor, such as a position curve. The ellipses represent the rigid bodies which the sensors are attached to in the common coordinate frame.	17
1.6	Thesis map.	20
2.1	Spectrum of common mathematical representations of 3D rotation. Note that although $so(3)$ is technically embedded in $\mathbb{R}^{3 \times 3}$ it can be parameterized by $\omega \in \mathbb{R}^3$ and is effectively dense.	31

2.2	The relationship between two 6DoF sensors has characteristic equation given by Equation 2.25. $\underline{S}$ and $\underline{Q}$ are the sensor reference frames. S_t and Q_t are the coordinate systems corresponding to the 6DoF pose of each sensor at time step t . $\mathbf{Y}$ and $\mathbf{X}$ are the rigid transforms linking the sensor reference frames and the sensor poses respectively. $\mathbf{A}_t$ and $\mathbf{B}_t$ are the 6DoF pose transforms of the two sensors at time step t .	33
3.1	Sensor matching and alignment concept.	51
3.2	The relationship between two 6DoF sensors has characteristic equation given by Equation 3.1. $\underline{S}$ and $\underline{Q}$ are the sensor reference frames. S_t and Q_t are the coordinate systems corresponding to the 6DoF pose of each sensor at time step t . $\mathbf{Y}$ and $\mathbf{X}$ are the rigid transforms linking the sensor reference frames and the sensor poses respectively. $\mathbf{A}_t$ and $\mathbf{B}_t$ are the 6DoF pose transforms of the two sensors at time step t .	53
3.3	Invariant Functional (IF) Method - in this hypothetical example, A and B are correlated, while C is not.	54
3.4	Calibration Error (CE) Method	56
3.5	Simulation sample-noise and sample-timing analysis for the calibration error (CE) method. The heat map displays the fraction of correct identifications of the simulated sensors as a function of angular noise and number of samples averaged over the trial period (<i>left</i>). The sample-timing linear least squares relationship was $\tau_m(N_s) \approx (0.0018 * N_s + 0.033)\text{ms}$ (<i>right</i>).	60
3.6	Simulated noise performance analysis of the calibration error (CE) and invariant functional (IF) matching algorithms. The mean matching performance is equal to the fraction of correct identifications of sensor dependencies over a 52 second simulation. Note that since the IF method only compares rotation data, it is expected to perform well independent of positional noise.	61
3.7	Data visualisation for the offline real sensor case study. <i>Left</i> shows the OptiTrack marker coordinate frames, transformed into the Kinect reference frame using ground truth (see Figure 3.8), and the detected user. A rigid body skeleton is generated with the OpenNI software platform version 1.5.7 (<i>centre</i>). The candidate Kinect rigid bodies have their corresponding coordinate frames drawn at their origins. The two 6DoF measurements from the sensors are matched with the one of six Kinect rigid body streams in real time (<i>right</i>). At this instance, the left knee (red) has been correctly identified, while the right shoulder has been misidentified as the left shoulder (green).	63
3.8	The motion capture setup for measuring the Kinect reference frame for ground truth verification (<i>left</i>) and an example sensor placed on the shoulder and tracked by an OptiTrack system (<i>right</i>).	64

3.9	The OpenNI Kinect skeleton tracking software has several instabilities which make it unsuitable for rigid body identification. For example, the knee orientation is ambiguous when outstretched; here the rigid body for the knee flips 180° discontinuously.	66
3.10	Tracking results for an OptiTrack marker (<i>left</i>) and a PS Move controller (<i>right</i>) from the perspective of the PS Eye camera. Note that the OptiTrack marker has been transformed to the PS Move sensor space for visualisation only; all computations use raw data in the native space.	67
3.11	Two markers (<i>top</i>) were matched with a PS Move controller. Moment of attachment (<i>bottom left</i>) and moment of identification of the rigid link (<i>bottom right</i> ; indicated by the white sphere) for the CE method. Typical time between the two events is 1-2 seconds, depending on the movement of the user (Figure 3.12). The amount of movement required to match here can be seen to be a 90° rotation and a translation of about 15cm. The IF method achieved similar results. . . .	68
3.12	An example trace where two motion capture markers are attached sequentially, one at a time, to the PS Move sensor. Here it is assumed that when the ground truth distance between a marker and the PS Move (<i>top</i>) is less than 30cm, the two are rigidly attached. The matching result for the CE method (<i>bottom</i>) can be seen to match well with the ground truth estimator (<i>centre</i>). The broken and dotted lines indicate the times at which a sensor is attached and removed respectively.	69
3.13	An example of real-time interactive sensor matching for the Microsoft Kinect. The user is tracked with OpenNI and NiTE software libraries and the sensor's true location (left shoulder) is drawn as 3D coordinate axes (<i>left</i>). Hypotheses are shown as small white spheres placed at the origin of each articulated body part, with the single larger sphere representing the most likely hypothesis (<i>centre</i>). After approximately 10 seconds of tracking, the correct location is determined and most other hypotheses are eliminated (<i>right</i>).	70
4.1	Visualisation of the proposed latency calibration method. The approach searches for values of latency l such that close points in $\mathbf{x}$ correspond to close points in $\mathbf{y}$. The algorithm uses the largest disagreement over all close points in $\mathbf{x}$ as the objective function.	76
4.2	Example of latency calibration applied to one dimensional signals x, y procedurally generated as described in Section 4.3.1, with parameters ($n = 1, m = 1, \sigma_x = 0, \sigma_y = 0, K_x = 10, K_y = 2, s_x = 2, s_y = 5, T = 1$ sec). The latency was successfully calibrated to within one frame of error. (<i>top left</i> : signals over time; <i>top right</i> : latency search; <i>bottom left</i> : signals before calibration of latency; <i>bottom right</i> : signals after calibration of latency)	81

4.3	Example of latency calibration applied to two 2 dimensional signals x, y procedurally generated as described in Section 4.3.1, with parameters ($n = 2, m = 2, \sigma_x = 0, \sigma_y = 0, K_x = 10, K_y = 2, s_x = 2, s_y = 5, T = 1$ sec). The latency was successfully calibrated to within one frame of error.	82
4.4	Example of latency calibration applied to two 10 dimensional signals $\mathbf{x}, \mathbf{y}$ procedurally generated as described in Section 4.3.1, with parameters ($n = 10, m = 10, \sigma_x = 0, \sigma_y = 0, K_x = 10, K_y = 2, s_x = 2, s_y = 5, T = 1$ sec). The latency was successfully calibrated to within one frame of error.	83
4.5	Example of latency calibration applied to two 10 dimensional signals as in Figure 4.4, but here the latency was not correctly recovered. This is likely due to a lack of close points in x -space.	84
4.6	Example of latency calibration applied to two 1 dimensional signals x, y procedurally generated as described in Section 4.3.1, with parameters ($n = 1, m = 1, \sigma_x = 0.05, \sigma_y = 0.05, K_x = 10, K_y = 2, s_x = 2, s_y = 5, T = 1$ sec). The latency was successfully calibrated close to the true value (<i>black dashed line</i>), though some error is present. . . .	85
4.7	Performance of latency calibration with varying dimension of the signals. For each value, 20 trials were performed each with random latency and procedural functions. Here, $T = 1$ second, and the periods of the procedural functions were $s_x = 0.5, s_y = 5$. Strong performance is achieved since the independent signal $\mathbf{x}(t)$ is periodic with period less than the sample time T , and so there are many high quality close points in $\mathbf{x}(t)$	86
4.8	Performance of latency calibration with varying dimension of the signals. For each value, 20 trials were performed each with random latency and procedural functions. Here, $T = 1$ second, and the periods of the procedural functions were $s_x = 2, s_y = 5$. Compared to Figure 4.7, performance is reduced by the fact that the function $\mathbf{x}(t)$ is not periodic on the domain $[0, T]$, and hence the quality and number of close points is reduced.	87
4.9	Performance of latency calibration with varying measurement noise of the signals for one dimensional (<i>left</i>) and two dimensional (<i>right</i>) signals. The red line indicates the boundary of 80% success. Here, n and m refer to the dimension of the independent signal $\mathbf{x}(t)$ and dependent signal $\mathbf{y}(t)$ respectively.	88
4.10	Error in determined latency for the motion example using the proposed method for various image down-sample rates. The down-sampled image pixel count represents the dimension of the dependent signal $\mathbf{y}(t)$	89

4.11	Error in determined latency for the long motion example using the proposed method for various image down-sample rates. The down-sampled image pixel count represents the dimension of the dependent signal $\mathbf{y}(t)$. Compared to Figure 4.10, the data spanned 300 frames (10 sec), and the increased information and self-intersections enables the solution to be determined at resolutions greater than 7×7 pixels.	90
5.1	The relation between two 6DoF sensors is characterized by Equation 5.1. $\underline{\mathbf{S}}$ and $\underline{\mathbf{Q}}$ are the sensor reference frames. $\mathbf{S}_t$ and $\mathbf{Q}_t$ are the coordinate systems corresponding to the 6DoF pose of each sensor at time step t . $\mathbf{Y}$ and $\mathbf{X}$ are the rigid transforms linking the sensor reference frames and the sensor poses respectively. $\mathbf{A}_t$ and $\mathbf{B}_t$ are the 6DoF pose transforms of the two sensors at time step t .	95
5.2	Software architecture of the fusion plugin.	98
5.3	State machine for calibration between two systems. States: Uncalibrated (U), Refining (R), Calibrated (C).	102
5.4	The three tasks used to assess the performance of the ambient calibration. In the sorting task, the user must sort the cubes (a) into their respective colors (b) on the platforms a few steps away. In the pointing task, the user must point to a series of targets while standing in place (c). The walking task involves the user stepping around the tracking space while facing the Kinect and slowly raising and lowering their arms (d).	106
5.5	Distributions of errors for Rift (a) and Vive (b) for three different ambient calibration scenarios. Confidence ellipses (95%) are shown as visual aids.	107
5.6	To measure the accuracy of the ambient calibration $\mathbf{Y}$ between the Kinect and the VR systems, an external gold-standard OptiTrack motion capture system was used to measure $\mathbf{K}$.	109
5.7	An example of the typical results for calibration with walking activity. An outline of the user's real body (<i>white edge</i>) from the Vive's passthrough camera is shown on the right, overlaid with the virtual scene rendered from user perspective. The controllers are outlined with a <i>green edge</i> . This calibration took 16 seconds, and had an error of 22.2cm and 5.25° .	110
5.8	An example calibration trace with calibration states (Figure 5.3) overlaid in color (Red = 'Uncalibrated', Yellow = 'Refining', White = 'Calibrated'). The Kinect is moved at the 55 second mark, giving it a rotation 8 degrees from its original configuration. The error is corrected within 40 seconds.	112
6.1	Software architecture of the fusion plugin.	120

6.2	Example recursive fusion of a position measurement relative to the root node. The nodes which are included in the fusion chain have their coordinate systems shown in yellow. The fusion chain only consists of the first three nodes from the end of the chain to give the required 3 positional degrees of freedom. The green nodes represent a possible final pose of the fusion graph if the measurement were to be trusted completely.	122
7.1	Software architecture of the fusion plugin.	138
7.2	Data flow of the Spooky Skeleton Fusion Plugin within Unreal Engine 4.	140
7.3	An example of adding a sensor measurement to Spooky through Unreal Engine 4 blueprints. This method is called whenever new sensor information is available. The sensor is configured to have a known node location - the left hand.	142
7.4	Example Animation Blueprint calls to retrieve the fused Spooky result.	143
7.5	Example configuration of a Spooky Graveyard for fusion of the Leap Motion, Perception Neuron and Oculus Rift (as used in Chapter 8). .	144
7.6	Spooky Graveyard class structure within a UE4 level. Items labelled in typewriter font are example objects owned by the surrounding box.	145
8.1	Oculus Rift with Leap Motion attached.	146
8.2	Default Perception Neuron glove (<i>right</i>) compared to the custom glove used in this study (<i>left</i>) as seen by the leap motion infrared cameras with tracking result overlaid. The default glove is not tracked at all.	148
8.3	The custom built glove for the right hand, allowing simultaneous Leap Motion and Perception Neuron hand tracking. Left image shows the sensor arrangement, and the right image shows the final glove appearance with outer-layer.	149
8.4	Example screenshot from Perception Neuron software, with the connected sensors shown on the left in green or yellow (depending on their connection strength). An example tracking result is shown on the right.	150
8.5	The avatar animated by the tracking data during the user study. . . .	151
8.6	Example grip action. If the distance between the thumb and any of the fingers d becomes less than 5cm, the closest object within 5cm of the midpoint of the engaged fingertips becomes gripped.	152
8.7	Software structure for the Leap Motion (LP) solution used for the user study, within Unreal Engine 4.	152

8.8	Software structure for the Perception Neuron solution for the user study, within Unreal Engine 4. The role of the Spooky Fusion Plant here is to retarget the animation from the PN skeleton to the UE4 Mannequin, and use the Oculus Rift data to position the avatar correctly.	152
8.9	Software structure for the Fused Tracking solution for the user study, within Unreal Engine 4.	153
8.10	Palm confidence as a function of angle θ_{palm} between the view direction and the palm normal direction.	155
8.11	Keyboard Task	156
8.12	Sorting Task	157
8.13	Throwing Task	159
8.14	Real-World training for the Sorting Task.	161
8.15	Real-World training for the Throwing Task.	162
8.16	Calibration poses performed before each task (marked with a checkmark). Since only the upper body of the participant was tracked, the participant was allowed to sit during calibration.	163
8.17	Example VR view and webcam footage recorded during the study for an example participant.	165
8.18	Participant scores (n=18). The plots display the inter-quartile range (box), median (red line), non-outlier range (whiskers), and outliers (circles). Outliers are points which deviate by more than $1.5 \times IQR$ from the 1st and 3rd quartiles. (LP = Leap Motion, PN = Perception Neuron, FT = Fused Tracking)	166
8.19	Sum hit density of participant throws for each tracking technology. The concentric rings represent the target, the white square and disk represent the ball table and spawn location, and the green cross represents the participant's location (see Figure 8.13).	170
8.20	Ball hit density for the Throwing Task summed over all participants. The vertical lines indicate the location and size of the target and its rings.	171
8.21	Subjective response metrics for each task and each technology. Strongly statistically significant results are marked with *. Maximum mean rank is 3, indicating the technology was always preferred first of the 3 options. Minimum mean rank is 1, indicating the technology was always the last preference.	173
8.22	Tracking technology mean preference rank over all tasks. This indicates that Fused Tracking is the most generally preferred technology.	175
8.23	Response sentiment for each tracking technology. Percentages reported are fractions of response opportunities; each participant had three opportunities to comment on each of the three technologies, once each task. However, the participant was blind to which technology they were talking about, and the prompt was open ended.	176

8.24	Equipment configuration for ground truth accuracy analysis of proposed fusion algorithm. OptiTrack motion capture markers were used to measure the ground truth position of the wrists relative to the head.	181
8.25	Test 1. Left (<i>orange</i>) and right (<i>blue</i>) wrist position measured relative to the head (<i>black dot</i>) viewed from the top. The <i>x</i> -axis points in the direction of the user's view (<i>red line</i>), and the <i>y</i> -axis points left-wards (<i>green line</i>). Grid lines are 20cm apart. Notice that the Fused Tracking case reproduces many of the ground truth curves more faithfully than the Leap Motion or Perception Neuron alone.	183
8.26	Test 2. See Figure 8.25 for semantic key. Note that the two large identifiable curves in the right hand data (<i>orange</i>) of the Fused Tracking and Ground Truth cases are less identifiable or missing from the individual tracking cases.	184
8.27	Test 3. See Figure 8.25 for semantic key. In this test, a throwing motion was performed repeatedly with the right hand (<i>orange</i>). The ground truth trace is drawn beneath each trace for comparison (<i>grey</i>). Note how the Fused tracking combines the high accuracy of the Leap Motion in front of the headset with the tracking coverage provided by the Perception Neuron.	185

List of Tables

1.1	Examples of common commercial and research tracking systems. See Table 1.2 for more information about tracker types.	3
1.2	Summary of tracking paradigms for determining real time position and orientation of objects.	5
2.1	Manual vs Automatic Calibration	24
3.1	Calibration error (CE) method - fraction of correct identifications for real sensor experiment	64
3.2	Invariant functional (IF) method - fraction of correct identifications for real sensor experiment	64
3.3	Confusion matrix of attachment errors for matching one marker to one of two possible locations A and B	69
4.1	Snapshots for the videos captured in the two examples.	88
5.1	System-node table example.	100
5.2	Transition conditions for the state machine shown in Figure 5.3. Each of the hard-coded values here are configurable in reality - these are the values used in this study, determined heuristically.	103
5.3	Summary statistics for errors of ambient calibration procedure compared to ground truth. Values are reported in ‘mean $\pm$ standard deviation’ format.	108
6.1	Structure of a node in the Fusion Graph	116
6.2	Types of Articulation. See Section 2.1.2 for definitions of the functions $\exp, T, S$ and more information on representing 3D transforms.	117
6.3	Measurement structure	118
6.4	Measurement types and associated data structures	119
6.5	Measurement flags for controlling measurement fusion behaviour	119
6.6	Available Degrees of Freedom for Articulation Types	124
6.7	Required Degrees of Freedom for Measurement Types	125
6.8	Update functions for articulations with velocity modelled.	126
8.1	Example Tracking Presentation Orders	160

8.2	Paired change in performance metrics - reported numbers are the amount of change in the metric when a given participant was using the Fused Tracking (FT) solution compared to the baseline of the other two solutions (Leap Motion (LP) or Perception Neuron (PN)).	168
8.3	Wilcoxon signed-rank test for performance metrics when using Fused Tracking (FT) solution compared to either Leap Motion (LP) or Perception Neuron (PN).	169
8.4	Significance of participant preferences using Kendall's W test statistic. For significant results ($p < 0.05$), Kendall's $W \in [0, 1]$ indicates the amount of agreement among the participants.	174
8.5	Summary of the conclusions drawn from the results.	177
8.6	Error statistics for each tracking solution relative to the ground truth, over the three trials.	182
A.1	Table of Complaint Responses Toward Leap Motion Tracking	203
A.2	Table of Complaint Responses Toward Perception Neuron Tracking	204
A.3	Table of Complaint Responses Toward Fused Tracking	205

Thesis Highlights:

- The problem of **ambient sensor fusion** is identified as a valuable research direction aimed at allowing researchers, professionals and hobbyists to create cheaper, high quality, modular sensor systems for VR (Chapter 1)
- A literature review is presented summarising the related work in the areas of sensor fusion and VR software middlewares (Chapter 2)
- The contributions of this thesis include four main theory areas:
 1. Ambient sensor identification (Chapter 3)
 2. Model-less calibration of latency between dependent sensor signals (Chapter 4)
 3. Ambient sensor calibration and alignment (Chapter 5)
 4. Modular sensor fusion for articulated bodies (Chapter 6)
- Additionally, two central practical contributions are discussed:
 1. An open-source plugin for Unreal Engine 4 / C++ implementing key sensor fusion algorithms (Chapter 7)
 2. A user study assessing the utility of the developed fusion algorithms (Chapter 8)