

2.9 Programming and Platforms (PRO)

CENS systems research strives to advance the state of the art in innovative in situ observing systems. To this end, our research efforts have focused on two critical themes that cut across our broad range of observing systems: the development of practical tools and platforms, and the design and evaluation of architectures and programming systems. Together, these two research directions will enable the Center to field sophisticated, rapidly reconfigurable, multi-user observing systems that support advanced sensing modalities.

Tools and Platforms for Observing Systems

A primary systems research thrust is the development of mature tools and platforms for deployments of observing systems. Our platform efforts have revolved around three activities: participatory sensing mobile to web platform mechanisms, the power aware LEAP platform, and time synchronization.

Participatory Sensing Platforms: The first are a suite of building blocks needed for the construction of robust Participatory Sensing systems. The growing real-world application experience with PS systems has clarified a set of technology challenges that span the three major components of participatory sensing systems: (1) data collection on handsets and other devices, (2) cloud-based processing, and (3) user interaction and system management on the web, as well as the underlying data models and cross-cutting security and privacy issues.

On the mobile devices themselves, research continues to better understand *handset usage models*, especially with respect to power consumption, where improved knowledge will support the goal of being able to continually run participatory sensing applications on everyday handsets. Additionally, we continue to explore embedding local processing to tighten the feedback loop with users. In general, as our deployments expand, we expect to dedicate more effort to user interface improvements and usability study.

On the data campaign management side, approaches to analyzing spatiotemporal patterns of participants are being developed to support semi-supervised *campaign recruitment*—the matching of users' availability and interests with data collection needs, as well as *path planning and automated sample requests* once participants have signed up for a collection campaign, and *performance metrics and incentive mechanisms* for managing and improving long-term participation. *Rapid campaign deployment and management tools*, borrowing from the experience of mobile crisis response systems like Ushahidi, will enable new data collection campaigns to be quickly created and deployed through the assembly and customization of a pre-existing web services and user interface components. Additionally, through collaboration with other institutions, CENS is exploring how to integrate other emerging platforms and standards for data collection, such as Open Data Kit (ODK).

As noted in the project details of this report, many participatory sensing applications combine continuous time-location "traces" gathered using mobile handset GPS and/or cell tower data combined with intermittent samples in other modalities. *Activity and place classification*, the generation of higher-level, semantically meaningful features from this basic participatory sensing data will reduce the need for participants to continually classify their own data manually, enabling them instead to make use of semantically-meaningful location and mobility data that is automatically defined in terms of place and path instead of coordinate points and time series. Handling of time-location traces themselves is being improved in applications like PEIR by using a path model rather than simply storing a collection of points.

The Spotlight project aims to develop an affordable, easy-to-use resource monitoring system that monitors fine-grained resource consumption in buildings to provide feedback to individuals about their life habits and impacts. The main goal of the project is to provide general users with an easy means to monitor their own resource consumption in their spaces. This year we developed an affordable easy-to-use appliance level power monitoring system by exploiting the fact that appliances emit measurable signals when they operate. Since appliances typically emit measurable signals when they are consuming energy, we can estimate their consumption using indirect sensing (Figure 7). The project team developed a fine-grained power monitoring system that furnishes users with an economical, self-calibrating tool that provides power consumption of virtually every appliance in buildings. The principle of operation is a network of wireless distributed sensors monitoring signals that appliances emit and forwarding them to a personal computer acting as a back-end fusion center. The fusion center collects data from the indirect sensors and measurements from the main power meter, and runs a model-based machine learning algorithm that automatically learns and estimates power consumption of every appliance on-the-fly.

Low Power Energy Aware Processing Platform (LEAP): In LEAP we have created a state of the art seismic data acquisition system based upon the successful second-generation low power, energy aware processing (LEAP) platform. This technology is now incorporated in the Reftek RT-155 platform from Refraction Technology Inc. in the form of the RT619 module that includes a micropower FPGA with LEAP's Energy Management and Preprocessing (EMAP) module and a new time synchronization module that provides sub-microsecond time accuracy from both

GPS and network timing sources. The LEAP enabled RT-155 is currently operating in field exercises with L-4 seismometers as well as acoustic microphones.

Time Synchronization for Sensing Platforms: High quality time information is crucial in embedded sensing networks, but is difficult to achieve in the presence of jitter in computational and communication latencies, time interval between resynchronizations, accuracy of time stamping network wireless packets, and quality of local clock source (quantization, frequency tolerance, aging, and drift). In Time Synchronization—Take 2 we have re-evaluated two aspects of time synchronization – sources of error in time synchronization and how they affect synchronization

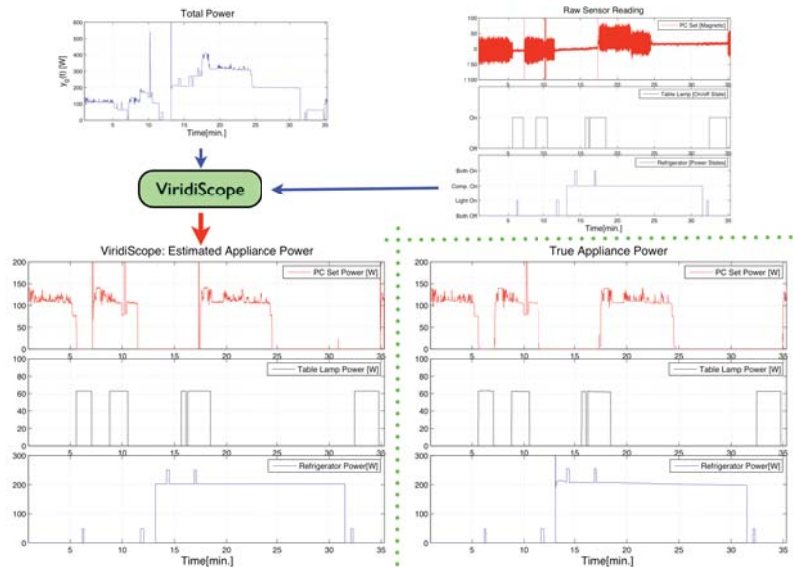


Figure 7. Non-intrusive Appliance-level power monitoring system evaluation

accuracy and energy consumption, and post-facto synchronization using sensor data. Based on that, we have developed three new mechanisms that collectively provide robust, stable, and high resolution time information at low energy costs: Temperature Compensated Time Synchronization (TCTS), Virtual High Resolution Time (VHT), and Data Driven Time Synchronization (DDTS).

Architecture and Programming Systems

Taking a sensing system that works in a small scale lab setting out into the real world might seem to be a conceptually simple step. We have repeatedly found to that this is not so; the effort to deploy a long-running reliable ENS observing system is currently perhaps an order of magnitude more than that of writing the application itself. Our ongoing research will change this: by developing visibility into a deployed system, by developing languages and tools that enable robust application development, and by re-architecting ENS software to ensure the development of manageable software, our research will enable more nimble observing systems.

Lowlog: Observing the internal execution of severely resource constrained wireless embedded devices remains a critical block to widespread adoption of embedded wireless sensing systems. Bandwidth limitations, constraining both data transfer and data storage, hinder the ability to observe runtime state in a deployed distributed embedded system using traditional logging mechanisms. LowLog, a logging framework capturing runtime control flow traces, attempts to attack this visibility problem. Our work over the past year formalizes the benefits of LowLog, and takes from this formalization the new observation that local token scoping is important for compact logging. Specifically, our work exploits statically derived runtime program behavior to create small local name spaces from which token identifiers are assigned. This results in identifiers having very small bit widths and thus minimizes the resulting log bandwidth.

Virgil: The failure of sensing device software can sometimes have life-threatening consequences. Medical monitors, or sensors that monitor the infrastructure, must be certifiably robust. Our Virgil project investigates programming language tools and software that can certify the robustness of sensing devices and software. It has developed a domain-specific language that encourages design for certifiability and make certification easier, and domain-specific tools for certifying the four key properties of space bounds, soft-real-time response, life time, and meaningful results. A tool maps a Virgil program to a timed automaton and uses a real-time model checker to check properties of a timed automaton that represents all program variables via transitions, and can capture timings of most operations.

Tenet: Finally, our Tenet project revisits the architectural foundations of the sensor network systems built and deployed by CENS. Tenet leverages the fact that every network we deploy has masters: 32-bit CPU-class nodes for which power can be engineered. Its architectural principle constrains multi-node fusion functionality to these relatively less constrained nodes. This year, the project has focused on expanding the expressivity of its tasking language. Using a general-purpose threads package and a dynamic loader for TinyOS, the project has explored an extension of its tasking language beyond linear data-flow programs. Now, tasks can contain arbitrary loop constructs and conditionals, expanding the scope of supported applications. Furthermore, the new tasking language supports pre-emption and therefore greater concurrency of execution.

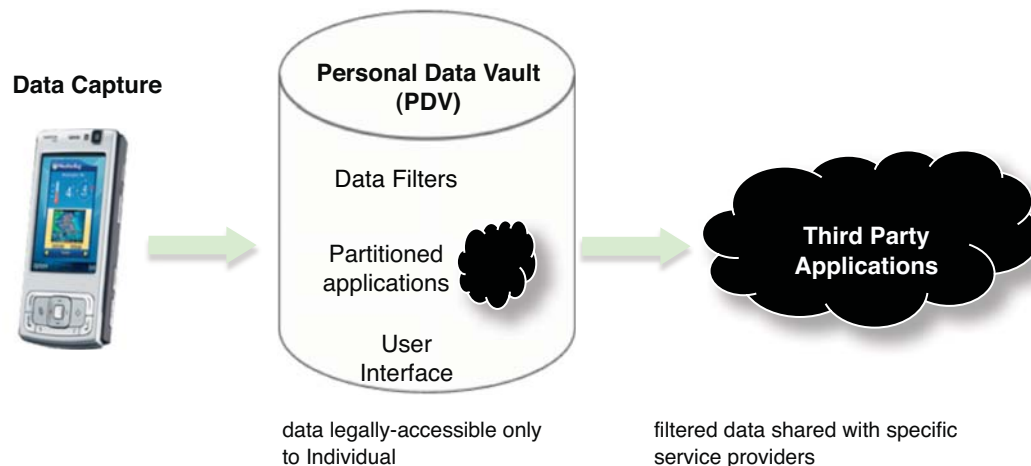
PRO 01 Personal Data Vault

PRO 01.1 Overview

Concerns about privacy are endemic to participatory sensing. Collecting individual's locations and activities is useful but sensitive. CENS is pursuing an approach to privacy that puts individuals in charge of their data, and helps them make informed decisions about selective data sharing.

Such decision-making about selective data sharing is complicated by the incomplete information about data collection usually available to participants and their often limited understanding of consequences [Acquisti & Grossklags, 2008]. But asymmetries in information and understanding can be reduced by design choices that illuminate data and support user decision-making, and compliance requirements that, like fair trade practices, encourage third parties to do the same if they wish to participate in this participatory sensing data ecosystem. These considerations lead us to four design principles: participant primacy, data legibility, longitudinal engagement, and parsimony.

Participant primacy means that individuals have control of their participatory sensing data. Any system that assists in participatory sensing should enable participants to own data generated by their devices and have complete control over data sharing decisions. Data and policy legibility means that individuals should be able to understand the data they have collected, and the policies which they have implemented to share that data. Long-term engagement means keeping participants involved with their data over time. Engagement allows individuals to check to see if their data is still visible and relevant, and make continuing, ongoing decisions about sharing policies. Finally, parsimony means sending as little data as possible out of the secure vault. By minimizing data flow out of the vault, we can reduce the viscosity of personal information while maximizing benefit of sharing to users.



PRO 01.2 Approach

We have outlined a minimal set of principles for secure storage and selective sharing of participatory sensing data. The centerpiece of our architecture to privacy for participatory sensing data is the existence of individually-controlled secure data repositories we call Personal Data Vaults (PDVs). The PDV decouples the capture and archiving of personal data streams from the sharing of that information. Instead of individuals sharing their personal data streams directly with services, we propose the use of secure containers to which only the individual has complete access. The Personal Data Vault would then facilitate the selective sharing of subsets of this information with various services over time. Selective sharing may take the form of exporting filtered information from specific times of day or places in space, or may import service computations to the data vault and export resulting computational outputs. Tools to audit information flows are also essential to support meaningful usage, and are a critical part of vault functionality.

These vaults, which could be made available to any interested individual as a public or private service, would provide secure archives of user-contributed data, and offer tools for managing and sharing subsets of that data for use by third party services according to specific filters approved by the individual on a per-service basis. Such a PDV would thus be comprised of a software system that (a) provides persistent, highly-available storage and management for spatiotemporally-tagged data, and (b) implements controlled sharing on behalf of the data owner.

PRO 01.3 System Description

The PDV is intended as an alternative to storing personal data streams directly with the third-party applications. However, storage alone is not sufficient to create an effective PDV. In order to facilitate third party data access with the permission of the user, the PDV must give users tools to boost participant primacy, data legibility, long-term

engagement and enable for data sharing parsimony. As described in detail below, the PDV will provide technical mechanisms to authenticate third parties and to control their access to a user's data. The system will also provide a user interface to enable users to view, interpret, delete and update their data.

| | |
|--------------------------------------|---|
| UI | Service interfaces |
| Computation | Management (audit, recovery, user authoring of policies) |
| Selective Sharing and Access Control | |
| Data store | Identity |

Figure 2. The PDV Platform Structure

Data store

The PDV data store has a number of functional requirements. It should be logically secure. This includes both best-practice security for the data held within the store, as well as security for the sharing rules defined by the user. The data store should be redundant to prevent data loss. The store should track data provenance and access to the data. It should also track changes to sharing rules over time.

Identity

While the PDV should support 'strong identity' that links data to the individual, it should also support anonymous and pseudonymous sharing by preventing third-parties from cross-referencing multiple streams, etc. to determine identity. (This doesn't prevent inferring identity from the data itself, which is addressed in a limited way through selective sharing)

Access Control and Selective Sharing

The PDV will provide access control, allowing users to select and limit who can see what kinds of data. The vault will also provide selective sharing, filtering the outgoing data streams according to service-specific rules. The PDV defaults to keeping all data private as they arrive. Access control and sharing mechanisms would allow users to change these default policies, setting new sharing policies for particular third parties.

Computation

Installing application-specific processing in the vault is important to meeting the design goal of parsimony and limiting the amount of data going out of the vault. One common way of protecting the privacy of location data is applying a high degree of corruption to data to disguise sensitive locations and preserve anonymity [Krumm 2007]. This technique could make many of participatory sensing applications unusable, however, because such corruption would alter application outputs to an unacceptable degree. By moving application computations that require exact location into the PDV, users can access detailed and accurate application outputs while protecting their location information.

Management

The PDV will need to support several data management functions. Users need to be able to view and manage their own data. Users should additionally be able to view and manage their sharing policies, and whether those policies are being honored by third party applications. Giving users feedback on who has viewed their data, and how frequently, is an important part of data and sharing legibility. Such feedback could enable users to update their sharing decisions should they become uncomfortable with an application's level of access.

A "Traceaudit" could be incorporated into the vault to provide this transparency. The Traceaudit, modeled after the Internet "Traceroute," would be designed to log and display transactions and transformations on a user's data. Users would contract with third-party applications that permit data owners to audit the applications' use of their data. (The application might then be rewarded with vault-certified or authorized status.) The Traceaudit would then audit the path that a certain type or particular bundle of data takes from its start on the phone, through the data vault, to third-party applications and their components. This would require a mechanism for tracking provenance through the ecosystem of vaults and services that all obey a certain set of authorized standards.

UI

A personal data stream from mobile devices could host diverse data types. Some data types are familiar to many users, but others, like accelerometer data, are not.

Users may not understand the consequences of sharing certain types of data. In addition, without technical understanding of participatory sensing applications, users find may find it difficult to understand how their data sharing choices affect application performance and accuracy.

PRO 01.4 Accomplishments

There are many possible instantiations of a PDV: a personal server housed in property owned by the user, a cloud service operated for profit (with the appropriate trust and ownership guarantees), a base virtual machine housed in a cluster in the infrastructure (also with the appropriate trust and ownership guarantees), and possibly others. For our initial implementation, we explore a complementary cloud-based option, using a bare virtual machine accessible only to user. In an exploratory implementation of this concept, the PDV is implemented using a virtual hosting platform such as OpenVZ. Each PDV runs a Linux image, and has Apache web server. Data from the mobile device is uploading using the web server, and stored in a schema-free database such as CouchDB.

Data Store

Each owner uploads data through secure means to his/her PDV. The data store is conceptually a collection of named databases, each possibly corresponding to an application or a data type. Simply using structured database like SQL might work for a short term. However, in the structure databases, as application needs evolve the schema and storage of the existing data must be updated. This often causes problems as new application needs arise, and make distributed upgrades a problem for every PDV that needs to go through a schema update. In addition, the applications may need different kinds of data at different times. We use a schema free database. Our implementation uses CouchDB. With CouchDB, no schema is enforced, and each database can contain many types of data.

Identity

Every access to the PDV must contain an authentication token. This token establishes the identity of the entity accessing the PDV, and the identity is used to determine what data can be shared. Initially, a user's identity is established using X.509 SSL certificates. Certainly, this choice is not perfect, given the lack of a widely deployed public key infrastructure. Other user authentication mechanisms based on emerging standards can ensure wider adoption: for example, using OpenID based identities, and OAuth based authentication.

Selective Sharing and Access Control

As an initial simple approach, our PDV will specify data sharing policies in JSON for each database created by the user. Our initial PDV makes spatio-temporally tagged data streams (e.g., traces from a GPS device) a first class data type. Users will be able to set up different policies for different groups of users by defining three kinds of spatial-temporal constraints: bound, precision and frequency. This is motivated both by the prevalence of location data in emerging participatory sensing systems, its potentially invasive character, and the suggestion in several user studies that spatiotemporal constraints often influence sharing decisions [Anthony, Kotz, & Henderson, 2007].

Users could configure privacy preference policies according to location or time by setting 'Bound' constraints. These constraints would specify the time interval(s) and the spatial locations for which the PDV could share data with a third party for a certain data set. Spatial bounds constraint can be specified in two ways: polygon and circle. Precision constraints are another implementation to support selective sharing. Precision constraints govern the precision of the time or location value reported to the user from this database. A PDV owner may allow a friend to access timestamps that are accurate up to a minute, or precise locations, but only allow a second acquaintance to access timestamps accurate to an hour and location accurate to a GSM or WiFi cell, a zipcode, or a city. Frequency constraints can also help users set selective sharing preferences. Frequency constraints ensure that the data shared with the user does not exceed a specified temporal frequency. Many activity recognition algorithms rely on the frequency of location information or other cues (for example, it may not be possible to accurately estimate highway driving trajectories with one location sample every 10 minutes). This service would attempt to provide data owners with some control on such use of the data. Users could make activity detection more difficult, or even impossible, by changing the frequency at which their data is shared.

Management

The basic PDV logs and displays transactions and transformations on a user's data. This can only exercise control over the first hop of data dissemination and cannot guarantee by itself control over what external third party applications do. We will mention possible advanced approaches to overcome this problem in the next section in more detail.

Service Interfaces

Both users and third-party applications communicate with our PDV implementation via HTTP and exchange data in JSON. The current PDV supports three kinds of operations under the following semantics:

Authentication

Authenticate: Verifies user/application's identify. Key value(k) has to be specified.

<http://pdv.cens.ucla.edu/authenticate?k=akdjalfadla>

Dealing with data

Upload: Allows the authenticated user to store her data at the PDV. Five arguments have to be specified: username(u), type(t), json protocol version(prv), phone version(phv), data(d). An example is a following.

http://pdv.cens.ucla.edu/upload?u=dsdfdlkjkef&t=prompt&phv=0.9&prv=0.9&d=%5Bdata_goes_here%5D

Pull: Allow the authorized third party application to read data from the PDV. Application name(u), start time(s), end time(e) information has to be specified. An appropriate access control policy in JSON is applied to data before sharing the data with the application. An example is:

<http://pdv.cens.ucla.edu/pull?u=andwellness&s=2009-11-01 01:57:14&e=2009-11-01 01:57:54>

Push: Allows a user to send data from the PDV to third party applications. The user can set up how often she wants to send data to the applications.

Subscription

Register: Register pre-configured application's filter to the PDV. Application name(u), application url(iurl), privacy policy(pcf), communication type(t) information should be specified.

http://pdv.cens.ucla.edu/register?u=andwellness&url=andwellness.cens.ucla.edu&t=pull&pcf=policy_goes_here

UI

Within the current PDV user interface, a user can view and delete uploaded data. The user can also view and delete privacy policies for each data collection and the samples of shared data. In addition, a user can look at all of the data transactions and review changes in privacy policy choices.

Deployments

To validate that it is possible to implement qualitatively different spatiotemporal data sharing application using the PDV, we have prototyped a social networking application, MobiLatitude, which lets user see the current location of all friends on a map And we have implemented a version of the PEIR [Mun 09] application using the PDV. In addition, we're develop health and wellness applications like Andwellness using the PDV.

PRO 01.5 Future Directions

The initial version of PDV platform will support data sharing policies expressed using spatiotemporal constraints for each data collection created by users. However, in reality, privacy is not a simple matter of filtering data by time and location. We expect to gain more insight into these questions as we explore a diverse suite of self-analytic applications, especially related to health and wellness. Our research team will work on identifying common use cases that have generalizable access control rules, and look for off-the-shelf query languages like SocialLite and Datalog to express the identified access control rules at a finer granularity.

In addition, To enlarge the privacy protection discussion beyond individual decision-making, our research team will pursue theoretical and empirical work into social and legal changes to fortify participant engagement with the PDV and the protection of data privacy. These include evaluating the usability of the PDV, work to situate the Vault in a viable market model, increasing the transparency of third party data practices, and legal arguments to secure a trade secrets-like legal privilege for data held in the Vault.

Reference

[Acquisti & Grossklags, 2008] A. Acquisti and J. Grossklags. What can behavioral economics teach us about privacy? *Digital Privacy: Theory, Technologies, and Practices*, 2008.

[Anthony, Kotz, & Henderson, 2007] Anthony, D., Kotz, D., & Henderson, T. (2007). Privacy in location-aware computing environments. *Pervasive Computing*, 6(4), 64-72.

[Krumm 2007] J. Krumm. Inference Attacks on Location Tracks. *Lecture Notes in Computer Science*, 4480:127, 2007.

[Mun 09] M. Mun et al. Peir, the personal environmental impact report, as a platform for participatory sensing systems research. In *MobiSys*, 2009.

PRO 02 Inferring Everyday Mobility States using Mobile Phones

PRO 02.1 Overview

Ambulation

Previously, CENS developed an accurate activity classifier for mobile phones using the accelerometer and GPS. Activity classifiers are mainly used for context sensing, which can serve either as data or metadata for participatory sensing campaigns. We developed an application, Ambulation, which uses activity classification to monitor activity traces. Ambulation is a tool for evaluating the health of patients who suffer from mobility-affecting chronic diseases such as MS, Parkinson's, and Muscular Dystrophy by monitoring how much they walk. Its classifier can distinguish between running, walking, biking, driving, or staying still. It also has a secondary classifier for indoor use that uses only the accelerometer and can classify being still, walking, or running. It uses a mobile client for collecting data and a web interface to review trends in the data over time. We also implemented energy usage optimizations to prolong the battery life of the phone.

Semi-supervised learning

We also analyzed and implemented a system for real-time activity classifiers on mobile phones to use semi-supervised learning to improve their accuracy over time, allowing them to adapt to users for whom the initial classifier may not perform well "out of the box."

PRO 02.2 Approach

Ambulation

Ambulation was developed on an Android Dev Phone 1 client and a Linux server. The mobile client, once activated, classifies the user's activity as long as the user keeps the phone on him/her. When GPS is available (e.g. outdoors), it uses both GPS and accelerometer data to infer the user's activity (still, walk, run, bike, or drive). However, when GPS is not available, it uses only the accelerometer and is limited to a more limited range of activities (still, walk, run). However, since it is designed for indoor use, these three classes of mobility are sufficient. To save energy, Ambulation uses two adaptive strategies, one for uploading data and one for managing GPS. The phone only uploads when it is plugged in or when the application first launches, so that the data is guaranteed to get uploaded as long as it is possible. Since uploading does not need to happen in real time for the purposes of the application, this avoids performing the energy-consuming operation while on battery power, extending the effective life of the battery. The other adaptive strategy is adaptive GPS, which deactivates the energy-expensive GPS when the user has been still for a few minutes. The GPS remains off until the accelerometer detects that the user has been moving consistently for the better part of a minute. This way the phone does not try to sample GPS during long periods when the user is stationary.

Semi-supervised learning

To determine the best semi-supervised learning (SSL) or active learning algorithm to automatically improve mobile activity classifiers, we analyzed the performance of self-learning, co-learning, and active learning algorithms, and implemented the most promising algorithm in a client-server system. With self-learning (Zhu, 2008) the phone adds the most confidently classified samples into the training set. Co-learning uses multiple classifiers that label samples for each other. We implemented two single-view (one set of features) co-learning algorithms based on the literature: democratic co-learning (Zhou, 2004) and En-Co-Training (Guan, 2007). We compared these SSL approaches to an active learning approach (Kapoor, 2008), where the user labels the least confidently classified samples. We tested each of the classifiers by training classifiers with subsets of data collected from 17 interns. Then, the SSL and active learning algorithms were run using data from 15 new users, and the improvement for each user was calculated.

PRO 02.3 System Descriptions

Ambulation

Ambulation has other components besides the mobile phone client: a database server (SensorBase) and a data processing, visualization, and web server, which cleans up noisy data and creates a visualization which can be accessed with a browser. The mobile phone uploads data to SensorBase, and the other server downloads from

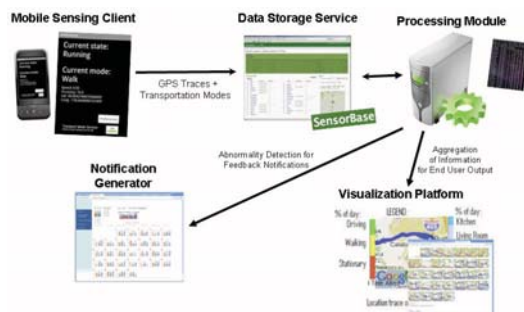


Figure 1. Ambulation System

SensorBase and generates visualizations. These are designed to intuitively show trends in a user's mobility patterns over time.

Semi-supervised learning

SSL uses a server along with the mobile client. SSL periodically generates a new classifier using the old training data and new labeled samples selected by the democratic co-learning algorithm. Since the growing training dataset must be used for this, classifier generation is performed on a server and then downloaded onto the mobile device in the form of model parameters, which the mobile device then can plug into the activity classifier. SSL data selection is performed on the phone so that only the necessary samples need be uploaded to the server for classification.

PRO 02.4 Accomplishments

Ambulation

The Ambulation system was created and able to track mobility data and display them in an intuitive format. This type of system is a way for caregivers to better monitor the progress of patients. Previously patients would have to report their ambulation manually and/or briefly walk during a periodic checkup. These methods are inaccurate and inadequate for monitoring the effectiveness of treatments or the progression of a disease. Ambulation automates collecting and displaying mobility data for higher accuracy and convenience.

Semi-supervised learning

We found that active learning and democratic co-learning can raise the accuracy of classifiers up from around 76-80% to about 90%. Democratic co-learning gave an accuracy of 1% less than active learning, but since it requires no input from the user, it is still preferable to active learning, which requires user input.

Using SSL, classifiers can be initially be trained on very little training data and still work well as the classifier learns new training data. Also, SSL improves accuracy for users who experience initial low accuracy, as sometimes happens even with normally robust classifiers.

PRO 02.5 Future Directions

Mobility classification will continue to be a part of other CENS projects, such as AndWellness. Further research may also be done in increasing the energy-efficiency of the classification algorithms and sensors.

PRO 03 Discovering semantically meaningful places

PRO 03.1 Overview

Recent advances in location technology and mobile devices have opened the door for many interesting mobile applications. These technologies offer different opportunities and limitations; the Global Positioning System (GPS) provides worldwide coverage except in buildings and underground, while technologies based on Wi-Fi and cellular signals can potentially provide relatively coarse location estimates anywhere wireless internet and voice services are available. Several commercial products have shown that a mixture of GPS and RF-beacon-based location can allow a device to compute its position ubiquitously and with high availability. The raw coordinates provided by these location systems enable location-aware applications such as navigation and emergency response that require absolute locations for only a short period of time.

Many emerging mobile applications, however, can benefit from places, which are colloquially labeled representations of locations such as “Home”, “My Office”, or “Joe’s plumbing store”, instead of a series of raw coordinates. Places can directly support applications ranging from simple location aware reminders to personalized mobile searches based on place preference. Automated place discovery can help studies of human spatial and temporal behavior, which have historically depended on laborious manual recording or direct observation. More generally, place information can help device intelligent algorithms for applications that capture and share a user’s context. Such applications collect streams of location, image, acoustic, or text data to continuously understand and record people’s activity and mobility patterns, report information about their environment (e.g. traffic, pollution levels), or exchange whereabouts among friends and family [15, 5, 16]. For participatory sensing, a place discovery technique can help reminding participants to collect data only when it matters and can effectively summarize the collected data.

PRO 03.2 Approach

Place learning algorithms attempt to find a locale that is important to an individual user and carries a semantic meaning. In this paper, an important locale is defined as a place where the user spends a substantial amount of time and/or visits frequently. Typically, the input to a place learning algorithm is a sequence of time-series sensor data (e.g. GPS coordinates, CDMA/GSM cell towers, Wi-Fi Access Point MAC addresses, etc.) and its output is a sequence of tuples (date, enter time, leave time, place name). A number of interesting place learning algorithms have been proposed both based on coordinates provided by location systems (GPS or Place Lab) or on raw RF-beacon (Wi-Fi Access Point or cell tower) fingerprints.

We designed the PlaceSense algorithm which is an evolutionary step in this line of research. PlaceSense collects Wi-Fi or cell tower radio fingerprints by scanning the environment, detects place entrance and departure using multiple successive scans. It cannot, of course, automatically assign semantically meaningful names to places, but accurately identifies place entry and exit to enhance recognizing when places are visited. It improves upon prior work in two ways: (1) it is more robust since it uses separate mechanisms for entrance and departure, and (2) is more responsive since it uses history information to rapidly detect subsequent visits. Newly-seen beacons trigger entrance determination, but the algorithm robustly avoids beacon instability (caused by weak beacons in hallways, for example). On the other hand, departure is determined by the disappearance of representative beacons seen in a place, but this disappearance is carefully made to avoid false positives.

PRO 03.3 System(s) Description and/or Experiments

Evaluating the performance of a place discovery technique is not an easy task. Unlike the localization problem where the evaluation metric is often the distance between a real coordinate and an estimated coordinate, a place is typically not a single point nor has a universal spatial shape or size. For initial evaluation, we conducted a scripted tour of 30 different places in 12 buildings and 4 outdoor plazas. Each data collector individually selected 10 places they go to often on the UCLA campus which included various building rooms, library floors, stores, gyms, patios, and food courts. Several places were within a single building and some places overlapped between participants. Three distinct visit durations (8, 10, and 15 minutes) were distributed to data collectors (10 visits per each). Distance between places varied from 1 to 10 minutes by a normal walk. For further validation, we collected 4 week-long location trace logs from each of three data collectors as they went about their normal lives. Each volunteer collected radio traces and kept a written diary of places they visited (which limited the number of real-life data we can collect). They mostly stayed within the local city limits, while a couple of traces were also collected in three different cities during a trip. All results are presented together as no significant difference was observed. Using these two sets of data, we compare PlaceSense’s effectiveness in discovering visited places and the accuracy of the detected entrance and departure time with previous work, and demonstrate that it outperforms the other methods in real-life applications. Intuitively, Place-Sense performs better than algorithms that use geographic coordinates since semantically meaningful places are often indoors where current location systems suffer in continuously providing accurate positions. Furthermore, it

outperforms RF-beacon fingerprint based algorithms by being more robust to inconsistent beacons, and more responsive in detecting short visits.

The results of PlaceSense on these real-life traces generated with a representative threshold $\text{rep} = 0.9$ and tolerance depth $\text{tmax} = 3$ (optimal configuration obtained from our initial evaluation) are shown in Figure 1 and is compared against BeaconPrint [1] and Kang et al. [2] using their suggested parameters. By focusing on representative beacons and buffering data for subsequent visits, PlaceSense reduces the number of missed and divided places while also increasing the number of interesting and false places compared to BeaconPrint (Figure 1). Kang et al. based on GPS

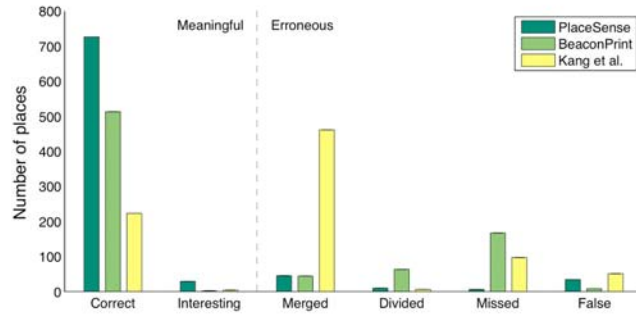


Figure 1

| | Aron | | | Bryan | | | Chris | | | All | | |
|-----------|------|------|------|-------|------|------|-------|------|------|------|------|------|
| | PS | BP | KA | PS | BP | KA | PS | BP | KA | PS | BP | KA |
| Cor. | 233 | 156 | 81 | 251 | 182 | 63 | 242 | 175 | 79 | 726 | 513 | 223 |
| Int. | 10 | 1 | 2 | 6 | 1 | 0 | 13 | 0 | 2 | 29 | 2 | 4 |
| Mer. | 6 | 14 | 138 | 23 | 15 | 185 | 16 | 15 | 138 | 45 | 44 | 461 |
| Div. | 2 | 21 | 1 | 2 | 30 | 3 | 6 | 12 | 2 | 10 | 63 | 6 |
| Mis. | 0 | 50 | 21 | 3 | 52 | 28 | 3 | 65 | 48 | 6 | 167 | 97 |
| Fal. | 6 | 2 | 10 | 14 | 2 | 20 | 14 | 4 | 21 | 34 | 8 | 51 |
| Recall | 0.97 | 0.65 | 0.34 | 0.90 | 0.65 | 0.23 | 0.91 | 0.66 | 0.30 | 0.92 | 0.65 | 0.28 |
| Precision | 0.95 | 0.81 | 0.36 | 0.87 | 0.80 | 0.23 | 0.88 | 0.85 | 0.33 | 0.89 | 0.82 | 0.30 |

Table 1

| | 5 -10 min | | | 10 - 30 min | | | 30 - 2 hrs | | | 2 - ∞ hrs | | |
|-----------|-----------|------|------|-------------|------|------|------------|------|------|-----------|------|------|
| | PS | BP | KA | PS | BP | KA | PS | BP | KA | PS | BP | KA |
| Cor. | 158 | 45 | 47 | 180 | 138 | 61 | 213 | 187 | 54 | 175 | 143 | 61 |
| Int. | 26 | 2 | 1 | 3 | 0 | 3 | 0 | 0 | 0 | 0 | 0 | 0 |
| Mer. | 12 | 15 | 97 | 13 | 12 | 102 | 17 | 12 | 158 | 3 | 5 | 104 |
| Div. | 0 | 0 | 0 | 3 | 7 | 1 | 5 | 27 | 1 | 2 | 29 | 4 |
| Mis. | 4 | 114 | 30 | 1 | 40 | 33 | 1 | 10 | 23 | 0 | 3 | 11 |
| Fal. | 33 | 6 | 38 | 1 | 2 | 12 | 0 | 0 | 1 | 0 | 0 | 0 |
| Recall | 0.91 | 0.26 | 0.27 | 0.91 | 0.70 | 0.31 | 0.90 | 0.79 | 0.23 | 0.97 | 0.79 | 0.34 |
| Precision | 0.80 | 0.69 | 0.26 | 0.92 | 0.87 | 0.36 | 0.91 | 0.83 | 0.25 | 0.97 | 0.81 | 0.36 |

Table 2

enabled cab is also discovered as a place. However, many of these cases are repeated and can be recognized so that they can be filtered out. Interesting places mostly are unrecorded brief visits to various place such as bus stop, gas station, copy room, parking lot, etc., that are recoverable during diary and map reviews.

Finally, to investigate the overall improvement of PlaceSense in recognizing revisited places by enhancing place discovery, we compare against BeaconPrint using the same recognition algorithm it uses. We focus on how well places that were actually visited are recognized. The first visits to a place discovered correctly by both algorithms are given to the algorithms for learning. Any subsequent visits are used to evaluate recognition accuracy. Our real-life data set included 143 places that have a valid learning visit, but only 63 places are visited more than once. Additionally, there are 62 places (49 were visited once) that only PlaceSense has a correct learning visit, and 6 places (5 are visited once)

Figure 2

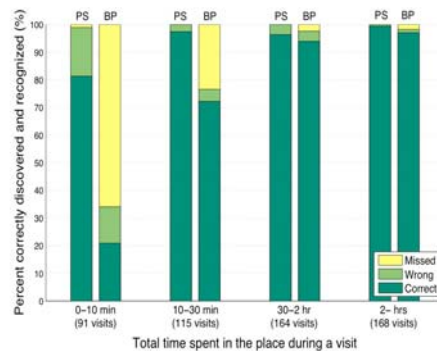


Figure 2

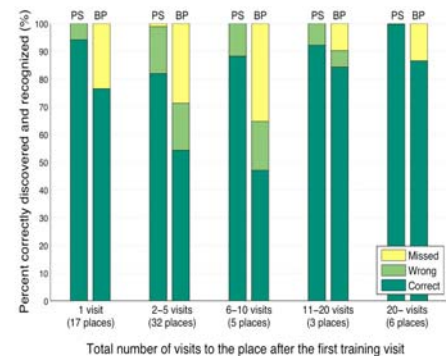


Figure 3

that only BeaconPrint has. 10 places have no valid learning visit for both algorithms. Kang et al. is excluded as its performance in correctly discovering places is significantly lower than the others. Places with no learning data are excluded. The percentage an algorithm correctly identifies a place is the ratio of the total number of places the algorithms correctly predicts to the total number of places the data collectors actually visited. Each error by the algorithms is further broken down to missed and wrong places. Figure 2 shows PlaceSense's notable strength in recognizing short visits. For both algorithms, similar percentage of places was incorrectly recognized as nearby places that shares similar beacons. Figure 3 implicates that frequently visited places are often visited for less than 30 minutes.

PRO 03.4 Accomplishments

Our results show that PlaceSense provides a significant improvement in the ability to discover and recognize places. Precision and recall with PlaceSense are 89% and 92% versus the previous state-of-the-art BeaconPrint approach at 82% and 65% precision and recall. Because it uses response rate to select representative beacons and suppresses the influence of infrequent beacons, PlaceSense's accuracy gains are particularly noticeable in challenging radio environments where beacons are inconsistent and coarse. PlaceSense also detects place entrance and departure times with over twice the precision of previous approaches thanks to judicious use of buffering and timing. It has the ability to overlap the departure fingerprint of one place with the arrival fingerprint of the subsequent place. Lastly, PlaceSense is accurate at discovering places visited for short durations (less than 30 minutes) or places where the device remains mobile. Accuracy in short-duration and transient places is a significant contribution because these types of places are valuable to emerging applications like life-logging and social location sharing.

PRO 03.5 Future Directions

Encouraged with the performance of PlaceSense, we plan to conduct additional data collections and user studies to understand about the everyday places people go. Data collected from a wider population for an extended duration will allow us to learn the number, type, and visit frequency of places people visit in more depth. Furthermore, we plan to devise an adaptive technique that minimizes the battery usage and also tracks paths between places using multiple sensors available on contemporary mobile phones.

References:

- [1] J. Hightower, S. Consolvo, A. LaMarca, I. E. Smith, and J. Hughes. Learning and recognizing the places we go. In *UbiComp '05*, pages 159–176, 2005.
- [2] J. H. Kang, W. Welbourne, B. Stewart, and G. Borriello. Extracting places from traces of locations. In *WMASH '04*, pages 110–118, New York, NY, USA, 2004. ACM.

PRO 04 Smart Power Management on Cellphones

PRO 04.1 Overview

Modern mobile phones are changing from single-purpose devices to multi-functional programmable computers. As a result, a plethora of mobile applications are emerging, many of which run in the background and collect context information to process locally or upload to a server. Most of the participatory sensing applications developed at CENS are of this type. With the presence of such applications mobile phones cannot rely on long low-power idle states to conserve energy. Therefore, such background applications negatively affect phone battery life and thus users' satisfaction of their phones. Our experiences with several such applications, including Campaignr, the Nokia Simple Context, PEIR, and Andwellness, indicate that they reduce the phone battery life to less than 12 hours, and therefore, many users stop running them. This is a major obstacle on wide deployment of Participatory Sensing mobile applications. The objective of the Smart Power Management project is to solve this problem and give users of such applications "reasonable" battery life.

Several efforts have focused on reducing power consumption of individual phone components. However, there is evidence that significant power saving potential lies at the application layer. This potential can be realized by exploiting the intrinsic trade-off between applications' quality of output and their energy consumption. In general, decreasing fidelity leads to lower energy consumption. For example, increasing the GPS sampling interval from 15 seconds to 30 seconds reduces the GPS energy consumption by 50%. The effect on quality of service, however, is application specific.

We argue that participatory sensing applications on smartphones should be able to adapt their operation rate or fidelity based on users' battery life expectations. Users cannot be expected to manage how applications run, or to keep track of the battery. Therefore, the power management system should monitor the system and plan ahead to meet the user's battery life expectation. Such system capabilities would rely on models of: battery life, user's charging behavior, energy consumed by "legacy applications" (e.g., calls), and the energy-performance trade-off of each adaptive application.

PRO 04.2 Approach

We peruse an experimental approach. Our first step in this approach is to understand how users use their mobile phones. Therefore, we need real usage traces from smartphone users to visualize and model. We use statistical modeling and learning techniques in the model building step of our research.

Along with our modeling efforts, we build systems that run on mobile phones to implement our proposed algorithms and policies. This gives us solid evaluation of the performance of different power management strategies and algorithms.

PRO 04.3 System(s) Description and/or Experiments

We initially started collecting system information on Nokia phones from a few volunteer smartphone users using a tool developed by Nokia Research Labs, named the Nokia Simple Context. We had a pilot study consisting of six volunteers with this tool. This pilot helped us understand what system level and usage events are most interesting and relevant to our study. We used the results of this study in a technical report on reducing the energy consumption of the mobile phone screen.

We took our experimental study to a next level by developing SystemSense. SystemSense is a logging software tool that was developed at CENS to log detailed system related information on Android smartphones. SystemSense keeps the logged records in a local database on the phone, and regularly uploads them to SensorBase. SystemSense has been designed to be as energy efficient as possible. We have made sure that the logging operation does not consume excessive CPU and network resources. SystemSense accesses low-level Android APIs that are only accessible to the operating system. Therefore, we can only run SystemSense on a developer phone. Following is the list of information that each version of SystemSense collects. We continue improving this tool.

Version 1.0:

- Detailed battery information:
- Battery level
- Battery voltage
- Battery temperature
- Battery health

- Charging status
- Screen status changes (on and off)
- Application usage times
- Network traffic per application
- CPU time used by each application

Version 1.1:

- GPS usage statistics per application
- WiFi interface status changes
- Call durations

Version 1.2:

- Accelerometer usage statistics per application

To minimize usage and battery perturbation SystemSense uploads the collected records only when the phone is plugged to the charger. To increase sampling accuracy SystemSense increases sampling frequency when the screen turns on (i.e., the user is interacting with the phone). During all other times, SystemSense samples resource usage counters at a fairly low frequency.

We also developed a logging tool for applications named SystemLog. SystemLog is an Android service developed at CENS to facilitate collecting performance related logs during deployments of CENS mobile applications. The SystemLog client runs as an Android Service on the phone. It defines a simple interface using the Android Interface Definition Language (AIDL). All other applications can send their log messages to SystemLog. SystemLog will augment log messages with information such as date and time and name of the logger application. The log records are kept in a local database on the phone. When SystemLog detects the phone is plugged to external power, it uploads the logged records to SensorBase.

PRO 04.4 Accomplishments

We deployed SystemSense during the summer of 2009 and collected a valuable data set. These traces are from 33 Android users. These users consisted of 16 knowledge workers and 17 high school students. All the participants worked at CENS during the summer. They were recruited to participate in our study by a staff member at CENS. As stated in our study consent form, the users' identity was never revealed to the researchers involved in this project. We are using these anonymized traces to find appropriate models of users interaction with their phones and also test the performance of several power management strategies and algorithms in trace-based simulations. However, we will not stop the evaluation at this stage. In the next section we explain how we will further evaluate our algorithms.

Several aspects of smartphone usage have been successfully modeled. We have characterized both intentional user activities—i.e., interactions with the device and the applications used—and the impact of those activities on network and battery resources. We uncovered a surprising level of diversity among users. Along all aspects that we studied, users differ by one or more orders of magnitude. For instance, the average number of interactions per day varies from 10 to 200, and the average amount of data received per day varies from 1 to 1000 MB (Figure 1). This level of diversity implies that any mechanism designed to improve user experience or energy consumption will likely be much more effective if it learns and adapts to the user behavior. Our paper that presents these results was accepted to the 8th Annual International Conference on Mobile Systems, Applications and Services.

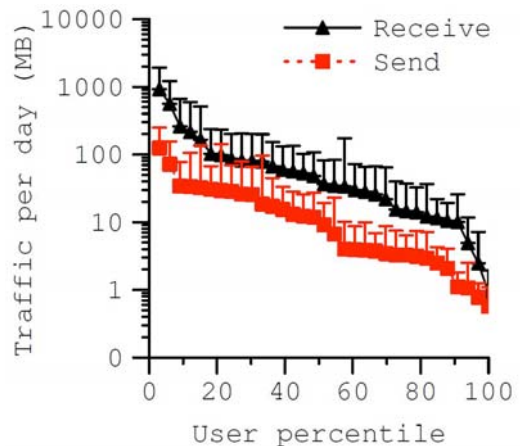


Figure 1. The mean and standard deviation of the traffic sent and received by the users

PRO 04.5 Future Directions

With our current understanding of smartphone usage, we are ready to start implementing the smart energy management system on the Android platform. We have finalized the design of the software and found the first project within CENS that is going to use our power management.

We are working with the members of the Andwellness project to incorporate energy adaptation techniques and mechanisms into their mobile application. We have a specific battery lifetime goal for Andwellness that is 20 hours of uninterrupted operation.

PRO 05 Participatory Sensing Campaigns for Sustainability: Tools for Recruitment and Management

PRO 05.1 Overview

Participatory sensing involves engaging a community of volunteers for distributed data collections using everyday mobile phones. In this project, we explored three “campaigns” involving sustainability issues on a university campus. These campaigns allowed us to create models and algorithms that can be used for both recruitment and management of participatory sensing data collections. Also, the campaigns helped evolve visualization techniques used for inferences made based on the data collected.

Project campaigns include: GarbageWatch, Asset Map, and Biketastic.

GarbageWatch is a project that involves members of the UCLA community to perform a coordinated waste audit using mobile phones. Volunteers take pictures of the tops of garbage bins and label them based on its contents - recyclables (paper, glass, aluminum, and plastic) or waste. Participants are also asked to identify recycle bin proximity. Analyzing the photo contributions along with the corresponding meta-data, allowed us to identify regions where recycling was low. A screenshot from the GarbageWatch website is presented as Figure 1.

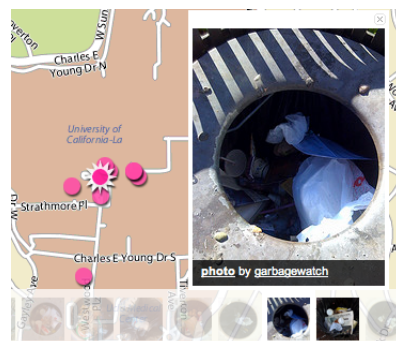


Figure 1. Screenshot of GarbageWatch website.

Asset Map is a campaign that asks volunteers at UCLA to take pictures of various sustainability assets on the campus. This includes objects such as bicycle racks and recycle bins. The information is geo-coded using the localization capabilities (GPS) of the mobile phones and the volunteers also add tags such as the number of open/available slots on the bike racks and the type of recycle bin. This information was given to the UCLA Asset coordinators to update their campus wide maps.

Biketastic is a system that enables bikers to share their routes to others. Using a mobile phone, volunteers record the location, bumpiness (accelerometer), noisiness (via microphone) of their bike routes. The route information is sent to a processing server where additional attributes such as elevation and reverse-geocoded addresses are added. By sharing the routes to the public, a organic bike map of a region can be made so that individuals can navigate from one area to another by taking frequently used routes with certain attributes in terms of quality.

PRO 05.2 Approach

In participatory sensing, one of the fundamental challenges that exists is the recruitment and management of campaigns largely due to the varied skills and interest levels of the individuals involved. We are creating a framework that is designed to help organizers of campaigns conduct data collections in an efficient manner so that campaign objectives can be maximized while considering certain resource constraints and the changing behavior of volunteers during the actual performance of the data collection.

We first analyze the recruitment problem where an organizer is interested in identifying which individuals to concentrate incentives, data collection resources, and training time on. We postulate that campaigns will have many participants that show interest but the organizer can only concentrate on a few folks to dedicate resources. Thus, organizers would be interested in finding volunteers that would maximize the amount and quality of contributions that would be made but still stay within the constraints of the campaign. The recruitment framework models participants based on three distinctive attributes: capabilities, availability, and user participation and performance. Capabilities represents the type of data that the participants are willing to collect and it is affected by the kind of device they use and the privacy levels established by the user. For instance, in GarbageWatch, if high resolution images are required with GPS granularity in terms of localization, then individuals that have low quality imagers and only share cell tower level location information would not be a good fit for the data collection. Availability represents the spatial and temporal contexts associated with a user's typical routines during a certain period of time. In Biketastic, if information about the routes in the Santa Monica region are needed during day-time hours, then the recruitment framework would look for participants that maximize this area and time of coverage when choosing individuals to target for the data collection. Finally, participation and performance relates to the likelihood, quality, and timeliness of contributions that are made by participants. If asset information needs to be collected on a daily basis to guide participants in terms of the mapping task that needs to be performed, participants that have network capabilities that are able to transmit collected data soon after capture might be desired.

system. For instance, one of the main drawbacks with our original data collection application on the phone was the long latency in taking samples. It not only involved many steps but also was cumbersome. Subsequent iterations of the mobile phone application made the data capture process streamlined and also more transparent in informing the user what was occurring during the process. Also, from surveys conducted after the campaigns, we learned the importance of visualizations - both for the campaign data itself and for how participants performed the data collections. This feedback was also incorporated into our overall campaign deployment systems by improving the analytics that was provided to participants as campaigns ran.

PRO 05.5 Future Directions

The next phase of campaign management involves studying the effect of different incentives on participation and performance of data collection campaigns by participants. We are considering exploring both bulk and micro-incentive payment schemes to see the characteristics and effectiveness in terms of participant data collection behavior. Furthermore, we are continuing to reform the system components used for data collection. Specifically, we are concentrating on making them more scalable and also easier to duplicate for fast replication.

Citations

[Reddy10a] S. Reddy, D. Estrin, M. Srivastava. "Recruitment Framework for Participatory Sensing Data Collections" to appear in the International Conference on Pervasive Computing (Pervasive), May 2010.

[Reddy10b] S. Reddy, K. Shilton, G. Denisov, C. Cenizal, D. Estrin, M. Srivastava. "Biketastic: Sensing and Mapping for Better Biking" to appear in ACM Conference on Human Factors in Computing Systems (CHI), April 2010.

[Reddy09] S. Reddy, K. Shilton, J. Burke, D. Estrin, M. Hansen, M. Srivastava. "Using Context Annotated Mobility Profiles to Recruit Data Collectors in Participatory Sensing" in the International Symposium on Location and Context Awareness (LoCA), May 2009

PRO 06 Tenet Architecture for Tiered Embedded Networks

PRO 06.1 Overview

This project revisits the architectural foundations of the sensor network systems built and deployed by CENS.

Sensor network researchers originally envisioned large networks of tiny wireless nodes with simple gateways to the Internet. Because of energy and network constraints, the tiny nodes themselves would collaboratively process data in-network in complex, application-specific ways. In reality, though, the sensor networks we deploy mostly perform continuous data acquisition, and incorporate little or no on-mote multi-node data fusion. We believe that two factors can explain this development. First, the introduction of masters: 32-bit CPU-class nodes for which power can be engineered. Masters are an integral part of every network we deploy, and rather than acting as mere Internet gateways, they participate in the functionality of the network. (Yet our current architectural principles take little advantage of them!) Second, the unexpectedly high complexity of mote-based multi-node data fusion makes implementing such functionality a bad tradeoff. By not optimizing for the system as a whole, we are missing the opportunity/have overlooked to build a software architecture that promotes on-board mote processing of its local time series in an adaptive and efficient manner.

Tiered data-collection networks are therefore here to stay. Unfortunately, only the original architectural principles are available to sensor network designers. When designers follow those principles the resulting systems are fragile and overly complex. Even worse, they are difficult to repurpose; and the master nodes on which network health depends remain underutilized.

An architecture is needed to guide the construction of scalable, evolvable, and replicable sensor systems that will serve the vast array of applications currently awaiting deployment. The Tenet project is developing such an architecture.

PRO 06.2 Approach

Many current large-scale sensor network deployments are tiered. The lower-tier, composed of motes, contains sensing and actuation functionality and enables infrastructure-less instrumentation of physical spaces and artifacts. The upper-tier, consisting of 32-bit nodes, masters, is free of energy constraints and provides increased network and computational capacity, enabling large-scale deployments.

The Tenet architecture prescribes a functional separation between motes and masters, with the goal of reducing overall system complexity. The architecture asserts that it is still desirable for the motes, which should be optimized

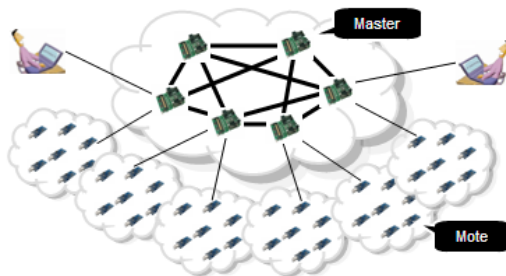


Figure 1. A Tiered Sensor Network

for low power operation, to do local aggregation, compression, and even filtering of its time series data. However, cross node aggregation, filtering, and processing is best done by the master.

Fundamentally, Tenet constrains collaborative multi-node in-network processing to be performed on the master nodes. In-network aggregation and fusion is inherently somewhat centralized in that data from multiple nodes is sent to a common node to be processed. The masters in a tiered architecture are natural fusion and aggregation points from the perspective of capacity (CPU, storage, bandwidth and energy). Constraining aggregation to be performed at masters results in a simpler architecture relative to one that

allows aggregation on topologically convenient, but resource-constrained motes. In Tenet, motes are tasked by applications running on masters, and can implement simple logical elements such as thresholds and compression, but any further computation takes place only on masters. Finally, masters can collaborate with one another to implement distributed applications for tracking, detecting spatio-temporal events, or (as in this proposal) multi-robot coordination (Figure 1).

PRO 06.3 System(s) Description and/or Experiments

The current Tenet system consists of several components: a tasking library which supports the composition and execution of small programs called tasks; a routing subsystem which uses a multi-sink version of a standard tree routing protocol for routing data from motes to masters; a task dissemination subsystem which ensures the reliable delivery of tasks from any master to all the motes, a transport subsystem which provides end-to-end reliable transmission of sensor data from motes to masters, and a time synchronization component which ensures that all the motes maintain a globally synchronized time.

Recently we have designed and implemented two versions of Tenet using the threads primitive, called TOSThreads, that has recently become available in TinyOS. Using threads allow Tenet to be robust even in applications with long running CPU intensive tasks. Tenet-T is a reimplementation of Tenet using the TOSThreads library that spawns one thread to service each Tenet task and replaces Tenet's original task scheduler with the TOSThreads thread scheduler. The main difference between Tenet-T and Tenet is that TOSThreads supports preemption. This allows each thread running a Tenet task to execute its tasklets one after another without regard for how long each of them might take. In the original Tenet, the Tenet task scheduler has to interleave tasklets from multiple tasks in order to keep them all responsive; individual tasklets run to completion and cannot perform long running computations. Tenet-T removes this limitation at the expense of a small increase in code size. Tenet-C is a reimplementation of Tenet that significantly increases the expressivity of the tasking language, yet does not require drastic modifications to the overall system. In Tenet-C, the user writes a C program, instead of a data-flow task description, and compiles it into a dynamically loadable binary object. The Tenet-C kernel is identical to that of Tenet-T except that it dynamically link and load application binaries. However, Tenet-C's API is significantly smaller because the functionality provided by many of the Tenet tasklets are already provided natively by C. Figure 2 provides a pictorial overview of these modifications. We have validated through experiments that our design correctly support timing-sensitive applications in the presence of long-running computations.

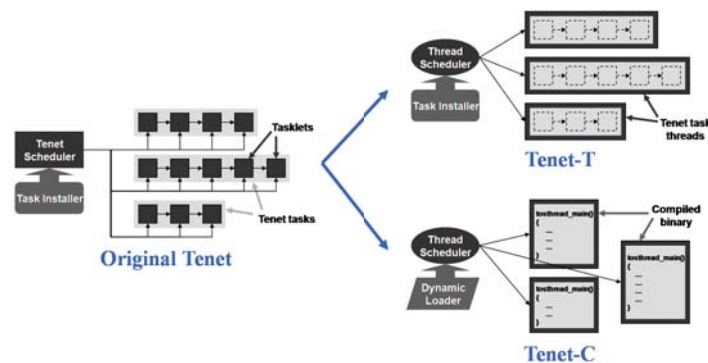


Figure 2 Implementation of Tenet using TOSThreads library

PRO 06.4 Accomplishments

Software:

We released the threaded version of Tenet (Threaded-Tenet) implemented on top of the TOSThreads and TinyLD library in TinyOS 2.1 software in June 2009. The instruction to download and use the software can be found at <http://enl.usc.edu/software.html>.

We have also updated Tenet 2.0 release to Tenet 2.1. This release includes robust version of many experimental features we developed in the past such as RCRT congestion control protocol.

Deployment:

In the past, we have implemented and deployed Pursuit Evasion Game and seismic monitoring network on a suspension bridge. Most recently, we have extended Pursuit Evasion Game into a multi-agent pursuit Evasion Game in which multiple robotic pursuers collectively determine the location of multiple evaders, and try to corral them.

PRO 06.5 Future Directions

Our work for next year will be focused on deployment of Tenet as well as evolution of the architecture and its components to allow Tenet to be used in a variety of settings.

We plan to deploy Tenet for seismic sensing of buildings using the imote-2 based platform. We are currently porting Tenet to this platform and designing the software to manage and coordinate the deployments.

To enable energy-efficient deployments of Tenet, we designed AEM to duty-cycle the radio. Although we have experimented with AEM in the in-door laboratory setting, we have not yet deployed Tenet with AEM in the field. We plan to take that step and use the lessons learnt to further improve the design of AEM.

PRO 07 LEAP—Low Power Energy Aware Processing

PRO 07.1 Overview

A broad range of embedded networked sensor (ENS) systems for important environmental monitoring and other applications now require advanced capabilities to support highly capable, high power sensor devices. Many of these applications also require support for on-demand high performance computing and communication for complex information processing. This includes image processing, statistical computing, and optimization algorithms required for selection of proper sensor sampling. In order to support these applications efficiently we require an ENS platform designed with integrated energy monitoring and scheduling features.

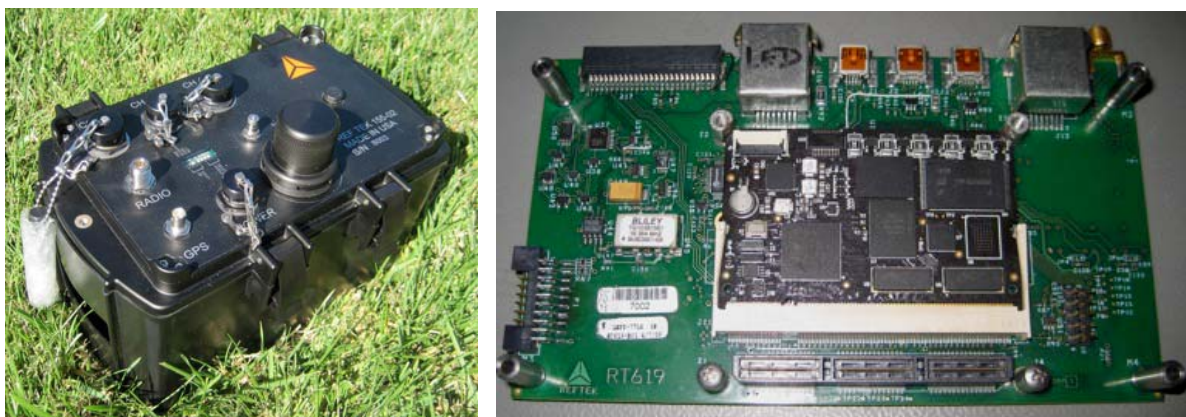


Figure 1: The RT-155 data acquisition system and RT619 with processor module

PRO 07.2 Approach

In collaboration with Refraction Technology Inc., we have developed a state of the art seismic data acquisition system based upon the successful LEAP2 platform. The LEAP2 platform is a second generation low power, energy aware processing (LEAP) ENS system that was designed specifically to allow high accuracy, low overhead energy measurement of platform resources at granularity levels previously unachievable. Additionally, energy measurement domains may be added to or removed from the platform through an expandable energy monitoring bus integrated into the LEAP2 stacking connectors. We note that LEAP enables energy aware applications through scheduling and energy profiling of high energy efficiency components including multiple wireless network interfaces, storage elements, and sensing capabilities. As ENS applications continue to increase in scale and complexity, energy profiling with high temporal resolution is now required to permit per process and per subsystem energy accounting.

At the heart of the LEAP system is its Energy Management and Preprocessing (EMAP) capability. On LEAP2 this is integrated into a dedicated ASIC implemented in a micro-power antifuse based FPGA. The EMAP2 ASIC performs continuous, real-time energy monitoring functions as well as sophisticated power scheduling across the entire LEAP2 platform. This allows for a new design approach that focuses on minimizing energy required for each individual sensing, computing, and communication task. Through the EMAP2 ASIC, LEAP2 peripherals may be scheduled for use only when needed and detailed energy information is gathered during their operation. Energy usage information for individual platform subsystems including computational resources, such as the PXA270 microprocessor, memory subsystems, such as the SDRAM, NOR flash, NAND flash and SRAM, and peripheral subsystems, such as the Ethernet, 802.11, USB, Imaging, Compact Flash, and external sensors modules are all available at millisecond accuracies. In addition, the EMAP2 ASIC's energy data and scheduling controls are available to the host processor through a high bandwidth memory bus interface, minimizing measurement overhead issues. This enables the host processor to obtain energy usage information across a wide range of devices at millisecond intervals and with a minimal overhead. These features provide LEAP2 with a unique platform monitoring and control capability that allows one to significantly reduce overall platform energy usage.

GeoNet

The Reftek RT-155, shown in Figure 1, incorporates LEAP2 technology to provide energy aware data acquisition enabling low average power dissipation. The LEAP technology is embedded in a new RT619 circuit board including a micropower FPGA. The FPGA contains logic from the EMAP2 ASIC as well as a new time synchronization module, proving sub microsecond time accuracy from both GPS and network timing sources. The FPGA dissipates less than 2mW while performing energy measurements from up to 24 energy domains as well as operating the power management scheduler and time synchronization algorithm. The RT619 module interfaces with the LEAP2 HPM

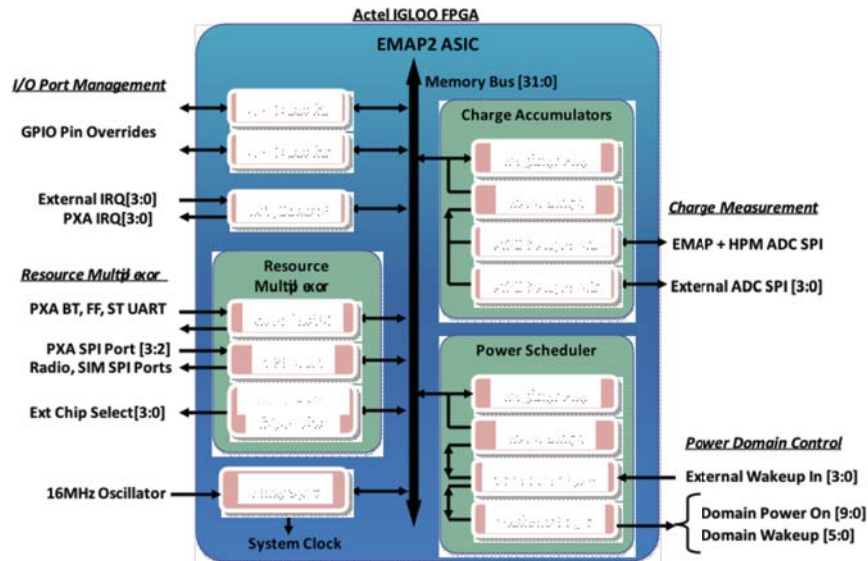


Figure 2: The RT619 micropower FPGA block diagram

(processor module) which runs a custom Linux kernel. The kernel integrates support for LEAP energy monitoring hardware and is able to accurately determine energy costs for most platform operations. The analog sampling engine may be configured and controlled via the Linux console through simple file system operations. This provides an easy to operate and robust platform for application development.

The RT-155 platform is a highly capable data acquisition system, capable of providing 6 channels of 24-bit data with sample rates up to 4 KHz. This is performed while maintaining average power dissipation less than 300mW. The platform is able to operate continuously in low power vigilant states while performing event detection. If events require, the platform can quickly transition to a fully vigilant mode where networking and additional platform resources can be enabled through the EMAP logic. The platform is able to continuously record data through these power mode transitions.

The LEAP enabled RT-155 is currently operating in field exercises with L-4 seismometers as well as acoustic microphones.

PRO 08 Time Synchronization—Take 2

PRO 08.1 Overview

The quality of time information at a node is affected by many factors: jitter in computational and communication latencies, time interval between resynchronizations, accuracy of time stamping network wireless packets, and quality of local clock source (quantization, frequency tolerance, aging, and drift). Much of the early work on time synchronization in sensor networks (including our own work at CENS) focused on the design of time synchronization protocols and regression mechanisms for coping with noisy time stamping of packets used during synchronization.

In this project, we re-evaluated two aspects of time synchronization – sources of error in time synchronization and how they affect synchronization accuracy and energy consumption, and post-facto synchronization using sensor data.

We investigate the fundamentals of clock drift and how drift due to changes in environmental temperature affects the overall synchronization accuracy. Using this knowledge, we developed a new synchronization protocol called Temperature Compensated Time Synchronization (TCTS). Another mechanism, called Virtual High Resolution Time (VHT) extends the duty-cycling principle often used in wireless communication to the clock subsystem of embedded devices. This allows power proportional time synchronization, where the power consumption scales with accuracy and the “read” and “write” operations of the timing subsystem.

We have developed a methodology called Data Driven Time Synchronization (DDTS) which provides post-facto time synchronization. DDTS works by using characteristics about the sensor data to provide time synchronization. We have used DDTS to recover incorrectly time synchronized data from our MASE deployment using microseisms. In the process we have made discoveries about how ocean effects such as wave height are directly correlated to microseism. We have also shown that there is potential to use DDTS for acoustic networks by applying similar data processing techniques we used for seismic DDTS.

PRO 08.2 Approach

Temperature Compensated Time Synchronization exploits the fact that almost every embedded microcontroller contains a temperature sensor. Thus, an embedded device can learn the clock characteristics with respect to changes in temperature by communicating with a master node that has access to an accurate time source. Using the communication capabilities, a node retrieves the current frequency error of the local clock source and measures the environmental temperature. Saving this “frequency error – temperature” pair in memory, the node can fall back on it later on when it re-encounters the same temperature. The node essentially becomes a temperature compensated clock since effects of temperature on the local clock get learnt over time and thus are eliminated.

One major problem in high-accuracy time synchronization in embedded systems is the need for high-frequency clock signals in order to achieve high-resolution time. A simple, though novel, approach to this problem is to use two clocks, one high frequency, one low frequency, to track time. During system sleep, only the low-frequency clock is on and tracks time. However, during active time, the high frequency clock is used to interpolate between the low frequency’s clock ticks, and thus keep high resolution time. The benefits of such a system is that during system sleep, no high frequency clock signal is using precious energy.

Our approach to developing DDTS was to use real data from the CENS seismic experiments and real data from deployments of the CENS Acoustic ENS Box. The ideal underlying characteristics to apply DDTS with are ones where the signal is not correlated to the features of the phenomena for which time synchronization is required and the characteristic should be either omnipresent or occur at regular intervals such that DDTS can be applied over the entire deployment.

There are two methods with which to apply DDTS and both depend on modeling the underlying characteristic in the data. The first method is to use an internal model. The internal model does not relate the sensor data to any other external factors, processes, or phenomena. The relation between sensors that the model builds is used to synchronize the data between stations. Typically the model is built around an effect that is in the signal at each sensor and can be related between sensors. Contrary to this, an external model for data-driven time synchronization

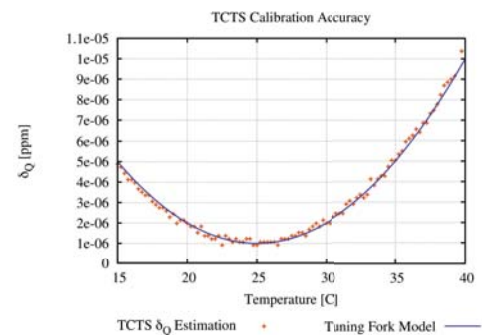


Figure 1.1 TCTS estimates the frequency drift function $g()$ for the current temperature, and stores it in a calibration table. In our simulation, $g()$ is a quadratic curve, as found in the tuning fork crystals.

will relate the data from each sensor to an external factor, process, or phenomena. In this case, the external factor, process, or phenomenon generates a signal that the sensor detects and understanding that factor, process, or phenomenon provides a means to synchronize the data.

PRO 08.3 System(s) Description and/or Experiments

We implemented TCTS in a simulator and on a real mote platform. The simulator was based on Castalia, which we enhanced with a clock model that incorporates changes in drift due to environmental temperatures. The hardware platform for the TCTS experiments was the regular TelosB mote using the MSP430 on-chip temperature sensor. This combination allows TCTS to be used by many researchers that are already using the TelosB motes in their deployments.

We implemented and extensively tested the concept of Virtual High-Resolution Time on the Epic sensor network platform. Additionally, we implemented a prototype VHT system on an FPGA and conceptually evaluated its feasibility.

We have applied DDTs to our seismic data from the Meso-American Subduction Experiment (MASE) using microseisms. A microseism is a type of seismic wave that is generated by the interference of oceanic surface waves. The interference creates enough pressure in the ocean floor to generate the seismic waves. Microseisms travel through the oceanic and continental crusts. The microseism period depends on the ocean depth and the oceanic surface wave period generated by the wind. Microseisms are omnipresent and exist in the 0.03 to 0.3 Hz frequency range, requiring the use of broadband seismometers. In our seismic data, the dominant period of the microseism energy traveling north through the array is 6 seconds. We have developed a model of the microseism propagation through the MASE array and can use that to recover the time synchronization.

We have also applied DDTs to acoustic data collected with the CENS Acoustic ENSBox. We used data from two deployments, one set from the rain forest in Chajul, Mexico the other from the Rocky Mountain Biological Laboratory (RMBL). We searched this data to determine if there was signal independent from the birds and marmots being studied with which we could apply DDTs.

PRO 08.4 Accomplishments

Over a 16 hour simulation where no resynchronization occurs, TCTS accumulates only 4ms of time synchronization error. This corresponds to a clock stability of <math><0.07\text{ppm}</math>. Figure 1.1 shows the calibration information TCTS collected during this experiment, and compares it to the actual clock model fed into the simulator itself. This plot shows that a time synchronization protocol can successfully temperature calibrate a local clock oscillator, and thus removes the factory calibration step necessary for a TCXO.

In evaluating the implementation we showed how a network of 5 VHT enhanced Epic nodes achieve an average synchronization accuracy of $0.125\ \mu\text{s}$ with a standard deviation of $0.645\ \mu\text{s}$ (Figure 1.2). Investigating the average power draw advantages of a VHT enabled node, we showed that the average radio power significantly overshadows the power of the VHT subsystem by a factor of 40 when duty cycling at 0.77%.

We applied DDTs to our seismic data and were able to detect that approximately 7% of the data collected by the network was incorrectly time synchronized. We were able to recover time synchronization and repair incorrectly time synchronized data to better than 0.2 seconds. We verified that this accuracy is at the desired limit by performing a sensitivity analysis for local earthquake localization. We also attempted to derive an external model for DDTs by using ocean data to estimate the amplitude and azimuth of the microseisms at each station. We were able to

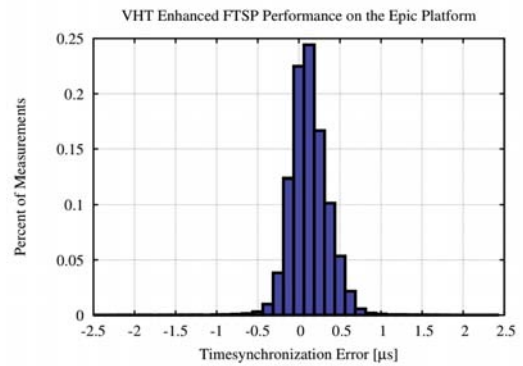


Figure 1.2 VHT synchronization accuracy histogram. The high-frequency 8 MHz virtual clock provides a maximal time-resolution of $0.125\ \mu\text{s}$. Using this clock, we achieved an average accuracy of $0.125\ \mu\text{s}$ (one tic) with a standard deviation of $0.645\ \mu\text{s}$.

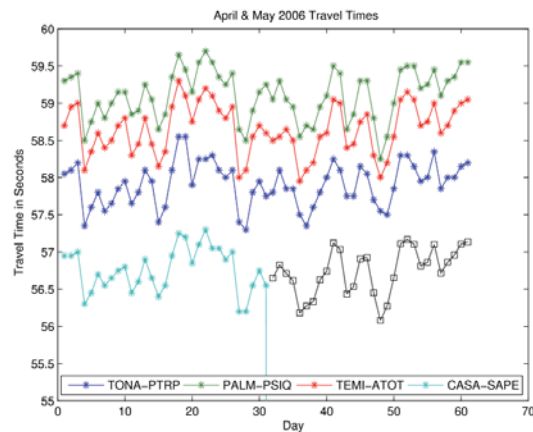


Figure 1.3 The travel time between 4 independent pairs of stations. The black line with squares represent the travel time repaired with DDTs. It correctly correlates with the other correctly time synchronized stations.

correlate ocean wave height data and developed an empirical model with which we could input the ocean wave data and determine the microseism travel time between stations to 0.3 seconds on average.

Applying DDTs to acoustic data, we determined that in both sets of data there existed underlying signals for which we could apply DDTs with. For the Chajul deployment, the signal came from a nearby water pump. For the RMBL deployment, the signal came from vehicles driving past the acoustic array on a nearby dirt road. With the RMBL data, using the same data processing techniques we applied to the seismic data, we were able to reconstruct the distances between stations to better than a meter and using these positions localize an artificial source to within 1 meter of its actual position. This translates to an average time synchronization error of 0.003. For acoustic deployments the acceptable time synchronization error is dependent on the desired accuracy of the source localization.

PRO 08.5 Future Directions

We plan to provide a standard implementation of TCTS and VHT for the TinyOS operating system. This will allow to easily port the two concepts to new microprocessor architectures, extending the usefulness of the concepts. Additionally, we want to apply the insights gained from our theoretic analysis of the impact of change in temperature on time synchronization accuracy on the root or master election algorithm of multi-hop time synchronization.

The future of seismic DDTs lies in applying it to more deployments, refining the models for microseism propagation, and developing new external models. For the acoustic DDTs we will develop experiments which introduce an artificial external signal to determine how accurately we can recover time in controlled environments. If successful this will enable future deployments to include an external signal and provide a backup method for both self localization and time synchronization.

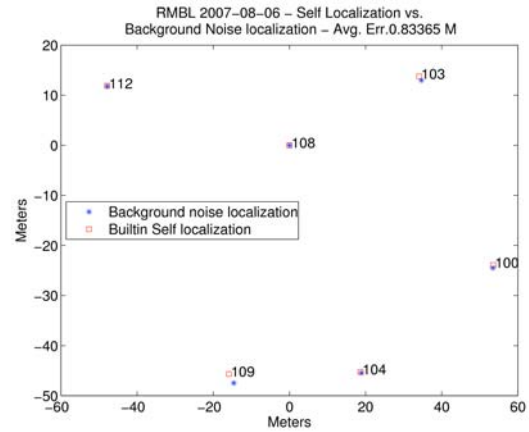


Figure 1.4 Self localization applied after we apply DDTs techniques to determine the ranges between stations. The self localization was off by less than a meter for each station.

PRO 09 Virgil: towards certified sensor nodes

PRO 09.1 Overview

A medical device should not crash or confuse. A device crash can be anything from inconvenient to life threatening, while confusing device behavior can lead a user to draw an incorrect medical conclusion. We envision a certification tool that can meet challenges

related to space bounds, soft-real-time response, life time, and meaningful results. We aim for both fundamental advances in programming language design and static error checking, as well as progress on how to do applications programming for medical monitoring devices. Our goal is to take a major step towards design for certifiability and to bring closer the day when the FDA will use static error checking tools frequently and routinely. A medical monitoring device collects data from the body, carries out local computation, and sends data to an external computer. Together, the device and the external computer form a small sensor network with a few sensors and one base station. We have an NSF-funded collaboration with Majid Sarrafzadeh at UCLA whose group has built software for four monitoring devices that were then tested by doctors and patients in the UCLA Medical School. Majid's devices monitor such things as pressure changes in the upper urinary tract, myotatic stretch reflex, neurological disorder, and diabetic foot ulcer (see Figure 1). The devices are small and the software is typically on the order of a few thousand lines of source code. Eventually we want to be able to certify the software in Majid's four devices.

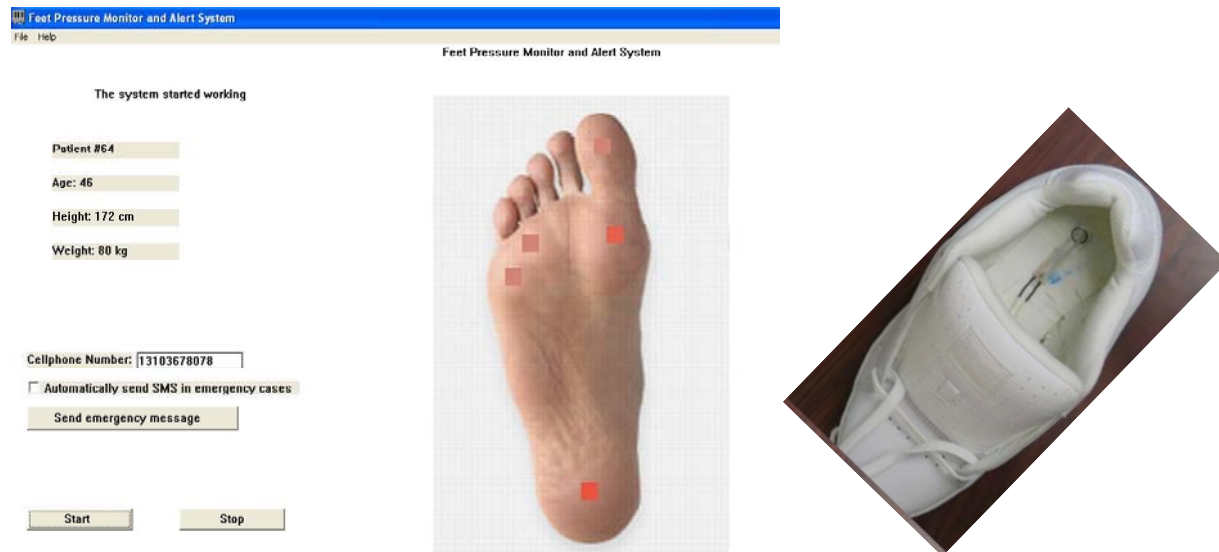


Figure 1. A device for monitoring diabetic foot ulcer.

PRO 09.2 Approach

Our goal is to develop

- a domain-specific language that will encourage design for certifiability and make certification easier, and
- domain-specific tools for certifying the four key properties of space bounds, soft-real-time response, life time, and meaningful results.

In particular, we want the tools to do static error checking, that is, certify the software without running the software. The certification tools will guarantee that certain problems cannot occur; and rigorous testing can then focus on problems that were left unaddressed by the certification tools. The certification tools will increase our confidence in medical devices and help decrease the scope, duration, and cost of the testing effort.

PRO 09.3 System Description and Experiments

Our own language Virgil is our starting point for designing a new domain-specific language. Virgil is a statically-typed, object-oriented language in the tradition of C++, Java, and C#. Virgil is designed specifically for high-level, type-safe systems programming, including programming of device drivers for sensors, radios, timers, analog-to-digital converters, etc., and is targeted to run on tiny devices such as sensor nodes.

PRO 09.4 Accomplishments

We have written drivers in Virgil for all the Mica2 devices. The final driver that we completed was the radio device driver which unsurprisingly turned out to be a major challenge. As a result, we now have the entire base functionality of TinyOS written in Virgil. Software can now control a sensor node using Virgil code alone. Additionally, we have rewritten one of our medical device software applications into Virgil. This has provided us with an excellent benchmark that we use in our day-to-day research. We have also studied approaches to extending Virgil from being a language for programming single nodes to become a language for programming an entire network of sensor nodes. We are focusing on X10, an object-oriented language from IBM. The X10 language contains two key constructs for concurrent programming called `async` and `finish` that we believe can be valuable for programming an entire sensor network.

We have made major progress on verification of Virgil programs. We have built a tool that maps a Virgil program to a timed automaton. We then use the UPPAAL real-time model checker to check properties of the timed automaton. Our timed automaton represents all program variables via transitions in the automaton. We represent timings of most operations, except machine level timings of such things as stack manipulation, heap allocation, etc. We don't support recursion and currently we handle procedure calls by duplicating the subautomaton for the callee.

Our benchmark Virgil program has 1094 lines of code, and the derived timed automaton has 628 states.

We have had great success with checking low-level properties

that are usually difficult to reason about, including:

- Atomic access to the ADC converter? We can catch nonatomic accesses, that is, make sure that consecutive calls to the ADC converter don't overlap. If there is access to the ADC converter when the ADC converter is active, then the automaton will move to an error state. The model checker verifies that there is no circumstance under which that could happen. If we change the configuration of the hardware timer such that it starts the ADC converter too often, then we will catch the error.
- Might we drop a sensor reading? In other words, is the timer period for controlling input from the sensor too short? We can catch such problems.

We have also made progress on checking stack overflow and on determining the worst-case time to execute the ADC converter from beginning to end.

PRO 09.5 Future Directions

We will continue our work on porting the software in Majid's four devices from NesC to Virgil.

We have started work on directed testing for Virgil programs that will enable us to find worst-case event sequences that stress test an application.