A Distributed Intelligent Sensing Approach for Environmental Monitoring Applications

THÈSE N^{O} 6673 (2015)

PRÉSENTÉE LE 31 JUILLET 2015

À LA FACULTÉ DE L'ENVIRONNEMENT NATUREL, ARCHITECTURAL ET CONSTRUIT
LABORATOIRE DE SYSTÈMES ET ALGORITHMES INTELLIGENTS DISTRIBUÉS
PROGRAMME DOCTORAL EN INFORMATIQUE ET COMMUNICATIONS

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE

POUR L'OBTENTION DU GRADE DE DOCTEUR ÈS SCIENCES

PAR

William Christopher EVANS

acceptée sur proposition du jury:

Prof. B. Faltings, président du jury
Prof. A. Martinoli, directeur de thèse
Dr G. Barrenetxea Kobas, rapporteur
Prof. S. Santini, rapporteuse
Dr H. Huwald, rapporteur

ÉCOLE POLYTECHNIQUE
FÉDÉRALE DE LAUSANNE

Suisse
2015

Acknowledgements

THIS manuscript has been made possible by a multitude of supporters. I am immeasurably grateful to each of my friends, family and colleagues, in Switzerland and abroad, for continuing to share ideas, aspirations, and laughter, throughout this difficult process. Thank you all.

My career is owed to a lifetime of excellent mentors. Jorge Castro, who gave his time, reputation, and patience to give me the opportunities and guidance I needed to cultivate my early abilities as an engineer. Christian Blanvillain, who has not only been a patient friend, but serves to me as the highest example of what it means to lead an intellectually curious life. Prof. Raki Prakash, who gave me my first chance to stretch my academic legs. And to those who have had the most fundamental effect on the person I am today, my parents, whose extreme attention paid to my education enabled me to pursue my passions from a very young age.

Most recently I am proud to add my advisor, Prof. Alcherio Martinoli, to this list of exceptional mentors, for his unconditional support during the last seven years. His incredible patience is endlessly humbling, and his vision has never failed to invigorate me in my research.

To my professional collaborators, I could not have done this work without you. I am especially grateful to Dr. Dominique Mazzi of Agroscope, Davis Daidié and François Ingelrest at Sensorscope, and Martin Günter and Daniel Zingg at Andermatt Biocontrol, for going well out of their way to support my efforts during the last several years.

I owe my sanity to my friends and colleagues. First I must thank the many researchers with whom I have shared an office during my time in DISAL—Ezequiel Di Mario, Amanda Prorok, Maria Boberg, Milos Vasic, Duarte Dias, and Steven Roelofsen. Honorable mention to Sven Gowal. Our office was never a quiet one, and I would not have had it any other way. To the rest of DISAL, with whom I shared many years of food, drink and good conversation: Chris Cianci, Riccardo Falconi, Massimo Mastrangeli, Albrecht Lindner, Alexander Bahr, without whom the hardware aspects of this work would have never been; Grégory Mermoud, who inspired me to fall in love with swarm intelligent systems; Adrian Arfire, Bahar Haghighat, Felix Schill, Ali Marjovi, Emmanuel Droz, Iñaki Navarro, Jorge Soares, Lorenzo Sarti, Zeynab

Talebpour and Alicja Wasik. I could not have asked for a more engaging and supportive mix of coworkers. Outside of the lab, thanks to Pierre-François Laquerre, Kamran Kalbasi, Yuliy Schwartzburg, Erinç Bayrakçı, Alex Pilchin, Alisa Yurovsky, Giorgi Gvianishvili, Nino Jejelava, Olivier Blanvillain, Louis Bliss, Mikaël Mayer, and Daniel Johansson—for all our memorable adventures.

I must thank Corinne Farquharson-Grandjean, Corinne Degott, Evelyn Duperrex, and the many other members of EPFL's secretarial staff for their tremendous patience through my administrative neglect.

I offer my deepest appreciation to the members of my thesis committee: Prof. Boi Faltings, Prof. Silvia Santini, Dr. Guillermo Barrenetxea Kobas, and Dr. Hendrik Huwald, for contributing their time and attention in their thorough review of my work.

Finally, to my two beautiful sisters, whom I have had the great pleasure of watching grow into young adults, in spite of geographical distance. Thank you for keeping me involved in your lives during my short absence.

Emily, Vreni, José: to some I owe too much for words. Thank you.

Lausanne, 05 June 2015 William C. Evans

Abstract

SCIENTIFIC reports from around the world present us with the undeniable fact that the global ecosystem is undergoing severe change. As this shift accelerates, it is ever more critical that we are able to quantify the local effects of such changes, and further, their implications, from our daily life to the biological processes that put food on our tables.

In this thesis, we study the application of sensor network technology to the observation and estimation of highly local phenomena—specifically at a scale between ten to several hundred square meters. Embedding knowledge about the observed process directly into the sensor nodes' behavior via dedicated resource management or control algorithms allows us to deploy dense networks with low power requirements.

Ecological systems are notoriously complex. In our work we must thus be highly experimental; it is our highest goal that we construct an approach to environmental monitoring that is not only realistic, but practical for real-world use.

Our approach is centered on a commercially available sensor network product, aided by an off-the-shelf quadrotor with minimal customization. We validate our approach through a series of experiments performed from simulation all the way to reality, in deployments lasting days to several months.

We motivate the need for local data via two case studies examining physical phenomena. First, employing novel modalities, we study the eclosion of a common agricultural pest. We present our efforts to acquire data that is more local than commonly employed methods, culminating in a six month deployment in a Swiss apple orchard. Next, we apply an environmental fluid dynamics model to enable the estimation of sensible heat flux using an inexpensive sensor. We integrate the sensor with a wireless sensor network and validate its capabilities in a short-term deployment.

Acquiring meaningful data on a local scale requires that we advance the state of the art in multiple aspects. Static sensor networks present a classical tension between resolution, autonomy, and accuracy. We explore the performance of algorithms aimed at providing

all three, showing explicitly what is required to implement these approaches for real-world applications in an autonomous deployment under uncontrolled conditions.

Eventually, spatial resolution is limited by network density. Such limits may be overcome by the use of mobile sensors. We explore the use of an off-the-shelf quadrotor, equipped with environmental sensors, as an additional element in a system of heterogeneous sensing nodes. Through a series of indoor and outdoor experiments, we quantify the contribution of such a mobile sensor, and various strategies for planning its path.

Keywords: environmental monitoring, sensor networks, mobile sensing

Résumé

Des études scientifiques du monde entier nous rapportent le fait indéniable que l'ensemble d'écosystème subi de sévères changements. Le phénomène s'accélérant, il est plus que jamais essentiel de quantifier les effets locaux de ces changements, et leurs implications, de nos vies quotidiennes jusqu'aux processus biologiques qui permettent de nous nourrir.

Dans cette thèse nous étudions la technologie des réseaux de capteurs et leurs applications pour l'observation et l'estimation de phénomènes hautement locaux, spécifiquement à une échelle entre dix et plusieurs centaines de mètres carrés. Intégration les connaissances des processus observé directement dans le logiciel du réseau de capteurs, à travers la gestion de ressources dédiée ou des algorithmes de contrôle, nous permettent de déployer des réseaux denses avec des besoins en énergie réduit.

Les systèmes écologiques sont notoirement complexes, raison pour laquelle notre travail est hautement expérimental ; notre but ultime n'est pas seulement de construire une approche réaliste de la surveillance de l'environnement, mais aussi applicable à un monde non académique. Notre approche utilise un système de réseau de capteurs disponible dans le commerce de même qu'un quadcoptère avec un minimum de modifications. Nous validons notre approche à travers une série d'expériences allant de la simulation au déploiement sur le terrain, avec des durées allant de quelques jours à plusieurs mois.

Nous motivons le besoin de données locales par deux études de cas examinant des phénomènes physiques. Premièrement, utilisant de nouvelles modalités, nous étudions l'éclosion d'un nuisible agricole commun. Nous présentons nos efforts pour collecter des données que ceux communément employés, débouchant sur un déploiement de 6 mois dans un verger de pommes en Suisse. Ensuite nous utilisons un modèle environnemental de dynamique des fluides permettant l'estimation de flux de chaleur sensible en utilisant un capteur peu onéreux. Nous intégrons the capteur dans un réseau sans fil et validons ses capacités dans un déploiement de courte durée.

Acquérir des données significatives à une échelle locale requière que nous fassions avancer

l'état de l'art dans plusieurs aspects. Les réseaux de capteurs statiques classiques mettant en tension les nécessités de résolution, autonomie et précision. Nous explorons la performance d'algorithmes visant à obtenir les trois, montrant explicitement ce qui est nécessaire pour implémenter ces approches dans des applications réelles dont le déploiement est autonome et les conditions ne sont pas contrôlées.

Éventuellement, la résolution spatiale est limitée par la densité du réseau. Ces limites peuvent être dépassées en utilisant des capteurs mobiles. Nous explorons l'utilisation d'un quadcoptère du commerce équipé avec des capteurs environnementaux comme un élément additionnel dans un système hétérogène de capteurs. À travers une série d'expériences, tant à l'intérieur qu'à l'extérieur, nous quantifions la contribution de ce capteur mobile et les diverses stratégies pour planifier son chemin.

Mots-clés : surveillance de l'environnement, les réseaux de capteurs, capteurs mobiles

Contents

Introduction Part I

1 Outline

SENSING and observation is the foundation on which modern science has built its understanding of our universe. The quest for this understanding is driven endlessly forward not just by our curiosity, but in the face of climate change, by will against regress.

Cydia pomonella, the codling moth, is an insect pest that afflicts fruit-bearing orchards around the world. Like most moths, they are primarily ectothermic, i.e., they rely on their environment for body heat, and their development speed and lifetime is thus heavily dependent on ambient temperature. The Valais region of Switzerland typically sees two full generations of the codling moth each year. That is, between the spring and fall, the codling moth will fly, mate, and reproduce (thus destroying precious fruit) twice before finally returning to hibernation.

Farmers in Valais take a wide range of precautionary measures to prevent the growth and spread of this pest, ranging from the application of pesticides to mating disruption, at risk of major crop losses. Codling moth management requires predicting the current life stage of the pest based on environmental indicators (primarily temperature) in order to deploy precisely timed countermeasures. In Switzerland, a national research organization called Agroscope performs this modeling, and issues an advisory to farmers, which, if followed, mitigates this source of crop damage.

However, the global environment is a complex dynamical system that is currently undergoing an accelerating shift. Rising global mean temperature is one of the most obvious symptoms of this change [1], and it is not without effect on our biological ecosystem. Indeed, a recent study has shown that at the current rate of change, increasing global temperatures will lead to Valais seeing *three* generations of the codling moth each year [2].

This is just an example of the many ways our changing environment affects biological processes [3], at least some of which the human race depends on for prosperity and even survival. These changes will test our understanding of these processes—and when our understanding fails, we must ultimately revert to sensing and observation.

1.1 Objective and Organization

Advances in sensing and communication technology during the last few decades have made it possible to directly observe or indirectly estimate a breadth of phenomena that we could previously reason about only theoretically. The falling costs and increasing autonomy of these technologies has led to additional interest in the study of local (spatial) variations in environmental processes.

This dissertation contributes to enabling the sensing of microclimate-scale environmental processes with high spatial density. We design, build and employ multi-sensor systems that—using intelligence embedded at the node level—observe and estimate real-world phenomena in such a fashion that takes into account system resources such as energy and per-station cost.

We divide this manuscript into five parts: an introduction, followed by three parts describing our experimental case studies, and finally concluding with a discussion and summary.

Part I: Introduction We first outline the core concepts and motivation behind our work. A qualitative framework is produced that incorporates the various themes of our research. Finally, this part describes the core toolset used throughout our work, including our hardware platforms, custom extensions, onboard software architectures and simulators.

Part II: Model-Driven Sampling Our first experimental part describes two concrete contributions of WSN technology to environmental monitoring, in which we leverage different physical models of a biological and an environmental fluid dynamic system to drive system design. We present and discuss field results from both applications.

Part III: Data-Aware Efficient Communication In this part we take a closer look at the cost of transmitting sensor data in a multihop WSN. We take a previously proposed efficient monitoring algorithm from theory to reality, and detail necessary modifications and optimizations needed to tailor it to a real-world environment.

Part IV: Hybrid-Mobility Sensor Networks In our final experimental case study, we consider the added utility of a mobile sensor operating in cooperation with static nodes. We develop a quantitative framework for modeling and estimating 2D spatial environmental fields with simple optical sensors, based on Gaussian process regression. We evaluate multiple strategies for guiding this sensor over an environmental field to fill gaps left by a static sensor deployment in a series of realistic simulations and real-world experiments.

Part V: Discussion The final part of this thesis gives a comprehensive discussion of the above case studies, providing lessons learned, summarizes our contribution and general outlook.

1.2 Contributions and Publications

Our primary contributions and associated publications are listed below.

Part II: Model-Driven Sampling We first explore the utility of environmental monitoring with high spatial density in two scenarios, each backed by an established physical model. The first is a biologically driven sensing modality—the presence of mating adult codling moths in an apple orchard. We present the integration of several technologies, in collaboration with Sensorscope, to build a system for not only detecting the moth's presence, but also predicting the life stage of the pest using locally gathered data. In collaboration with Agroscope, Switzerland's national agricultural research organization, we validate our models against local trap catches, and present a discussion of the limitations of our approach. Second, we use a recently proposed statistical technique to perform sensible heat flux estimation with high spatial density, fully integrating the approach with a commercial sensor network system. An underlying physical model of the phenomenon is used to drastically reduce the cost per node. We note that the reference deployment was performed in part by Hendrik Huwald, Chad Higgins, and Marc Parlange. Alexander Bahr assisted with hardware design and field deployment in both of the above case studies. Benjamin Bejar developed the moth recognition and counting algorithms used above.

- **W. C. Evans**, B. Bejar, A. Bahr, M. Vetterli, and A. Martinoli, "Monitoring and predicting the presence of the adult the codling moth with a camera-based sensor network.", 2015, in preparation.
- H. Huwald, C. Higgins, A. Bahr, **W. C. Evans**, M. Parlange, and A. Martinoli, "Sensible heat flux from wireless environmental sensor networks", *Journal of Atmospheric and Oceanic Technology*, to be resubmitted.
- A. Bahr, **W. C. Evans**, A. Martinoli, H. Huwald, C. Higgins, and M. Parlange, "Measuring sensible heat flux with high spatial density", in *2012 IEEE Sensors Applications Symposium (SAS)*, Feb. 2012, pp. 1–6.

Part III: Data-Aware Efficient Communication This chapter contains multiple experimental contributions in the area of efficient communication in sensor networks. We bring a selected efficient monitoring algorithm—CONCH—from literature to a real-world platform, detailing modifications to the algorithm necessary to make it feasible outside of simulation. To the best of our knowledge, we present the first results using this algorithm not only on real data, but in a real deployment. We proceed to present and evaluate DCONCH, a reworking of the original algorithm that is able to more quickly adapt to changing environmental patterns. Using a custom power monitoring extension board developed in collaboration with Alexander Bahr, in conjunction with calibrated simulations, we show that despite significantly reducing the total data transmitted, there is no measurable power savings, illustrating that suppression-based algorithms

are unlikely to be useful on real-world systems without major changes to commonly used network protocols.

- **W. C. Evans**, A. Bahr, and A. Martinoli, "Distributed spatiotemporal suppression for environmental data collection in real-world sensor networks", in *9th IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS)*, Cambridge, MA, USA, May 2013, pp. 70–79.
- **W. C. Evans**, A. Bahr, and A. Martinoli, "Evaluating efficient data collection algorithms for environmental sensor networks", in *Proceedings of the 10th International Symposium on Distributed Autonomous Robotic Systems*, ser. Springer Tracts in Advanced Robotics, vol. 83, Springer Berlin Heidelberg, 2013, pp. 77–89.
- **W. C. Evans**, A. Bahr, and A. Martinoli, "A flexible in situ power monitoring unit for environmental sensor networks", in *IEEE/RSJ IROS Workshop on Environmental Monitoring (WREM)*, Oct. 2012.
- A. Prorok, **W. C. Evans**, and A. Martinoli, "An adaptive field estimation algorithm for sensor networks in dynamic environments", in *Workshop on Robotics for Environmental Monitoring (WREM); International Conference on Intelligent Robots and Systems (IROS)*, Sep. 2011.

Part IV: Hybrid-Mobility Sensor Networks In this chapter we propose a novel testbed for evaluating the usefulness of a mobile sensor operating among static nodes in a WSN. We again select a commercially available platform, and design a comprehensive approach for using said platform to construct a two-dimensional field estimate with a given sensor. We compare the performance of various path planning strategies in cooperation with preexisting static nodes in a calibrated simulator and in our indoor testbed. Both Steven Roelofsen and Jnaneshwar Das assisted with the experimental aspect of this effort.

- **W. C. Evans**, S. Roelofsen, P. Flikkema, and A. Martinoli, "Environmental field estimation with a low resolution sensor", in *Proceedings of the International Conference on Intelligent Robots and Systems (IROS 15)*, Hamburg, Germany, Sep. 2015, submitted.
- J. Das, **W. C. Evans**, M. Minnig, A. Bahr, G. Sukhatme, and A. Martinoli, "Environmental sensing using land-based spectrally-selective cameras and a quadcopter", in *Proceedings of the Thirteenth Int. Symp. on Experimental Robotics*, ser. Springer Tracts in Advanced Robotics, vol. 88, Springer International Publishing, 2013, pp. 259–272.

2 Approach and Challenges

DENSE static networks and mobile sensors present complementary means of sensing and estimating phenomena with small-scale spatial variability. In evaluating the application of these means to various scenarios, our foremost goal is to keep our approach not only grounded in reality, but practical for implementation and reuse. Thus, whenever possible, we build our systems around inexpensive, commercially-available platforms, and evaluate our work in real-world environments.

This chapter proceeds as follows. In Section 2.1, we provide background on the various themes in our work. In Section 2.3, we describe the guiding principles of the approach we take throughout the rest of the manuscript.

2.1 Background

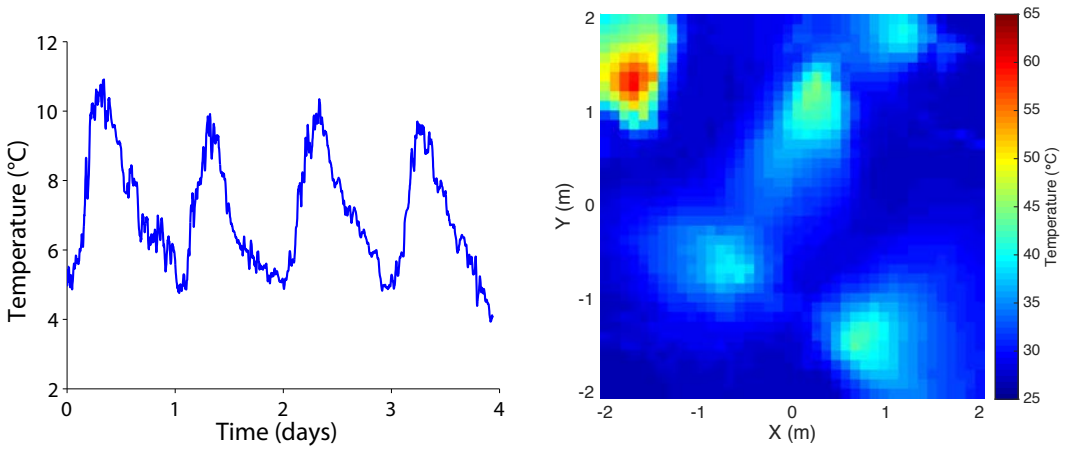
Our work incorporates two primary themes: environmental monitoring and wireless sensor networks.

2.1.1 Environmental Monitoring

By definition, environmental processes have an inherent spatial structure—consider Wardo Tobler's famous quote, "Everything is related to everything else, but near things are more related than distant things." [13].

2.2 Sensor Classification

See Figure 2.2. We classify sensors as remote versus *in situ*, active versus passive. Passive sensor—a photographic camera—versus its active counterpart, a camera with a flash.



(a) (b)

Figure 2.1 – Environmental data is by definition well-correlated over space and time. For example, (a) ambient temperature changes smoothly throughout the day, and (b) over space, composite thermal image of an arrangement of electrical heaters.

	Remote	*In situ*
Active	• RADAR • SONAR • Laser spectroscopy • Flash photography	• Ultrasonic anemometers • Sap flow meters • **Mobility**
Passive	• Photographic cameras • Infrared thermometers	• Thermistors • Thermocouples • Elecrochemical gas detectors • pH electrodes

Figure 2.2 – Sensors can be classified as either active or passive, remote or *in situ*.

In situ

In situ sensors are defined by their physical contact with the phenomena under measure.

Remote

Remote sensing, in contrast to the above, measure information about a phenomena from a distance, often by collecting reflected light.

Active remote sensors operate by emitting a kind of signal into the environment and measuring its reflection, e.g., as in the case of flash photography. Passive remote sensors collect a signal transmitted from an external source, such as sunlight.

2.3 Approach

In this thesis, we take a distributed intelligent sensing approach to measuring environmental process with high spatial density.

2.3.1 Distributed Sensing

Traditionally, much of environmental monitoring is done at a large spatial scale, using (possibly extremely) expensive and high quality sensors, e.g. the aforementioned MeteoSwiss weather stations or Earth observing satellites. In this section we discuss a distributed sensing approach: the use of many less expensive (and generally less accurate) sensors, distributed over a smaller area.

Particularly for immobile, *in situ* sensors, distributed sensing offers the only route to achieving any kind of spatial resolution. Remote sensors may also benefit—multiple nodes can contribute additional coverage or improved resolution.

However, this shift to a distributed architecture involves multiple tradeoffs. First and foremost, in order to feasibly deploy a large number of stations, they must be proportionally less expensive as compared to their monolithic counterparts. This puts a great pressure on minimizing energy storage and energy harvesting—active, power hungry sensors are thus difficult to justify in our proposed scenario. Radio usage must be optimized. Perhaps counterintuitively, this may lead to increased complexity due to for instance multihop networking with multiple radios (long range versus short range), ultra low power protocols, power-aware sensing strategies, and/or in-network data compression.

While energy harvesting techniques, such as solar power collection, may be employed, minimizing the power consumed by a sensor node is of critical importance. Again, particularly when considering the deployment of a dense network of many nodes, equipping each with a solar panel and large battery may be cost prohibitive. Further, it may make the stations too

Figure 2.3 – The SwissEx Wannengrat deployment. Harsh and inaccessible environments make it difficult to adjust the network over time.[1]

bulky to deploy in some scenarios, such as remote areas reachable only by helicopter (i.e., due to limited or expensive cargo space, as in the Wannengrat deployment, see Figure 2.3).

2.3.2 Intelligent Sensing

As hinted above, high node density often implies increased node complexity. As more nodes are concentrated in a single area, embedded intelligence is required due to power concerns. Intelligence is cheap compared to actuating a costly sensor or a radio—making local decisions about when to take a sample and what to communicate about that sample (or indeed, whether to communicate anything at all) is vital.

Intelligent nodes are nearly a necessary condition for the distributed approach—nodes must cooperate in order for communication to be scalable (in terms of power efficiency/cost).

Mobility and its associated costs change the picture dramatically. Typically speaking, and certainly for all of the modalities we consider in this thesis, the energy consumed by the act of self-locomotion is orders of magnitude greater than that consumed in the act of sensing. Thus, when we consider the optimization of a mobile sensor, we are no longer interested in when we sample, but where. Such devices may have an autonomy as low as fifteen minutes, during which time they must sample an area that is very large, or very dynamic, as compared to their speed. For mobile sensors, we are thus interested in computing a path that maximizes the utility of the sensor data collected during this brief period.

[1] Photograph by Walter Steinkogler.

3 Core Tools

THE experimental approach necessitates that we adopt a stable yet flexible set of tools. These two qualities, stability and flexibility, are often in competition—building a system from scratch, for example, allows us to exactly tailor our platform to our research trajectory. However, developing and relying on a bespoke solution not only incurs significant cost, it also limits the impact of the achieved result. We thus constrain our search to commercially available and mature platforms that still allow us to modify and extend their internals.

Ultimately, we select one static and one mobile sensor platform for our work: Sensorscope, a commercially available static sensor network product [14], and the Ascending Technologies (AscTec) Hummingbird [15], [16], a miniature quadrotor capable of carrying both a small sensor payload and extra computational resources. The rest of this section is devoted to motivating these choices with respect to their alternatives, and detailing the ways in which they have been extended for our needs.

3.1 Static Platform: System Overview

Sensorscope is a commercial platform for environmental monitoring which has seen extensive use in scientific deployments (e.g., [17]–[19]). At the time we chose to build on the Sensorscope platform, it was still rapidly evolving; in the interest of stability we froze on Sensorscope V2.3.0 (2010 era), and the rest of the section describes this particular revision of the Sensorscope system. We were privileged with hardware schematics and source code for the stations and server, which enabled us to extend the platform to suit our needs.

We note that even static sensor networks have extremely broad applications, and limiting our approach to any single platform is thus unlikely to yield universal results. We discuss this issue in further depth and describe the specific limitations imposed by our choice in platform in Section 3.1.2. We mitigate this issue by choosing a platform that has seen repeated use in actual scientific deployments in a variety of environments.

Figure 3.1 – Sensorscope station on the rooftop of EPFL's GR building.

3.1.1 Sensorscope Station

Here we describe Sensorscope's software and hardware architecture as far as it relates to our work. A broader look at the Sensorscope system can be found in [20], [21].

Individual stations are typically deployed with a number of environmental sensors (see Figure 3.1) attached in a daisy chained fashion from a central collection and logging board, henceforth referred to as the *datalogger*. The datalogger also features a short-range radio, which it may use to form a multi-hop network with other nearby stations, and optionally a long range GSM transceiver for sending data out of the network. Sensor detection and network configuration is performed autonomously when a station is activated, providing out-of-the-box operation, a key feature for environmental scientists.

Typical deployments number in the range from one to twenty stations, however, Sensorscope has seen short-term scientific deployments of up to 92 nodes [17], and currently maintains a multi-year 52 station deployment in Valais, among others. Nodes in the network organize in a tree topology descending from the nearest station with a GSM radio, i.e., the sink, with which they time-synchronize and to which they forward all collected sensor data.

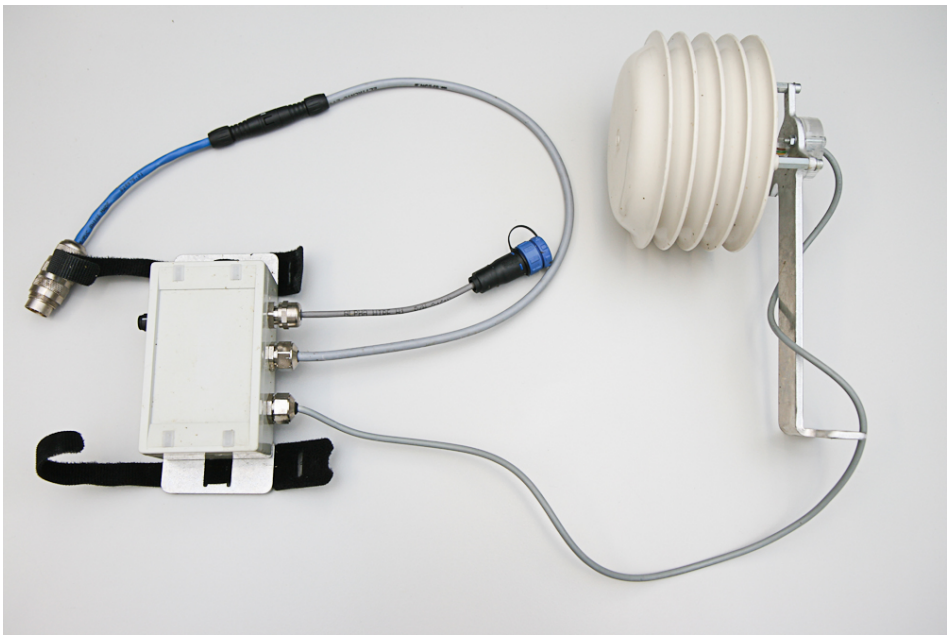


Figure 3.2 – (left) Sensor module and (right) its attached Sensirion SHT75 digital temperature/humidity sensor.

Hardware Description

The stations are built around the datalogger, a central processing board that is responsible for collecting data from its sensor bus while simultaneously maintaining a multi-hop network with its neighbors. The datalogger contains ShockFish TinyNode 584 [22], running a software suite based on TinyOS 2.x. Local communication is performed in the 868 MHz band using a Semtech XE1205 radio transceiver, with a range approaching one kilometer given clear line of sight. Optional long range communication is performed via a Telit GM862-GPS GPRS modem. Sensor measurements may be stored to an onboard microSD card. The electronics are protected by an environmentally sealed container with external connectors for the antenna, sensor bus and solar panel.

Sensor Loadout

Each sensor is attached to its own module, separate from the datalogger, which contains its own microprocessor that supports components for sensor sampling and hardware for driving the SPI bus (see Figure 3.2). This architecture imposes very little constraints on the sensor itself, allowing sampling and intermediate processing to be done asynchronously with the operation of the datalogger. Periodically, all sensors are polled for their latest measurements, which are sent to the datalogger for storage and transmission out of the network.

Sensorscope supports a variety of sensors out of the box. Here we summarize the characteristics of the sensors used in this thesis, barring custom built sensors we describe in later

chapters. We use the Davis solar radiation sensor, Davis anemometer, the Sensirion SHT11 or SHT75 digital temperature/humidity (for availability reasons). The sensor itself is housed in a layered radiation shield that serves to protect the sensor from the elements while still allowing accurate sensing (see Figure 3.2). The corresponding sensor module runs on TI's MSP430F149 microprocessor.

In-network Algorithm

Sensorscope employs two types of stations: slaves, which communicate only over their short range radio, and masters (i.e., sinks), which collect data from all connected slaves and transmit that data to an off-site server via a cellular network. In the following section we describe the in-network algorithm that enables slaves to forward their data to the sink, and the operation of the long range radio in transmitting data out of the network.

Each station, slave or master, executes the same algorithm for sensor sampling and local communication. The temperature sensor is sampled every 60 seconds. In order to mitigate per-transmission overhead, these sensor values are aggregated into a single packet until either the packet is full (maximum 48 byte payload) or until the packet is ten minutes old (i.e., due to infrequent sampling). Sensor values are timestamped by the datalogger, storing the packet's first measurement's full timestamp (i.e., seconds since the epoch), while only storing the delta with subsequent samples. In our system, in order to get a clear picture of the effect of data suppression, only the temperature is transmitted over the network. Other sensed data, such as network statistics, power consumption, and battery status are exclusively written to the onboard microSD card.

The short-range radio is duty-cycled, turning on every two minutes for a variable window of time. This window is between 4 and 60 seconds long (with two seconds of padding), depending on how much the station and its neighbors usually send each cycle. The window length is adjusted by maximally one second per cycle, until between 60–80% of the frame is spent either sending or receiving packets. Sensorscope uses the Carrier Sense Multiple Access (CSMA) MAC protocol for packet transmission, and all sensor data is sent from station to station in a reliable fashion (i.e., with packet acknowledgments).

Each node with a route to the sink sends a beacon at the beginning of every radio cycle. The beacon contains information for routing, time synchronization and adjustment of the frame length, and is additionally used to estimate link quality with neighboring stations. Each station tracks the number of beacons received by its neighbors during the last 16 radio cycles. A station must have received at least 11/16 of the most recent beacons from a neighbor to consider it for routing purposes.

GPRS

Each network may contain multiple master stations, each with a GPRS module for off-site data transmission. Every fifteen minutes, master stations activate their GPRS modules and connect to a preprogrammed server via the Internet. On successful connection, the station authenticates itself to the server using a pre-shared key. It receives a global time reference, and updates its clock accordingly. Data packets are then combined and sent as 3500 byte blocks, but are not interleaved, i.e., each subpacket contains sensor data from a single station maximally spanning over ten minutes, and each subpacket is packed back-to-back. The station waits as long as 10 seconds to receive an acknowledgment from the server before sending its next block.

GPRS operation is done out of phase with the short-range radio. Operating in our suburban environment, the GPRS module typically takes 25–40 seconds after being powered on to connect to the external server, which leaves about between 20–90 seconds for data transmission before the next short-range radio phase begins. We observe a typical speed of 2–3 kbps, allowing about 6000 sensor measurements to be transmitted per cycle per master (assuming a 4 byte payload).

3.1.2 Limitations

Sensorscope is a well-suited platform for environmental monitoring with high spatial density in the scenarios we consider in this manuscript. However, it may not suit applications that require a high datarate, or that have real-time constraints on latency.

3.2 Static Platform: Power Monitoring

While software performance logging allows us to monitor an algorithm's performance (in terms of the reduction in transmitted data, for example), we are ultimately interested in the actual power consumed by each station. We have thus developed an extension board for the Sensorscope datalogger that allows for *in situ* power monitoring, henceforth referred to as the *power board* [9].

3.2.1 Related Work

Power monitoring is a highly sought-after feature for WSNs. iCount [23] and SPOT [24] provide energy monitoring functionality. Aveksha [25] is a more featured extension board developed for the TelosB platform, providing energy monitoring and software tracing. By making minor modifications to the TelosB mote, they are able to take advantage of the microcontroller's debug features to provide this functionality in a non-intrusive fashion.

Our approach is unique in that it is closely integrated with a system designed for use outside

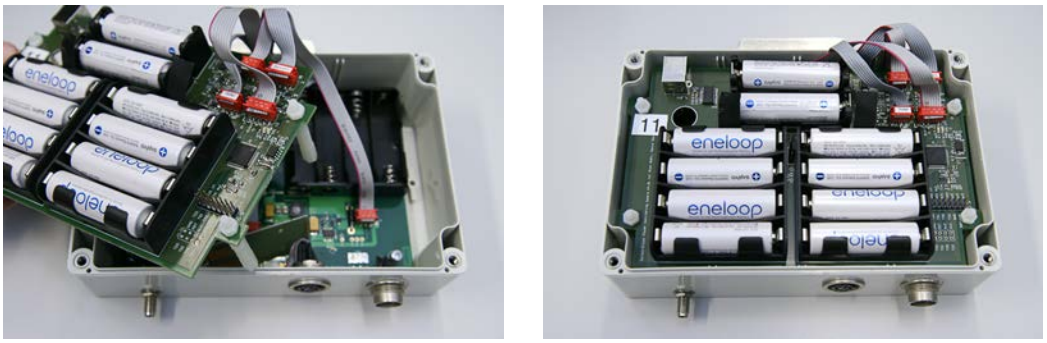


Figure 3.3 – The power board fits inside the datalogger over the original board.

of laboratory settings, with the capability to separately monitor incoming and outgoing power for different subsystems.

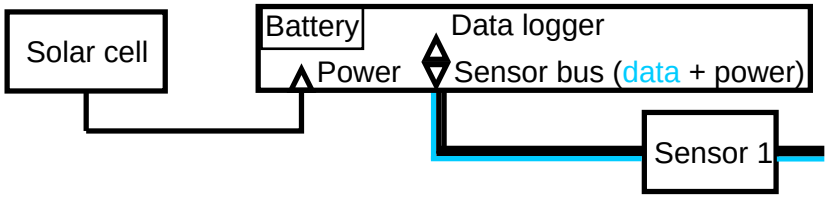
3.2.2 Power board

We have developed a power monitoring extension board that integrates seamlessly with Sensorscope's existing architecture. The board logs power consumption data to local storage as well as over the network (like any other sensor in the Sensorscope system). It is also capable of providing real-time information to the station itself, which has applications beyond performance evaluation, such as power-aware networking algorithms or advanced charge controllers.

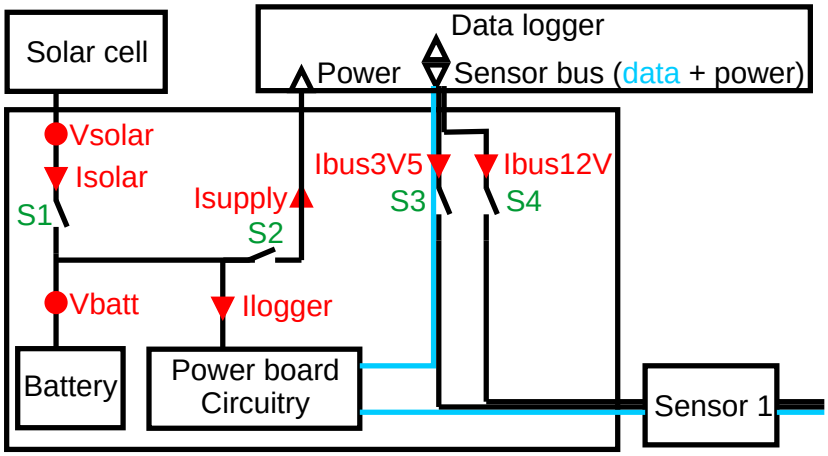
Sensorscope Integration

The power board fits inside the datalogger's enclosure, resting over the main board, intercepting the ribbon cables connecting its power source and sensor chain (see Figures 3.3 and 3.4). This approach allows the power board to be used without requiring any mechanical or electrical modification to existing stations. By measuring the current passing through these ribbon cables, the power board is able to monitor the station's complete energy budget, including incoming energy from the station's solar panel, and energy used by the datalogger, sensor chain, and the power board itself. In addition to power monitoring, the power board is able to disconnect the attached solar panel, allowing the implementation of a software-based charge controller.

The power board has its own microcontroller (TI's MSP430F1611), which allows it to function similarly to other sensors on the bus. It reports measured power data to the datalogger via the SPI bus, as per any other sensor. However, although current measurements are taken at 5 Hz, this bus was not intended for streaming sensor data. Thus, the power board reports the mean, minimum and maximum values over each reporting period (one minute in our case). For *ex situ* experimentation, a UART port is included on the power board, allowing one to record high resolution data (at 115200 baud) as is shown later in this document.



(a) Sensorscope datalogger.



(b) Sensorscope datalogger with power board.

Figure 3.4 – Block diagram of the Sensorscope datalogger (a) alone and (b) with the power board intercepting incoming and outgoing connections.

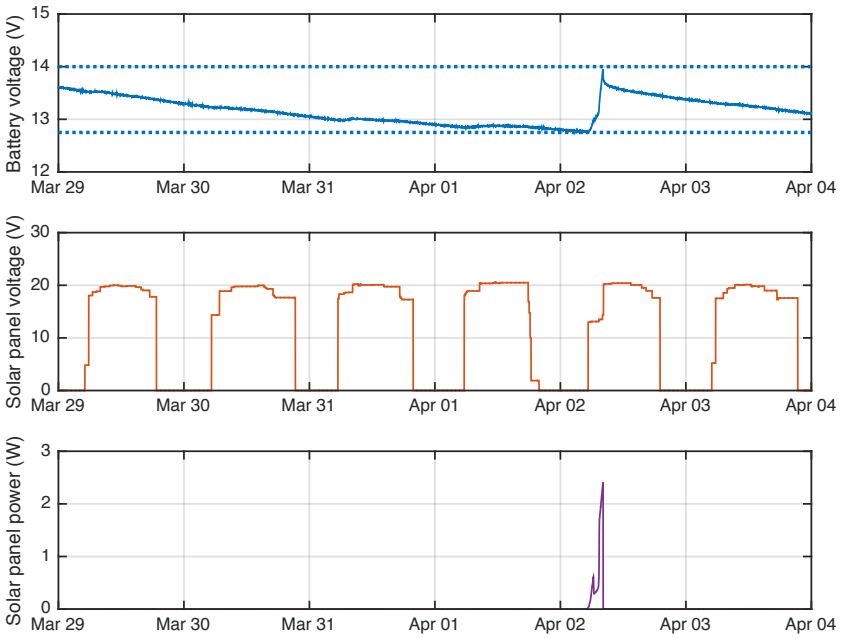


Figure 3.5 – Power board data from one station during normal deployment. Here we observe the behavior of our voltage-threshold charge controller. Upper and lower thresholds marked as the dotted lines at 14.0 V and 12.8 V, respectively. The charging period corresponds to the spike in incoming solar power.

We performed a ten station deployment around EPFL's campus in order to evaluate the power board's robustness. Selected results from this deployment can be seen in Figure 3.5. When the battery voltage falls below the charging threshold, the power board connects the solar panel and solar current increases dramatically as the batteries recharge. The upper threshold is reached in under an hour, and the solar panel is again disconnected.

We have also used the board to characterize the power consumption of various station activities, as seen in Figure 3.6. Note that the consumption measured by the power board closely matches that measured by current probe. The power board has limited temporal resolution compared to our oscilloscope, and does not capture the effect of the datalogger's switching voltage regulator, among other factors.

Validation

While the power board was designed to give quantitative results regarding algorithms deployed in real-world environments, such experiments are time consuming and may yield dramatically different results between runs. It is thus critical that we leverage power board data to estimate power consumption during repeated simulations over the same datasets.

The above Sensorscope characterization allows us to add realistic power consumption estimates to TOSSIM [26] simulations. In Figure 3.7 we present a comparison of simulated power

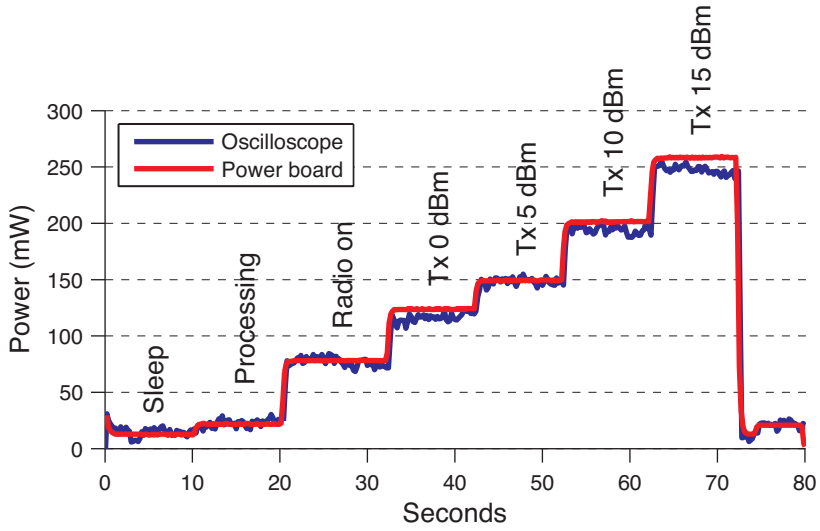


Figure 3.6 – Power consumed by the datalogger while sleeping, processing, radio on, and cycling through various transmission power levels. The power board has limited temporal resolution compared to the oscilloscope, and does not capture the effect of the datalogger's switching voltage regulator (note the ripple in the oscilloscope curve above), among other factors.

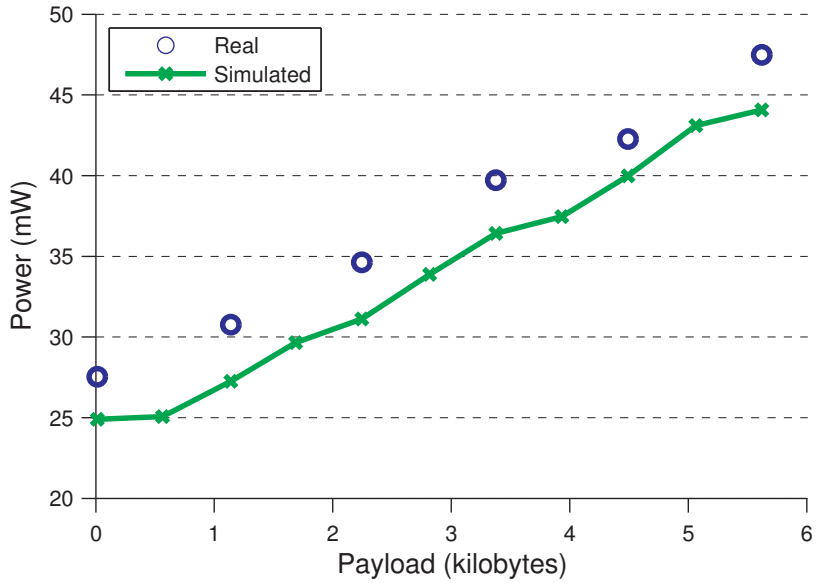


Figure 3.7 – Comparison of real-world versus simulated power consumption as a function of per-frame payload. Each data point represents the power consumed during one entire radio cycle (two minutes), averaged over a 14 minute period.

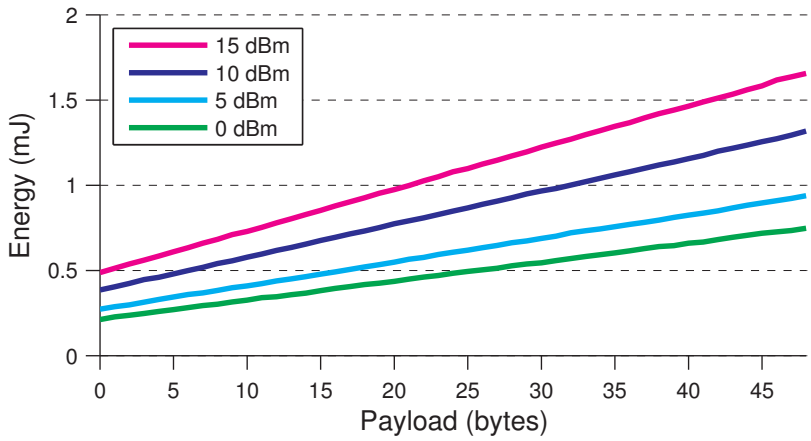


Figure 3.8 – Energy consumed by TinyNode radio transmission by packet payload size and transmission power. Packets of increasing payload size from 0 to 48 bytes were sent via the TinyNode's Semtech XE1205 radio, while power was logged by oscilloscope.

consumption versus data measured by the power board on a real station. The station under measure operated as a typical Sensorscope slave station (i.e., without GPRS), transmitting between zero and six kilobytes of sensor data per cycle. The monitored station is one hop from the sink, with no other network neighbors. We observe a strong match between simulated and real-world performance.

3.2.3 Station Characterization

In order to facilitate realistic simulation and to give context to our testbed results, we first characterize the power consumption of the Sensorscope system.

An overview of the cost of various station activities, as measured by the power board, can be seen in Figure 3.6.

We used a LeCroy WaveSurfer 24Xs-A oscilloscope to measure energy usage during packet transmission, as the power board's maximum effective sampling rate of 5 Hz is far too low to capture it. Due to the presence of large capacitors onboard the data logger, we measured the energy used by the TinyNode in isolation. Using this data we were able to compute the energy used during packet transmission as a function of payload size and transmission power (see Figure 3.8).

3.3 Static Platform: Software and Simulation

Simulation is critical to algorithmic performance evaluation for multiple reasons. First, debugging sensor network algorithms is extremely difficult *in situ*. As our target environment is remote, features such as wireless reprogramming and live debugging are difficult to realize.

Second, in order to truly compare different network architectures, it is important that we have an environment that allows us to test algorithmic variations under repeatable (but realistic) circumstances.

3.3.1 TinyOS

TinyOS is an event-based programming language [27], commonly described as a superset of C. Both Sensorscope and the power board are programmed using TinyOS, and thus both also benefit from its simulation framework, TOSSIM [26].

3.3.2 Simulation Environment

We added simulation support to the Sensorscope codebase using TOSSIM, a realistic simulator for TinyOS [26]. TOSSIM only natively supports the MICAz platform, so adding simulation support meant essentially porting the Sensorscope framework to the MICAz. As a result of this work, we are able to run the exact same high-level code both in simulation and on the real stations.

In this thesis, our simulations are driven by real data collected during our outdoor experiments—i.e., by "replaying" network connectivity and sensor data as logged during our real deployments.

Calibration

In order to estimate the efficiency of different network algorithms, we implemented a simple method for simulating power consumed by the radio. We characterized the TinyNode's short-range radio (see Section 3.2.3 for details), and input to the simulation of the energy required to transmit a packet depending on the length of its payload. We added a layer to the TinyOS radio module that records each duration the radio is turned on along with the size of each packet transmission and the corresponding station's identifier. This calibration process enables us to accurately estimate the per-station radio-related power costs.

3.4 Mobile Platform: System Overview

Mobile sampling requires the integration of a number of technologies. In this section we describe and motivate our choices for each piece of our platform—from the quadrotor-based mobile node to its sensor loadout, from the real-world experimental arena to our realistic simulator, to the software architecture that encompasses the whole.

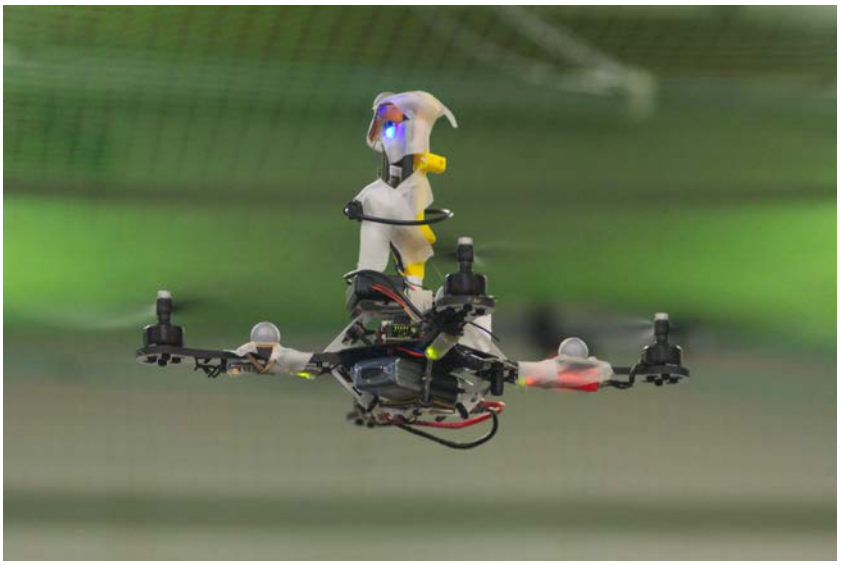


Figure 3.9 – AscTec Hummingbird in flying arena. The vertical tower is used to mount a Gumstix Overo Airstorm, which is used for general computation, communication, and sensor data collection. Reflective spheres are used for indoor localization.

3.4.1 AscTec Hummingbird

Our mobile sensor is built on the Ascending Technologies Hummingbird quadrotor [16]. Quadrotors make for versatile research platforms. Their maneuverability allows them to fly easily in both indoor and outdoor environments. Quadrotors have very few moving parts (e.g. as compared to a helicopter), which makes them remarkably robust to crashes. When they do take damage, parts are easily replaced.

The Hummingbird is a small, commercially available aircraft equipped with four brushless motors for flight (see Figure 3.9). It weighs approximately 510 g with battery out of the box, and has a flight time of 15–20 minutes, and has a maximum airspeed of 15 m/s. It is capable of carrying a payload of up to 200 g.

We chose the Hummingbird particularly for its usability and extensibility. It benefits from Markus W. Achtelik's `asctec_mav_framework` package [28], which allows it to interface with any computing hardware that supports the Robot Operating System (ROS) [29]. ROS is a widely-used robotics middleware that primarily provides hardware abstraction and message-passing based composition of various software components.

Two computers are present on the stock Hummingbird, termed the low-level processor (LLP) and high-level processor (HLP). The LLP primarily runs the motor control loop, while the HLP controls locomotion at a longer timescale, e.g., by directing the LLP to a series of GPS waypoints. Using the aforementioned `asctec_mav_framework` package, we have interfaced our computational platform—a Wi-Fi enabled, ARM-based Gumstix board—directly to the HLP via serial link. This configuration allows us to log the LLP/HLP's internal state, such as

the quadrotor's estimated pose and GPS position, as well as to issue movement commands on the fly, e.g. as generated by a path planning algorithm.

3.4.2 Limitations and Alternatives

While the quadrotor makes for a flexible research platform, it has a number of limitations. Foremost, the limited payload and flight time are somewhat inherent to the platform—their rotor blades have a fixed pitch, and are all facing the same direction, so both lateral thrust and yaw rotation are generated by changing the speed of individual rotors. Larger rotors quickly become inefficient due to their greater momentum during flight.

There exist a number of multirotor craft that offer larger payload capacity by simply increasing the number of onboard rotors.

Fixed-wing aircraft are very efficient, sacrificing maneuverability for longer flight time and simplicity of construction. The senseFly eBee [30] is one such popular drone, used largely for aerial surveys. However, controlled indoor testing is obviously difficult.

The Hummingbird is an appropriate platform for our envisioned use case: microclimate-scale sensing over an area one square kilometer or less.

3.5 Mobile Platform: Onboard Sensing

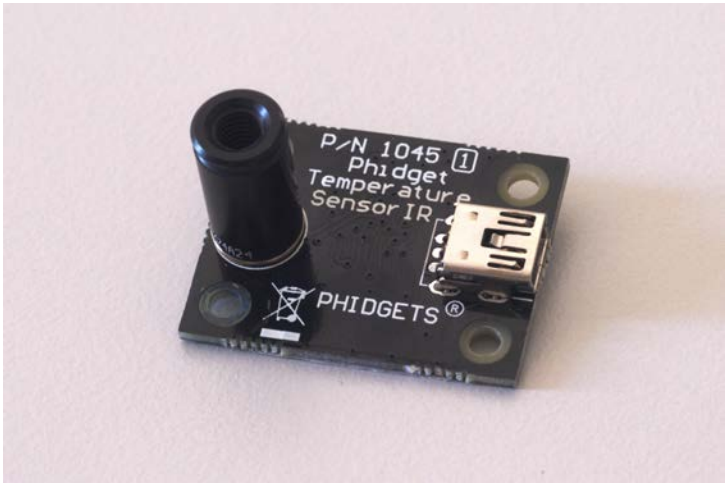
We integrate two different optical sensors with our mobile platform: an infrared thermometer and a miniature spectrometer. Both sensors quantify collected light over a defined field of view (FOV). Both sensors have been integrated with our flying platform, as seen in Figure 3.13.

3.5.1 Infrared Thermometer

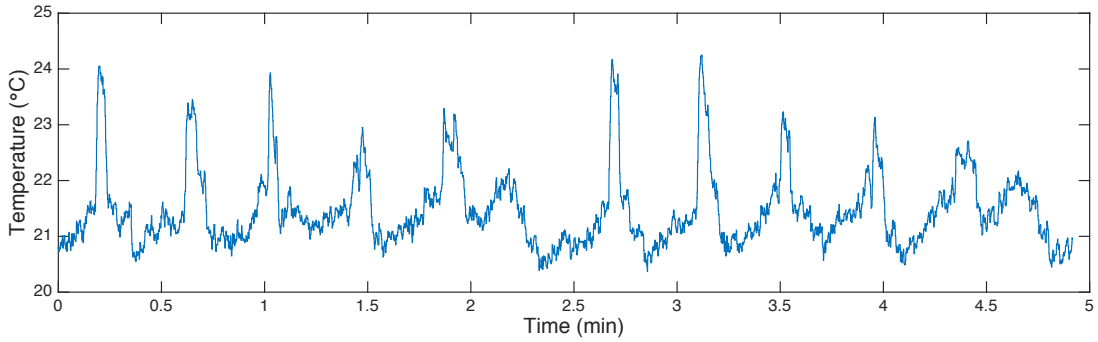
Phidget 1045 infrared temperature sensor [31]. Provides scalar surface temperature measurements at 31 Hz. Interfaced with the onboard Gumstix via USB. 10° FOV, with a listed error of ±2 °C.

In order to characterize the sensor, we attached it to a traversing system alongside the thermal camera (see Figure 3.11a). We performed a systematic perpendicular scan over a small heat source and measured the Phidget's response as a function of the lateral distance between the them. We found that the sensor's response—in remote sensors such the Phidget, this is often referred to as the *point spread function* (PSF)—could be modeled as a Gaussian weighted average over the area covered by the sensor (see Figure 3.11b). This matches published literature on PSFs for electro-optical sensors [32].

Note that in the absence of highly accurate global positioning, i.e., in an outdoor scenario, position error could be integrated into the sensing model, by increasing the width of the

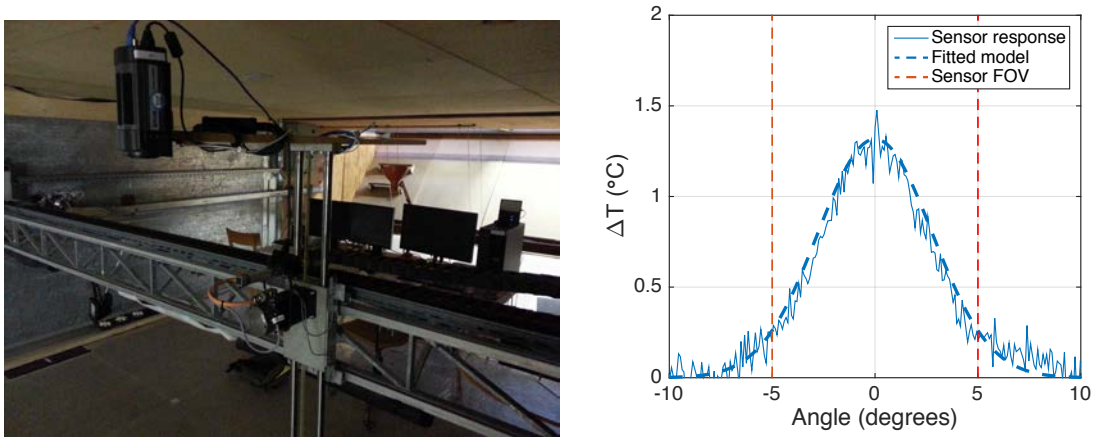


(a)



(b)

Figure 3.10 – (a) Phidget 1045 Infrared Thermometer and (b) example of measured data over the course of an indoor experiment.



(a) (b)

Figure 3.11 – (a) Thermal camera and Phidget infrared thermometer mounted on traversing system for sensor characterization. The traversing system allows for a fine scan with minimal vibration. (b) Sensor response while downward-facing and swept over a heat source at 1.62 m height. The x-axis gives the angle between the sensor and the heat source. The vertical red lines represent the sensor's FOV as stated in its datasheet.

weighting function.

3.5.2 Miniature Spectrometer

Hamamatsu C12666MA miniature spectrometer. We model this sensor as above, simply with an FOV of 25.4°, as defined by the sensor's datasheet [33]. The C12666MA measures incoming light between 340–780 nm—across the visible spectrum, and into the near infrared, with a spectral resolution of approximately 15 nm (see Figure 3.12).

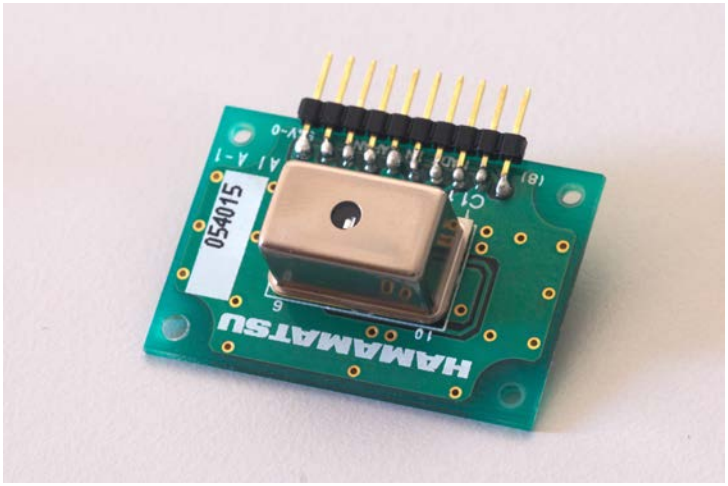
3.6 Mobile Platform: Software and Simulation

In this section we present our software framework for controlling the quadrotor, as well as our simulation environment.

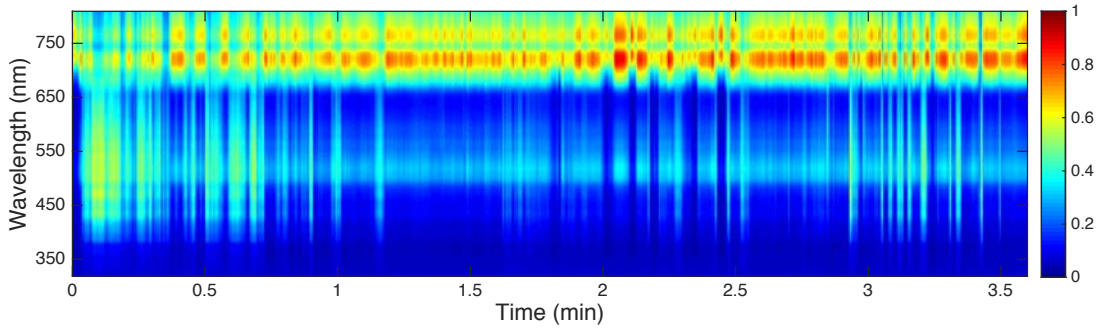
3.6.1 Software Framework

We have designed and implemented a modular MATLAB-based estimation and planning framework, with the goal of supporting rapid iteration on algorithmic design by reusing code as much as possible between simulation, controlled experimentation, and outdoor flights.

Our software platform is comprised of five modules that each encapsulate a different stage of the measurement and estimation process: the strategy, the sensor state, the sensing model,



(a)



(b)

Figure 3.12 – (a) Hamamatsu C12666MA miniature spectrometer and (b) example of measured data over the course of an outdoor flight. Color axis represents normalized spectral intensity.

(a) Mounted infrared thermometer, marked in red.

(b) Mounted spectrometer, marked in red. Intermediate sensor board marked in green.

Figure 3.13 – Sensors mounted on flying platform. (a) The infrared thermometer reports measurements directly via USB, while (b) the spectrometer is sampled via a custom protocol, which we implemented on a PRismino. The PRismino serves as an intermediate sensor board, in turn reporting spectrometer data to the quadrotor's Gumstix over USB.

the ground truth and the estimator. In brief, the ground truth represents the underlying phenomena, which the sensing model observes at the current location of the sensor, as given by the sensor state. The estimator is then responsible for fusing these observations into a single cohesive estimate. Finally, the strategy guides the mobile sensor by choosing a target position for the sensor to move toward. Below, we describe each of these modules and give examples of the various implementations of each. A comprehensive illustration of the process is given in Figure 9.1.

Ground Truth The ground truth module represents the underlying environmental field under measure by a discretized grid of points. During execution, the ground truth module responds to spatial queries from the sensing model—returning a discretized list of locations that match the query, and their corresponding field values. Depending on implementation, these field values may be taken from the mobile sensor itself, in real-time, or from a simulated or previously measured field.

Sensing Model The sensing model relates the ground truth to the sensor as a function of its pose. This module submits spatial queries to the ground truth, and returns the relationship between points on its discretized grid and the current measurement, i.e., the observation matrix.

Estimator Finally, the estimator module uses the observation matrix to attribute a measured value to grid locations, where it is fused with any prior collected data. This module has two outputs: the estimated value at each estimated grid point, and an associated variance.

Strategy The strategy module is responsible for planning the sensor's path. The implementations we propose may depend on the state of the estimator, e.g., we might choose to sample in areas where the estimation variance is high. In some cases, they may function only on their own internal state—its progress through a fixed-grid sampling pattern, or in the case of a random walk.

The selected position is sent to the sensor state module as its new target.

Sensor State The sensor state module represents the position and orientation of the sensor. During an experiment, it receives asynchronous updates from the mobile sensor over the network in real-time. It may alternatively be read from the filesystem, or it may be updated as the output of a motion model, in the case of a simulation.

3.6.2 Realistic Simulation

Real-world experiments with mobile sensors are expensive. The measured phenomena are highly stochastic, and configuring each experiment takes considerable time and effort—between registering the ground truth, charging and replacing batteries, and processing the resulting data, each individual experiment may take upwards of one hour.

Given that the processes we intend to measure are highly stochastic in nature—and further, that the difficulty of generating a statistically diverse set of fields to measure—a high quality simulation environment is absolutely necessary in order to quantify the performance of our algorithms. In the state of the art, authors often rely on a small number of datasets collected via remote sensing or static networks, then interpolating the data in order to produce a virtual ground truth to be sensed by simulated mobile nodes. This anecdotal flavor of experimental validation is difficult to generalize, as it is unclear how e.g. a mobile sensing strategy might perform in many of the diverse applications for which it may be useful.

Our simulation engine was built around the core software framework described in Section 3.6.1. The design of our simulator specifically focuses on a flexible sensing model and support for generating statistically diverse random environmental fields. Note that we did not invest effort in accurately simulating the sensor's dynamics in terms of mobility—it is our intuition that in our intended use case, both the physical size of the measured area and the correlation scale of the phenomena under measure are large as compared to constraints on the motion of our flying platform.

Gaussian random fields. Gaussian random fields (GRFs) present a flexible means of generating interesting ground truth data. R, a popular programming language for statistical computing, has available a toolbox capable of generating GRFs with a wide range of underlying covariance functions—Schlather et al.'s *RandomFields* package [34], [35]. Our module interfaces with R in order to generate GRFs, subject to specified parameters, and pulls them into the MATLAB environment. Given the prevalence of GRFs in spatial statistics applications, simulating algorithmic performance over such fields makes our results directly generalizable to many different scenarios.

Existing datasets. Ground truth data may also be drawn from previously collected sensor data—either from our field deployments, or from remote sensing data.

Part II Model-Driven Sampling

4 Biological Model-Driven Sampling

CODLING moths (l. *Cydia pomonella*) are present on six continents, and are capable of doing severe damage to many fruiting plants, including apple, plum, apricot and peach trees. Their larvae are the infamous apple worm, which burrow into the fruit immediately upon hatching, and live in its core for several weeks, ultimately destroying most of the fruit. Fruit farmers carefully monitor their orchards for the codling moth's presence via pheromone-based traps, and typically have access to regional resources that advise them on when to deploy various pest control strategies.

Given proper timing, codling moth populations can be controlled via a number of means, including chemical pesticides, pheromone-based mating disruption, or granulovirus. Each of these must be precisely targeted in terms of the moth's life cycle. For instance, the *Cydia pomonella* granulosis virus is one effective means of killing the codling moth during its larval stage. The treatment must be applied just before the moth's eggs hatch, so that it may affect the larvae during their brief travel to the fruit, where they are safe.

Modeling and predicting the life cycle of the codling moth is thus essential to fruit production around the world. Insect development is typically primarily determined by the temperature that they experience; in warmer climates, the same species will develop proportionally faster.

In Switzerland, a national organization—Agroscope—produces codling moth life cycle estimates using a model driven by a network of fourteen weather stations deployed across the country. In this chapter we explore the use of WSN technology to monitor relevant environmental indicators with greater locality—i.e., on the orchard itself—with the ultimate goal of improving the accuracy of this phenological prediction. In collaboration with both Sensorscope and Agroscope, we implement and deploy WSN-enabled pheromone traps in order to monitor the development of the codling moth via real-world moth catches. We present results from a six month long experimental campaign, and use state-of-the-art biological modeling tools to evaluate the predictive quality of our locally gathered data.

Figure 4.1 – Codling moth.[1]

4.1 Related Work

Codling moth lifecycle prediction is of great economic importance to agriculture around the world. A great amount of research has been produced regarding the moth's phenology, and these days both governmental and corporate entities distribute relevant information to fruit farmers in order to guide deployment of their pest control strategies.

RIMpro-Cydia [36] is a commercial decision support system that simulates non-linear, dynamic growth of codling moth populations based on continuous incoming weather data. Farmers are able to license the software directly, or use online services that serve as a wrapper around it, such as fruitweb [37].

A subscription to fruitweb costs farmers about two hundred euros per year, but their subscription can be discounted if they purchase and connect a weather station to the online service. In this way, fruitweb is able to grow their network of weather stations, ever improving their prediction quality.

In Switzerland, a national organization develops their own codling moth model, and distributes predictions to Swiss farmers. See the following section for details.

4.2 Methods

We use two methods in this chapter to model the codling moth's lifecycle: a simple, but widely-used model based on accumulated temperature—degree days—and a closed source model, called SOPRA, that is maintained by a Swiss governmental organization.

[1] "Cydia pomonella (Linnaeus, 1758)" by Olaf Leillinger, used under CC BY-SA 2.5

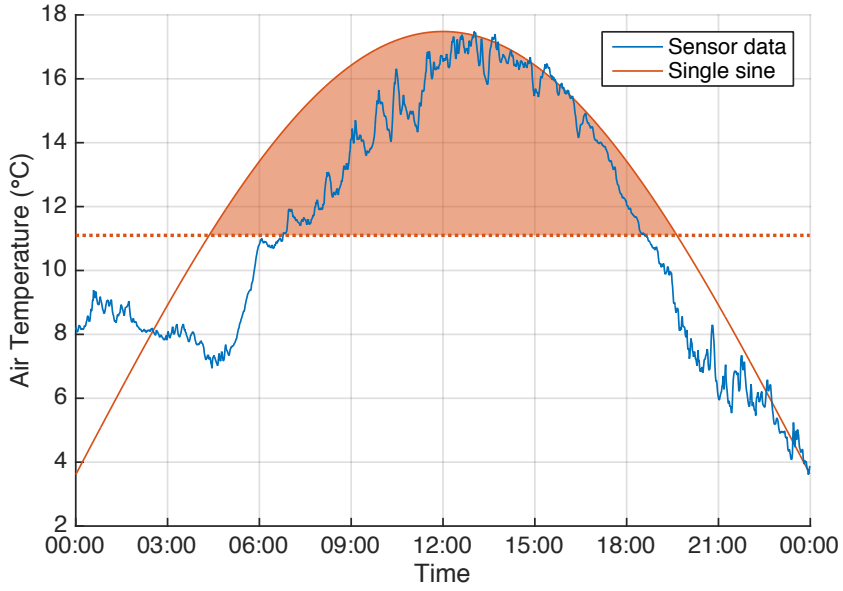


Figure 4.2 – Single sine degree-day model applied to one day of actively ventilated air temperature data from our Sion 2014 deployment. The horizontal line at 11.1 °C marks the lower developmental threshold, and the shaded area represents the area that would be integrated to obtain the degree day estimate under this particular model.

4.2.1 Degree-Day Modeling

Degree-day models are commonly employed in insect pest prediction. Most insects are primarily ectothermic, and as such their rate of development is almost entirely dependent on the temperature that they experience. Degree-day models accumulate heat units over the course of the season, and depending on local calibration, provide a rough estimate of the target insect's development in a given season. Degree-day models are extremely versatile, and can also be targeted for mites [38], bacteria [39] and fungi [40]. They are frequently used not only as a predictor, but as a general analytic tool.

Degree days are obtained by integrating temperature that falls between so-called *developmental thresholds*: minimum and maximum temperatures between which the target biological process is assumed to develop at an optimal rate. A variety of methods are used to approximate this integration; the most basic of which might be to take the mean of the daily minimum and maximum temperatures as the value for a given 24 hour period. Another model, proposed in [41], imitates the daily rise and fall of temperature at a sine wave, centered at noon, with minimum and maximum temperatures fixed to the daily measured values. This model has been used to estimate the codling moth lifecycle (e.g., in [42]), with developmental thresholds between 11.1–34.4 °C (see Figure 4.2).

Pest lifecycle prediction is performed by accumulating degree days starting from a *biofix*, which is traditionally an observable biological event, such as catching an adult moth in a trap,

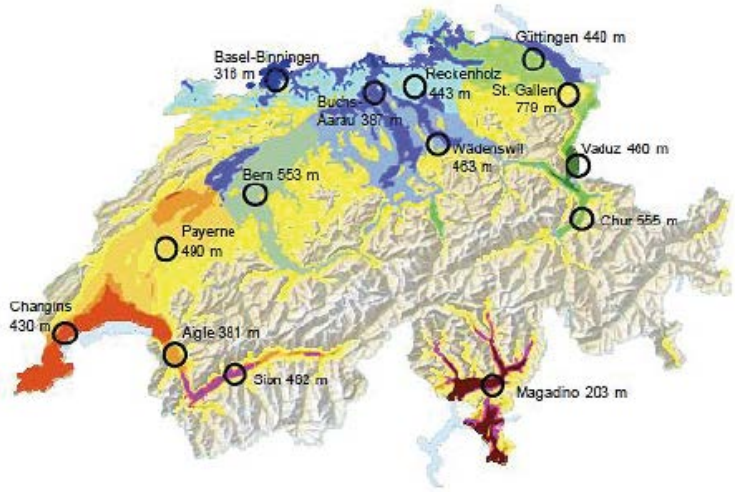


Figure 4.3 – SOPRA makes separate predictions for each of fourteen different agricultural regions in Switzerland. For a given region (denoted by a shaded area on the map), predictions are made by running data from one representative MeteoSwiss weather station through SOPRA.

but may also be a date or in itself an accumulation of a fixed number of degree days from the beginning of the year. In [42], the authors propose that the codling moth larvae will emerge from their eggs 263.9 degree days from the biofix, which is in this case given as the first day of the year that traps have caught adult moths two nights in a row, while temperature at sunset was simultaneously above 16.7 °C.

Note that degree-day models are not typically generalizable, but are instead calibrated specifically for a particular geographic location.

4.2.2 SOPRA

As noted above, agricultural pests and blights are the cause of immense crop losses worldwide. In many countries, national institutions take the role of predicting their presence and distribution, and often subsequently disseminating this information to regional groups whom in turn interpret these predictions for local farmers.

In Switzerland, these predictions are made by Agroscope, a federal organization responsible for research in the areas of food, agriculture, and the environment. A subdivision of Agroscope, Agroscope Changins-Wädenswil (ACW), has developed and now maintains a prediction model for agricultural pests called SOPRA [43]. During the season, on a weekly basis, measurements of various environmental indicators are processed by SOPRA in order to produce an estimate of the optimal time to deploy countermeasures. Such estimates are produced separately for fourteen different regions of the country (see Figure 4.3), and for each of ten different pests. Examples of SOPRA's output can be seen in Figure 4.4, or online at [44].

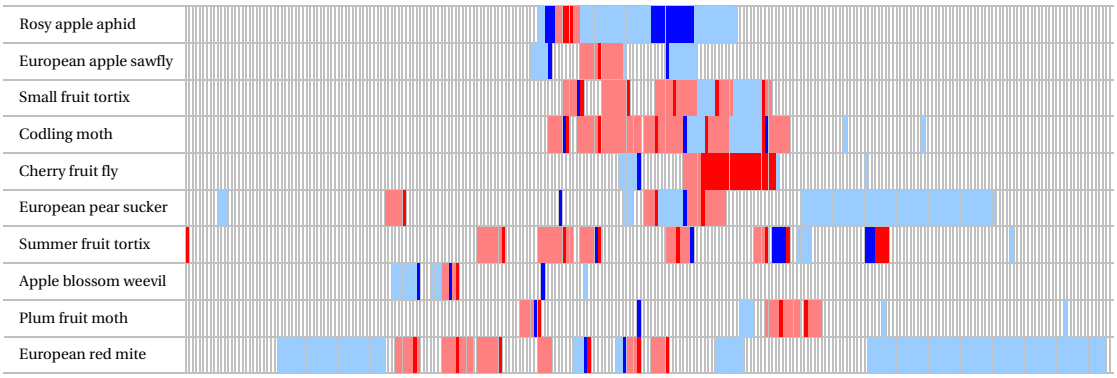


Figure 4.4 – Suggested schedule for pest monitoring and the deployment of countermeasures in Sion, Switzerland, during 2015, as an output from SOPRA's predictions. Pest monitoring is denoted in blue, and countermeasures in red. Light-colored cells represent days where action is recommended, and dark-colored cells where it is urgent. Image taken from Agroscope's website [44].

Internally, SOPRA functions by simulating the life cycle of a population of insects. The modeled pest cycles through various life stages according to a linear rate dependent on the temperature—this rate is tuned specifically for a particular life stage. Note that our input to this model is the temperature *experienced by the insect*—this quantity also depends on which stage of its life cycle the insect is in. The codling moth, for example, spends its early life living in a cocoon under the bark of an apple tree, and is thus primarily affected by the temperature of the tree trunk. SOPRA estimates the trunk temperature using air temperature and an estimate of the incident radiation on the trunk, the latter of which changes throughout the season as the trees begin to grow leaves, and as summer approaches, with the angle to the sun. From the point at which the larva pupates onwards, air temperature is used directly.

4.3 Experimental Setup

This section describes the various tools we use to predict and verify our predictive models.

4.3.1 Pest Detection

Farmers often hang pheromone-based traps from trees in their orchard in order to detect the presence of adult moths themselves, as well as to estimate changes in moth population from year to year. These traps require frequent visual inspection for moth catches, and the lure and trap paper must be replaced every one to two months. For this reason, the predictions made by SOPRA are often used as a rough guide as to when to deploy the traps, which subsequently provide a precise measurement of the onset of moth season. One such trap can be seen in Figure 4.5.

While farmers are recommended to deploy two to four traps per hectare [45], the farmers we

Figure 4.5 – Andermatt Biocontrol's PheroNorm pheromone trap. A pheromone lure is fixed inside the trap, drawing male codling moths to the sticky paper that lines the interior. The trap must be periodically visually checked for new moth catches. Image courtesy of Andermatt Biocontrol AG.

encountered typically deployed one or two traps over their entire orchard, and we understood that this is a typical practice. Traps may be checked infrequently or ill-maintained, which can ultimately lead to crop damage or improperly timed countermeasures.

In the interest of reducing the burden on farmers, we collaborated with Sensorscope Sarl and EPFL's Audiovisual Communications Laboratory (LCAV) in order to produce the "Smartrap" (see Figure 4.6). The Smartrap functions much like Biocontrol's PheroNorm, using the same trap paper and pheromone lure. However, moth catches are counted autonomously, using an internally mounted camera. While monthly trap maintenance is still required in order to change the trap paper and so on, the number of moth catches is updated hourly via cellular radio, without user intervention, and is accessible via an online interface.

4.3.2 Environmental Indicators

Each Smartrap is deployed colocated with a traditional Sensorscope station, equipped with a variety of environmental sensors.

Accurate temperature sensing is a high priority—the phenology models we use are primarily driven by ambient temperature measurements, and any errors introduced in sensing will accumulate over the season and may ultimately lead to significant prediction errors. Agroscope uses data from MeteoSwiss weather stations to drive SOPRA, which are equipped with highly accurate THYGAN VP6 ventilated temperature sensors [46]. The primary feature of these sensors is this active ventilation—in order to mitigate radiative error (see Figure 4.7 for an example), they are equipped with a fan that circulates air across the interior thermocouple. We note that active ventilators do not fully eliminate such errors [47].

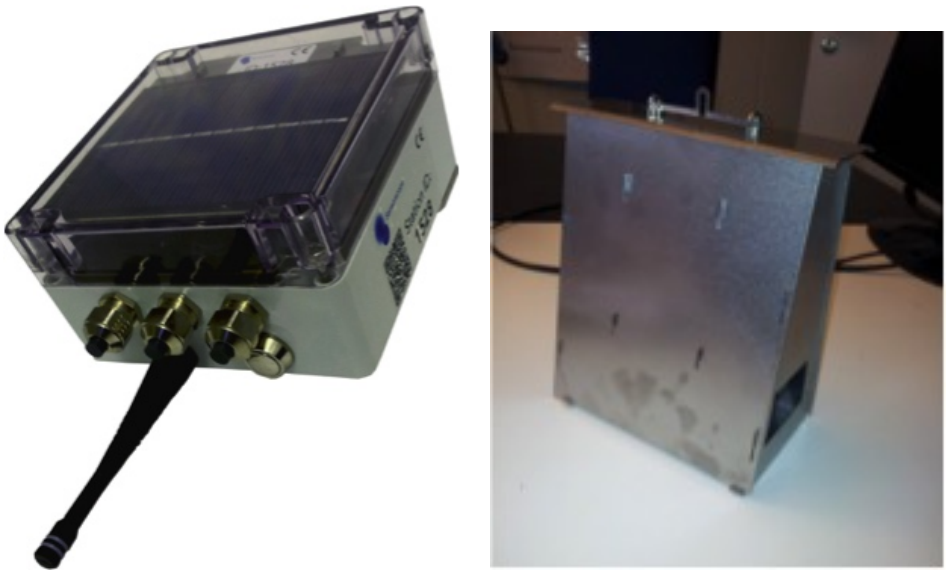


Figure 4.6 – Sensorscope's DS3 datalogger (left) and its attached smart trap (right).

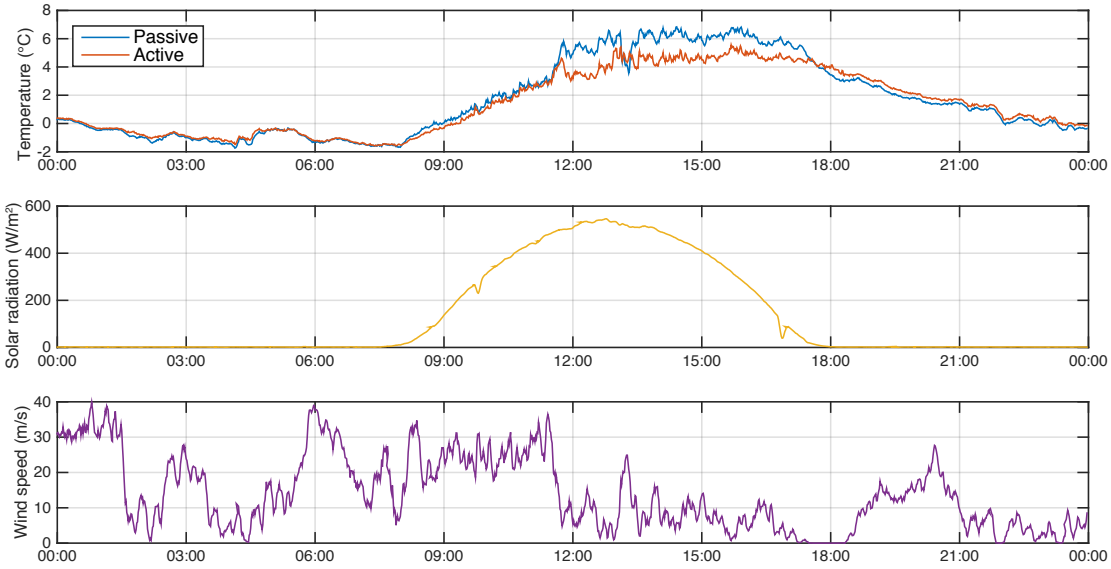


Figure 4.7 – Passively shielded temperature sensors experience significant radiative errors when wind speed is low. Observe that the passively shielded temperature sensor's value overshoots that of the actively ventilated sensor by up to two degrees when the solar radiation is high and the wind speed is low, e.g. at noon. Data taken from rooftop deployment in February of 2015.

Figure 4.8 – Two Sensorscope stations were deployed in an apple orchard in Sion.

The stations deployed in this chapter are equipped with two Sensirion SHT75 temperature sensors—one with a passive radiation shield, and one with the Young 43502H active ventilator [48].

Actively ventilated temperature sensors, while mitigating the above source of error, come at a high price. Not only financially—our selected ventilator continuously draws 6 W of power, orders of magnitude more than any other component in the system. As described in more detail later in this chapter, drawing enough energy via solar panel to support continuous operation of this sensor is difficult, particularly under the constraints of the agricultural environment.

Each station is also equipped with an anemometer, measuring wind speed and direction, and a solar radiation sensor, both from Davis Instruments.

4.3.3 Sion Deployment

We coordinated with the Varone & Consorts apple orchards in Sion in order to deploy two environmental sensing stations and two Smartraps on site (see Figures 4.8 and 4.9). Each station logs anemometry, temperature (with both unventilated and ventilated probes), humidity and solar radiation measurements once per minute. Next, we coordinated with Agroscope to get access to the SOPRA model, which we were then able to run with our own locally collected sensor data.

This experimental setup allows us to study prediction performance across different levels of

Figure 4.9 – Sensorscope stations deployed in Sion.

locality—first at foremost, to look at the difference between regional predictions (as are the standard in Switzerland), and local predictions, using data from our stations on the orchard itself. Further, we are able to compare the difference between predictions made using data in different locations on the same orchard, as the stations are about one kilometer apart.

4.4 Results

In a feasibility study, we developed a basic model of the development of the codling moth as an alternative to SOPRA which uses local temperature data to compute the eclosion time of the pest. The model takes into account the integral of the temperature above a specific threshold over time (measured in “accumulated degree days”) and is simple enough to be run on the nodes themselves, hypothetically allowing one to calculate the pest’s mating period using entirely local resources. In Figure 4.10 we compare the results of our model using 2013 temperature measurements from Sensorscope stations across Sion with the corresponding MeteoSwiss station. These preliminary results indicated that the codling moth’s eclosion time may vary by as much as two weeks inside of one region.

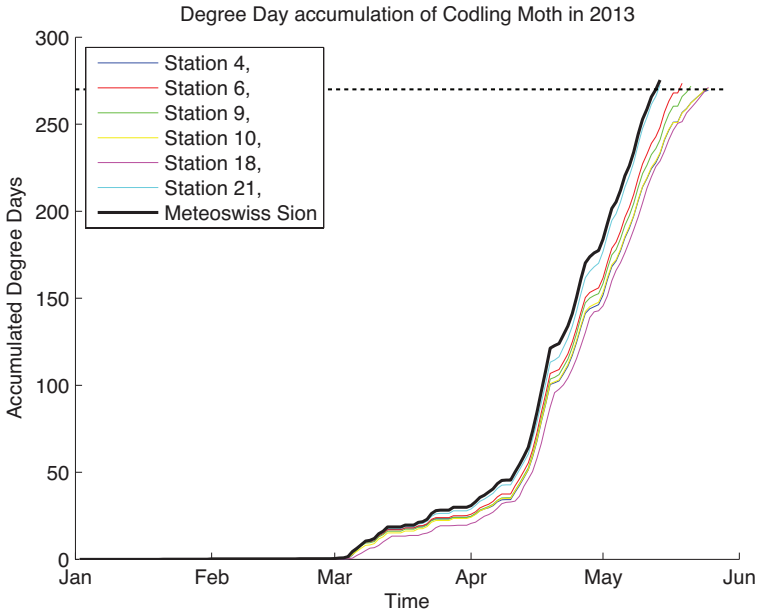


Figure 4.10 – Predicted time of eclosion (horizontal line) in different orchards during 2013.

4.4.1 Sion Deployment

In Figure 4.11, we demonstrate that running SOPRA using regional data and the data collected from our two monitoring stations each yields significant differences in the predicted population of adult moths. Note that the eclosion date is nearly identical for all datasets—due to difficulties in negotiating the exact location of the deployment, we were unable to deploy our monitoring stations until after the codling moth had begun its development, in mid-March. As shown in Figure 4.12, missing data was filled using Sion's MeteoSwiss station. This leads to a heavy bias towards the regional data early in the model prediction, however, as we collected more local data, the models diverged to more accurately represent the effects of local phenomena.

Figure 4.12 also shows an example of mid-deployment issues. In early June, due to expanding tree cover, Station 6 was no longer receiving sufficient solar power to operate continuously. Observe that normal operation is resumed following an adjustment to its position in the beginning of July. Both stations required multiple such adjustments during the season due to growing foliage, and resulted in short-term data gaps.

Finally, in Figure 4.13 we compare actual trap catches, as counted by the smart traps, to the prediction given by SOPRA. We note that for both stations, the smart trap was able to observe the general trend of the moth's lifecycle—that is, two peaks, one representing each adult lifecycle of the moth. However, SOPRA predicted an earlier than observed moth presence for each station by one full month. This large error seems at least partially due to the fact that even once the moth has developed, it requires exact climatic conditions before it will seek a

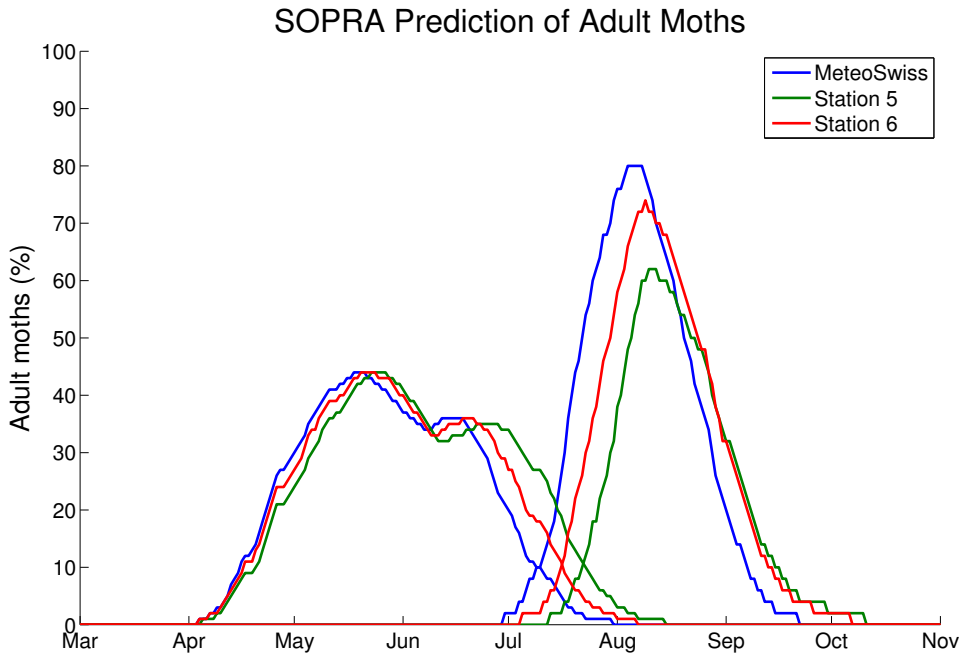


Figure 4.11 – The SOPRA-predicted portion of male population in the adult stage of the codling moth lifecycle.

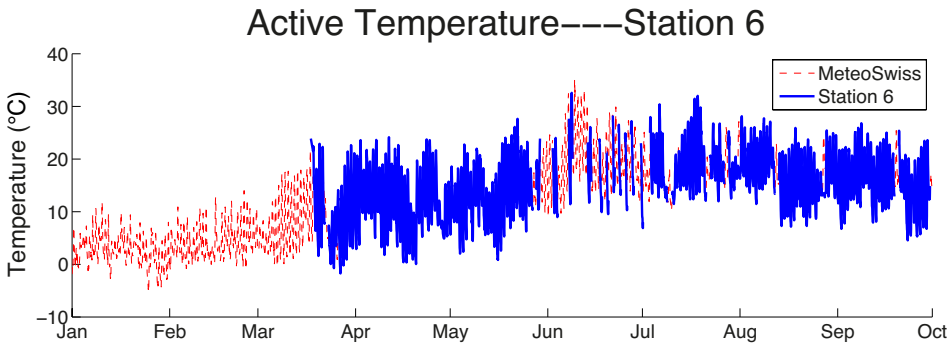


Figure 4.12 – Example of data gaps in active temperature measurements, filled by MeteoSwiss data.

mate. The codling moth requires a warm evening temperature to fly—Andermatt Biocontrol has suggested that if temperatures are below 16°C at 21h00, no mating will occur. Further, the presence of precipitation can also disrupt mating behavior. Indeed, the days leading up to our first moth catches were particularly cold and rainy.

We note that this observation is useful for resource control, in that the camera need not be sampled on days where the above requirements are not met. Throughout the season, we rarely observed moth catches on days not matching these criteria.

4.4.2 Discussion

The modeling suite used for the above predictions, SOPRA, actually predicts the lifecycle of ten different insect pests, including the codling moth. Through informal discussions with Agroscope, we have learned that the underlying model is nearly identical for all pests (based on temperature accumulation), simply with different parameters for each, tuned through years of testing. This leads us to believe that our general approach, improved prediction accuracy through local sensing, is applicable to other insect pests and plants.

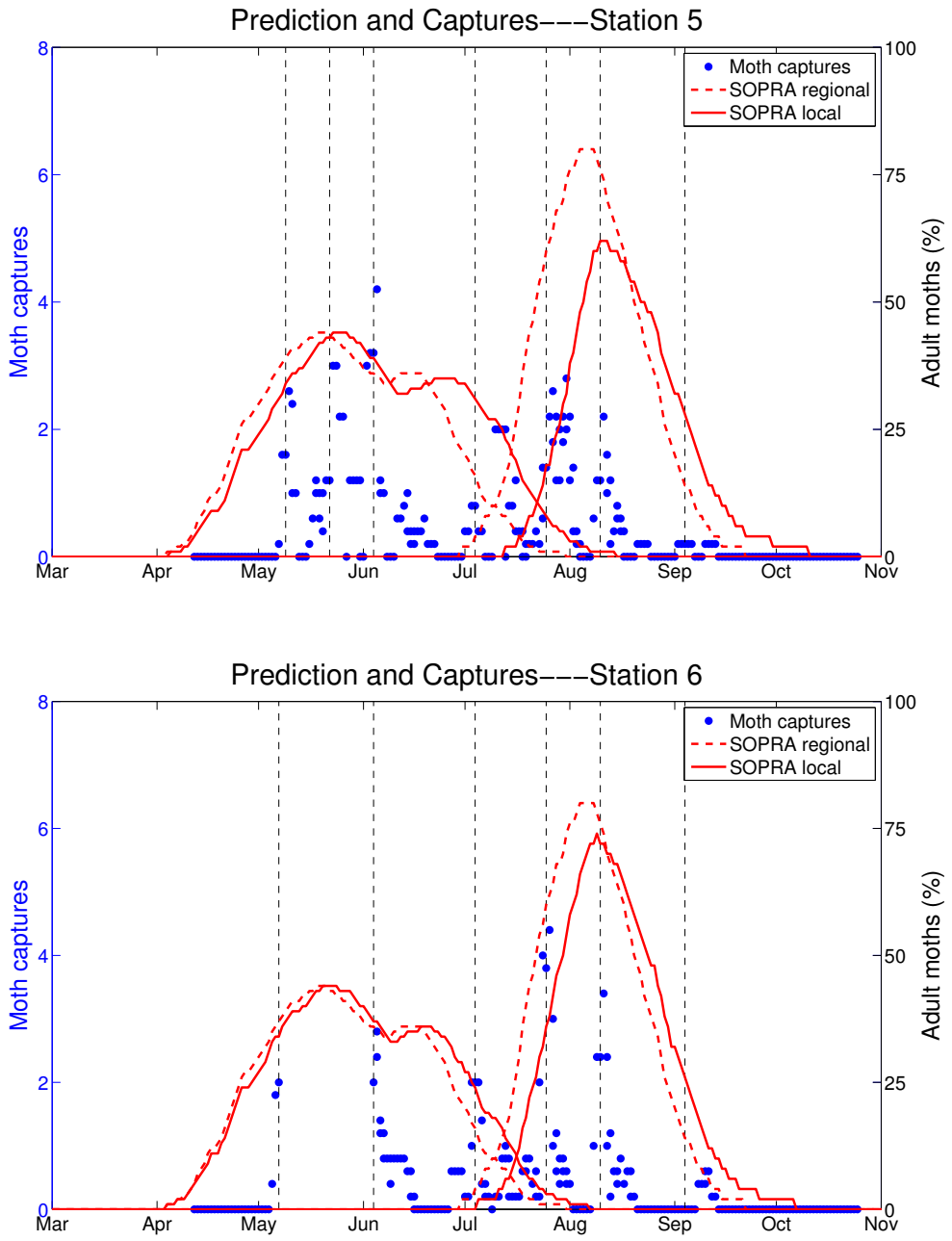


Figure 4.13 – Moth captures per day (five day moving average) plotted in comparison to SOPRA's prediction of the male adult moth population.

5 Environmental Model-Driven Sampling

DETERMINING the surface energy balance at high spatial density is essential in order to establish boundary conditions for many hydrological and atmospheric boundary layer models. Energy exchange at the surface of the earth consists of four components: turbulent heat (sensible) flux, moisture (latent) heat flux, radiative flux and conductive flux.

Of these four quantities, only the radiative component can be measured directly, for instance via a pyranometer. Such sensors are able to capture most of the radiative flux over a surface, and they are typically affordable enough to be deployed with sufficiently high spatial density. The sensible and latent heat fluxes are typically estimated indirectly, using the *eddy covariance* (EC) method. For the sensible heat flux specifically, this method requires the simultaneous measurement of both vertical wind speed and the temperature fluctuations of a small sample volume at high frequency (20 Hz), both of which can be obtained using a 3D sonic anemometer. Estimating the latent heat flux additionally requires a high frequency humidity measurement.

A typical sensor deployment for the study of boundary layer conditions usually consists of two groups of sensors. A large number of nodes equipped with low-cost sensing modalities such as air temperature, humidity, wind speed and solar radiation, in combination with a much smaller set of expensive sensors such as 3D sonic anemometers [49]. Our goal in this chapter is to implement a novel method from the literature for estimating sensible heat flux using an inexpensive sensor, thus allowing sensible heat flux estimation with high spatial density. We fully integrate this sensor into the Sensorscope system by implementing the estimation algorithm in-network, on the sensor nodes themselves, effectively summarizing the high frequency sensor data via this proposed environmental model.

The remainder of this chapter is organized as follows. Section 5.1 briefly outlines conventional methods for estimating sensible heat flux, and Section 5.2 introduces the σ_T-method that we implement in our approach. Section 5.3 describes our integration of both the necessary sensor and estimation algorithm with the Sensorscope platform, and proposes two experiments that validate the effectiveness of the approach. In Section 5.4, we discuss our experimental results,

and in Section 5.5 we conclude with a chapter summary.

5.1 Related Work

This section describes two methods for indirectly measuring sensible heat flux. The first, eddy covariance, is the modern standard, but requires the use of an expensive sensing equipment in order to characterize vertical wind flow. The second method, two-point profiling, applies Monin-Obuknov similarity theory [50] in order to estimate sensible heat flux without direct wind measurements, using only surface and air temperature.

5.1.1 Eddy Covariance

Eddy covariance (EC), also referred to as the eddy flux method, is the standard technique used to measure the sensible heat flux at the earth's surface. The technique is well-established [51], [52], and has been used to study the transport of mass, momentum, and energy over a broad range of applications such as agricultural crops [53], cities [54], forests [55], and snow-covered regions [56].

Achieving acceptable accuracy for sensible heat flux measurement with EC requires high frequency measurements of the vertical wind velocity w and air temperature T (i.e., 10 Hz or greater). The high frequency data are then used to compute the covariance (in time) between the vertical wind velocity and temperature fluctuations $\overline{w'T'}$ to obtain the sensible heat flux as follows:

$$H_{EC} = \rho c_p \overline{w'T'} \tag{5.1}$$

where the over-bar represents a time average, ρ is the air density, and c_p is the specific heat capacity of air at constant pressure. Such velocities and temperature measurements are typically acquired with 3D sonic anemometers [57], [58]. For the remainder of the chapter, EC will be the standard sensible heat flux to which our approach is compared.

The error associated with EC has been investigated by [59]–[61], and determined to be on the order of 10%.

5.1.2 Two-Point Profiling

Two-Point Profiling (TPP) techniques based on air and surface temperature measurements [62]–[65] indirectly obtain surface fluxes under ideal conditions through the application of Monin-Obuknov similarity theory [50], [63], [64], [66]. Surface temperature measurements are easily obtained via commodity infrared temperature sensors, making this approach potentially

suitable for spatially dense estimation.

Under steady state conditions over a homogeneous, flat land surface with zero-mean vertical fluid flow, the vertical profiles of wind speed and temperature can be described by,

$$\overline{u} = \frac{u_*}{\kappa}\left[\ln\left(\frac{z-d_0}{z_0}\right) + \Psi_m\left(\frac{z-d_0}{L}\right) - \Psi_m\left(\frac{z_0}{L}\right)\right] \tag{5.2}$$

$$\overline{T_s} - \overline{T_a} = \frac{H_{TPP}}{u_*\kappa\rho c_p}\left[\ln\left(\frac{z-d_0}{z_{0h}}\right) + \Psi_h\left(\frac{z-d_0}{L}\right) - \Psi_h\left(\frac{z_{0h}}{L}\right)\right] \tag{5.3}$$

where u_* is the friction velocity, κ is the Von Karman constant, z is the measurement height, z_0 is the surface roughness, d_0 is the displacement height, $\overline{T_s}$ is the mean land surface temperature, $\overline{T_a}$ is the mean air temperature, H_{TPP} is the sensible heat flux, ρ is the density of air, c_p is the specific heat capacity of air at constant pressure, z_{0h} is the scalar roughness for temperature, Ψ_m and Ψ_h are the stability correction functions for momentum and heat, respectively, and L is the Obukhov length given by,

$$L = -\frac{\rho c_p u_*^3 \overline{T_a}}{\kappa g H_{TPP}} \tag{5.4}$$

where g denotes the gravitational acceleration. Given measurements of $\overline{u}$, $\overline{T_s}$, $\overline{T_a}$, z_0, z_{0h}, and d_0 Equations 5.2 and 5.3 can be solved iteratively for the sensible heat flux and the friction velocity. However, out of these values only air temperature, surface temperature and wind speed are easily measured; therefore, to use this method, values for z_0, z_{0h}, and d_0 must be taken from the literature or estimated [67], [68].

5.2 Methods

Given the challenges of the above methods, it is useful to consider other approaches that are more amenable for widescale deployment. In this section, we describe a method from the literature that is able to estimate sensible heat flux using only high-frequency temperature measurements, the σ_T-*method* [69].

5.2.1 The σ_T-method

Similar to TPP methods, the σ_T-method applies the Monin-Obuknov similarity theory in order to estimate surface fluxes. As before, applying this principle requires steady state conditions over a homogeneous, flat land surface with zero-mean vertical fluid flow. The σ_T-method

Resistance at 25°C R0	2000 Ω
Operating temperature	−60°C to 300°C
Beta	3068
Diameter d	0.36 mm
Time constant in air τ	0.5 s
Dissipation constant K	0.1 mW/°C

Table 5.1 – Relevant product specifications for the Honeywell 111 NTC.

further reduces the number of parameters to be measured by additionally applying free convective scaling, which requires the dominant forcing of atmospheric motions to arise from buoyancy. This occurs mainly during the daytime.

For this special case Albertson et al. [69] show that the sensible heat flux can be computed using Equation 5.5, which only requires the first and second moment of the temperature measured with a sufficiently high sampling rate.

$$H_{\sigma_T} = \sigma_T^{3/2} \overline{T_a}^{-1/2} \rho c_p C_1^{-3/2} (\kappa g z)^{1/2} \tag{5.5}$$

Here σ_T^2 is the temperature variance, $\overline{T_a}$ is the mean air temperature, c_p is the specific heat capacity of air at constant pressure, $C_1 = 0.97$ is a constant taken from the literature, κ is the Von Karman constant, g is the acceleration due to gravity, and z is the measurement height above the surface. The advantage of the σ_T method is that it only requires a temperature fluctuation measurement at a single location, and there are fewer parameters to estimate.

5.3 Experimental Setup

The section describes the integration of a fast temperature sensor—suitable for use with the σ_T method—with the Sensorscope system, and two deployments thereof, first validating this approach against conventional techniques, then demonstrating the feasibility of the approach during a several week long WSN deployment.

5.3.1 Sensor

To measure the temperature we decided to use a very small Negative Temperature Coefficient thermistor (NTC) from Honeywell's 111 series. The key specifications are listed in Table 5.1. The small size of the sensor leads to a very short time constant τ, thus allowing temperature measurements with sufficiently high frequency.

5.3.2 Sensorscope Integration

For the purpose of this deployment, we integrated the NTC with an existing Sensorscope sensor module, by relying on in-network computation to seamlessly add sensible heat flux to the existing set of measurements.

We drilled an additional port to connect the NTC internally, attaching it to an unused analog-to-digital converter (ADC) of the standard SHT75 temperature/humidity sensor provided by Sensorscope (see Figure 5.1). The supply voltage for the NTC is provided by a high precision 2.5 V reference which is part of the sensor module. Due to the power dissipated as a result of a current flowing through it during measurements, the NTC heats up. This increase in temperature due to self-heating $\widetilde{T}_{NTC}$ is directly related to the current I flowing through the NTC, its dissipation constant K and its diameter d as follows:

$$\begin{aligned} \widetilde{T}_{NTC} &\propto I^2 \\ \widetilde{T}_{NTC} &\propto \frac{1}{K} \\ K &\propto d^2 \end{aligned} \tag{5.6}$$

When averaging temperatures measured with the NTC over long time periods, $\widetilde{T}_{NTC}$ can be considered as a constant bias which can be removed from T_{NTC} to retrieve T_a. For short time intervals considered in our application however the NTC acts like a high frequency hot-wire anemometer and the turbulent wind flows around the sensor greatly increase σ_T. Due to the small size d of our NTC this effect is very pronounced and thus I needs to be as small as possible.

In addition to physically integrating the NTC with the sensor modules, we also modified their firmware. Our replacement software samples the NTC at 10 Hz. However, in a large deployment, it may be undesirable to send all collected samples over the network. We thus implement an online algorithm for computing the necessary statistics on these values during each reporting period: Algorithm 1, originally presented in [70] which continuously updates $\overline{T_{NTC}}$ and σ_T. After each time interval Δt_q the sensor is queried by the datalogger. It then reports the actual value for $\overline{T_{NTC}}$ and σ_T which are then reset for the next reporting period.

Finally, our modified firmware piggybacks $\overline{T_{NTC}}$ and σ_T on the same payload as the original sensor.

Algorithm 1 Online computation of $\overline{T_{NTC}}$ and σ_T.

$\overline{T_{NTC}} \leftarrow 0$ {reset mean}
$\sigma_T^2 \leftarrow 0$ {reset variance}
k $\leftarrow 0$ {reset sample counter}
t $\leftarrow 0$ {reset timer}
while $t \leq T_q$ **do**
  k $\leftarrow$ k + 1
  $T_{NTC} \leftarrow$ getNtcSample()
  $\overline{T_{NTC}}' \leftarrow \overline{T_{NTC}} + \frac{T_{NTC} - \overline{T_{NTC}}}{k}$
  $\sigma_T'^2 \leftarrow \sigma_T^2 + (T_{NTC} - \overline{T_{NTC}})(T_{NTC} - \overline{T_{NTC}}')$
  $\overline{T_{NTC}} \leftarrow \overline{T_{NTC}}'$
  $\sigma_T^2 \leftarrow \sigma_T'^2$
  t $\leftarrow$ t + Δt
end while
return $\overline{T_{NTC}}, \sigma_T^2$

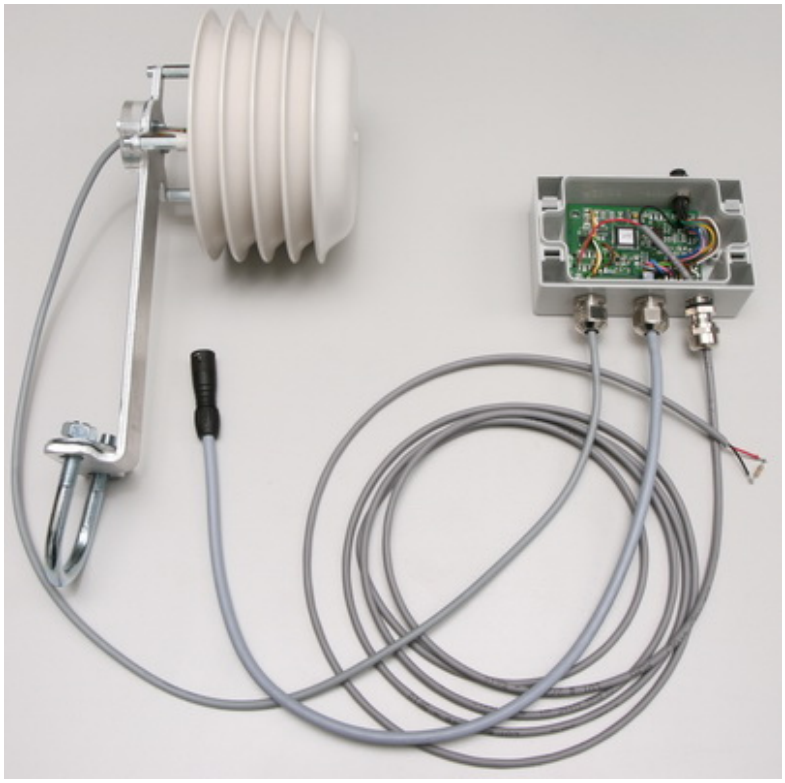


Figure 5.1 – A modified sensor module from a Sensorscope station: the original temperature/humidity sensor (left) attached to the leftmost port of the sensor module, the connector for the sensor bus (center port), cable with fast NTC, here substituted with a fixed resistor (rightmost port).

Figure 5.2 – Reference deployment setup. Sonic anemometer deployed in an open, grass-covered field, with the NTC extending into its sample volume on a probe.

5.3.3 Deployments

We proceed to perform two deployments of the NTC. First, we validate the σ_T approach itself by comparing it directly to sensible heat flux as estimated via the conventional EC method. We follow up by evaluating the NTC's performance during an autonomous Sensorscope deployment.

Reference Deployment

Our first goal is to directly compare sensible heat flux as calculated via EC versus the σ_T-method. An ultrasonic anemometer (the Campbell Scientific CSAT3)—capable of measuring the 3D wind velocity vector and the temperature of the air within its sample volume—was mounted on a thin but rigid structure 3 m above a flat, open, grass-covered surface. The NTC was mounted on a probe extending into the sample volume. Raw data from both sensors was collected over a one week period.

Figure 5.2 shows the entire setup and Figure 5.3 shows a close-up of the sample volume.

Extensive details concerning the reference deployment can be found in [5].

Figure 5.3 – Close-up of sonic anemometer's sample volume. The anemometer's transducers are visible along the top and bottom of the photo, and the NTC is visible in the center.

Sensorscope Deployment

Finally, we perform a several week long experiment in order to evaluate the feasibility of using the σ_T method to measure spatial heat flux during a real-world sensor network deployment. We installed five Sensorscope stations around EPFL's GR building, each station equipped with both an integrated NTC and SHT75 humidity/temperature sensor (see Figure 5.4).

5.4 Results

5.4.1 Reference Deployment

Data from all sensors were collected at 20 Hz and stored using a Campbell Scientific CR5000 data logger. Data from the sonic anemometer were then processed using the EC method and data from the NTC were processed using the σ_T-method. The results in Figure 5.5 show a strong similarity throughout the experiment.

5.4.2 Sensorscope Deployment

Notably, all of the NTCs remained physically intact for the duration of the deployment, despite their small size and exposed position.

Figure 5.4 – Sensorscope station during outdoor experiment. Modified sensor module (top left) and deployed NTC (top right) are visible.

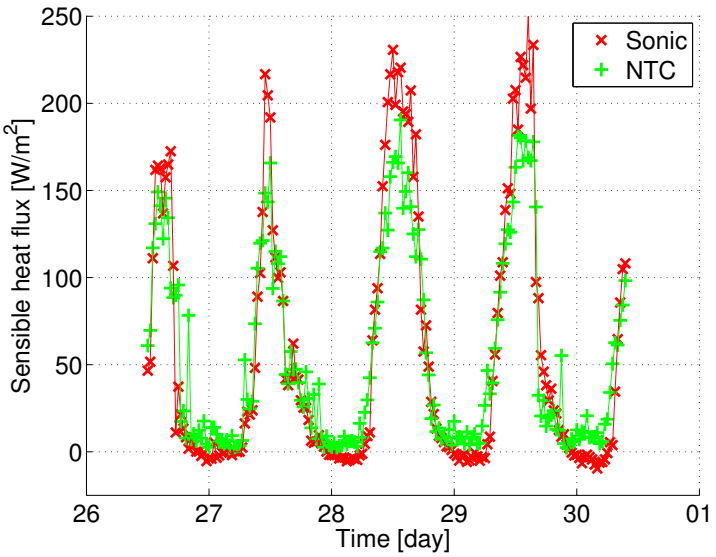


Figure 5.5 – Comparison between sensible heat flux determined by EC method (red) and σ_T-method (green).

Data Analysis

Figure 5.6 shows the sensible heat flux as computed from the data gathered during our deployment. We also validated the proper operation of the NTC through a comparison between the mean temperature of the NTC collected during a five minute observation interval and the instantaneous measurement of the SHT75 (Figure 5.7). Note that the differences in Figure 5.7 may stem from either the NTC's self heating and the radiative heating of the SHT75's enclosure.

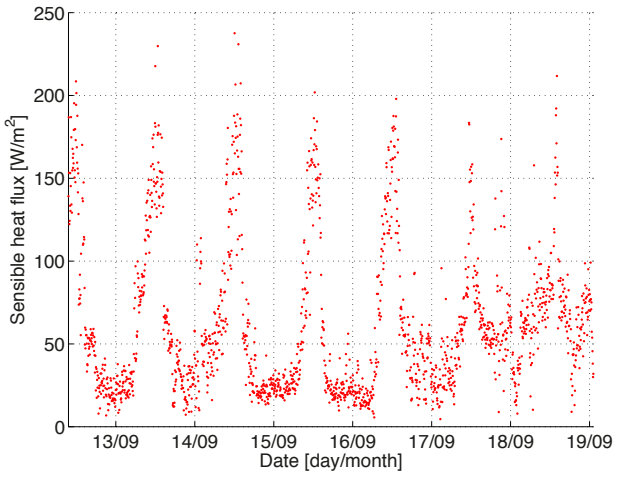
Most WSN technology, including Sensorscope reduces power consumption by minimizing on-time in order to satisfy the strict power requirements imposed by 24 hour, all-weather operation with only a moderately sized solar cell. As the σ_T-method requires the sensor and the processor to be continuously active, it was particularly important to keep the power consumption low. Adding the NTC and maintaining the corresponding microcontroller in an active state increased the power consumption by 5 mW, which may be reduced further by cutting the NTC excitation current.

The total price of all additional components for integrating sensible heat flux measurement is about 30 USD. As a result our sensor/method is affordable enough to become standard on all stations during a deployment.

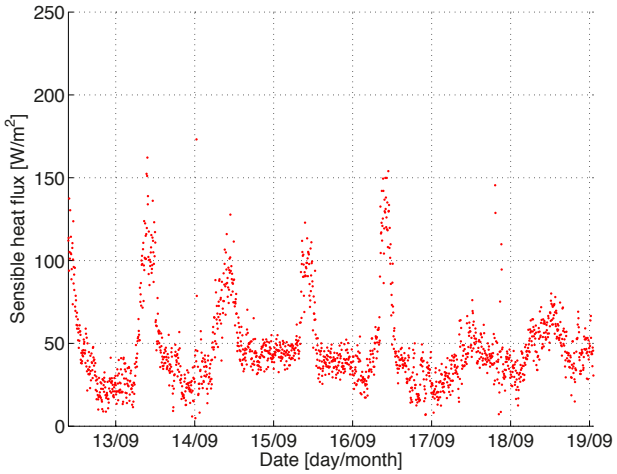
5.5 Summary

Our goal in this chapter was to obtain sensible heat flux—a parameter with high spatial variability which is important in investigating earth's surface energy budget—with high spatial density. We extended Sensorscope's infrastructure to support a custom sensor, physically integrating it with the platform, and writing a custom software module that computes the running mean and variance of the temperature needed by the σ_T method to reduce traffic sent across the network.

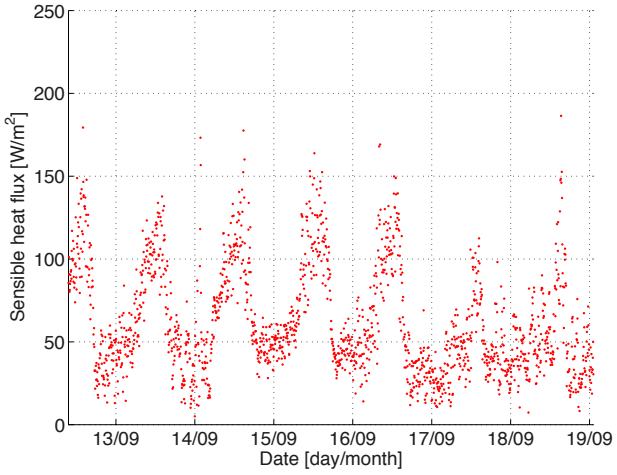
The performance of a single setup was first compared against a well-established reference technique. In a subsequent deployment of five stations we showed that the equipment used in this approach is robust enough to resist meteorological conditions over an entire summer season, yet cost-effective enough that existing sensing stations can be upgraded to include sensible heat flux measurements at negligible costs.



(a) Station 1



(b) Station 2



(c) Station 3

Figure 5.6 – One-week excerpt of sensible heat flux from three different stations as calculated from NTC measurements over a period of one week (moving average with five minute window).

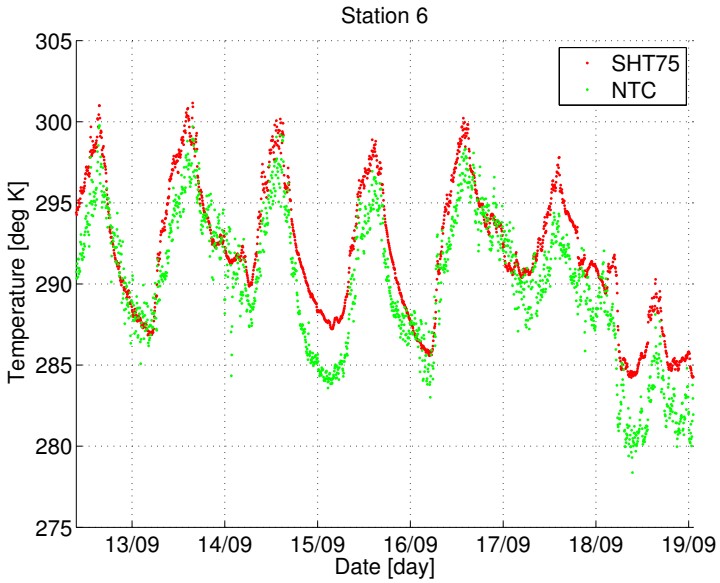


Figure 5.7 – Temperature measurements from the Sensirion SHT75 (green) and averaging the NTC data over 5 minute intervals

Data-Aware Efficient Communication Part III

6 Constraint Chaining

ENVIRONMENTAL processes are often severely oversampled. As sensor networks become more ubiquitous in purpose, increasing network longevity becomes ever more important. Radio transceivers in particular are a great source of energy consumption, and many networking algorithms have been proposed that seek to minimize their use. Traditionally, such approaches are often *data agnostic*, i.e., their performance is not dependent on the properties of the data they transport. In this chapter and the following, we explore algorithms that exploit environmental relationships in order to reduce the amount of transmitted data while maintaining expected levels of accuracy. We employ a realistic testing environment for evaluating the power savings brought by such algorithms, based on a combination of commercially available equipment and custom-built hardware and software.

Retrieving data from a remote deployment in an energy-efficient fashion is a difficult problem, and while solutions have been proposed in literature, real-world systems typically implement robust though inefficient methods. In an effort to bring efficient monitoring techniques to real-world environmental sensor networks, we seek to now quantify the performance brought by these algorithms in practical terms, i.e., by their resulting reduction in overall station energy consumption.

We implement and test a suppression-based data collection algorithm from literature that to our knowledge has never been implemented on a real system, and in doing so propose modifications that make it more suitable for real-world conditions. Using a custom extension board developed for in situ power monitoring, we show that while the algorithms greatly reduce the amount of energy spent on transmitting packets, they have no effect on the real system's overall power consumption due to its preexisting network architecture.

6.1 Related Work

Efficient monitoring techniques can be largely classified as operating on the basis of either spatial or temporal suppression. Both methods attempt to avoid sending redundant data to the sink, preferring instead to have the sink infer these values based on other received data. Temporally focused techniques operate local to a node, examining its history of measured values. The most basic example is simply not reporting a value if it has not changed since the last measure. In contrast, purely spatial approaches seek to suppress a nodes value if it can be inferred given the value of nearby nodes.

The Probabilistic Adaptable Query (PAQ) system is an approach to temporal suppression based on time series forecasting [71]. It uses autoregressive models maintained locally per sensor in order to keep from sending data directly to the sink. Instead, nodes communicate model parameters as necessary in order to keep the sink's predictions within some defined error bound. Tulone and Madden extend this work with their Similarity-based Adaptive Framework (SAF) [72], adding robustness to quick changes in data trends as well as a location-independent clustering technique that allows the detection of redundant nodes.

Many spatial suppression algorithms attempt to detect and deactivate sets of redundant nodes. Prorok et al. study hierarchical network topologies based on spatial clustering [10], [73]. In this approach, cluster heads may choose to prune their children if the part of the monitored field they represent is highly isotropic as defined by some statistically computed threshold. Arici and Altunbasak propose using a first-order model to determine the predictability of particular nodes [74]. They define some of the nodes in the network as *macronodes* which attempt to fit a plane over their neighbors' positions and data, commanding easily predictable nodes to stop reporting measurements for some period of time.

Outside of continuous monitoring, many approaches to efficient data collection seek to reduce overall message volume by eliminating uninteresting data in-network. TinyDB [75] provides such functionality, returning sensor data to the sink in response to simple aggregation queries such as SUM or MAX. Other techniques use in-network triggers to decide when data should be sent to the sink. Yang et al. present a Two-Phase Self-Join scheme that accepts complex monitoring queries from the user and informs the sink should an appropriate event be detected [76].

Constraint Chaining (CONCH) is another algorithm that provides real-time sensor data from a network while implementing spatiotemporal suppression [77]. CONCH operates by detecting highly correlated neighbors and monitoring a suitably defined edge constraint between them instead of their individual values. Unlike similar algorithms which may require functionality not provided by the current Sensorscope network architecture (e.g., intra-network or geographic routing), CONCH imposes very few constraints on the system in which it operates. Our approach to efficient monitoring builds on CONCH, as its few system requirements make it applicable to a wider range of real-world systems.

To our knowledge, none of the algorithms listed in this section have undergone implementation outside of a controlled laboratory environment. A key contribution of this chapter is the implementation and evaluation of a suppression-based monitoring algorithm on an outdoor commercial sensor network widely used for scientific data collection.

6.2 Methods

Exploiting the spatial and temporal relationships inherent to environmental processes is an obvious place to look for energy savings in environmental sensor networks. Simulations over datasets spanning a wide variety of environmental conditions have shown that simply suppressing sensor measurements that have not significantly changed since their last transmission (i.e., by an amount greater than their noise) may reduce the amount of reported data in scientific deployments by more than half [8]. Taking advantage of spatial relationships is much less obvious. In this section, we review one such spatiotemporal algorithm, CONCH, and address various difficulties in implementing it as part of an established network architecture.

6.2.1 Constraint Chaining

Silberstein et al. propose CONCH, a data suppression algorithm that operates by monitoring the difference between correlated neighbors in the network [77]. The network goes through an iterative training process, during which sensor relationships between neighboring stations are tracked. This data is used to periodically construct a spanning tree over the network such that each edge represents a pair of neighboring stations with values that are highly correlated. This spanning tree is called a *Conch plan*, and its construction is detailed below. Stations that are neighbors in the plan share their sensor values, and transmit only the difference between their values to the sink. These differential quantities are never transmitted unless they have changed more than by a user-defined threshold (in our case, we anchor this value to the sensor noise). The server is able to reconstruct the values at every station by walking along the aforementioned spanning tree. The flow of data in and out of the network is described in Figure 6.1.

This approach is well-suited to implementation on Sensorscope stations as it does not require advanced network functionality. Time synchronization, while not strictly necessary, eases implementation by providing a basis by which neighbors may compare their measurements. Some basic neighborhood discovery must be employed in order for stations to coordinate with related nearby nodes. However, most critically, the network must have some centralized means of generating the CONCH plan, i.e., access to the data and neighborhood history of all stations, and a way to then distribute the plan throughout the network. Sensorscope provides all of these features, with the exception of a means by which to send the plan from an off-site server back into the network. We have implemented naive reliable broadcast in-network and added an extra phase during the GPRS cycle for this purpose.

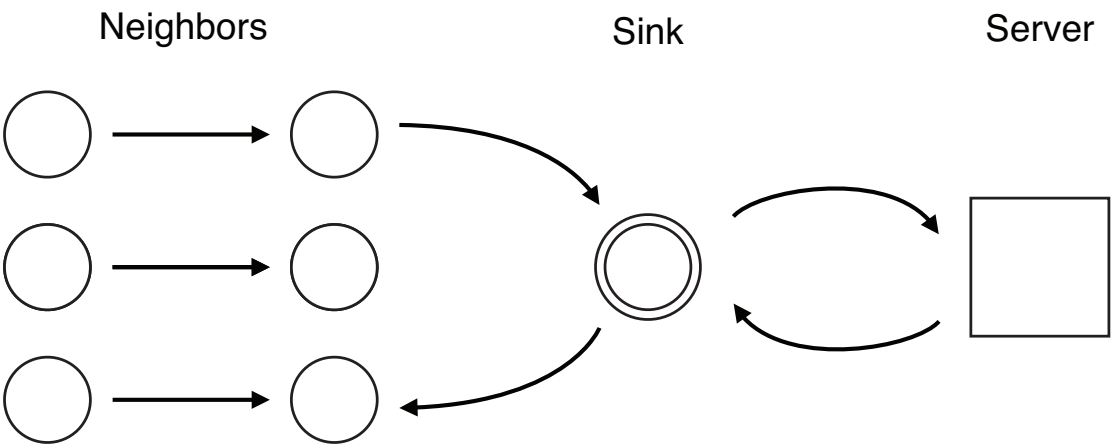


Figure 6.1 – Data flow in CONCH. Neighboring nodes transmit their sensor difference and neighborhood information to the sink via short-range radio, and the sink sends this data to an off-site server via a cellular link. Periodically, a new CONCH plan is calculated and sent from the server to the sink, which then uses a reliable flooding technique to spread it throughout the network.

Spatial Coordination

Spatial suppression in CONCH is enabled by coordination between pairs of nodes referred to as *updaters* and *reporters*. Each updater/reporter pair monitors one edge in the CONCH plan. Each radio cycle, updaters may transmit their new sensor measurements to a neighboring reporter. The reporter compares its updater's measurement with its own, and if the *difference* between these two values has changed since the last pair of measurements, that difference is transmitted to the sink via normal routing methods.

Packet aggregation works normally as compared to the original Sensorscope algorithm. Both updater and reporter messages are aggregated until a packet is either filled or becomes older than ten minutes. As such, measurements are not assumed to have been suppressed (i.e., because they have not changed) until this time period has passed.

In the case that a node is not found to be correlated with any of its network neighbors, it is given the special role of *direct reporter*. Direct reporters send their absolute sensor values directly to sink (again only when they change by more than a certain threshold).

Plan Construction

We refer to the choice of monitored edges and assignment of updater and reporter roles on each link as the CONCH plan. This structure is calculated periodically at the off-site server, and is based on historical data reported by the sensor network.

The first step is to choose the cheapest set of edges in the network to monitor such that these edges form a spanning tree on the network graph. As the server will only receive the difference

in value along each of these edges, if the graph is not connected, the value of disconnected nodes will be unknown. We proceed by assigning a cost to each network edge e equal to $dist(e) * freq(e)$, where $dist(e)$ is the number of hops between the closest node on that edge to the sink, and $freq(e)$ is the number of times the difference along that edge has changed during the previous planning period. Direct reporters are chosen by adding a fake edge from each node v directly to the sink, with cost $dist(v) * freq(v)$. To obtain the set of monitored edges, we simply calculate the minimum spanning tree (MST) over this graph.

Next we must assign updaters and reporters on each of the edges that do not indicate a direct reporter. In the algorithm's original formulation (see [77], the authors propose a solving a linear program in order to approximate their optimal assignment with respect to transmission costs. However, this linear program contains in its objective function a number of variables that increases exponentially with the number of time steps and nodes in the network. As previously stated, it is important to optimize over a length of time that is representative for the phenomenon being measured, e.g., one day/night cycle. In our deployments, Sensorscope stations are configured to take a sensor reading once per minute. For a modestly sized network of 10 nodes over one day of measurements at the aforementioned rate, the objective function contains over 25,000 variables, with an even larger number of constraints. The result is a completely intractable linear program that is simply not solvable in a time frame appropriate for this problem.

Consider that only reporters are responsible for communicating data to the sink, and an optimal CONCH plan will thus be likely to place reporters closer to the sink than their corresponding updaters. Bearing this in mind, we simply iterate through all edges in the plan, marking the adjacent node closest to the sink as its reporter, and the other as its updater. This simplification may be cause excessive energy usage if the node chosen as updater changes value significantly more often than its reporter, however we do not observe such scenarios in our experiments.

We examine the performance of a scheme that executes its replanning phase using the previous N hours of sensor data, repeating this process after another N hours have passed. One could imagine more dynamic schemes in which CONCH plan generation is triggered by some condition detected at the sink, however disseminating a new plan is fairly expensive, so for the sake of simplicity we leave such approaches to future work.

6.3 Experimental Setup

Having implemented the previously described modified CONCH algorithm in the Sensorscope system, we proceed to evaluate its performance via a series of deployments and realistic simulations. For reference, a complete overview of the Sensorscope and power monitoring hardware used below can be found in Sections 3.1 and 3.2, and a description of our simulator capabilities in Section 3.3.

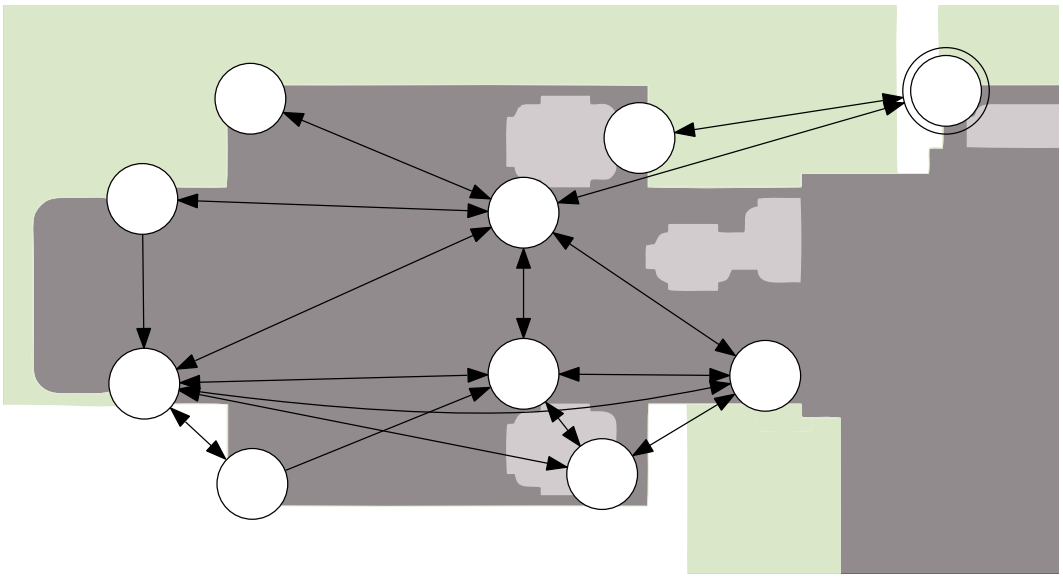


Figure 6.2 – Map of deployment with high quality links shown. The sink is marked with two concentric circles. Note that some nodes appear very close to each other but do not have a connection—this is often due to vertical separation.

6.3.1 Outdoor Testbed

CONCH has been shown to perform well in simulation over synthetic [77] and real-world datasets [8]. In the rest of this chapter, our goal to is to evaluate the algorithm in a real-world setting, i.e., as part of complete sensor network architecture, in an uncontrolled outdoor environment.

Deployment

We deployed a network of ten Sensorscope stations (nine slaves and one GPRS master) around the rooftop of EPFL's GR building and the surrounding balconies below (see Figure 6.2). Each station was equipped with an SHT75 digital temperature/humidity sensor and power monitoring board. The short-range radio's transmission power was set to 15 dBm in order to have a power consumption profile similar to a normal deployment. We used 20 dB attenuators attached to each station's antenna in order to limit the range such that a multi-hop network was formed, with a typical maximum distance to the sink of three.

6.3.2 Realistic Simulation

While our testbed is only able to evaluate one algorithm at a time, we log all sensor and networking data to a per-station SD card. This global dataset can be run through our TOSSIM-based simulation environment in order to realistically compare the default Sensorscope algorithm's performance to CONCH over the same conditions.

Method	Algorithm	Suppression	Tx Energy	Error
Testbed	Default	0.0 %		0.0 °C
	CONCH	45.3 %		0.21 °C
Simulation	Default	0.0 %	1.38 mJ	0.0 °C
	CONCH	51.1 %	0.66 mJ	0.30 °C

Table 6.1 – Algorithm performance

Note that our simulation results estimate power consumption using our characterization of the datalogger's power profile described in Section 3.2.3.

Server Integration

Under Sensorscope's current network architecture, the only data transmitted from the off-site server to GPRS master stations is a global time reference. While this can easily be faked in simulation, as monitoring algorithms become more complex some out-of-network planning may become necessary (and indeed this is the case for CONCH). For this reason, we implemented a simulated GPRS driver that is able to connect to a normal Sensorscope server via a local network socket. This approach allows us to simulate the operation of our modified server simultaneously with the station code.

6.3.3 CONCH Parameters

In the experiments that follow, we trigger a CONCH plan update every $N = 5$ hours, and set the updater/reporter transmission threshold according to the SHT75's stated sensor noise: 0.3 °C.

6.4 Results

In this section we present the results obtained in our experiments. First, we give simulation results over the data obtained during our real-world deployments. We proceed to describe the outcome of our testbed experiments, and conclude with a discussion of our results. A performance summary is provided in Table 6.1, listing the percentage of measured data that was suppressed by our CONCH implementation, along with the total energy spent transmitting packets and the mean error of the resulting temperature estimate.

6.4.1 Simulation Results

The default Sensorscope algorithm gives a simulated average power consumption of 25.50 mW over our two month dataset. CONCH reduces the number of reported measurements by half (51.1%). Ultimately, despite high rates of suppression, due to various sources of overhead inherent to the Sensorscope system, no algorithm shows a significant difference in overall

energy consumption as compared to the default algorithm.

6.4.2 Testbed Results

Default Algorithm

We first performed a ten day deployment using the original Sensorscope algorithm on our outdoor testbed. We observed high reliability, with 100% of all temperature measurements correctly logged to the onboard SD card, and only two sensor packets dropped over the ten day period (out of nearly 100,000). The dataloggers consumed 19.81 mW on average (σ = 6.44 mW).

Constraint Chaining

We performed a four week CONCH deployment during the month of April 2012. However, the network suffered from severe connectivity issues due to rain, causing all but one station to be partitioned from the sink for several days. This resulted in nodes almost always existing as direct reporters, as no links were deemed reliable enough over the five hour replanning period to become monitored edges. Additionally, due to implementation issues, our planning framework was not robust enough to recover from this extended period of unreliability, and thus did not function correctly near the end of the month when the rain had stopped.

Regardless, we extracted five days of useful data from this experiment. During this time we observed a 45.3% reduction in the number of sensor measurements sent, with an average temperature error of 0.21 °C (σ = 0.45 °C). Note that the temperature error is tied to the reporting threshold as described in Section 6.2.1 (0.30 °C in our case). While our result is agreeable with previous simulation results for this algorithm, we observe that the average datalogger power consumption is similar to that of the original algorithm (μ = 23.21 mW, σ = 4.84 mW). The slight increase in power consumption is likely due to stations losing network connectivity and subsequently disabling radio duty cycling, permanently turning on their radios while they wait to receive a beacon.

6.4.3 Discussion

We observe a significant decrease in energy spent on data transmission, however, the algorithm performs worse than expected, failing to measure up to 60%+ suppression rates reported on simulations over other scientific datasets [8], and far short of the synthetic scenarios evaluated by the original authors [77]. Given that our stations are deployed in a relatively dense urban environment, and further, around the edges of a building, it is plausible that the algorithm suffers from low planning frequency relative to the transience of temporal and spatial relationships in the observed data. However, flooding the network with a new CONCH plan is a relatively expensive operation in terms of traffic.

In Chapter 7, we attempt to overcome these issues by proposing the use of more powerful constraints between nodes, combined with in-network computation of the CONCH plan.

Note that despite suppressing about half of all data transmissions, we see no discernible decrease in power consumption. We defer discussion of this observation to the end of the next chapter.

6.5 Summary

In this chapter we examined the real-world performance of a suppression-based sensor network monitoring algorithm that—despite promising prior results in simulated scenarios—had yet to be tested in a real-world environment. We fully integrated it with the Sensorscope system, detailing necessary modifications to both the algorithm and the Sensorscope architecture itself.

In a series of calibrated simulations and real-world deployments monitoring ambient temperature, we observe that the algorithm's performance is worse than expected. In the following chapter, we propose further modifications to the core algorithm that allow it to adapt more quickly to changes in network configuration and the environment.

7 Distributed Constraint Chaining

In this chapter, we continue our effort to adapt CONCH to the real world by altering the core of the algorithm to better handle network instability and unpredictable changes in environmental patterns. We propose and implement modifications which replace centralized operation in favor of a distributed approach, allowing in-network optimization and increasing robustness in realistic (i.e., unstable) environments.

In its original form, the ability of CONCH to adapt to changes in the network or its environmental conditions is limited by the expense of transmitting a new CONCH plan to every node in the network. Naive reliable flooding requires the transmission of $O(|E|)$ messages, where E is the set of edges on the network graph, and less expensive approaches require additional routing structure to be maintained, imposing node-level complexity that may be undesirable.

In this chapter we present Distributed CONCH (DCONCH), a modification to the original algorithm in which MST calculation is done in a distributed fashion, allowing individual nodes to quickly adapt to dynamic conditions by making local adjustments to the CONCH plan.

We conclude this chapter with a critical lesson about energy optimization in sensor networks—despite often being one of the most expensive operations a node can perform, reducing energy spent on transmission does not necessarily lead to significant real-world power savings.

7.1 Methods

In this section, we describe existing approaches to distributed MST calculation, identify a method suitable for our network architecture, and use it as the basis for plan generation in DCONCH. Finally, we identify the differential edge constraint as an area of improvement, and propose an alternative based on autoregressive models.

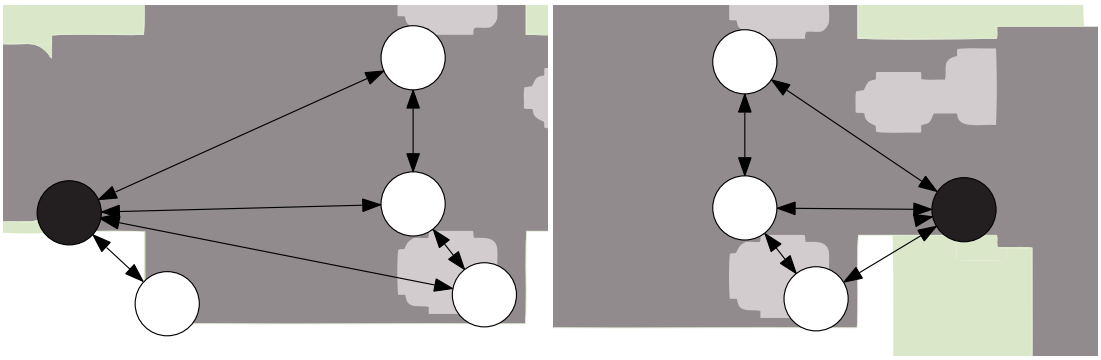


Figure 7.1 – Examples of one-hop neighborhoods from DCONCH. Reference nodes in black.

7.1.1 Distributed Minimum Spanning Tree

The greatest hurdle in generating the CONCH plan in-network is calculating the MST in a distributed fashion. A number of algorithms have been proposed both for calculating the exact MST (e.g., [78], [79]) and its approximation (e.g., [80]).

Li et al. propose the Local Minimum Spanning Tree (LMST) algorithm for topology control in WSNs [81]. Their algorithm approximates the exact MST as follows: each node in the network calculates the MST over the subgraph formed by its one-hop neighbors. An edge between two nodes exists in the LMST if and only if both nodes have each other in their one-hop MST. This approach has seen considerable acceptance in the sensor networks community as it only requires one-hop communication. Cartigny et al. develop an LMST-based approach to reliable flooding [82]. Ovalle-Martínez et al. propose a method to convert LMSTs into globally optimal MSTs by breaking cycles in the network graph [83].

We choose LMST for use in DCONCH as it is well-studied, inexpensive, and frequently used in the context of sensor networks.

7.1.2 Plan Adjustment

CONCH plan adjustment in DCONCH proceeds as follows. During each radio frame, nodes overhear sensor data sent between their neighbors. These measurements are compared with the eavesdropping station's own data, and it counts the number of times the difference between these values changes. The cost of monitoring the link between these two stations is equal to this value multiplied by the minimum distance between either of these nodes to the sink (note that this information in particular is already exchanged via periodic beacons).

Note that the form in which a station transmits sensor data depends on its role. If the node is an updater or a direct reporter, it will transmit its actual sensor value. However, if a node is a reporter, it will only send the difference between its own value and that of its paired updater. If only the reporter is in range, i.e., the eavesdropping station is unable to overhear

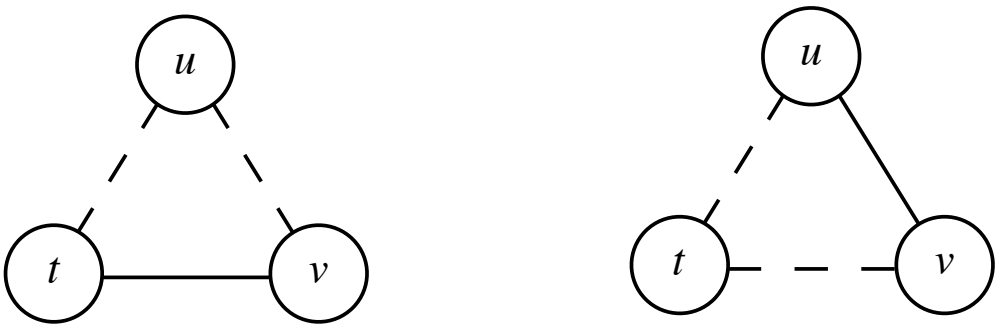


Figure 7.2 – The contentious edge problem in DCONCH. Consider the network from the perspective of node u (left) and node t (right). Dashed lines represent network connectivity, and solid lines represent plan monitored edges. If the costs on various edges are estimated to be similar, it is possible that nodes in this graph expect the other to take an edge connecting to v. If neither does, v will be orphaned and its value will not be calculable.

the updater's value, it is impossible to determine the reporter's actual value. For this reason, we only consider updater transmissions during edge cost calculation.

One issue in implementing DCONCH is that due to network losses, and the unreliable nature of eavesdropping (i.e., the sender does not expect eavesdropping nodes to acknowledge the transmission), the cost assigned to each edge is only an estimate. Different nodes may have different cost estimates for the same network edge. LMST is not designed to operate under these conditions, and does not guarantee connectivity unless all nodes share the same weighted graph. One might imagine solving this problem by having all nodes periodically share their edge weight estimates with their neighbors. However, one-hop agreement is not sufficient. In order to guarantee that the LMST preserves connectivity, all nodes who can see an edge must agree on its weight. On a unit disk graph, two nodes observing the same edge are maximally two hops away, however in an asymmetrical network like we observe in our testbed (recall Figure 6.2), they may be even farther. It is certain that performing per-edge consensus between nodes that may be an arbitrary number of hops from each other is considerably more expensive than the original CONCH algorithm. Also consider that periodically sharing these estimates would limit replanning to the same period, hampering the ability of DCONCH to adapt to changing conditions, much like the original algorithm.

In order to mitigate this problem, we must first identify edges that have a high risk of being contentious, i.e., that may significantly impact the network topology based on small differences in cost estimation between nodes (see Figure 7.2). An edge $\overline{tv}$ considered potentially contentious if a node u estimates the cost on $\overline{tv}$ to be similar to an edge between a node adjacent to that edge and itself, such as $\overline{uv}$. In this case, it is possible that both u and t believe that the other node is responsible for connecting v to the network. We proceed by estimating the maximum possible divergence between edge cost estimates. Thus, we consider an edge $\overline{vt}$

to be contentious if and only if, for any node u in range of both v and t

$$|cost_u(u,v) - cost_u(v,t)| < \sum_t (1 - qos(u,v)) \tag{7.1}$$

where $cost_u(v,t)$ is the cost of the edge $\overline{vt}$ as estimated by u, and $qos(u,v)$ is the link quality estimate at u for v (i.e., it represents the probability that v will send a message not heard by u, see Section 3.1.1 for details).

This problem is solved in DCONCH by forcing contentious edges to be included in every node's plan. This guarantees connectivity at the expense of redundant edges, (i.e., a poorer approximation of the actual MST).

7.1.3 Model-based Constraints

In order to improve the performance of CONCH's spatial and temporal constraints, we replace the basic differential approach described above with a model-based approach. Instead of transmitting sensor readings whenever they have changed over some threshold, we modify the algorithm to maintain a statistical model of local sensor behavior and instead transmit the model's parameters. Thus, we must only ensure that the sink has accurate model parameters for every node in the network.

Autoregressive Models

More specifically, we employ autoregressive (AR) models, maintained both at each network node and at the sink. AR models are a special case of autoregressive integrated moving average (ARIMA) models. An AR model of order p, denoted AR(p), takes $\mathbf{X}$ to be a stationary process formulated as follows:

$$\mathbf{X}_t = C + \sum_{i=1}^{p} \varphi_i \mathbf{X}_{t-i} + \epsilon \tag{7.2}$$

where C and φ_i are parameters of the model, and ϵ is white noise with variance σ^2. Further details, including the derivation of the autoregressive model, can be found in e.g. [84].

While more powerful, fitting an ARIMA model is potentially intractable on limited hardware, and may require solving an infinite system of equations. AR models only require solving a strictly linear system of equations, and for small orders can be solved at a low computational cost even on modest hardware (e.g., using Levinson recursion [85]).

We choose $p = 3$—an AR model of order three—in our work, as it has previously been shown to perform well on environmental datasets while still being computationally feasible in this application [71]. This yields four model parameters that replace our previous differential constraint: C, φ_1, φ_2, and φ_3.

Changes to the underlying algorithm are straightforward. Each time a node takes a new measurement, it recalculates its AR model parameters and, if they have changed significantly, transmits the new coefficients along any links on which it is an updater. For each link on which the node serves as a reporter, it then calculates the difference between its own vector of model parameters and those that it received from each updater, and if the difference between any two of these two sets of parameters has changed, they are transmitted to the sink. Note that while we are now operating on four values instead of the sensor's direct value, we have found that the model has excellent predictive capabilities on our datasets, therefore resulting in a reduced constraint breakage. As a consequence, its deployment results in a higher suppression rate than the original approach when combined with a high planning frequency, as seen later in this chapter.

7.2 Experimental Setup

In this chapter we present the results of calibrated simulations, using the simulation framework described in Section 3.3, over the datasets collected in Chapter 6. We compare the performance of CONCH and DCONCH with both the differential and autoregressive (i.e., AR-CONCH and AR-DCONCH) edge constraints.

As before, CONCH (and now AR-CONCH as well) constructs a new plan every $N = 5$ hours.

7.3 Results

We present a comprehensive table of results in Table 7.1.

7.3.1 DConch Performance

We found that in our experiments DCONCH monitored on average 1.42 times as many edges compared to CONCH. Due to its ability to replan locally (i.e., more frequently), the spatial suppression rate in DCONCH is significantly higher (28.8% versus 12.9%). However, the increase is not enough to compensate for the extra monitored edges.

7.3.2 Simulation Results

CONCH and AR-CONCH reduce the number of reported measurements by 51.1% and 57.2% respectively. DCONCH shows considerably lower performance (32.3%) than CONCH on our two

Method	Algorithm	Suppression	Tx Energy	Error
Testbed	Default	0.0 %		0.0 °C
	CONCH	45.3 %		0.21 °C
Simulation	Default	0.0 %	1.38 mJ	0.0 °C
	CONCH	51.1 %	0.66 mJ	0.30 °C
	DCONCH	32.3 %	0.98 mJ	0.39 °C
	AR-CONCH	57.2 %	0.62 mJ	0.29 °C
	AR-DCONCH	63.9 %	0.51 mJ	0.33 °C

Table 7.1 – Algorithm performance

month dataset due to redundant monitored edges created by our spanning tree approximation. However, AR-DCONCH is able to take advantage of fast replanning times to achieve a high rate of spatial suppression, overcoming the performance of AR-CONCH with a suppression rate of 63.9%.

Ultimately, despite high rates of suppression, no algorithm shows a significant difference in overall energy consumption as compared to the default algorithm.

7.3.3 Discussion

As we show below, it is clear from our results that Sensorscope's current network architecture does not benefit from suppression-based approaches to efficient monitoring. Many ultra low-power network architectures have been proposed, however they often have extreme data latencies (e.g., Koala [86], Dozer [87]) or heavily rely on the typical periodic nature of environmental sampling (e.g., DISSense [88]). A low power protocol that can take advantage of variable amounts of data per cycle must be identified before we can observe any real-world gains.

Overall Savings

We performed simulations in MATLAB in order to determine at what point the algorithmic choice has a significant impact on station lifetime. We precisely modeled the Sensorscope network frame (as described in Section 3.1.1), using values for power consumption, station density, and packet loss as measured during our outdoor deployments. Note that this model approximates average performance, and does not account for the bursty nature of unacknowledged packets.

Figure 7.3 shows that suppressing sensor data has an insignificant effect unless the station is transmitting a large amount of data per frame (i.e., more than about 2 kB), while the busiest node in our deployment was only responsible for about 30 bytes per frame. While transmitting is indeed considerably more expensive than other station activities (recall Figure 3.6), the datalogger still uses a significant amount of power even while the radio is not transmitting.

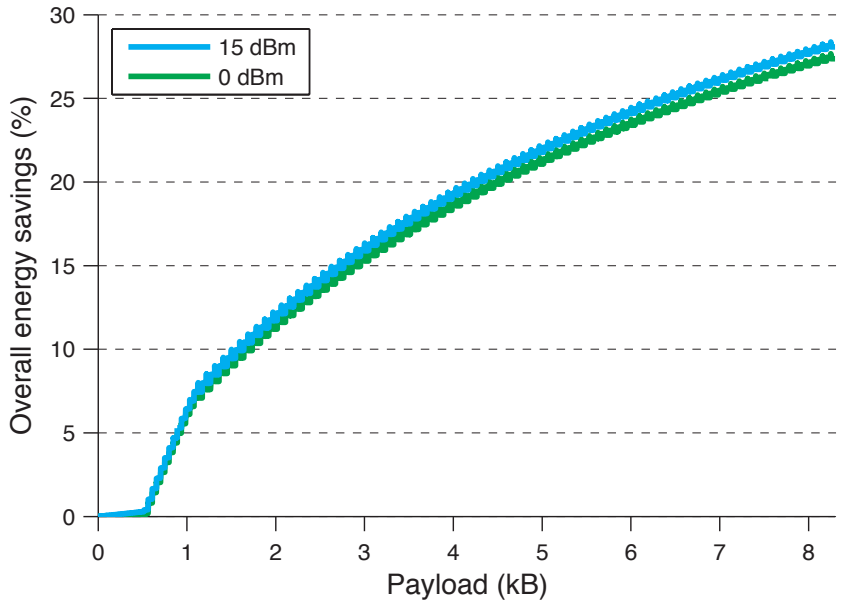


Figure 7.3 – Modeled effect of suppressing half of all sensor measurements on power saved, as a function of the total number of bytes to transmit per frame. We observe a slow increase until the minimum frame length is reached (one hundred bytes), followed by periodic jumps that correspond to the need for additional packets.

Once this overhead is surpassed by increasing demands on the radio, message suppression begins to have a significant effect. We obtain a maximum overall power reduction of nearly 30% at the maximum frame length. Note that minimum versus maximum transmission power makes only a small difference, as radio receive durations during CSMA backoffs dominate the radio's power budget.

Note that this plot will vary depending on the aforementioned factors. As the network becomes more dense and packet loss increases, the positive effect of packet suppression may increase.

The effectiveness of this approach is visualized in Figure 7.4. We observe that, in the absence of overhead, the performance of the differential constraint drops off very quickly as the replanning period increases. A more powerful edge constraint must be used to leverage DCONCH's ability to replan more frequently. The AR model constraint benefits greatly from reduced replanning time, yielding results more favorable to DCONCH by lessening the impact of redundant edges and increasing the suppression rate overall. Ultimately, this change results in AR-DCONCH sending 10.5% fewer messages than AR-CONCH in our simulations.

These results could be further improved if a model of the exact process under measure is known. Any temporal model could potentially be used as an edge constraint as long as the network nodes have the requisite processing power.

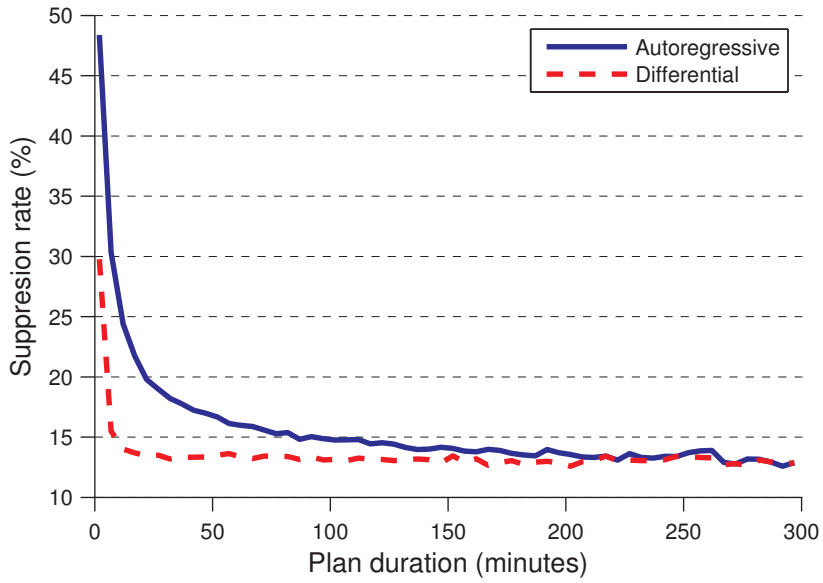


Figure 7.4 – Performance with respect to plan duration for both the AR model and original differential edge constraints. We observe that the AR model constraint is able to take greater advantage of high replanning frequencies. Note that this plot ignores the replanning overhead, i.e., the message cost of flooding the network with a new plan.

7.4 Summary

During the course of these past two chapters we took a suppression-based sensor network monitoring algorithm from literature and implemented it on a real system. We modified the algorithm significantly in order to make it feasible on real hardware. We found that it showed poor performance under the dynamic conditions typical in sensor networks, and proposed a fully distributed approach that is better able to adapt to unstable network topologies and changing environmental conditions. Finally, we used a custom-developed power monitoring board to show that even though these algorithms greatly reduce network traffic, they have no effect on Sensorscope's overall power consumption.

While in this work we studied algorithmic performance on a typical sensor at its typical sampling rate (as configured by Sensorscope), performance estimates must be tied to the characteristics of the underlying environmental process before they can be applied more generally. Consider that the more severely a process is oversampled, the better suppression-based algorithms will appear to perform. There exists a body of literature that explicitly examines environmental relationships in the context of sensor networks (e.g., [89], [90]), and in future works these relationships could be used as a predictor of algorithmic performance.

Part IV Hybrid-Mobility Sensor Networks

8 Spatial Field Estimation

As a first step toward our hybrid sensor network, we develop a Bayesian approach to combining observations of a field to form a two dimensional estimate. We consider a class of sensor that is not able to make spatially precise measurements of its environment, but instead observes a weighted average of measurements inside its field of view—therefore, measurements taken at a greater height incorporate more of the environment into their value.

8.1 Introduction

The remarkable accessibility of modern flying robots makes them an attractive platform for environmental sensing. Quadrotor aircraft in particular have seen a recent surge in consumer popularity due in large part to their robustness, stability and price. Autonomous and semi-autonomous fixed-wing aircraft have seen increasing use in environmental sensing and mapping, with the rise of companies such as senseFly [91]. Indeed, mobile environmental sensing offers to ameliorate a common issue in static sensor deployments—limited coverage.

However, this approach is not without its drawbacks. Commercially available autonomous flying platforms are still very limited in terms of flight time, and must be launched and landed manually. Furthermore, at least at the time of this publication, most areas of the developed world require that autonomous aircraft be supervised at all times by a human operator. Ultimately, the combination of these issues suggests that while mobile platforms offer a means of sensing with high spatial density, they suffer from limited overall autonomy, thus limiting the frequency with which they might collect data.

We thus conclude that current technology requires using static and mobile sensing nodes in tandem to achieve high measurement density and accuracy in both time and space. In this chapter and the following (Chapter 9), we construct an approach to integrating a mobile node with a static network, leveraging existing infrastructure to guide its data collection to best complement measurements from static nodes.

The rest of this chapter is structured as follows. In Section 8.2 we describe our technical approach, including our sensing model and estimation algorithm. In Section 8.3, we proceed to describe the tools used to evaluate our approach. In Section 8.4, we present our experimental results, and offer some discussion. We conclude the chapter with a summary in Section 8.5.

8.2 Methods

This section describes the technical background and details of our approach, including our model of the underlying environmental field and how it relates to sensor measurements under our chosen class of sensors, i.e., our sensing model.

8.2.1 Spatial Model

Gaussian processes are commonly used to represent spatially varying processes, and form the underpinnings of Kriging, a widely used interpolation technique in environmental science.

Gaussian Processes

A stochastic process $\mathbf{X}$ is a *Gaussian process* (GP) if and only if for every member of some index set $i \in \Lambda$, $\mathbf{X}_i$ is a Gaussian random variable, and further, for every k-length $\lambda \subseteq \Lambda$ with finite k, $\mathbf{X}_{\lambda_1,\ldots,\lambda_k}$ is a multivariate Gaussian.

In this chapter, we use GPs to model a spatially varying environmental process on a plane. We model this process by discretizing it into a grid of points, p_i, for each $i \in \Lambda$. That is, each $\mathbf{X}_i$ can be mapped to a point on a grid, with position p_i.

Homogeneity

GPs have many desirable properties that result from their underlying normal distribution, e.g. they are straightforward to manipulate and are computationally tractable, as we will see in the rest of this chapter. Toward this end, we will restrict our model even further, to only consider GPs which are both *stationary* and *isotropic*.

Any GP is completely described by its mean function μ, and covariance function K, the latter of which is sometimes called its kernel. Given $\mathbf{X} \sim \mathrm{GP}(\mu, K)$, $\mathbf{X}$ is considered to be stationary if and only if μ is constant inside Λ, and critically, if K_{ij} is solely a function of the vector $p_i - p_j$. $\mathbf{X}$ is furthermore isotropic if and only if K depends only on the distance between points p_i and p_j, d_{ij}. A GP that is both stationary and isotropic is called *homogeneous*.

Examples

While our homogeneity assumption may seem restrictive, such GPs allow quite some variety in spatial structure. The following covariance functions all meet aforementioned formulation. For some scaling parameter a,

$$\text{Exponential} \quad K_{ij} = \exp\left(-\frac{d_{ij}}{a}\right) \tag{8.1}$$

$$\text{Gaussian} \quad K_{ij} = \exp\left(-\frac{d_{ij}^2}{a}\right) \tag{8.2}$$

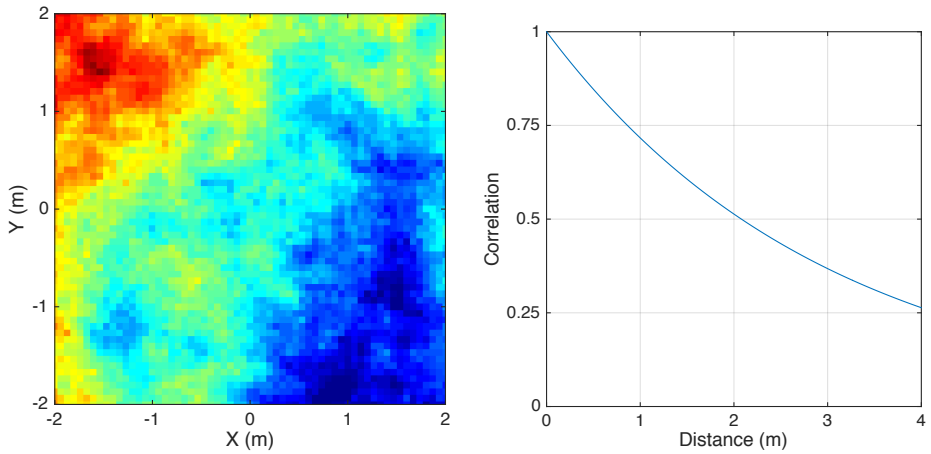
$$\text{Spherical} \quad K_{ij} = \begin{cases} 1 - \frac{3d_{ij}}{2a} + \frac{d_{ij}^3}{2a^3} & \text{if } d_{ij} \leq a \\ 0 & \text{if } d_{ij} > a \end{cases} \tag{8.3}$$

We can proceed to draw instantiations of GPs driven by these functions, as shown in Figure 8.1.

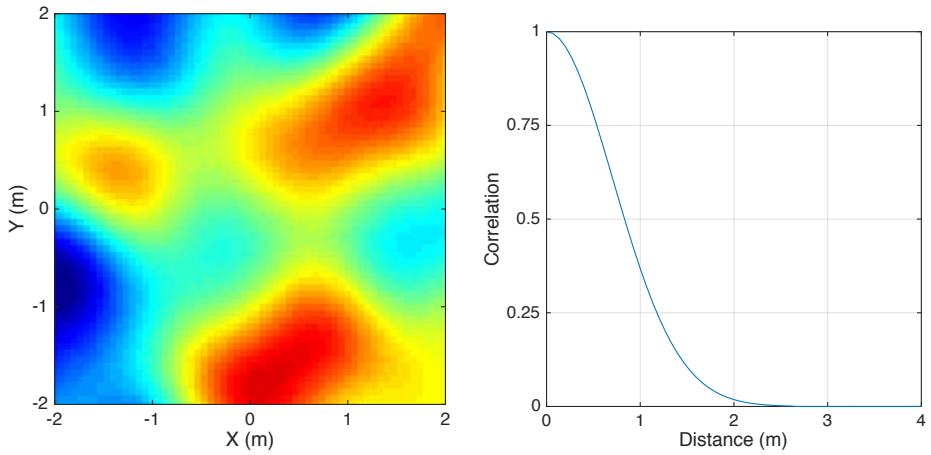
8.2.2 Sensing Model

Robotic platforms offer a particular advantage for measuring certain environmental quantities with unprecedented spatial and temporal resolution. Previously, common options for remote sensing were either the use of relatively low resolution satellite data, or hiring a manned aircraft, e.g., as in the Léman-Baïkal survey [92]. Some information simply cannot be sampled remotely, such as odor distribution [93]. Even when remote data is available for a field of interest, such as surface temperature (e.g., via satellite) it may be of insufficient resolution (in both time and space) to for example drive detailed models of microclimatic phenomena. Flying robots offer the ability to take in situ measurements with a sensor of choice with high spatial density.

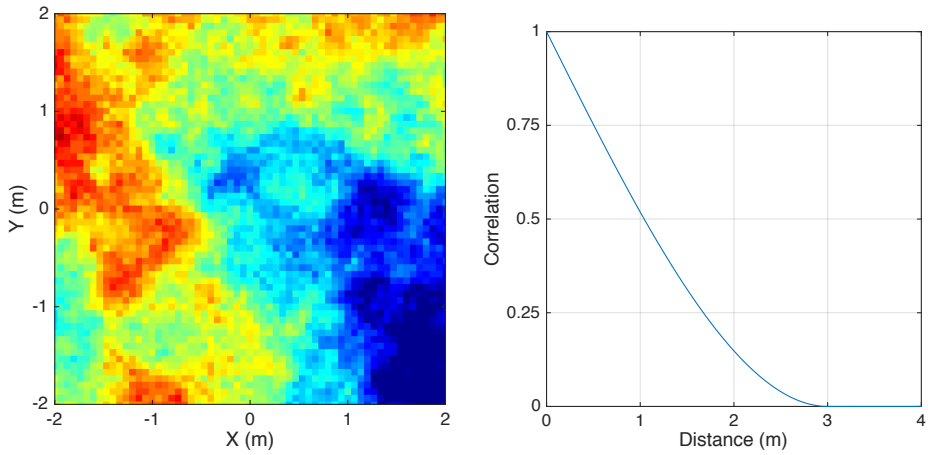
However, inexpensive flying robots can only carry a modest payload, and many useful sensors are simply too heavy to be mounted, e.g., high-resolution hyperspectral cameras [94], [95] or thermal cameras [95], [96]. In this chapter we develop an approach for using a low resolution (i.e., single pixel) thermal sensor to approximate the field measured by a high-quality thermal camera. Our algorithm exploits *a priori* knowledge of a field's statistical structure in order to rapidly build an accurate estimation. In uncontrolled environments (i.e., field deployments), one must make an approximation of such structure, either based on data collected under similar conditions, simulation, expert knowledge, and/or an already deployed static network that the mobile sensor is intended to complement. This structure, along with a model of the thermal sensor's response, is used to perform Gaussian process regression after each measurement.



(a) Sample from GP (left) with exponential covariance function, $a = 3$ (right).



(b) Sample from GP (left) with Gaussian covariance function, $a = 1$ (right).



(c) Sample from GP (left) with spherical covariance function, $a = 3$ (right).

Figure 8.1 – Samples from homogeneous GPs with various covariance functions.

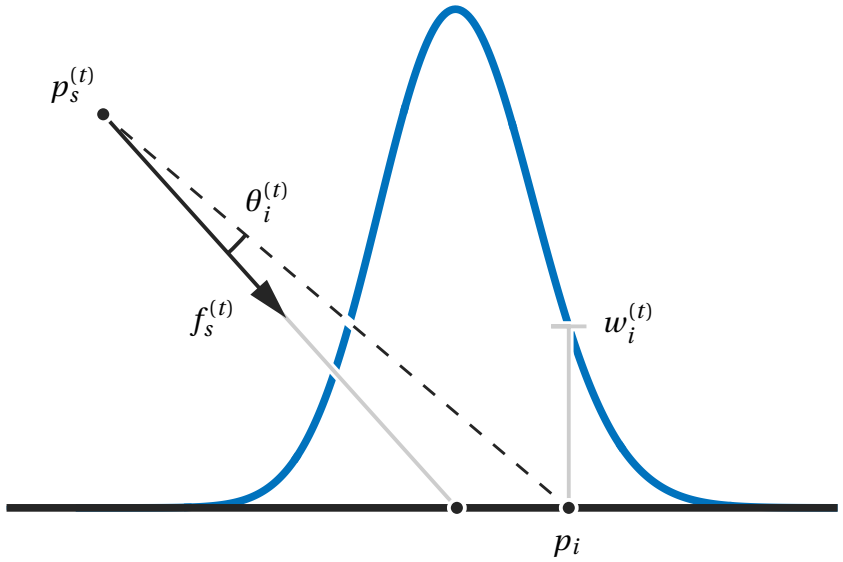


Figure 8.2 – Sensing model. Consider the sensor to be at $p_s^{(t)}$. The value returned by the sensor is the Gaussian-weighted sum of points inside its field of view. Point $\mathbf{X}_i$ is given a weight $w_i^{(t)}$ depending on its angle $\theta_i^{(t)}$ to the direction the sensor is facing, $f_s^{(t)}$.

The sensors we consider in our work operate as single pixel cameras, and can be modeled by a Gaussian weighted average with the angle between a particular point on the target surface and the sensor's orientation (see Figure 8.2). Thus, at time t the weight a given point $\mathbf{X}_i$ contributes to the sensor's observation is:

$$w_i^{(t)} = \begin{cases} \mathcal{N}(\theta_i^{(t)}, 0, \sigma_s) & \text{if } \theta_i^{(t)} < r_s \\ 0 & \text{otherwise} \end{cases} \tag{8.4}$$

with sensor range r_s and shape parameter σ_s. $\theta_i^{(t)}$ is the angle between the facing direction of the sensor, $f_s^{(t)}$, and the vector between the position of the sensor, $p_s^{(t)}$, and the position of the respective element of the GP, p_i:

$$\theta_i^{(t)} = \cos^{-1} \frac{f_s^{(t)} \cdot \left(p_s^{(t)} - p_i\right)}{\left|f_s^{(t)}\right| \cdot \left|(p_s^{(t)} - p_i)\right|} \tag{8.5}$$

Under this model, the sensor observes a low-pass filtered version of the underlying field, and the width of the blur increases with sensor height (see Figure 8.4). Let x denote the

instantiation of some GP $\mathbf{X}$. The value returned by the sensor is thus:

$$v_s^{(t)} = \sum_i^{\Lambda} w_i^{(t)} x_i + \epsilon \tag{8.6}$$

For some Gaussian noise $\epsilon \sim \mathcal{N}(0, \sigma_n^2)$. Note that a weighted sum of Gaussians is also Gaussian.

8.2.3 Incorporating Observations

We use Gaussian process regression (GPR) to fuse observations from our sensor into an estimate of the underlying field. GPR is a well-studied technique that has benefited from popularity in the machine learning community—and as such, is reasonably performant, even for a large number of states.

We can derive the conditioned GP by standard Bayesian methods, which yield the following equations for our posterior mean vector and covariance matrix, given observations from time $t_1, ..., t_k$:

$$\mathbb{E}\left[\mathbf{X} | v_s^{(t_1,...,t_k)}\right] = \mathrm{Cov}\left[\mathbf{X}, v_s^{(t_1,...,t_k)}\right] \left(\mathrm{Cov}\left[v_s^{(t_1,...,t_k)}\right] + \sigma_n^2 I\right)^{-1} v_s^{(t_1,...,t_k)} \tag{8.7}$$

$$\mathrm{Cov}\left[\mathbf{X} | v_s^{(t_1,...,t_k)}\right] = K - \mathrm{Cov}\left[\mathbf{X}, v_s^{(t_1,...,t_k)}\right] \left(\mathrm{Cov}\left[v_s^{(t_1,...,t_k)}\right] + \sigma_n^2 I\right)^{-1} \mathrm{Cov}\left[v_s^{(t_1,...,t_k)}, \mathbf{X}\right] \tag{8.8}$$

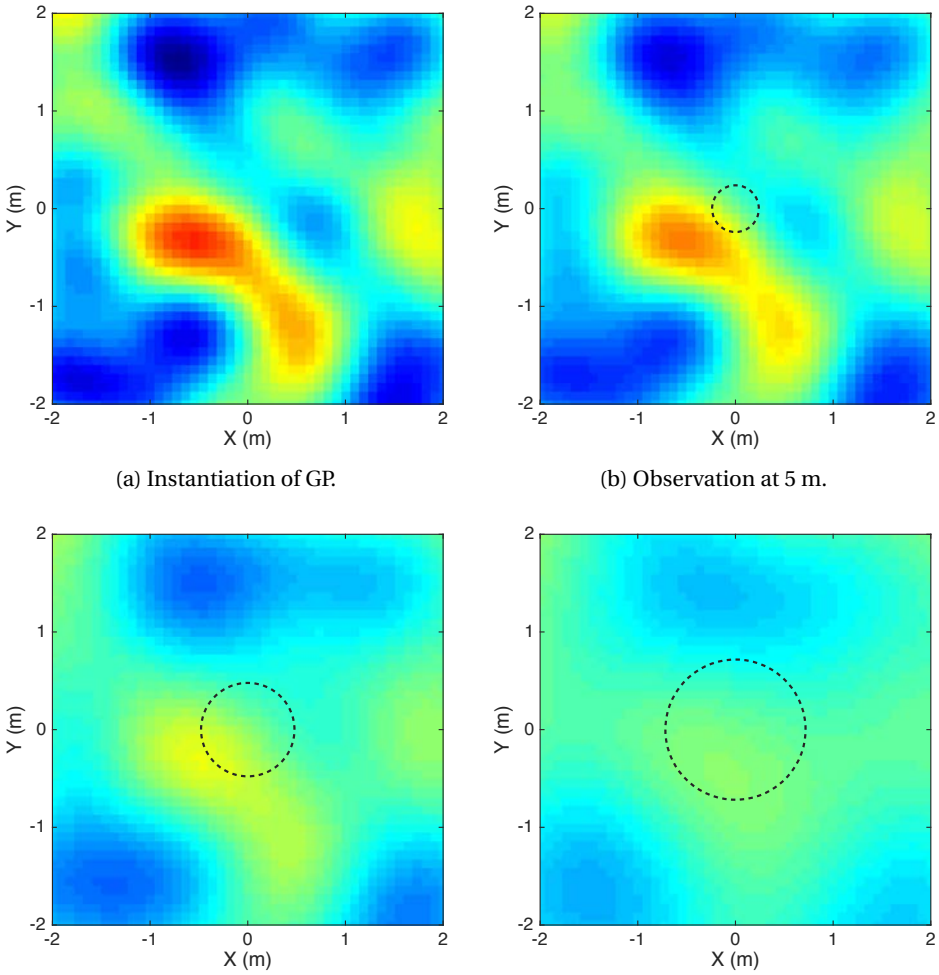
With the covariance between an observation and a single point:

$$\mathrm{Cov}\left[v_s^{(t)}, \mathbf{X}_i\right] = \sum_j^{\Lambda} w_j^{(t)} K_{ij} \tag{8.9}$$

And the covariance between observations at times t, u:

$$\mathrm{Cov}\left[v_s^{(t)}, v_s^{(u)}\right] = \sum_i^{\Lambda} \sum_j^{\Lambda} w_i^{(t)} w_j^{(u)} K_{ij} \tag{8.10}$$

Note that the posterior covariance (Equation 8.8) does not depend on the observed values, but only their statistics. This is a key feature of GPs, and it is of critical importance when it comes to path planning (Chapter 9).



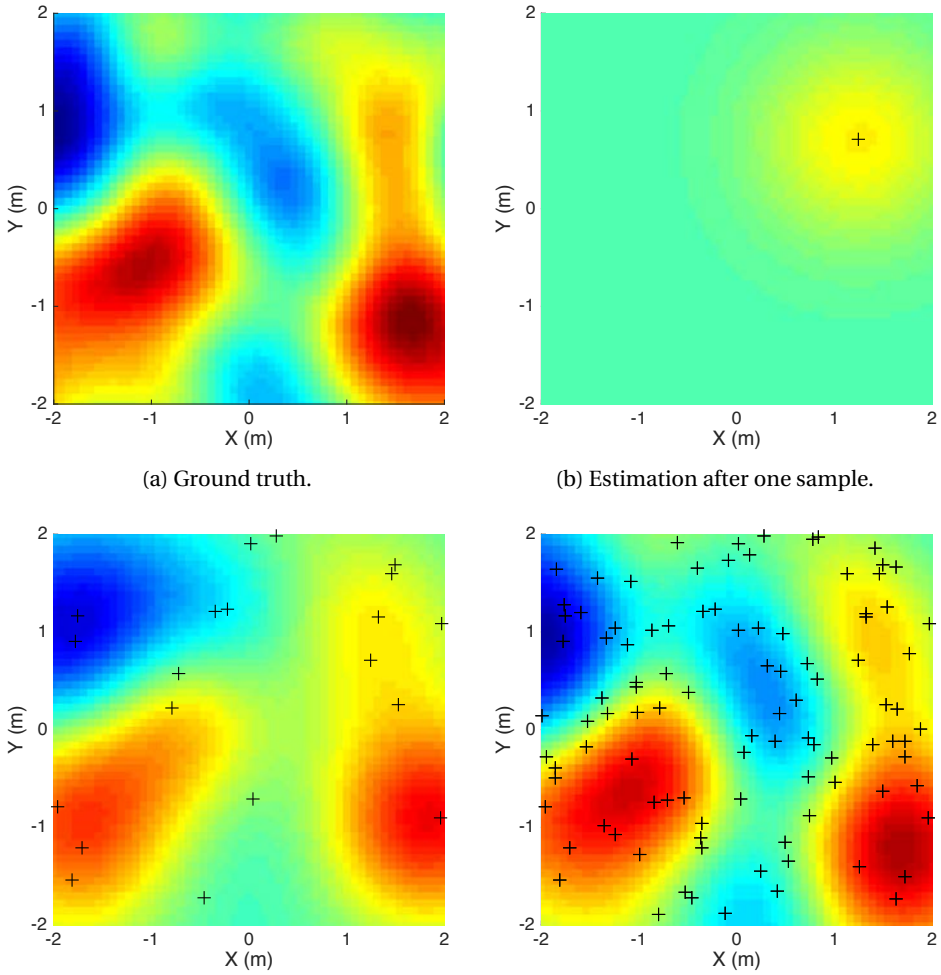
(a) Instantiation of GP.

(b) Observation at 5 m.

(c) Observation at 10 m.

(d) Observation at 15 m.

Figure 8.3 – (a) Instantiation of GP with Gaussian covariance function (Equation 8.2) and $a = 1$ m. (b–d) Observed value at each point in the field, with f_s perpendicular to the ground plane, at various heights. Black dashed circle represents width of the weighting function's standard deviation with respect to the ground plane, given $\sigma_s = 2.74°$.



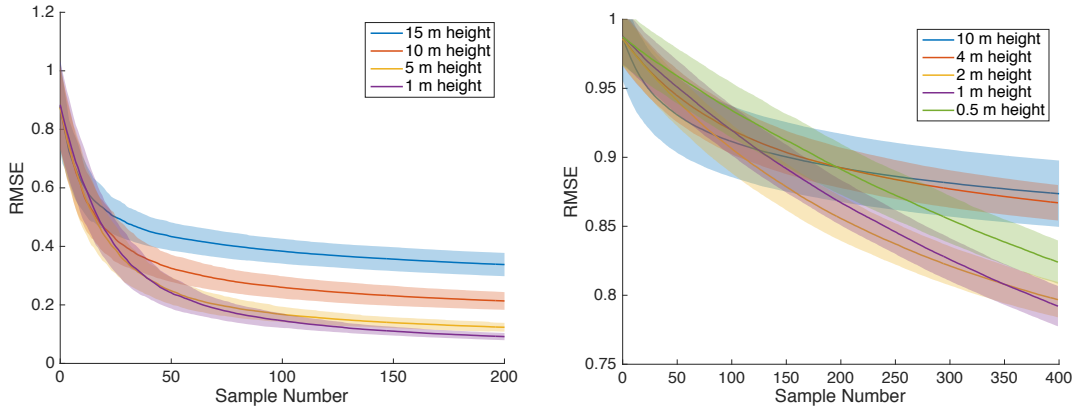
(a) Ground truth.

(b) Estimation after one sample.

(c) Estimation after 20 samples.

(d) Estimation after 100 samples.

Figure 8.4 – (a) Instantiation of GP with Gaussian covariance function (Equation 8.2) and $a = 1$ m. (b–d) Estimation after taking progressively more randomly placed samples, with f_s perpendicular to the ground plane, at a height of 5 m, and $\sigma_s = 2.74°$.



(a) Gaussian covariance function, $a = 1$.

(b) Exponential covariance function, $a = \frac{1}{10}$.

Figure 8.5 – Mean (line) and standard deviation (shaded area) of estimator performance taking samples at random, aggregated over 200 simulations. Field size 4 x 4 m.

8.3 Experimental Setup

We proceed to explore the performance of our estimation approach in both simulation and reality. Our simulation results were gathered on our custom-built simulator on our computational cluster, both of which are described in Section 3.6.1.

In simulation, we use $\sigma_s = 2.74°$ from the infrared thermometer's sensing model (Section 3.5.1). Our outdoor flight uses the quadrotor-mounted spectrometer (Section 3.5.2), which has $\sigma_s = 6.67°$.

8.4 Results

In this section we present quantitative simulation results, and a qualitative outdoor experiment.

8.4.1 Simulation Results

Here we compare the performance of the estimation process as a function of sensor height. In Figure 8.5a, the correlation range is relatively large with respect to the field size, and sampling at lower heights generally gives a better performance. However, at a height of 1 m, we can already observe a tradeoff—early performance is slightly worse than sampling from height, and only yields the best performance after 40 samples. In Figure 8.5b, the small correlation range results in a much more diverse performance across heights. Here we can see a general trend—early, it is better to sample at high altitude, then descending over the course of the estimation process.

8.4.2 Outdoor Flight with Spectrometer

Having validated our approach quantitatively in simulation, we now explore its use in hand with a more exotic sensor, in an uncontrolled environment. The Hamamatsu mini-spectrometer resolves incoming light to very precise spectral bands across the visible spectrum and into the near infrared, effectively functioning as a single pixel hyperspectral camera.

Figure 8.6 presents the outcome of a five minute flight over a catch basin adjacent to a farm plot near EPFL's campus, during which we logged lateral position via GPS, height via barometric pressure sensor, estimated pose by IMU, and measurements from the spectrometer. The quadrotor was flown manually at a height of approximately ten meters.

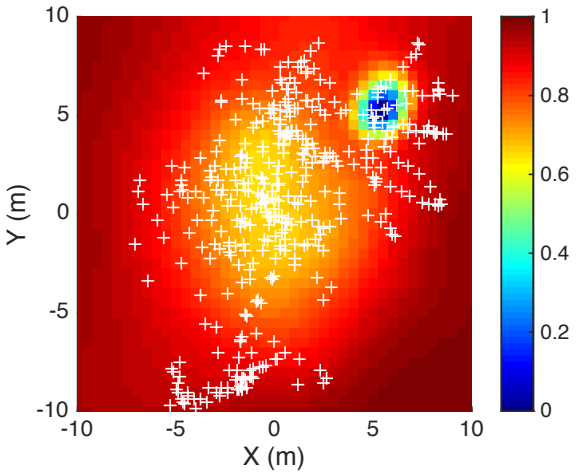
8.5 Summary

In this chapter we presented a GP regression-based approach to fusing measurements from a single-pixel optical sensor to form a two-dimensional map. We developed a flexible simulation environment to evaluate estimation performance as a function of the spatial characteristics of the field under measure.

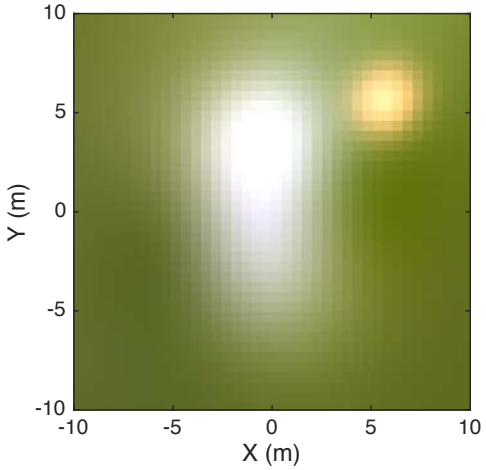
After validating our approach in simulation, we performed a qualitative evaluation using data from a single-pixel spectrometer acquired by mounting it on a quadrotor for a manual outdoor flight.

Note that we defer any discussion of strategies for choosing *where* to sample to the following chapter (Chapter 9).

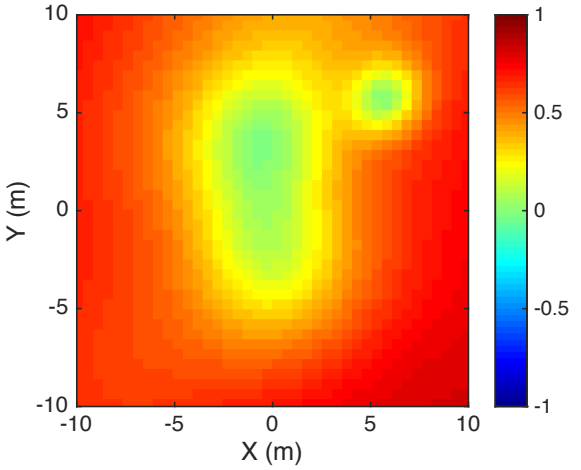
(a) Experimental site. Catch basin adjacent to farm plot. Note the red hydrant to the right of the catch basin.



(b) Normalized posterior variance at the end of the experiment. Measurement locations marked by white crosses.



(c) Visible spectrum reconstruction. Catch basin and hydrant are both clearly visible.



(d) NDVI reconstruction. Positive values correspond to vegetation.

Figure 8.6 – Data from outdoor flight with mounted spectrometer.

9 Informative Path Planning

In this chapter, we examine methods for deciding *where* to best make observations of a particular field in attempting to maximally improve our estimation of it. We implement and evaluate approaches both for sensor placement, i.e., where best to place static sensors, and path planning, i.e., how a mobile sensor should choose to move through the space, ultimately combining the two to form a heterogeneous sensor network.

Under our proposed approach, we leverage an *a priori* estimate of the statistical spatial structure of the quantity under measure in order to plan our next observation. We implement our algorithms on a real-world flying robot, equipped with an infrared thermometer, and assess their performance both in simulation and a novel indoor testbed.

In order to evaluate our approach, we propose the use of electrical heaters to create a repeatable thermal gradient. The heaters are placed with uniformly random position and orientation inside the area of flight, producing a field that can be statistically characterized. Ground truth is provided using a high-resolution thermal camera, and is compared to a flying robot's estimation produced by scanning the area with an infrared thermometer.

9.1 Related Work

Environmental sensor placement and control is an active area of research. Assuming that the field under measure can be modeled as a Gaussian process (GP), researchers have proposed multiple criteria for predicting the "informativeness" of a sampling position, most commonly either the variance of the GP or mutual information. Mutual information has been shown to be a more effective measure of sample utility [97], and has subsequently been used for both sensor placement [98] and the control of mobile sensors [99]–[101]. Alternative methods have been proposed to better cope with a lack *a priori* information about the field under measure. Ouyang et al. propose a tradeoff between gathering data that improves estimation accuracy versus data that better reveals the correlation structure of the process [102].

Some of the works listed above use more nuanced statistical models and target more complex

fields (e.g., anisotropic, nonstationary) than the approaches we develop in this chapter; however, to the best of our knowledge, none of them have been systematically validated via real experiments. Our goal in this chapter is specifically to develop an approach that lends itself well to real-world experimentation on a robotic platform. We thus evaluate our approaches exclusively on homogeneous random fields, so that we might build a baseline understanding of the performance of these approaches on a real system.

9.2 Methods

We implement and compare three approaches to guiding the quadrotor's sampling path: a naive "fixed grid" algorithm, which, given a period of time, evenly distributes its samples as a grid over the field, and two greedy algorithms that seek to maximize an information-theoretic quantity at each step.

9.2.1 Background

Our sampling strategies depend on two information theoretic quantities: entropy and mutual information.

Entropy

Entropy can be described as a measure of the uncertainty about a random variable. For a given discrete random variable, X, its entropy is

$$H(X) = -\sum_i P(X = x_i) \log(P(X = x_i)) \tag{9.1}$$

We can further define the *conditional entropy* between two random variables to represent the the amount of uncertainty that remains once the other's value is known.

$$H(X|Y) = \sum_{i,j} P(X = x_i, Y = y_j) \log\left(\frac{P(Y = y_j)}{P(X = x_i, Y = y_j)}\right) \tag{9.2}$$

Mutual Information

Mutual information is a measurement of the mutual dependence between two random variables. It can be defined in terms of entropy, as follows:

$$\begin{aligned} I(X;Y) &= H(X) - H(Y|X) \\ &= H(Y) - H(X|Y) \end{aligned} \tag{9.3}$$

for two random variables X and Y. Intuitively, mutual information is a measure of how much we learn about one variable, knowing the value of the other. If X and Y are independent, $I(X;Y) = 0$.

9.2.2 Sampling Strategies

Later in this chapter we will build hybrid networks, where a mobile node operates in part with a static sensor network. Intelligently placing static nodes not only has a great effect on estimation quality, but may also determine the path of the quadrotor.

Static Sensor Sampling Strategy

Static sensor placement is well-studied, and has seen use in a number of applications. It is a key element of the design of an environmental sensing deployment. In the remainder of this chapter, static nodes are always placed according to a greedy mutual information algorithm, as proposed in [98].

However, given that our sensors are not point sensors as in the above work, but measure an aggregate of the space inside their field of view (FOV), their full pose must be known in order to determine precisely what area of the space they are observing. We can then compute the mutual information as specified in [98], given that the covariance between two sensor observations in our model is given by Equation 8.10.

Mobile Sensor Sampling Strategy

Fixed grid Our naive approach simply distributes measurements evenly over the field, in a grid pattern. While simple, under our homogeneous GP assumption and for a fixed length of time, it is difficult to outperform.

Greedy information-theoretic We propose two information-theoretic strategies, one that moves in the direction of maximum entropy, and another that moves in the direction that maximally increases information about the underlying GP.

Here we formulate the problem a bit differently than the static case above. Instead of

an *a priori* fixed set of possible sampling locations as in [98], at each step we evaluate positions on the surface of a disk centered on the sensor's current position. The sensor moves in the direction that maximally increases the entropy or mutual information measure.

Formally, we first consider the mutual information case. The mobile sensor builds up a set of measured positions over the course of an experiment, A. At each step, we choose the point on the disk y that maximally increases our information about the set of points in the underlying GP, S. That is, we seek to maximize $I(A \cup y; S) - I(A; S)$.

Again, in a similar fashion to [98], we can derive

$$\begin{aligned} I(A \cup y; S) - I(A; S) &= H(A \cup y) - H(A \cup y \mid S) - [H(A) - H(A \mid S)] \\ &= H(A \cup y) + H(S) - H(A \cup S \cup y) - [H(A) + H(S) - H(A \cup S)] \\ &= H(A \cup y) - H(A) - [H(A \cup S \cup y) - H(A \cup S)] \\ &= H(y \mid A) - H(y \mid A \cup S) \end{aligned} \tag{9.4}$$

As our quantity to optimize. The maximum entropy strategy simply drops the second part of the equation, sampling to maximize $H(y \mid A)$.

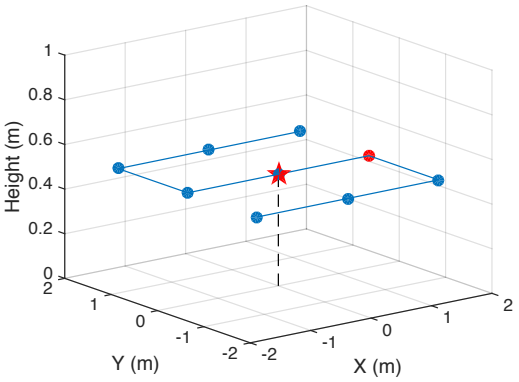
9.3 Experimental Setup

We present a performance comparison between the three aforementioned path planning strategies in both simulation and in a controlled indoor testing environment. Our unified software framework (and simulation environment) is described in Section 3.6.1, and the estimation process can be reviewed in Figure 9.1.

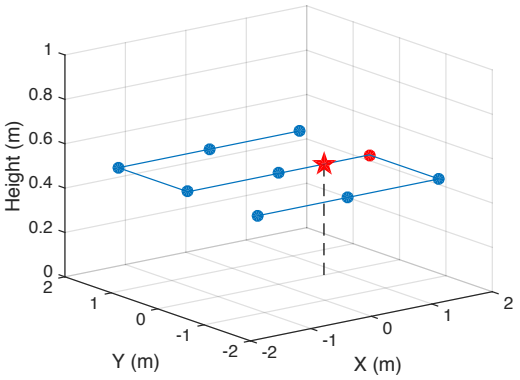
Our indoor testbed provides centimeter-accuracy localization, and a repeatable physical field to be sensed by the flying craft.

We validate our approach in a real-world setting using a flying robot in a semi-controlled environment. Specifically, we mount an infrared thermometer downward-facing on a quadrotor (see Section 3.5.1 for details). The quadrotor is flown in an indoor arena, with absolute positioning provided by a camera-based motion capture system. Electrical heaters are placed on the floor of the arena, tilted downward at a slight angle, generating a thermal gradient for the mobile sensor to sense and estimate.

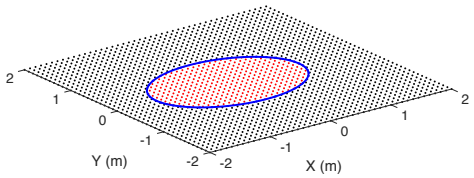
This method is flexible in that it allows for repeatable testing by rearranging the heaters in an arbitrary fashion. Performance evaluation is made possible by our choice of modality—a thermal camera is used to construct a ground truth to which the quadrotor's field estimation can be compared.



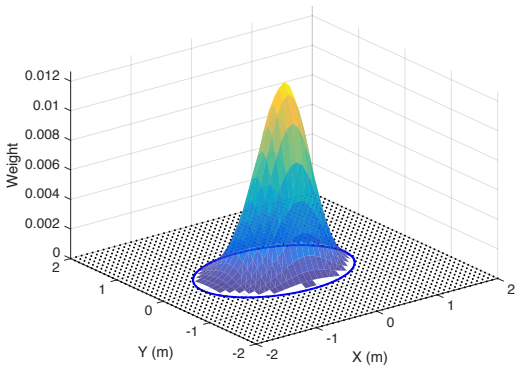
(a) The sensor position, marked by the red star, is stored in the sensor state module. Here, the fixed-grid strategy module implementation sets the sensor's target position as the next point in the grid, marked by the red circle.



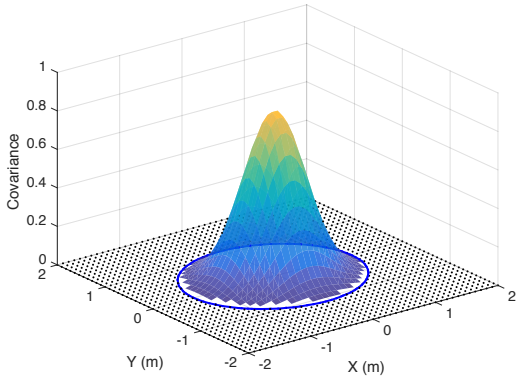
(b) As the sensor moves toward the target, its intermediate position is stored in sensor state.



(c) The sensing model queries the ground truth for grid locations observable from the sensor's current position and orientation. Here, it is tilted at a 45° angle with respect to the ground plane.



(d) Relevant elements of the GP are identified by applying a weight function to the query result.



(e) The estimator updates the posterior variance of the field based on this new measurement. Note that the shape becomes smoother in this phase due to the underlying isotropic covariance function.

Figure 9.1 – Walkthrough of estimation and planning framework.

Figure 9.2 – Experiment in progress. Electrical heaters on the ground produce a surface temperature gradient to be estimated by the quadrotor.

Our arena is equipped with a MotionAnalysis Osprey [103] motion capture system, comprised of 20 cameras distributed around the space. The system provides real-time positioning at 100 Hz with millimeter precision, over a total volume of 3×3×2 m established for our experiments. Position data is streamed to the quadrotor over Wi-Fi, where it is fused with measurements from its inertial measurement unit (IMU) in order to perform precision navigation.

Due to the limited space of our flying arena, and the narrow FOV of our mounted sensor, we performed all our scans at a fixed height of 1.8 m.

9.3.1 Environmental Field

We generate a thermal gradient to be sensed by the quadrotor's mounted infrared thermometer. At the start of each experiment, we place five electrical heating units, powered between 1–2 kW, around the floor of the arena. Each heater is tilted downward by approximately 10° to increase the effect on the arena floor. See Figure 9.2 for an example of the setup.

Given that we have chosen to model our field as stationary, by placing the heaters inside the arena with uniform position and orientation the field can also be considered isotropic.

We measured a set of ten fields generated as such using a thermal camera (see Section 9.3.2), calculated their empirical covariance and found a good fit using the Gaussian covariance model (Equation 8.2) with $a = 0.646$ m. Again, given the assumptions outlined above, this function fully characterizes the spatial structure of the field.

An anisotropic field could be generated by orienting all heaters in the same direction, or we could model existing anisotropy by dropping the assumption of stationarity. Both our

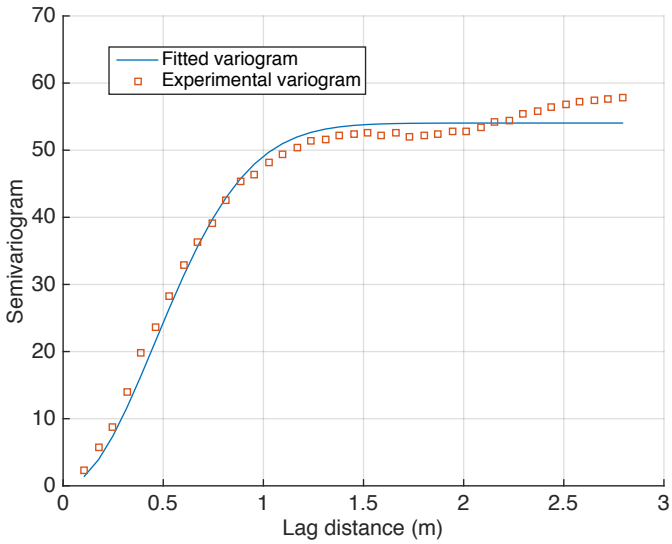


Figure 9.3 – Experimental and fitted variogram for a set of ten of our ground truth surface temperature maps, each representing an instantiation of our random electrical heater-based thermal gradients. The fitted covariance model is Gaussian, with $a = 0.646$ m.

estimation and our path planning algorithms are able to deal with this non-homogeneity, but given the amount of time required to perform these experiments, we only evaluate them on fields with homogeneous covariance structures.

9.3.2 Ground Truth Construction

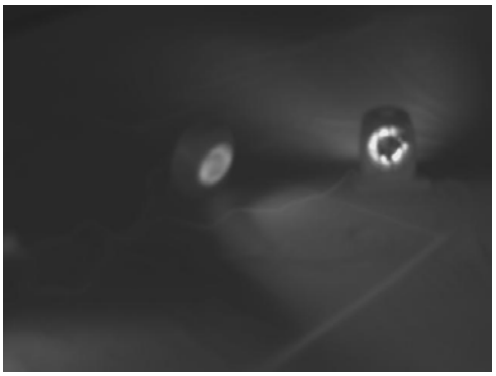
A FLIR A320 thermal camera (see Figure 9.4) is used to precisely measure the thermal gradients generated by various placements of our electrical heaters. This surface temperature map is then used to characterize the field, as above, and to evaluate the performance of our sampling strategies.

Due to the camera's limited 25° FOV, and the low ceiling in the experimental arena, it is not possible to take a picture of the whole field at once, and it must instead be constructed from a series of thermal images taken from different positions. In this section we explain this construction process.

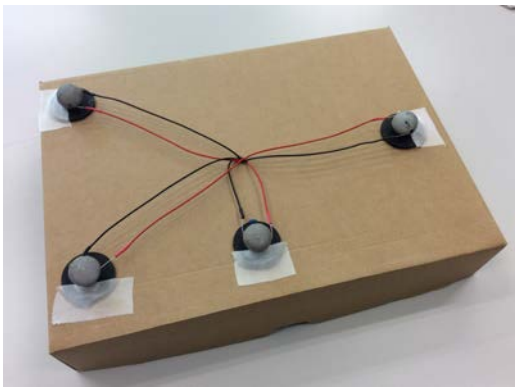
Assuming the pose of the thermal camera is known, and that the ground is flat, we can project the thermal image into the ground plane. As a first step in this projection, the three main axes of the camera are computed as

$$a_x = R_b \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, a_y = R_b \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, a_z = R_b \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \tag{9.5}$$

(a) FLIR A320 thermal camera with infrared reflectors for absolute positioning.



(b) Thermal image taken by the FLIR thermal camera during the ground truth construction process.



(c) Calibration frame with heated infrared markers.

(d) Calibration frame as seen by the thermal camera.

Figure 9.4 – The thermal gradients produced by our electrical heaters are measured using a thermal camera. Each image taken by the camera is associated with its position and orientation, obtained by correcting motion capture system pose estimates with a corrective transform, obtained by calibrating the camera with a handmade frame.

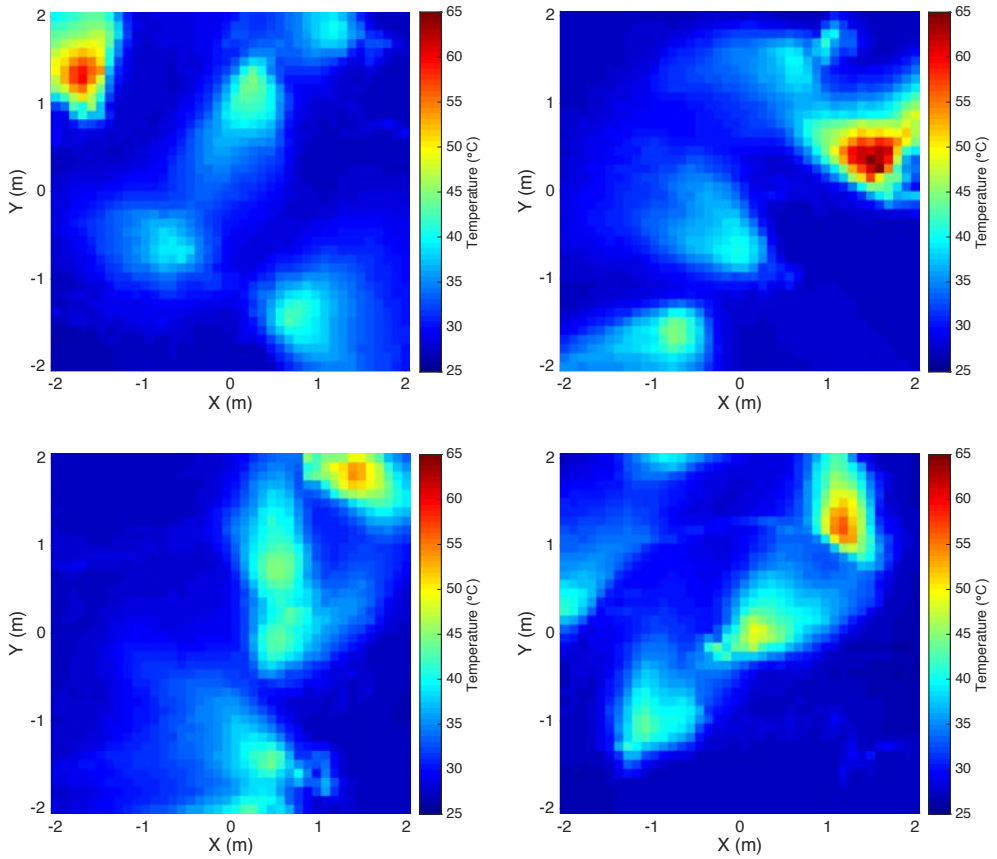


Figure 9.5 – Examples of ground truth surface temperature fields constructed using the thermal camera.

with R_b the world rotation to body rotation matrix. Assuming that there is no distortion in the image, the direction of incoming infrared light for pixel i, j at position p_x, p_y is

$$q_{ij} = a_z + \frac{\left(\frac{s_x}{2} - p_x\right) FOV}{90\frac{s_x}{2}} a_x + \frac{\left(\frac{s_y}{2} - p_y\right) FOV}{90\frac{s_x}{2}} a_y, \tag{9.6}$$

with s_x and s_y a pixel's horizontal and vertical position respectively, and FOV the camera's field of view. q_{ij} is then projected on the horizontal plane:

$$r_{ij} = p_{c,z} + \|v\|^2 \frac{q_{ij}}{q_{ij}^T v}, \tag{9.7}$$

with $v = \left[0, 0, p_{c,z}\right]^T$ the camera position's vertical component. The projection r_{ij} is performed for each pixel of each image and is associated with the temperature t_{ij} encoded in the pixel's value. To reduce the amount of data points, the points are aggregated in a grid. For each cell in the grid, the average temperature of all pixels that are inside the cell is computed. Examples of the resulting ground truth image can be seen in Figure 9.5.

Thermal Camera Calibration

Even with the motion capture system in place, estimating the exact pose of the thermal camera is not straightforward. The motion capture system returns the centroid of the positions of the infrared markers placed on the camera (i.e., the markers seen in 9.4a), with an arbitrarily aligned orientation. Given that the thermal map is constructed by taking images from many different positions inside the arena, it is critical that we accurately estimate the camera's pose.

To this end, we built a calibration frame that allows us to collect 2D–3D point correspondences between the thermal camera imagery and the pose estimated by the motion capture system, respectively. The frame consists of four infrared markers, heated from the inside by a resistor (see Figures 9.4c and 9.4d). Using standard image processing techniques, we can compute a corrective transformation between these correspondences, which we then use to obtain the camera's body rotation matrix, R_b, from the pose estimated by the motion capture system.

9.3.3 Experimental Procedure

The above calibration of the thermal camera is performed once at the start of each experimental session—which may consist of several subsequent experiments. The calibration is valid until the motion tracking system's cameras shift, e.g. due to the metal frame to which

they are mounted expanding or contracting according to temperature changes. In general, we recalibrate once every 24 hours.

Each experiment then proceeds as follows.

1. Heater placement is determined. A script is executed that generates a random position and orientation for each heater inside the arena. The heaters are placed accordingly, and we allow several minutes to pass in order for the arena's temperature to reach equilibrium.

2. Ground truth registration is performed (as in Section 9.3.2). Care must be taken during this step not to alter the positions of heating elements in the arena, including the power strips and cables running to each of the heaters.

3. Quadrotor flights are performed according to precomputed paths. Sensor data and position information is collected during the entire experiment. Each algorithm is run for each possible static sensor configuration, for a total of seven flights. Each flight lasts approximately two minutes, and the quadrotor's battery must typically be changed halfway through the seven flights.

9.4 Results

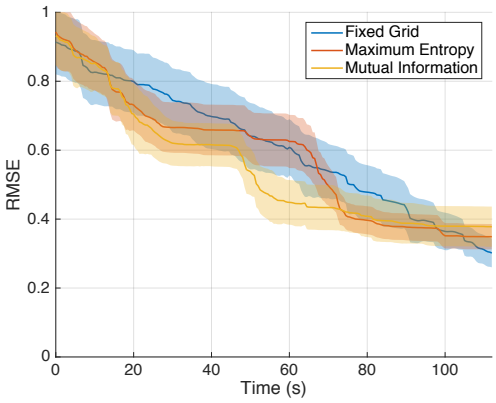
We evaluate all three algorithms—fixed grid, greedy maximum entropy, and greedy mutual information—both in simulation and our indoor testbed.

Recall that for a given experimental configuration—i.e., field size, statistical structure, and static node placement—the path generated by all three of the above algorithms can be computed *a priori*. The simulation results use the same underlying covariance function that was fit to the testbed field in the previous chapter, for sake of comparison.

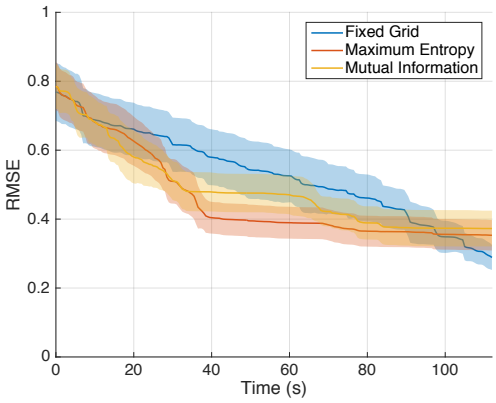
For each algorithm, we present performance results for 1) a mobile sensor operating alone, 2) a mobile sensor operating over a field of ten previously deployed static nodes, and 3) the same, finally with twenty static nodes. As described above, the static nodes are placed by a greedy mutual information strategy. In the testbed experiments, the values measured by these static nodes are simulated.

9.4.1 Simulation

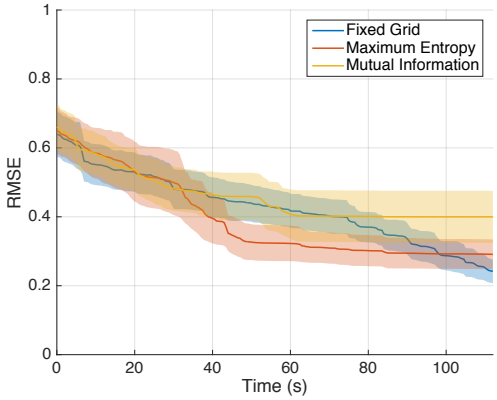
We present simulation results in Figure 9.6. Both greedy approaches perform better early in the experiment, but tend to become stuck in local extrema. The grid-based strategy perfectly distributes the given number of samples across the sampling area, so as the run continues, it eventually wins out. Note that the greedy approaches maintain a longer lead as we increase the number of static sensors placed beforehand.



(a) Without static sensors.

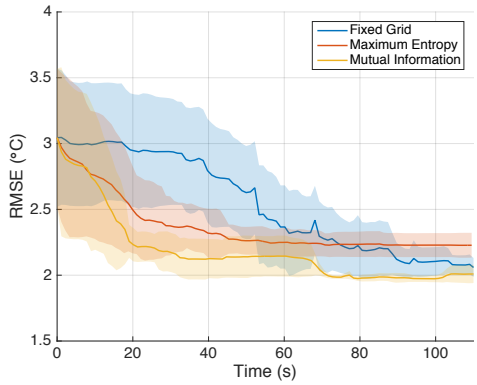


(b) Ten static sensors.

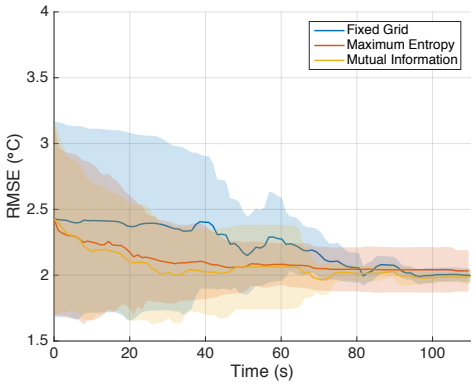


(c) Twenty static sensors.

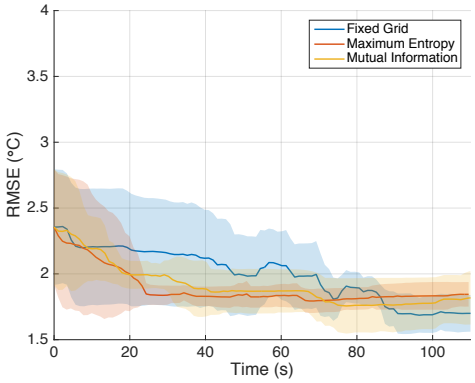
Figure 9.6 – Performance over the course of simulation for all three algorithms. Solid line represents the mean RMSE of the estimate, while the shaded area represents its standard deviation. Static sensors are placed before the start of the experiment (i.e., before sample number zero above). Each algorithm was evaluated over two hundred runs, and each run used a uniquely generated random field (Gaussian covariance with $a = 0.646$ m).



(a) Without static sensors.



(b) Ten static sensors.



(c) Twenty static sensors.

Figure 9.7 – Performance over the course of real-robot experiment for all three algorithms. Solid line represents the RMSE of the estimate, while the shaded area represents the standard deviation. Static sensors are placed before the start of the experiment (i.e., before time zero above).

Also notable is that while the mutual information-based strategy performs almost strictly better than the maximum entropy strategy in an open arena, the latter is better able to find pockets of information in the field in the presence of previously deployed static nodes.

9.4.2 Indoor Testbed

We performed five real-robot flights in our indoor testbed for each algorithm and for each possible sensor configuration. A comparison of estimate RMSE over the course of the each experiment is given in Figure 9.7.

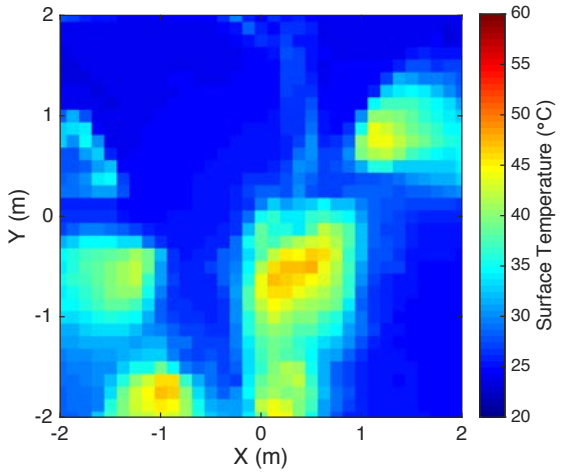


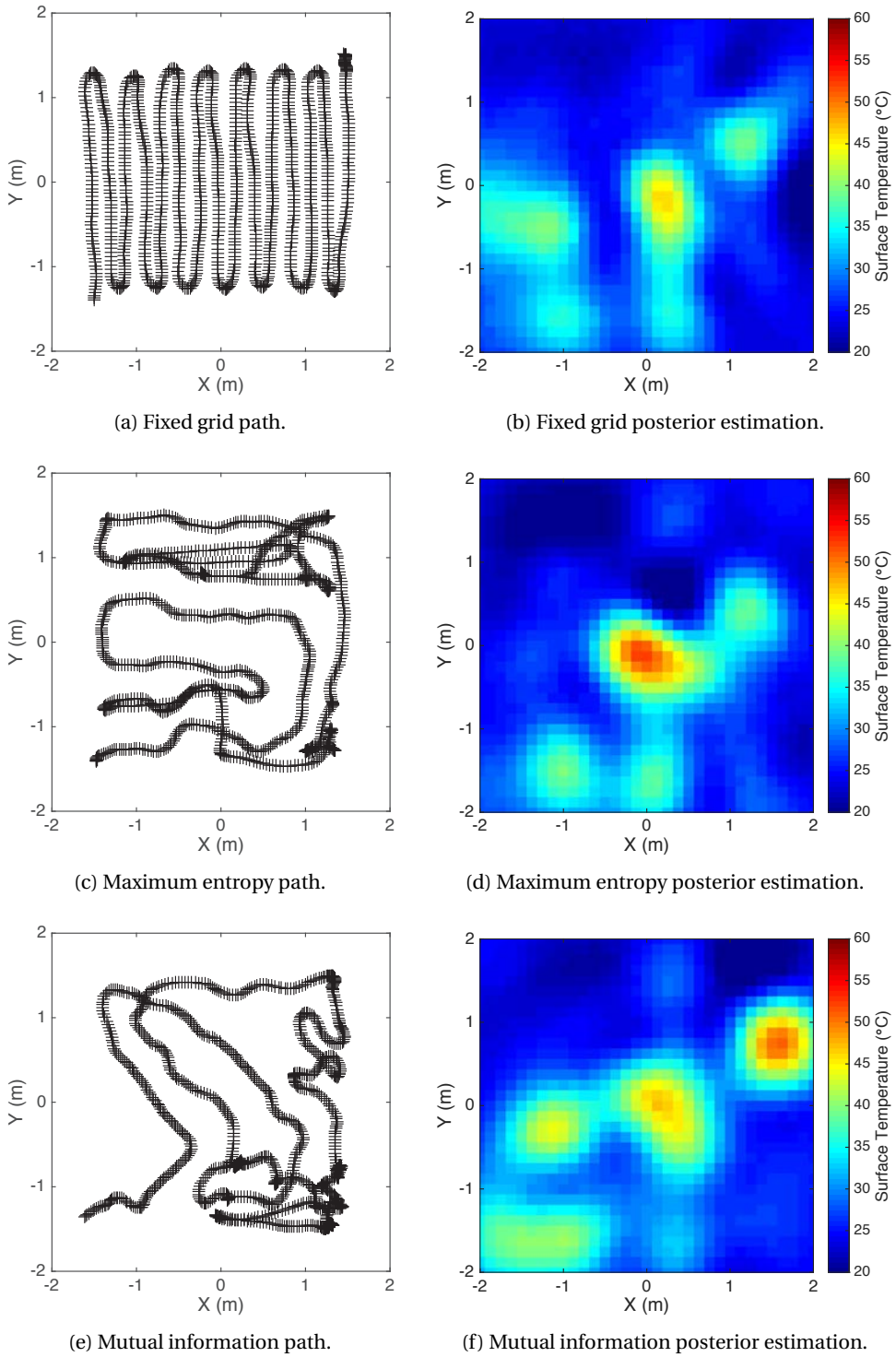
Figure 9.8 – Ground truth used in the real experiments on the following pages, as reconstructed using thermal camera imagery.

9.4.3 Discussion

First, we note a strong qualitative matching between our real-robot experiments in our indoor testbed (Figure 9.7) and our simulation results (Figure 9.6). In both simulation and reality, the greedy information-theoretic algorithms take an early lead, seeking out the most informative areas of the field, while the fixed grid approach, finishes with a higher-quality estimate—note that due to our assumption of field homogeneity, the fixed grid approach optimally distributes the samples for a given time budget.

As the number of static sensors increases, note that the information-theoretic algorithms stay competitive with the fixed grid approach later into the experiments. Early in the estimation process, the fixed grid approach takes many measurements that are redundant with those taken by the static sensors. The adaptive algorithms are able to seek out more useful measurements by avoiding areas informed by the static sensors. This effect is magnified by increased static sensor density. Again, we observe this trend in both simulation and reality.

It is difficult however to quantitatively compare the simulations with real robot results. The real-robot experiments suffer many sources of error which are not explicitly modeled, and ultimately this leads to less accurate estimates than we observe in our simulation results. Such error sources include vibration of the sensor due to the quadrotor's flight, which may increase the area observed by the sensor; changes in the temperature of the sensor itself, which has a large influence on its measured surface temperature; or airflow from the rotors disrupting the thermal gradient on the floor of the arena, resulting in measurements that do not match the ground truth. Quantifying and otherwise addressing these issues will improve the overall estimation and should ultimately lead to results in line with the ideal case, as in our simulations.



(a) Fixed grid path.

(b) Fixed grid posterior estimation.

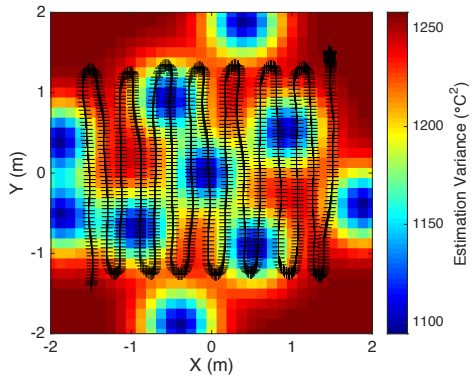
(c) Maximum entropy path.

(d) Maximum entropy posterior estimation.

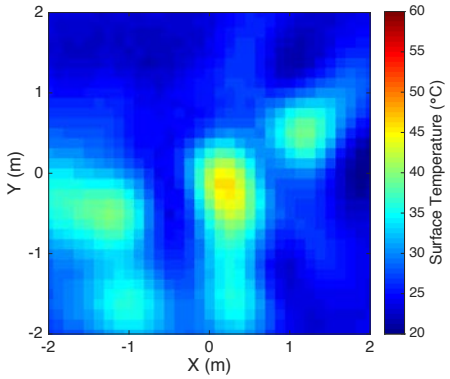
(e) Mutual information path.

(f) Mutual information posterior estimation.

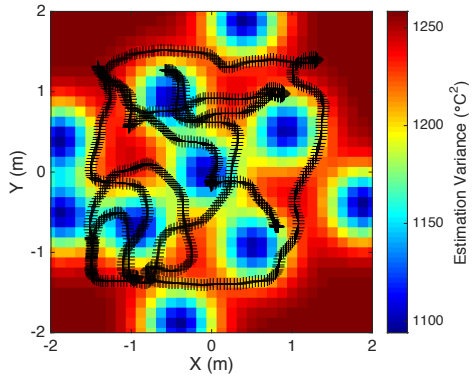
Figure 9.9 – Real-robot trajectories and estimation results for mobile sensors acting alone (i.e., without a supporting static network). (a), (c), (e) Actual trajectory of quadrotor executing the fixed grid, maximum entropy, and mutual information strategies, respectively. (b), (d), (f) Example of final field estimate upon concluding each strategy.



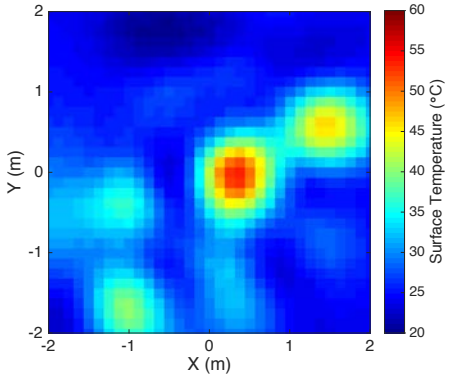
(a) Fixed grid path over static network posterior variance.



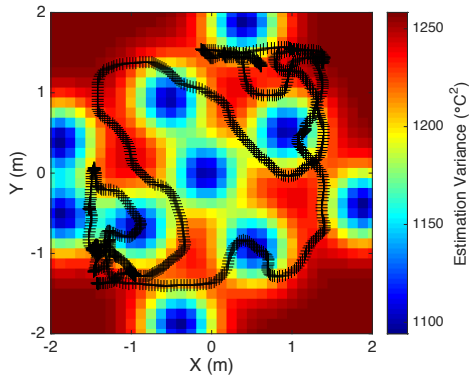
(b) Fixed grid posterior estimation.



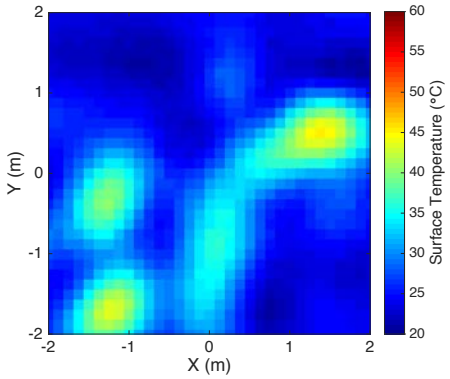
(c) Maximum entropy path over static network posterior variance.



(d) Maximum entropy posterior estimation.

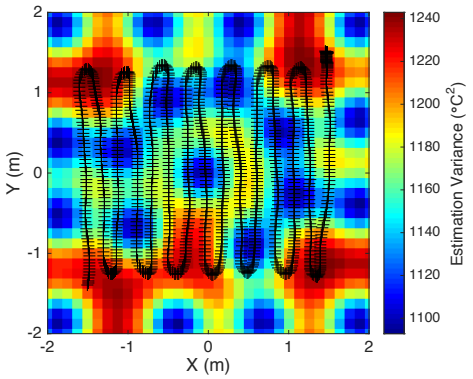


(e) Mutual information path over static network posterior variance.

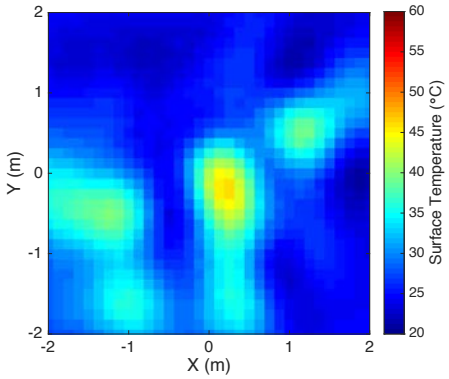


(f) Mutual information posterior estimation.

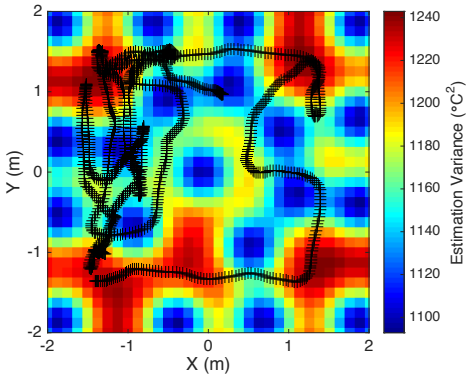
Figure 9.10 – Real-robot trajectories and estimation results for mobile sensors cooperating with a network of ten static nodes. (a), (c), (e) Actual trajectory of quadrotor executing the fixed grid, maximum entropy, and mutual information strategies, respectively. Overlaid on the posterior estimation variance for the *static network only*. The blue low variance patches thus correspond to the ten static sensors placed. The two information-theoretic strategies tend to measure more frequently where the estimation variance is higher. (b), (d), (f) Example of final field estimate upon concluding each strategy.



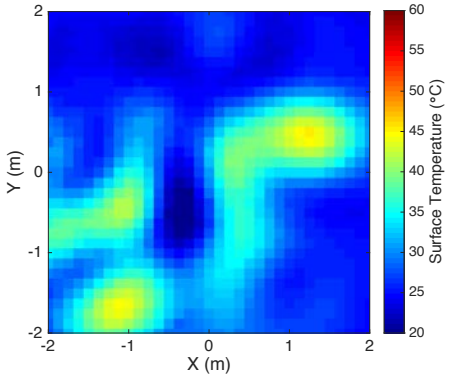
(a) Fixed grid path over static network posterior variance.



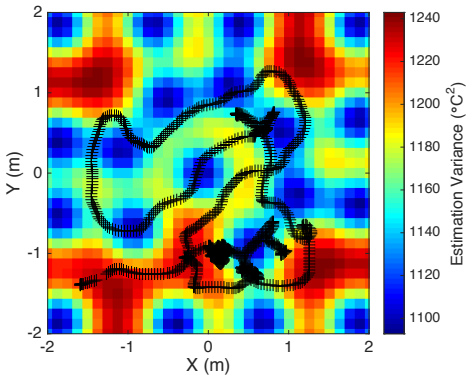
(b) Fixed grid posterior estimation.



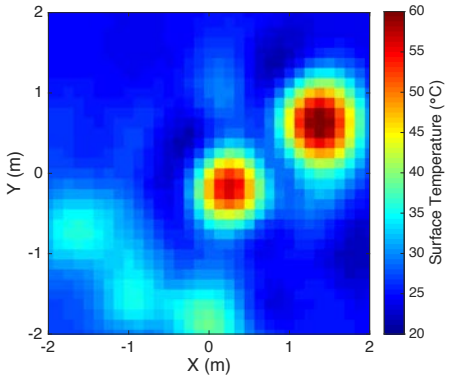
(c) Maximum entropy path over static network posterior variance.



(d) Maximum entropy posterior estimation.



(e) Mutual information path over static network posterior variance.



(f) Mutual information posterior estimation.

Figure 9.11 – Real-robot trajectories and estimation results for mobile sensors cooperating with a network of twenty static nodes. (a), (c), (e) Actual trajectory of quadrotor executing the fixed grid, maximum entropy, and mutual information strategies, respectively. Overlaid on the posterior estimation variance for the *static network only*. The blue low variance patches thus correspond to the twenty static sensors placed. The two information-theoretic strategies tend to measure more frequently where the estimation variance is higher. (b), (d), (f) Example of final field estimate upon concluding each strategy.

Future Directions

Given the greedy controllers' tendency to become stuck in local extrema, coupled with the fact that under our current model, their paths may be precomputed entirely, it is clear that long term planning would be beneficial to their performance. Given that choosing an optimal path in terms of entropy/mutual information is in the general case NP-hard [98], [104], we are currently exploring the application of traveling salesman approximation techniques to path planning in this domain.

Many environmental fields have local nonstationarity—as is the case in our indoor testbed. While the general process by which we generate our thermal gradients can only be described as stationary, local anisotropies exist, for instance following the direction of a heat vent. Ouyang et al. propose an approach to mobile sampling that is "piecewise-stationary", i.e., they fit local stationary kernels to different parts of the field in an online fashion, as measurements are taken [102]. This line of approach is able to represent much more complex spatial structures.

Due to the sparse distribution of heaters in our experiments, the testbed field often has large spaces with zero gradient. The presence of such areas makes it difficult to fit a single stationary model to the field. Interestingly, this problem also appears in the study of rainfall prediction: large areas of the space may be dry, while areas experiencing nonzero rainfall are somehow well-related. Schleiss et al. propose a sort of masking technique, in which areas not experiencing the phenomenon are treated separately from those that are [105].

9.5 Summary

In this chapter, we compared three algorithms for guiding a mobile sensor taking samples of an environmental field as part of a hybrid-mobility sensor network. We implemented and evaluated our approach in both simulation and an indoor real-robot testbed.

Our indoor testbed features controlled thermal gradients, configured by randomly placing electrical heaters pointing at the floor. The mobile node used an infrared thermometer to sample various points in the field, as governed by one of the three aforementioned algorithms, which were ultimately combined to form a two-dimensional estimate of the field.

Discussion Part V

10 Conclusion

TODAY there exists a large gap between literature surrounding the sensor networks community and real systems found "in the wild". In this thesis, we attempted to bridge this gap for very targeted scenarios. Each one of our experimental case studies required the integration of numerous technologies, and involved many different actors with widely varying fields of expertise. Real-world implementation and application of sensor network technology is often a great and expensive effort; it is our hope that in this manuscript we can communicate the color and uniqueness of results that may be obtained by following this methodology.

This chapter proceeds with reflections on our approach, practical lessons from our experimental work, a summary of our contributions, and pointers to directions of future work.

10.1 Approach

This section revisits the approach that we laid out in Chapter 2. Our primary goal in this thesis was to explore the use of distributed intelligent systems in observing and estimating environmental processes with high spatial density. We specifically sought out opportunities for experimental work, i.e., implementing various algorithms and models as part of a real-world system; and interdisciplinary projects, so that we might leverage domain expertise to contribute new sensing capabilities to other fields.

10.1.1 Distributed Sensing

Distributed sensing is vital to our aforementioned primary goal—sensing with high spatial density. For *in situ* sensors, a distributed approach is simply the only option, ignoring mobility, which may or may not be applicable depending on the time scale of the process being measured. Even in remote sensing one faces conflict between coverage, resolution, and freshness—companies such as Skybox Imaging [106] seek to augment traditional Earth observation satellites with a denser network of less expensive microsatellites, in order to collect

satellite imagery with greater coverage and freshness.

In Chapter 4, we deployed a monitoring and prediction system for a common agricultural pest. This project captures the spirit of distributed sensing in two ways. First, by simply deploying environmental sensors closer to the phenological phenomena they seek to predict—ultimately, a denser network of such stations may lead to a better understanding of the pest. Second, replacing manually checked traps with an automated pest detection system may enable greater overall trap density. According to leadership at Andermatt Biocontrol AG, farmers typically vastly underdeploy labor-intensive pest detection measures. Such an application of this technology may be a first step towards fixing these issues, thus further mitigating crop losses to these pests.

In Chapters 5, 6 and 7, we specifically investigated reducing the component-level costs of deploying dense networks. In Chapter 5, we exploited an environmental fluid dynamics model to enable the estimation of sensible heat flux with a sensor that is orders of magnitude cheaper than the alternative. Chapters 6 and 7 examine a data-aware suppression algorithm intended to reduce per-station energy usage—the idea of which has a number of benefits related to network scalability and cost.

Finally, in Chapters 8 and 9, we evaluate the performance of a hybrid-mobility sensor network for high density sampling of both simulated and real-world environmental fields.

10.1.2 Intelligent Nodes

Embedded intelligence at the node level is a key ingredient in densely deployed networks—avoiding power hungry and potentially disruptive actions such as extended use of onboard radios often requires increased software complexity.

The biological lifecycle models discussed in Chapter 4 are certainly tractable for implementation on embedded hardware. While it was not done in the course of this project, predicting for instance the eclosion of the codling moth could inform a remotely monitored trap about when to active its camera. On the other hand, embedding more basic biological knowledge of the targeted pest may also be useful. The codling moth flies to mate only one time per day, under specific conditions—again, a node could use this knowledge to avoid activating the trap's camera when no new catches are likely.

In Chapter 5, while it is a simple example, computing and transmitting only the running mean and variance of the measured temperature from the sensor node itself allows us to save significant traffic on the network. Continuing in this theme, Chapters 6 and 7 gave a more general approach to preventing unnecessary data transmission in sensor networks. However, despite suppressing a considerable amount of traffic, we did not observe real-world power savings—the underlying system constraints must be carefully considered, otherwise optimization is unlikely to bring any advantage.

While the paths computed for the mobile node in Chapter 9 could be precomputed in our case, due to properties of the Gaussian model that underlies our chosen statistical framework, algorithms that learn the covariance structure in an online fashion will require on-the-fly calculation onboard the quadrotor itself, as would a node cooperating with other unpredictable mobile sensors.

10.1.3 Experimental Methodology

An experimental approach is critical for ensuring the real-world feasibility and applicability of new algorithms and platforms. This section describes the real-world experiments performed during the course of this thesis, and their impacts.

In Chapter 4 we deployed four stations to two experimental sites from March 2014 to September 2014. Both sites were located on an apple orchard under the management and with the cooperation of Varone & Consorts. Each site was comprised of a Sensorscope "Smartrap" station and a separate Sensorscope station equipped with various environmental sensors. We found it quite difficult to manage a power-hungry station in an apple orchard. Our equipment had to be kept in line with the orchard's trees in order to avoid interfering with farming machinery, and once the leaves grew in, we experienced intermittent failures throughout the remainder of the season.

In Chapter 5, we integrated a novel method of estimating sensible heat flux with the Sensorscope platform, allowing us to perform a several-week heat flux monitoring deployment without the use of expensive hardware.

In Chapter 6, we deployed ten stations on the rooftop of an EPFL building for five weeks. We attached attenuators to each station's short-range radio antenna in order to artificially increase the diameter of the network over a small space. However, this led to the network being easily disrupted by weather effects, and in the end we only managed to retrieve 15 days of useful data.

In Chapter 9, we constructed a repeatable testbed for mobile sampling in an indoor flying arena. Using a motion capture system, thermal camera, and several electrical heaters, we were able to perform mobile estimation experiments using a mounted infrared thermometer to measure a potentially repeatable thermal gradient.

10.1.4 Interdisciplinarity

While each of the case studies in this thesis required some degree of collaboration, those in Part II were the most heavily interdisciplinary. In Chapter 4, we invested considerable effort in reaching out to and working with domain experts at Andermatt Biocontrol AG, for practicalities with respect to trap design and testing, and deployment best practices; Agroscope, for state-of-the-art knowledge on codling moth modeling and prediction; Varone & Cohorts for station

placement issues with respect to farming equipment and longevity; and Sensorscope Sarl for trap construction and online data retrieval.

In Chapter 5, the fluid dynamics model application and validation via reference deployment was due to researchers at EPFL's Laboratory of Environmental Fluid Mechanics and Hydrology (EFLUM), and sensor design and physical integration with the Sensorscope system was performed by Alexander Bahr.

Chapter 6, power board electrical and mechanical design was performed by Alexander Bahr.

10.2 Lessons Learned

During the course of the work involved in this thesis, we performed years worth of continuous data collection across multiple experimental platforms. Often we managed multiple remote deployments simultaneously, with different hardware or software configurations. This section describes some of the practical techniques we developed to ease this process.

- Keep a photographic record of all deployed hardware. We found that our practice of storing photographs of every deployed sensor station alongside collected data to be crucial to recalling not only their exact hardware configuration at the time of deployment, but also to reason about the data *a posteriori*. Particularly in exploratory work, it is difficult to predict what environmental features of the experimental site might be explanatory to the observed data—sensor height and relative position to tall objects (e.g., trees or walls) are two that proved interesting to consider when analyzing the temperature data from our codling moth deployment.

- Report software revision on station boot. In addition to recording the hardware configuration of deployed stations, the software configuration is also critical. We leveraged existing Sensorscope functionality that reports a software version number on station boot—modifying it to report a version number tied to our revision control system. This allowed us to better diagnose anomalous data post-deployment, and during deployment to map issues with particular stations to software revisions, ultimately improving our ability to iterate and improve our code.

- Real-time monitoring tools. As we accumulated more expertise in real-world sensor network deployments, we found time spent developing simple tools for monitoring station health was well invested. While triggered alerts—such as email notifications for low battery voltage and outright station failure—were somewhat useful, other failures are difficult to detect automatically, in particular individual sensor failure or malfunctioning. We found it immensely useful to have a visualization dashboard for each ongoing deployment, displaying as much time series data as possible in such a fashion that unusual events are detectable at a glance.

10.3 Summary

Here we summarize the various contributions and themes of this thesis.

- Via the integration a number of technologies, in collaboration with several partners, we built and deployed a system for detecting the presence of an agricultural pest, the codling moth, using an automated moth-counting trap. We proceed to use this trap as a ground truth for assessing the effectiveness of using locally gathered data to predict the moth's lifecycle using state-of-the-art phenological models.
- In collaboration with domain experts, we use a recently proposed statistical technique to perform sensible heat flux estimation with high spatial density, fully integrating the approach with a commercial sensor network system. An underlying physical model of the phenomenon is used to drastically reduce the actual cost per node and power consumption.
- We modified CONCH, a data suppression algorithm from the literature, such that it was feasible to implement on a commercial sensor network platform, and evaluated its affect on the actual power consumed by the station via a custom developed power monitoring extension board.
- We proposed further modifications to improve the ability of the above approach to operate in unreliable networks where centralized control may hurt performance, creating DCONCH, and evaluated its performance via calibrated simulation.
- We construct an approach for fusing multiple measurements from a single-pixel optical sensor to form a cohesive estimate, taking into account both the sensor's point spread function and *a priori* knowledge of the field under measure.
- We proposed a novel indoor testbed for evaluating the usefulness of a real-robot mobile sensor operating among static nodes in a WSN over a repeatable thermal gradient.

10.4 Future Work

Each of the case studies undertaken in this effort suggests interesting lines of future work. We group them into three categories, for each of the experimental parts of this thesis.

Model-Driven Sampling Our codling moth field study suggests opportunities for WSN technology to contribute to agricultural pest monitoring. A wide range of insects, blights, fungi and so on can be modeled using easily accessible environmental indicators, such as ambient temperature, soil temperature, and solar radiation. Agricultural areas are often underequipped to monitor the presence of these pests, and traps that are deployed may not be checked frequently. WSN approaches can enable reliable pest detection with much higher response time as compared to traditional means.

Data-Aware Efficient Communication Given the quantity of literature surrounding suppression-based approaches to efficient communication, it may be interesting to pursue low power network stacks that put the communication burden square on the transmitting node. Low-power listening is one such approach for doing so, but it has seen limited use in real-world environments due to accompanying practical issues. A suppression-based approach that takes into account the full network stack may stand to deliver significant power savings in ultra-low-power scenarios.

Hybrid-Mobility Sensor Networks Autonomous mobile sampling is a rapidly evolving field—recent papers have proposed and validated in simulation a range of techniques supporting massively scalable distributed coordination, online learning of spatial structure, and so on. We suggest, based on the experience accumulated in the production of this manuscript, that implementation on a real-robot platform, sensing real phenomena, may bring additional insights.

Bibliography

[1] D. Hartmann, A. Klein Tank, M. Rusticucci, L. Alexander, S. Brönnimann, Y. Charabi, F. Dentener, E. Dlugokencky, D. Easterling, A. Kaplan, B. Soden, P. Thorne, M. Wild, and P. Zhai, "Observations: atmosphere and surface", in *Climate Change 2013: The Physical Science Basis. Contribution of Working Group I to the Fifth Assessment Report of the Intergovernmental Panel on Climate Change*. Cambridge University Press, 2013, pp. 159–254.

[2] S. Stoeckli, M. Hirschi, C. Spirig, P. Calanca, M. W. Rotach, and J. Samietz, "Impact of climate change on voltinism and prospective diapause induction of a global pest insect – *Cydia pomonella* (l.)", *PLoS ONE*, vol. 7, no. 4, e35723, Apr. 2012.

[3] W. H. Schlesinger and E. S. Bernhardt, *Biogeochemistry: An analysis of global change*. Academic Press, 2013.

[4] **W. C. Evans**, B. Bejar, A. Bahr, M. Vetterli, and A. Martinoli, "Monitoring and predicting the presence of the adult the codling moth with a camera-based sensor network.", 2015, in preparation.

[5] H. Huwald, C. Higgins, A. Bahr, **W. C. Evans**, M. Parlange, and A. Martinoli, "Sensible heat flux from wireless environmental sensor networks", *Journal of Atmospheric and Oceanic Technology*, to be resubmitted.

[6] A. Bahr, **W. C. Evans**, A. Martinoli, H. Huwald, C. Higgins, and M. Parlange, "Measuring sensible heat flux with high spatial density", in *2012 IEEE Sensors Applications Symposium (SAS)*, Feb. 2012, pp. 1–6.

[7] **W. C. Evans**, A. Bahr, and A. Martinoli, "Distributed spatiotemporal suppression for environmental data collection in real-world sensor networks", in *9th IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS)*, Cambridge, MA, USA, May 2013, pp. 70–79.

[8] **W. C. Evans**, A. Bahr, and A. Martinoli, "Evaluating efficient data collection algorithms for environmental sensor networks", in *Proceedings of the 10th International Symposium on Distributed Autonomous Robotic Systems*, ser. Springer Tracts in Advanced Robotics, vol. 83, Springer Berlin Heidelberg, 2013, pp. 77–89.

[9] **W. C. Evans**, A. Bahr, and A. Martinoli, "A flexible in situ power monitoring unit for environmental sensor networks", in *IEEE/RSJ IROS Workshop on Environmental Monitoring (WREM)*, Oct. 2012.

[10] A. Prorok, **W. C. Evans**, and A. Martinoli, "An adaptive field estimation algorithm for sensor networks in dynamic environments", in *Workshop on Robotics for Environmental Monitoring (WREM); International Conference on Intelligent Robots and Systems (IROS)*, Sep. 2011.

[11] **W. C. Evans**, S. Roelofsen, P. Flikkema, and A. Martinoli, "Environmental field estimation with a low resolution sensor", in *Proceedings of the International Conference on Intelligent Robots and Systems (IROS 15)*, Hamburg, Germany, Sep. 2015, submitted.

[12] J. Das, **W. C. Evans**, M. Minnig, A. Bahr, G. Sukhatme, and A. Martinoli, "Environmental sensing using land-based spectrally-selective cameras and a quadcopter", in *Proceedings of the Thirteenth Int. Symp. on Experimental Robotics*, ser. Springer Tracts in Advanced Robotics, vol. 88, Springer International Publishing, 2013, pp. 259–272.

[13] T. W., "A computer movie simulating urban growth in the detroit region", *Economic Geography*, vol. 46, no. 2, pp. 234–240, 1970.

[14] Sensorscope. Official website, [Online]. Available: http://www.sensorscope.ch/ (visited on 05/28/2015).

[15] D. Gurdan, J. Stumpf, M. Achtelik, K.-M. Doth, G. Hirzinger, and D. Rus, "Energy-efficient autonomous four-rotor flying robot controlled at 1 KHz", in *2007 IEEE International Conference on Robotics and Automation*, Apr. 2007, pp. 361–366.

[16] A. T. GmbH. Asctec hummingbird, [Online]. Available: http://www.asctec.de/en/uav-uas-drone-products/asctec-hummingbird/ (visited on 05/28/2015).

[17] D. F. Nadeau, W. Brutsaert, M. B. Parlange, E. Bou-Zeid, G. Barrenetxea, O. Couach, M.-O. Boldi, J. S. Selker, and M. Vetterli, "Estimation of urban sensible heat flux using a dense wireless network of observations", *Environ. Fluid Mech.*, vol. 9, pp. 635–653, 2009.

[18] S. Simoni, S. Padoan, D. Nadeau, M. Diebold, A. M. Porporato, G. Barrenetxea, F. Ingelrest, M. Vetterli, and M. Parlange, "Hydrologic response of an alpine watershed: Application of a meteorological wireless sensor network to understand streamflow generation", *Water Resources Research*, vol. 47, W10524, 2011.

[19] C. Ranquet Bouleau, T. Baracchini, G. Barrenetxea, A. Repetti, and J.-C. Bolay, "Low-cost wireless sensor networks for dryland irrigation agriculture in Burkina Faso", in *Technologies for Development*, S. Hostettler, E. Hazboun, and J.-C. Bolay, Eds., Springer International Publishing, 2015, pp. 19–31.

[20] G. Barrenetxea, F. Ingelrest, G. Schaefer, and M. Vetterli, "The hitchhiker's guide to successful wireless sensor network deployments", in *Proceedings of the 6th ACM conference on Embedded Network Sensor Systems*, Nov. 2008, pp. 43–56.

[21] F. Ingelrest, G. Barrenetxea, G. Schaefer, M. Vetterli, O. Couach, and M. Parlange, "SensorScope: application-specific sensor network for environmental monitoring", *ACM Transactions on Sensor Networks*, vol. 6, no. 2, 17:1–17:32, Mar. 2010.

[22] H. Dubois-Ferrière, L. Fabre, R. Meier, and P. Metrailler, "TinyNode: A comprehensive platform for wireless sensor network applications", in *Proceedings of the 5th International Conference on Information Processing in Sensor Networks*, Apr. 2006, pp. 358–365.

[23] P. Dutta, M. Feldmeier, J. Paradiso, and D. Culler, "Energy metering for free: Augmenting switching regulators for real-time monitoring", in *Proceedings of the 7th International Conference on Information Processing in Sensor Networks (IPSN)*, Apr. 2008, pp. 283–294.

[24] X. Jiang, P. Dutta, D. Culler, and I. Stoica, "Micro power meter for energy monitoring of wireless sensor networks at scale", in *Proceedings of the 6th International Conference on Information Processing in Sensor Networks (IPSN)*, Apr. 2007, pp. 186–195.

[25] M. Tancreti, M. S. Hossain, S. Bagchi, and V. Raghunathan, "Aveksha: a hardware-software approach for non-intrusive tracing and profiling of wireless embedded systems", in *Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems*, Nov. 2011, pp. 288–301.

[26] P. Levis, N. Lee, M. Welsh, and D. Culler, "TOSSIM: accurate and scalable simulation of entire tinyos applications", in *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems*, Nov. 2003, pp. 126–137.

[27] J. Hill, R. Szewczyk, A. Woo, S. Hollar, D. E. Culler, and K. S. J. Pister, "System architecture directions for networked sensors", in *Architectural Support for Programming Languages and Operating Systems*, Cambridge, MA, USA, Nov. 2000, pp. 93–104.

[28] ROS Wiki. asctec_mav_framework, [Online]. Available: http://wiki.ros.org/asctec_mav_framework (visited on 05/28/2015).

[29] M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "ROS: an open-source robot operating system", in *ICRA Workshop on Open Source Software*, May 2009.

[30] senseFly. Sensefly ebee, [Online]. Available: https://www.sensefly.com/drones/ebee.html (visited on 05/28/2015).

[31] I. Phidgets. Phidget 1045 infrared temperature sensor, [Online]. Available: http://www.phidgets.com/products.php?product_id=1045 (visited on 05/28/2015).

[32] C. Huang, J. R. Townshend, S. Liang, S. N. Kalluri, and R. S. DeFries, "Impact of sensor's spread function on land cover characterization: assessment and deconvolution", *Remote Sensing of Environment*, vol. 80, pp. 203–212, 2002.

[33] H. P. K. K. Micro-spectrometer c12666ma, [Online]. Available: http://www.hamamatsu.com/resources/pdf/ssd/c12666ma_kacc1216e.pdf (visited on 05/28/2015).

[34] M. Schlather, A. Malinowski, M. Oesting, D. Boecker, K. Strokorb, S. Engelke, J. Martini, F. Ballani, P. J. Menck, S. Gross, U. Ober, K. Burmeister, J. Manitz, P. Ribeiro, R. Singleton, B. Pfaff, and R Core Team, *RandomFields: simulation and analysis of random fields*, R package version 3.0.62, 2015.

[35] M. Schlather, A. Malinowski, P. J. Menck, M. Oesting, and K. Strokorb, "Analysis, simulation and prediction of multivariate random fields with package RandomFields", *Journal of Statistical Software*, vol. 63, no. 8, pp. 1–25, 2015.

[36] M. Trapman, H. Helsen, and M. Polfliet, "Development of a dynamic population model as a decision support system for codling moth (cydia pomonella l.) management", in *Ecofruit – The 13th International Conference on Cultivation Technique and Phytopathological Problems in Organic Fruit-Growing*, Weinsberg, Germany, Feb. 2008, pp. 247–251.

[37] fruitweb GmbH. Official website, [Online]. Available: http://www.fruitweb.info/ (visited on 05/28/2015).

[38] J. C. Bergh and G. Judd, "Degree-day model for predicting emergence of pear rust mite (acari: eriophyidae) deutogynes from overwintering sites", *Environmental Entomology*, vol. 22, no. 6, pp. 1325–1332, 1993.

[39] R. Fukui, H. Fukui, and A. M. Alvarez, "Effect of temperature on the incubation period and leaf colonization in bacterial blight of anthurium", *Bacteriology*, vol. 89, no. 11, pp. 1007–1014, Nov. 1999.

[40] C. Roubal and P. C. Nicot, "Apple scab: numerical optimization of a new thermal time scale and application for modelling ascospore release in southern France", *Plant Pathology*, 2015. DOI: 10.1111/ppa.12398.

[41] J. C. Allen, "A modified sine wave method for calculating degree days", *Environmental Entomology*, vol. 5, no. 3, pp. 388–396, 1976.

[42] C. Pickel, R. S. Bethell, and W. W. Coates, "Codling moth management using degree-days", in *University of California Statewide IPM Project. Publication #4*, 1986.

[43] J. Samietz, B. Graf, H. Höhn, L. Schaub, H. U. Höpli, and E. Razavi, "Web-based decision support for sustainable pest management in fruit orchards: development of the Swiss system SOPRA", *Efficient Decision Support Systems - Practice and Challenges From Current to Future*, pp. 373–388, 2011.

[44] A. Changins-Wädenswil. Sopra output, [Online]. Available: http://www.sopra-acw.admin.ch/.

[45] Andermatt Biocontrol AG. PheroNorm, instructions of use, [Online]. Available: http://www.export.biocontrol.ch/media/pdf/products/monitoring-systems/pheronorm/pheronorm_instructions_of_use.pdf (visited on 05/28/2015).

[46] Meteolabor AG. Ventiliertes thermo-hygrometer VTP 6 (Thygan), [Online]. Available: http://www.meteolabor.ch/fileadmin/user_upload/pdf/meteo/WX/vtp6_d.pdf (visited on 05/28/2015).

[47] H. Huwald, C. W. Higgins, M.-O. Boldi, E. Bou-Zeid, M. Lehning, and M. B. Parlange, "Albedo effect on radiative errors in air temperature measurements", *Water Resources Research*, vol. 45, no. 8, 2009. DOI: 10.1029/2008WR007600.

[48] R. M. Young Company. Instructions, compact aspirated radiation shield, model 43502, [Online]. Available: http://www.youngusa.com/Manuals/43502-90(C).pdf (visited on 05/28/2015).

[49] S. Simoni, S. Padoan, D. Nadeau, M. Diebold, A. Porporato, G. Barrenetxea, F. Ingelrest, M. Vetterli, and M. B. Parlange, "Hydrologic response of an alpine watershed: application of a meteorological wireless sensor network to understand streamflow generation", *Water Resources Research*, vol. 7, no. 10, 2011. DOI: 10.1029/2011WR010730.

[50] A. S. Monin and A. M. Obukhov, "Basic laws of turbulent mixing in the surface layer of the atmosphere", *Tr. Akad. Nauk. SSSR Geophiz. Inst.*, vol. 24, no. 151, pp. 163–187, 1954.

[51] D. Baldocchi, E. Falge, L. Gu, R. Olson, D. Hollinger, S. Running, P. Anthoni, C. Bernhofer, K. Davis, R. Evans, J. Fuentes, A. Goldstein, G. Katul, B. Law, X. Lee, Y. Malhi, T. Meyers, W. Munger, W. Oechel, K. T. Paw, K. Pilegaard, H. P. Schmid, R. Valentini, S. Verma, T. Vesala, K. Wilson, and S. Wofsy, "FLUXNET: a new tool to study the temporal and spatial variability of ecosystem-scale carbon dioxide, water vopor, and energy flux densities", *Bulletin of the American Meteorological Society*, vol. 82, no. 11, pp. 2415–2434, 2001.

[52] X. Lee, W. Massman, and B. Law, *Handbook of Micrometeorology: A Guide to Surface Flux Measurement and Analysis*. Kluwer Academic Publishers, 2004.

[53] H. Soegaard, N. Jensen, E. Boegh, C. Hasager, K. Schelde, and A. Thomsen, "Carbon dioxide exchange over agricultural landscape using eddy correlation and footprint modelling", *Agricultural and Forest Meteorology*, vol. 114, no. 3, pp. 153–173, 2003.

[54] M. Rotach, R. Vogt, C. Bernhofer, E. Batchvarova, A. Christen, A. Clappier, B. Feddersen, S.-E. Gryning, G. Martucci, H. Mayer, V. Mitev, T. Oke, E. Parlow, H. Richner, M. Roth, Y.-A. Roulet, D. Ruffieux, J. Salmond, M. Schatzmann, and J. Voogt, "BUBBLE - an urban boundary layer meteorology project", *Theoretical and Applied Climatology*, vol. 81, no. 3–4, pp. 231–261, 2005.

[55] D. Baldocchi, "Assessing the eddy covariance technique for evaluating carbon dioxide exchange rates of ecosystems: past, present and future", *Global Change Biology*, vol. 9, no. 4, pp. 479–492, 2003.

[56] M. L. Reba, T. E. Link, D. Marks, and J. Pomeroy, "An assessment of corrections for eddy covariance measured turbulent fluxes over snow in mountain environments", *Water Resources Research*, vol. 45, 2009. DOI: doi:10.1029/2008WR007045.

[57] J. C. Kaimal and J. J. Finnigan, *Atmospheric boundary layer flows: their structure and measurement*. Oxford University Press, 1984.

[58] C. Tropea, A. L. Yarin, and J. F. Foss, *Springer handbook of experimental fluid mechanics*. Springer-Verlag, 2007.

[59] D. H. Lenschow and B. B. Stankov, "Length scales in the convective boundary-layer", *Journal of the Atmospheric Sciences*, vol. 43, pp. 1198–1209, 1986.

[60] D. H. Lenschow, J. Mann, and L. Kristensen, "How long is long enough when measuring fluxes and other turbulence statistics", *Journal of Atmospheric and Oceanic Technology*, vol. 11, pp. 661–673, 1994.

[61] S. Salesky, M. Chamecki, and N. L. Dias, "Estimating the random error in eddy-covariance based fluxes and other turbulence statistics: the filtering method", *Boundary-Layer Meteorology*, vol. 144, no. 1, pp. 113–135, Jul. 2012.

[62] W. Brutsaert, *Evaporation into the Atmosphere: Theory, History, and Applications*. Reidel, 1982.

[63] T. Foken, "The energy balance closure problem: an overview", *Ecological Applications*, vol. 18, pp. 1351–1367, 2008.

[64] J. R. Garratt, *The Atmospheric Boundary Layer*. Cambridge University Press, 1992.

[65] R. B. Stull, *An Introduction to Boundary Layer Meteorology*. Klewer Academic Publishers, 1988.

[66] G. G. Katul and M. B. Parlange, "Determination of average field scale soil surface-temperature from meteorological measurements", *Soil Science*, vol. 155, pp. 166–174, 1993.

[67] W. Brutsaert, "Roughness length for water-vapor, sensible heat, and other scales", *Journal of the Atmospheric Sciences*, vol. 32, pp. 2028–2031, 1975.

[68] A. T. Cahill, M. B. Parlange, and J. D. Albertson, "On the Brutsaert temperature roughness length model for sensible heat flux estimation", *Water Resources Research*, vol. 33, pp. 2315–2324, 1997.

[69] J. D. Albertson, M. B. Parlange, G. G. Katul, C.-R. Chu, H. Stricker, and S. Tyler, "Sensible heat flux from arid regions: a simple flux-variance method", *Water Resources Research*, vol. 31, no. 4, pp. 969–973, 1995.

[70] B. P. Welford, "Note on a method for calculating corrected sums of squares and products", *Technometrics*, vol. 4, no. 3, pp. 419–420, 1962.

[71] D. Tulone and S. Madden, "PAQ: Time series forecasting for approximate query answering in sensor networks", in *Proceedings of the 3rd European Conference on Wireless Sensor Networks (EWSN)*, 2006, pp. 21–37.

[72] ——, "An energy-efficient querying framework in sensor networks for detecting node similarities", in *Proceedings of the 9th ACM International Symposium on Modeling Analysis and Simulation of Wireless and Mobile Systems*, 2006, pp. 191–300.

[73] A. Prorok, C. Cianci, and A. Martinoli, "Towards optimally efficient field estimation with threshold-based pruning in real robotic sensor networks", in *IEEE International Conference on Robotics and Automation*, May 2010, pp. 5453–5459.

[74] T. Arici and Y. Altunbasak, "Adaptive sensing for environment monitoring using wireless sensor networks", in *IEEE Wireless Communications and Networking Conference*, Mar. 2004, pp. 2347–2352.

[75] S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong, "Tinydb: an acquisitional query processing system for sensor networks", *ACM Transactions on Database Systems*, vol. 30, no. 1, pp. 122–173, Mar. 2005.

[76] X. Yang, H. B. Lim, T. M. Özsu, and K. L. Tan, "In-network execution of monitoring queries in sensor networks", in *Proceedings of the 2007 ACM SIGMOD International Conference on Management of Data*, 2007, pp. 521–532.

[77] A. Silberstein, R. Braynard, and J. Yang, "Constraint Chaining: on energy-efficient continuous monitoring in sensor networks", in *Proceedings of the 2006 ACM SIGMOD International Conference on Management of Data*, Jun. 2006, pp. 157–168.

[78] R. Gallager, P. Humblet, and P. Spira, "A distributed algorithm for minimum-weight spanning trees", *ACM Transactions on Programming Languages and Systems*, vol. 5, no. 1, pp. 66–77, Jan. 1983.

[79] M. Elkin, "A faster distributed protocol for constructing a minimum spanning tree", in *Proceedings of the Fifteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, Jan. 2004, pp. 359–368.

[80] M. Khan, G. Pandurangan, and V. S. A. Kumar, "A simple randomized scheme for constructing low-weight k-connected spanning subgraphs with applications to distributed algorithms", *Theoretical Computer Science (Elsevier)*, vol. 385, no. 1–3, pp. 101–114, Oct. 2007.

[81] N. Li, J. C. Hou, and L. Sha, "Design and analysis of an MST-based topology control algorithm", *IEEE Transactions on Wireless Communications*, vol. 4, no. 3, pp. 1195–1206, May 2005.

[82] J. Cartigny, F. Ingelrest, D. Simplot-Ryl, and I. Stojmenović, "Localized LMST and RNG based minimum-energy broadcast protocols in ad hoc networks", *Ad hoc Networks (Elsevier)*, vol. 3, no. 1, pp. 1–16, Jan. 2003.

[83] F. J. Ovalle-Martínez, I. Stojmenović, F. García-Nocetti, and J. Solano-González, "Finding minimum transmission radii for preserving connectivity and constructing minimal spanning trees in ad hoc and sensor networks", *Journal of Parallel and Distributed Computing (Elsevier)*, vol. 65, no. 2, pp. 132–141, Feb. 2005.

[84] P. Brockwell and R. Davis, *Introduction to time series and forecasting*. Springer, 1994.

[85] J. Durbin, "The fitting of time series models", *Rev. Inst. Int. Stat.*, vol. 28, pp. 233–243, 1960.

[86] R. Musaloiu-E., C.-J. M. Liang, and A. Terzis, "Koala: ultra-low power data retrieval in wireless sensor networks", in *Proceedings of the 7th International Conference on Information Processing in Sensor Networks (IPSN)*, Apr. 2008, pp. 421–432.

[87] N. Burri, P. von Rickenbach, and R. Wattenhofer, "Dozer: ultra-low power data gathering in sensor networks", in *Proceedings of the 6th International Conference on Information Processing in Sensor Networks (IPSN)*, Apr. 2007, pp. 450–459.

[88] U. M. Colesanti, S. Santini, and A. Vitaletti, "DISSense: an adaptive ultralow-power communication protocol for wireless sensor networks", in *2011 International Conference on Distributed Computing in Sensor Systems and Workshops (DCOSS)*, Jun. 2011, pp. 1–10.

[89] M. C. Vuran and Ö. B. Akan, "Spatio-temporal characteristics of point and field sources in wireless sensor networks", in *IEEE International Conference on Communications*, Jun. 2006, pp. 234–239.

[90] X. Dong and M. C. Vuran, "Spatio-temporal soil moisture measurement with wireless underground sensor networks", in *The 9th IFIP Annual Mediterranean Ad Hoc Networking Workshop*, Jun. 2010, pp. 1–8.

[91] senseFly. Official website, [Online]. Available: https://www.sensefly.com/ (visited on 05/28/2015).

[92] Y. Akhtman, D. Constantin, M. Rehak, V. Nouchi, G. Shinkareva, D. Bouffard, N. Pasche, S. Chalov, U. Lemmin, and B. Merminod, "Leman-baikal: remote sensing of lakes using an ultralight plane", in *6th Workshop on Hyperspectral Image and Signal Processing*, Lausanne, Switzerland, 2014.

[93] T. Lochmatter, "Bio-inspired and probabilistic algorithms for distributed odor source localization using mobile robots", PhD thesis, IC, Lausanne, Switzerland, 2010.

[94] C. C. D. Lelong, P. C. Pinet, and H. Poilvé, "Hyperspectral imaging and stress mapping in agriculture: a case study on wheat in beauce (france)", *Remote Sensing of Environment*, vol. 66, no. 2, pp. 179–191, Nov. 1998.

[95] P. J. Zarco-Tejada, V. González-Dugo, and J. A. J. Berni, "Fluorescence, temperature and narrow-band indices acquired from a UAV platform for water stress detection using a micro-hyperspectral imager and a thermal camera", *Remote Sensing of Environment*, vol. 117, pp. 322–337, 2012, Remote Sensing of Urban Environments.

[96] D. Stajnko, M. Lakota, and M. Hočevar, "Estimation of number and diameter of apple fruits in an orchard during the growing season by thermal imaging", *Computers and Electronics in Agriculture*, vol. 42, no. 1, pp. 31–42, 2004.

[97] W. F. Caselton and J. V. Zidek., "Optimal monitoring network designs", *Statistics and Probability Letters*, vol. 2, no. 4, pp. 223–227, 1984.

[98] A. Krause, A. Singh, and C. Guestrin, "Near-optimal sensor placements in gaussian processes: theory, efficient algorithms and empirical studies", *Journal of Machine Learning Research*, vol. 9, pp. 235–284, Jun. 2008.

[99] B. J. Julian, M. Angermann, M. Schwager, and D. Rus, "Distributed robotic sensor networks: an information-theoretic approach", *International Journal of Robotic Research*, vol. 31, no. 10, pp. 1134–1154, 2012.

[100] N. Cao, K. H. Low, and J. M. Dolan, "Multi-robot informative path planning for active sensing of environmental phenomena: a tale of two algorithms", in *Proceedings of the 2013 International Conference on Autonomous Agents and Multi-agent Systems*, St. Paul, MN, USA, May 2013, pp. 7–14.

[101] G. Hoffmann and C. Tomlin, "Mobile sensor network control using mutual information methods and particle filters", *IEEE Transactions on Automatic Control*, vol. 55, no. 1, pp. 32–47, Jan. 2010.

[102] R. Ouyang, K. H. Low, J. Chen, and P. Jaillet, "Multi-robot active sensing of non-stationary gaussian process-based environmental phenomena", in *Proceedings of the 2014 International Conference on Autonomous Agents and Multi-agent Systems*, ser. AAMAS '14, Paris, France: International Foundation for Autonomous Agents and Multiagent Systems, 2014, pp. 573–580.

[103] M. A. Corporation. Osprey digital realtime system, [Online]. Available: http://www.motionanalysis.com/pdf/Osprey_System.pdf (visited on 05/28/2015).

[104] C. Ko, J. Lee, and M. Queyranne, "An exact algorithm for maximum entropy sampling", *Operations Research*, vol. 43, pp. 684–691, 1995.

[105] M. Schleiss, S. Chamoun, and A. Berne, "Nonstationarity in intermittent rainfall: the 'dry drift'", *Journal of Hydrometeorology*, vol. 15, no. 3, pp. 1189–1204, 2014.

[106] I. Skybox Imaging. Official website, [Online]. Available: http://www.skyboximaging.com/ (visited on 05/28/2015).

[107] **W. C. Evans**, G. Mermoud, and A. Martinoli, "Comparing and modeling distributed control strategies for miniature self-assembling robots", in *Proceedings of the 2010 IEEE International Conference on Robotics and Automation*, Anchorage, AK, USA, 2010, pp. 1438–1445.

[108] G. Mermoud, U. Upadhyay, **W. C. Evans**, and A. Martinoli, "Top-down vs bottom-up model-based methodologies for distributed control: a comparative experimental study", in *Proceedings of the 12th International Symposium on Experimental Robotics*, ser. Springer Tracts in Advanced Robotics, New Delhi, India, 2010.

[109] G. Mermoud, L. Matthey, **W. C. Evans**, and A. Martinoli, "Aggregation-mediated collective perception and action in a group of miniature robots", in *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS-2010)*, Nominated for CoTeSys Best Robotics Paper Award, Toronto, Canada, 2010, pp. 599–606.

[110] A. T. Bradley, **W. C. Evans**, J. L. Reed, S. K. Shimp, and F. D. Fitzpatrick, “TEM cell testing of cable noise reduction techniques from 2 MHz to 200 MHz”, in *2008 Asia-Pacific Symposium On Electromagnetic Compatibility And 19th International Zurich Symposium On Electromagnetic Compatibility*, Singapore, 2008, pp. 610–617.

Curriculum Vitae

William Christopher Evans

chris@nsevens.com

Education

2009–2015	Ph.D. Computer Science *École Polytechnique Fédérale de Lausanne (EPFL)*
2012	M.S. Computer Science *École Polytechnique Fédérale de Lausanne (EPFL)*
2004–2008	B.S. Computer Science *University of Texas at Dallas (UTD)*

Experience

Summer 2014	Software Engineering Intern *Google Inc., Madison, Wisconsin, USA*
2008–2009	Research Assistant *Distributed Intelligent Systems and Algorithms Lab. (DISAL), EPFL*
2007–2008	Research Assistant *Distributed Systems Laboratory, UTD*
Summer 2007	Research Assistant *NASA Langley Research Center, Hampton, Virginia, USA*
2004–2006	Software Engineering Intern *INET/Tektronix, Richardson, Texas, USA*

Honors and Awards

2011	EPFL ENAC Research Day, best poster award (3rd)
2009–2010	EPFL EDIC Departmental Fellowship
2008–2009	EPFL Excellence Fellowship

Publications

1. H. Huwald, C. Higgins, A. Bahr, **W. C. Evans**, M. Parlange, and A. Martinoli, "Sensible heat flux from wireless environmental sensor networks", *Journal of Atmospheric and Oceanic Technology*, to be resubmitted.

2. **W. C. Evans**, S. Roelofsen, P. Flikkema, and A. Martinoli, "Environmental field estimation with a low resolution sensor", in *Proceedings of the International Conference on Intelligent Robots and Systems (IROS 15)*, Hamburg, Germany, Sep. 2015, submitted.

3. **W. C. Evans**, A. Bahr, and A. Martinoli, "Distributed spatiotemporal suppression for environmental data collection in real-world sensor networks", in *9th IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS)*, Cambridge, MA, USA, May 2013, pp. 70–79.

4. **W. C. Evans**, A. Bahr, and A. Martinoli, "Evaluating efficient data collection algorithms for environmental sensor networks", in *Proceedings of the 10th International Symposium on Distributed Autonomous Robotic Systems*, ser. Springer Tracts in Advanced Robotics, vol. 83, Springer Berlin Heidelberg, 2013, pp. 77–89.

5. **W. C. Evans**, A. Bahr, and A. Martinoli, "A flexible in situ power monitoring unit for environmental sensor networks", in *IEEE/RSJ IROS Workshop on Environmental Monitoring (WREM)*, Oct. 2012.

6. A. Bahr, **W. C. Evans**, A. Martinoli, H. Huwald, C. Higgins, and M. Parlange, "Measuring sensible heat flux with high spatial density", in *2012 IEEE Sensors Applications Symposium (SAS)*, Feb. 2012, pp. 1–6.

7. J. Das, **W. C. Evans**, M. Minnig, A. Bahr, G. Sukhatme, and A. Martinoli, "Environmental sensing using land-based spectrally-selective cameras and a quadcopter", in *Proceedings of the Thirteenth Int. Symp. on Experimental Robotics*, ser. Springer Tracts in Advanced Robotics, vol. 88, Springer International Publishing, 2013, pp. 259–272.

8. A. Prorok, **W. C. Evans**, and A. Martinoli, "An adaptive field estimation algorithm for sensor networks in dynamic environments", in *Workshop on Robotics for Environmental Monitoring (WREM); International Conference on Intelligent Robots and Systems (IROS)*, Sep. 2011.

9. **W. C. Evans**, G. Mermoud, and A. Martinoli, "Comparing and modeling distributed control strategies for miniature self-assembling robots", in *Proceedings of the 2010 IEEE International Conference on Robotics and Automation*, Anchorage, AK, USA, 2010, pp. 1438–1445.

10. G. Mermoud, U. Upadhyay, **W. C. Evans**, and A. Martinoli, "Top-down vs bottom-up model-based methodologies for distributed control: a comparative experimental study", in *Proceedings of the 12th International Symposium on Experimental Robotics*, ser. Springer Tracts in Advanced Robotics, New Delhi, India, 2010.

11. G. Mermoud, L. Matthey, **W. C. Evans**, and A. Martinoli, “Aggregation-mediated collective perception and action in a group of miniature robots”, in *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS-2010)*, Nominated for CoTeSys Best Robotics Paper Award, Toronto, Canada, 2010, pp. 599–606.

12. A. T. Bradley, **W. C. Evans**, J. L. Reed, S. K. Shimp, and F. D. Fitzpatrick, “TEM cell testing of cable noise reduction techniques from 2 MHz to 200 MHz”, in *2008 Asia-Pacific Symposium On Electromagnetic Compatibility And 19th International Zurich Symposium On Electromagnetic Compatibility*, Singapore, 2008, pp. 610–617.