Research article

GRAIL: a general purpose localization system

Yingying Chen
Department of ECE, Stevens Institute of Technology, Hoboken, New Jersey, USA

Gayathri Chandrasekaran
Department of Computer Science, Rutgers University, Piscataway, New Jersey, USA

Eiman Elnahrawy
Kordinate LLC, Metuchen, New Jersey, USA

John-Austen Francisco and Konstantinos Kleisouris
Department of Computer Science, Rutgers University, Piscataway, New Jersey, USA

Xiaoyan Li
Department of Computer Science, Lafayette College, Easton, USA, and

Richard P. Martin, Robert S. Moore and Begumhan Turgut
Department of Computer Science, Rutgers University, Piscataway, New Jersey, USA

Abstract
Purpose – The purpose of this paper is to describe a general purpose localization system, GRAIL. GRAIL provides real-time, adaptable, indoor localization for wireless devices.
Design/methodology/approach – In order to localize as diverse a set of devices as possible, GRAIL utilizes a centralized, anchor-based approach. GRAIL defines an abstract data model for various system components to support different physical modalities. The scalable architecture of GRAIL provides maximum flexibility to integrate various localization algorithms.
Findings – The authors show through real deployments that GRAIL functions over a variety of physical modalities, networks, and algorithms. Further, the authors found that a centralized solution has critical advantages over distributed implementations for handling privacy concerns.
Originality/value – A key contribution of this system is its universal approach: it can integrate different hardware and software capabilities within a single localization framework. The deployment of such a system in academic and research environments allows researchers to explore issues beyond algorithms and investigate effects in real deployments.

Keywords Wireless, Localization, Sensors, Local area networks

Paper type Research paper

1. Introduction

Technology trends have reduced the cost of wireless networking to the point where it can be added to nearly every computing device. Indeed, wireless networking devices include laptops, PDAs, small sensors, active RFID tags, and even cameras and printers. The inclusion of wireless networking in such a broad range of devices opens an opportunity for a new computing service: positioning devices in physical space. A generic service of this kind would enable a host of additional applications, ranging from such diverse areas as asset management, disaster recovery, inventory tracking, geometry-based routing, and perimeter-based security. Using the same wireless traffic for both communication and positioning would provide tremendous cost and deployment savings over an independent localization infrastructure.

In this paper, we present the GRAIL system, which has the explicit goal of providing location information about any wireless transmitter (http://grailrtls.sourceforge.net). The field of localization covers a broad range of topics, from lower-layer physical problems to high-level application services. Within this extensive area, GRAIL focuses on performing general purpose localization in building-sized environments. The GRAIL system is designed to have the following properties:

- *General purpose*. A primary goal of the GRAIL system is that it should work over a variety of physical modalities, networks, and algorithms. Just as networking systems support multiple media access layers, routing protocols, and applications, a general purpose localization system should support multiple physical modalities and localization methods in order to support a diverse set of applications. GRAIL is designed to localize using any

The current issue and full text archive of this journal is available at
www.emeraldinsight.com/0260-2288.htm

Sensor Review
28/2 (2008) 115–124
© Emerald Group Publishing Limited [ISSN 0260-2288]
[DOI 10.1108/02602280810856679]

wireless network that supports physical layer measurements of packet data. It currently supports the use of received signal strength (RSS) as the physical modality and can easily be extended to support angle-of-arrival (AOA), and time-of-arrival (TOA).

- *Real-time*. Latency is a key property of localization systems because it defines the maximum mobility that can be supported. Our system can return results in less than 1 s, allowing us to support both stationary devices as well as those moving at walking speeds (about 1 m/s).
- *Adaptable*. Indoor environments are especially challenging due to reflections, refractions, and scattering, which result in substantial multi-path effects. GRAIL is designed to manage the uncertainty of these effects by expanding or contracting the confidence in the position estimate in accordance with the quality of the underlying observations. In addition, GRAIL supports dynamic feedback in the position estimates in order to dynamically scale the position uncertainty to the quality of recent observations.
- *Indoor*. GRAIL is designed to scale to the set of indoor environments controlled by a single organization; e.g. a campus. Integration of the location information with outdoor systems, such as GPS, is the responsibility of higher layers.

The rest of this paper is organized as follows: we first provide a system overview of GRAIL in Section 2. Then, in Sections 3, 4, 5, 6 and 7, we describe each of the system components in GRAIL, including landmarks, the server, solvers, the database, and the web server. Further, we present our experiences in deploying GRAIL in Section 8. In Section 9, we place our work in the context of the broader localization literature. Specifically, we compare our work with other existing localization systems. Finally, we conclude our work in Section 10.

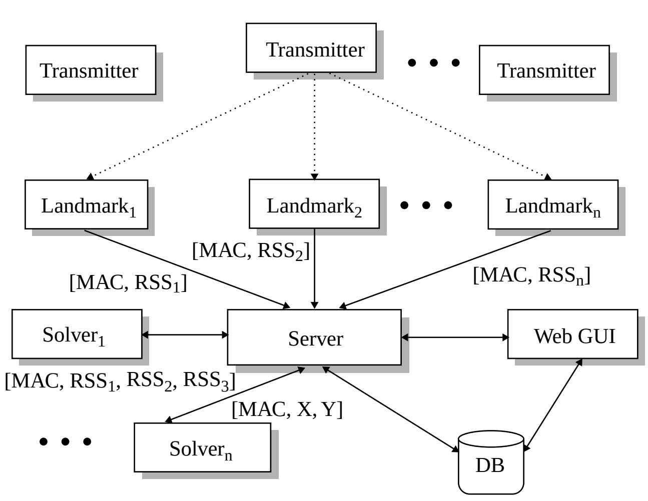
2. Overview of the GRAIL system

2.1 System architecture

Figure 1 shows the architecture of the GRAIL system. The main system components are transmitters, landmarks, the server, solvers, the database, and the web server. In the figure, each square is realized as a separate process. Note that in this work, we use the OSI/IP model when referring to communication layers.

In order to localize as diverse a set of devices as possible, GRAIL utilizes a centralized, anchor-based approach.

Figure 1 Grail system architecture



The anchor-based strategy removes the need to install special-purpose software on the devices to be localized. It also allows users to rapidly incorporate new devices. Specifically, we have used the GRAIL system to simultaneously localize multiple devices running 802.11 (WiFi, http://standards.ieee.org/getieee802/802.11.html), 802.15.4 (ZigBee, http://standards.ieee.org/getieee802/download/802.15.4-2003.pdf) and Roll-Call (http://inpointsys.com) protocols, with little to no software modifications. We compare and contrast our strategy to others, (e.g. hop-by-hop, Niculescu and Nath, 2003) in Section 9.

In this approach, special nodes called landmarks are deployed at known locations. They continuously monitor the channels' traffic at the packet-level and forward their observations to a central entity, the server. The localization process begins once the transmitters send packets. Landmarks are sometimes called anchor points or access points. In GRAIL, however, we use the term landmark, because access points provide access to the wired network while in GRAIL landmarks need only observe radio packets.

The next step in the localization process is for the landmarks to extract the time when the packet was received, RSS, and possibly AOA, and send them to the server along with other packet header information.

Upon receiving the RSS readings and timestamps from each landmark, the server collects the complete set of RSS readings for each transmitter, cleans up the data samples, decides on which localization algorithm to use, and then forwards the information to an instance of the solver. The solver estimates the locations and sends them back to the server, which it then stores in the database. Multiple localization algorithms can be plugged into the solver.

2.2 Key abstractions

The GRAIL system uses an abstract data model of various system components. Understanding this model is critical for understanding the localization process. GRAIL divides the model into two levels:

2.2.1 Low-level

The low-level abstractions describe raw packet information. This kind of data includes packet arrival time, signal strength, and AOA. These abstractions, including transmitters, timestamps, RSS values, AOA information, and samples are utilized by the system for higher-level processing. For instance, a transmitter is a network device that transmits radio frequency packets, and is identified by its layer-2 or layer-3 address. The timestamp represents the time at which a packet is observed. The RSS is the RSS reading embedded in a packet, while AOA is for the reported angle. A sample aggregates a group of these low-level observations, including a timestamp, RSS, and potentially AOA, for some transmitter.

The landmarks are responsible for transforming the modality- and network-dependent data into these low-level abstractions. For example, a landmark would have to read a packet header and extract the TOA and RSS of the packet. The landmark then forwards samples, built from these low-level abstractions, to the server.

2.2.2 High-level

The GRAIL system further models higher-level objects built from the low-level abstractions. These are regions, fingerprints, training data, testing data, and locations. These are realized as objects in the server.

A region object represents the area of interest for performing localization, and can be a section of, or a floor inside, a building. A fingerprint is a summary of a set of data samples from a suite of landmarks. Typically, this is a vector of averaged data samples from the landmark set, or the median of the same data. The data samples can be RSS readings or AOA. In GRAIL, a fingerprint vector always carries its set of associated landmarks.

Building on fingerprints are the concepts of training data and testing data. Training data consists of fingerprints collected at known locations, that is, the coordinate positions are known. Testing data, on the other hand, are fingerprints from devices with unknown positions. These terms have their roots in machine learning terminology. Many solver algorithms use the training data to extract parameters which are then applied to the testing data to infer the unknown positions.

Finally, in the GRAIL system the representation of a location is flexible and can be a single point, an ellipse, or a set of tiles. Supporting both point- and area-based representations of a location (Elnahrawy *et al.*, 2004) provides the user with the flexibility to trade accuracy for precision depending on an application's requirements.

Splitting the data model into two levels helps to separate the physical modality-dependent part from the high-level objects and makes the localization framework more generic. When incorporating different physical modalities such as RSS, AOA and TOA, only the low-level abstraction portion needs to be augmented. Our database design captures this data abstraction model, which is described in Section 6.

2.3 System components

We now give a brief overview of the system components and their functionality within the GRAIL system. A more detailed description of each system component is provided in subsequent sections and also in Chen *et al.* (2007a, b):

- *Transmitter.* Any device that transmits radio packets, called a transmitter, can be positioned, including laptops, palmtop devices, WiFi-enabled cell phones, and active RFID devices. Packet-level localization is very important because it allows us to localize using existing traffic. This gives GRAIL tremendous advantages, in both generality and ease of use, over other approaches. For example, often the application code does not need to be altered on a sensor node in order to localize it. Most devices also support a polling functionality, (e.g. ICMP ping), which we have successfully used in many contexts to localize devices that are not actively generating traffic. GRAIL also supports a special type of transmitter, called a stationary transmitter, which has a known and fixed position (e.g. wireless desktops and access points). Stationary transmitters can be used to generate online training data. The goal of the GRAIL system is to localize any transmitters.
- *Landmark.* Landmarks passively monitor the existing network traffic and listen to incoming packets. The current landmarks support RSS measurement, and landmarks with an AOA capability are under development. Landmarks can forward raw data samples to the server or they can send summarized information about multiple packets, reducing network bandwidth requirements. By limiting the computational requirements of the landmarks, they can be deployed on low-cost networking devices (e.g. inexpensive wireless routers).
- *Server.* A localization server is a centralized moderator that collects data samples from the landmarks, summarizes and cleans the observed information, and then passes the data to solvers to compute the positions of all the devices that sent packets over a period of time. The server interacts with our web server to accept user requests for localization at a specified time and reports the location estimations back to the web server. The server stores all the related packet traffic information and localization results in the database based on our information abstraction model.
- *Solver.* Solvers take inputs from the localization server, compute the location estimation using server specified localization algorithms, and return localization results back to the server. GRAIL's architecture is flexible with regards to the separation of server and solver; multiple solvers can run simultaneously against a single server. This allows the system to run multiple localization algorithms, as well as perform load balancing between solvers.
- *Database*. The database schema is based on our information abstraction model. It interacts with the localization server and is a repository for storing the hard state of the GRAIL system. It contains the localization results and fingerprints computed from data samples, which can be used for further offline data analysis and summary, as well as maintaining information about landmarks, regions, and transmitters that have been used for previous location estimations.
- *Web server.* The web server provides a front-end to the GRAIL system, and is used as the main point of interaction between the system and higher-layer applications, including the current web-based GUI. The web server enables authentication mechanisms and a single point of interaction between the GRAIL system and users or applications. A set of APIs are provided for higher-layer applications built on top of the GRAIL system. The current web-based GUI allows users to view previous localization results stored in the database, adjust server and system settings, and request manual localization of devices.

Much like the Hypertext Transfer Protocol (HTTP) used on the web and the Transaction Language 1 used as a standard protocol in the telecommunications equipment industry, all communications between the system components use a simple text-based protocol over TCP sockets. Although this is not an optimal use of network bandwidth, the resulting compatibility gains are substantial. Using a binary or XML-based protocol would have forced us to develop additional software components, with little to no benefits, whereas the current system is simple, extensible, and facilitates interchangeable components.

3. Landmark

The primary job of a landmark is to convert network- and device-specific observations into the low-level abstractions. That is, landmarks passively observe the radio channel and report information, such as RSS, TOA, AOA, and packet headers to the server, which has knowledge of the landmark's location, for aggregation and processing. GRAIL assumes the transmitter of a packet is identifiable via a unique ID. The current system uses layer-2 addresses as a unique identifier. Landmarks are stateless, that is, landmarks do not hold any permanent state needed for localization. Keeping the landmarks stateless greatly simplifies their design.

Typically, a landmark is implemented using off-the-shelf radio technologies that are placed in "monitor" or "scanning" modes. These modes often forgo processing the layer-2 protocols, and simply pass all packets on a given channel to the higher layer, along with metadata such as RSS and TOA. More sophisticated hardware configurations can simultaneously pass copies of the packets to the landmark and act as a regular layer-2 device, although these devices are the exception rather than the rule. If simultaneous monitoring and protocol operations are possible, the landmark can act as both a landmark and a wireless access point.

A landmark can also support multiple radio technologies. Figure 2 shows a picture of a landmark that can simultaneously capture 802.11, ZigBee and Roll-Call packets. The landmark software is divided into a device dependent and device independent layers. The device-dependent layer handles sample creation and the independent layer manages the samples.

A single-board 133 MHz 486 computer running Linux hosts the different radio technologies, demonstrating that the landmarks can operate effectively with little processing power compared to the other system components. Landmarks can support different physical layers and report to which physical layer each network ID belongs. Landmarks currently support two methods of reporting the observed data back to the server: per-packet or aggregated:

1. Per-packet samples are used when the landmark has a higher-bandwidth connection to the server than the wireless channels from which it is collecting data. For example, an 802.11 listener with a 100 Mbps wired connection, or a Roll-Call listener with an 11 Mbps 802.11b connection to the server can send per-packet samples. Using this method of reporting, a sample is a packet summary, e.g. the network ID and RSS of a packet, followed by the first N bytes (typically 256) of the packet header.
2. Aggregate samples send a summary of a series of packets for each network transmitter. The current protocol enables landmarks to send per-layer-2 ID histograms of RSS values measured over the last k s. That is, the landmark sends the set of value/count pairs it received over the last k s.

Figure 2 A GRAIL landmark that simultaneously monitors packets from WiFi, ZigBee, and Roll-Call

In both cases, the landmarks need to perform some interpretation of the packet headers. As a minimum, the landmark must extract the layer-2 address of the transmitter and the RSS value from the packet header. Additional information is optional, but can be utilized by the server if available.

4. Server

4.1 Characteristics

The server acts as a central point of coordination in the GRAIL system. It interacts with all the other system components, and its main characteristics include:

- *Soft state.* The server maintains a soft state, keeping only the most recently-collected data internally, and storing the rest of the data, including landmark information, localization results, and transmitter details, in the database. This allows the server to restart without adversely affecting the localization system. After a server restart, landmarks and solvers automatically reconnect, and landmarks resume sending samples. In order to support localization on past data, the server sends fingerprints to the database, where they can be recalled later to enable further data analysis. Raw packet headers can also be stored, although we do not use this feature for normal localization.
- *Authentication.* The current implementation provides simple authentication using named accounts and clear-text passwords. Landmarks and solvers must be programmed with the correct usernames and passwords in order to connect to the server. Further access restrictions can be implemented using IP address filtering to allow only the web server, landmarks, and solvers to connect to the server. Given that GRAIL's external interface is through the web server, and the server is behind the organization's firewall to the external world, we view this level of security as sufficient.
- *Supporting data abstraction.* The server is centered around the high-level data abstractions including regions, landmarks, transmitters, samples, fingerprints, training data, testing data, and locations. The server manages multiple regions, each region being independent of the others, though not necessarily disjoint, and potentially representing the same physical location. Each region keeps track of the landmarks and transmitters located within its boundaries. Within the server, transmitters are identified by their layer-2 addresses, and as samples are collected by landmarks for each transmitter, fingerprints can be generated for use in localization.
- *Decoupled operation.* Decoupling the server and solvers, i.e. having them run as separate processes or on separate hosts, has several advantages. It allows seamless integration of any new solver implementations into the GRAIL framework. The server can launch multiple solver instances under periods of high load, for example if it needs to simultaneously localize hundreds of transmitters. Decoupling also makes it easier to scale the hardware and software resources dedicated to running the localization algorithm. For example, a parallel machine could be

dedicated to running a parallelized solver, leaving a machine with a slower processor to handle the coordination tasks of the server.

The disadvantage of the decoupled approach is that it increases localization latency because messages must be exchanged between processes, or possibly even over a network, in order to localize. However, we have found that the advantages of decoupling the server and solvers outweigh the cost of increased latency:

4.2 Privacy mechanisms

The centralized server entity makes enforcing contracts and privacy policies more tractable. Currently, the only privacy mechanisms that are supported in GRAIL are positive and negative filters on network addresses. Positive filters describe network transmitters which are allowed to be localized, while negative filters describe transmitters to be discarded. The filters can be applied in two places:

1 when data arrives from a landmark; and
2 just before data is sent to a solver.

Filtering in the latter approach allows the system to retain observations for network devices, even if immediate localization of these devices is not desired.

4.3 Localization process

The server controls when localization occurs, and it can be manual or automatic. A manual localization is performed at the request of a user through the web-based GUI on the web server. Automatic localization can be further refined into three subcategories: time-based, fingerprint-based, and sample-based.

Time-based localization is the most common trigger. It takes place when a timer with a predefined wait period expires, and can ensure that a localization takes place every n s. Fingerprint-based localization takes place whenever the computed fingerprints for a transmitter exceed a predefined threshold, and allows the server to localize moving transmitters more often than stationary devices. Further, a sample-based localization occurs when a predefined number of samples is received for a transmitter. This enables the server to localize a transmitter after every k samples have been received, and allows the system to track transmitters generating more network traffic more closely than those that generate less traffic.

5. Solver

5.1 Supporting different algorithms

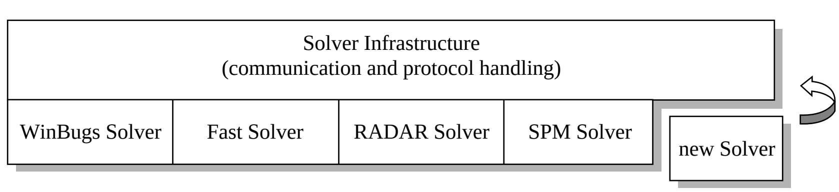
The major role of the solver component is to perform position estimation utilizing information sent from the server. The solver is flexible, scales easily, and can support multiple localization algorithms. The solver has two layers as shown in Figure 3. The top layer interfaces with the server and handles communication and protocol-related information. The bottom layer is where the different localization algorithms are plugged in, and is independent of the communication protocols and the external interface.

The current GRAIL system supports two solvers that utilize Bayesian networks (BNs) for localization: the WinBugs Solver and the Fast Solver. Our BNs have no closed-form solution and, thus, we turned to Markov Chain Monte Carlo (MCMC) simulation to solve them. This family of approaches uses statistical sampling to explore the probability density functions (PDFs) of the variables in the networks with unknown value. The WinBugs Solver uses the statistical WinBugs tool (www.mrc-bsu.cam.ac.uk/bugs/). WinBugs is a general purpose comprehensive Bayesian analysis tool for solving complex statistical models using MCMC methods. It usually encounters high-computational cost, since it does not account for any special properties of wireless networks.

As a result, we were motivated to develop our own customized Bayesian solvers, the Fast Solver (Kleisouris and Martin, 2006). The MCMC method we used was Gibbs sampling. We found that the Gibbs sampling method used to estimate the X and Y coordinates of a device to be localized dominates the execution time of our solvers. Thus, we implemented a variation of this method that takes advantage of the observation that the PDFs of X and Y in the BNs are relatively flat. This variation, which we call whole domain sampling, samples uniformly over the whole domain of X and Y, as opposed to carefully choosing only parts of the domain to sample from. We found whole domain sampling is computationally efficient and provides fast convergence to the values of X and Y that we want to estimate. In addition, the method requires no tuning, making it an attractive approach, since it constitutes a black-box sampler for our networks. This approach achieves computational efficiency and simultaneously provides good ratio of accuracy vs time. The Fast Solver uses a novel real-time sampling technique which reduces the computational cost significantly and solves BNs 9-17 times faster than the WinBugs Solver. It localizes 1-10 transmitters in less than half-a-second, and scales to 50 transmitters simultaneously, with no location information in the training data, within 6 s.

In addition to BNs, the current GRAIL system supports two additional solvers: a version of simple point matching (SPM) (Elnahrawy *et al.*, 2004) and the K Nearest Neighbor in Signal Space, also known as RADAR (Bahl and Padmanabhan, 2000). The inclusion of multiple solvers demonstrates GRAIL's flexibility. As shown in Figure 3, WinBugs, Fast, RADAR, SPM, and ABP solvers can coexist within the solver system component. During real time localization, different localization algorithms can be used by the server, either specified in each user request, or from an internal default setting.

Figure 3 Layered architecture in solver



Sensor Review
Volume 28 · Number 2 · 2008 · 115–124

5.2 Bayesian modeling

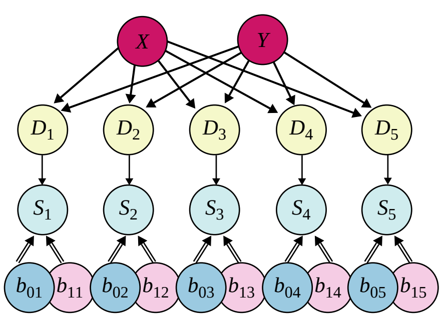
The BN is a graphical model that encodes dependencies and relationships among a set of random variables. The vertices of the graph correspond to the random variables and the edges represent dependencies. Bayesian inference in conjunction with BNs offers an efficient and principal approach for avoiding the over-fitting of data. In BN, the overall joint density of *x*?X, where *x* is a random variable, only depends on the parents of *x*, denoted pa(x):

$$p(X) = \prod_{x \in X} p(x|\mathrm{pa}(x))$$

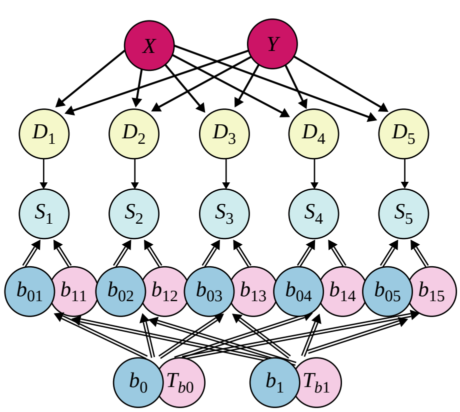
Once $p(X)$ is computed, the marginal distribtution of any subset of the variables of the network can be obtained as it is proportional to overall joint distribution.

In the GRAIL system, we developed several Bayesian graphical models to encode therelationship between the RSS and the location based on signal-to-distance propagation model. We have built both non-hierarchical (M1), as well as hierarchical (M2) Bayesian graphical models as shown in Figure 4(a) and (b). We use five landmarks as an example in Figure 4. The vertices X and Y represent location of a transmitter; the vertex S_i is the RSS reading from the ith landmark; and the vertex D_i represents the Euclidean distance between the location specified by X and Y, and the ith landmark. The value of S_i follows a signal propagation model $S_i = b_{i0} + b_{i1} \log D_i$, where b_{i0}, b_{i1} are the parameters specific to the ith landmark. The distance:

Figure 4 BNs used in the solver component



(a) Non-hierarchical Bayesian graphical model



(b) Hierarchical Bayesian graphical model

$$D_i = \sqrt{(X - x_i)^2 + (Y - y_i)^2}$$

in turn depends on the location (X, Y) of the measured signal and the coordinates (x_i, y_i) of the ith landmark. The network models noise and outliers by modeling the S_i as a Gaussian distribution around the above propagation model. The Bayesian graphical model assumes that X and Y are marginally independent with the following distribution for each vertex in M1:

$$X \sim \mathrm{uniform}(0, \mathrm{length}),$$
$$Y \sim \mathrm{uniform}(0, \mathrm{width}),$$
$$S_i \sim N(b_{i0} + b_{i1} \log D_i, \tau), \quad i = 1, 2, 3, 4, 5,$$
$$b_{i0} \sim N(0, 0.001), \quad i = 1, 2, 3, 4, 5,$$
$$b_{i1} \sim N(0, 0.001), \quad i = 1, 2, 3, 4, 5,$$

with length and width denotes the dimension of the indoor environment. The transmitter that GRAIL is localizing is within this indoor environment. Further, we seek to borrow strength across different regression models; and the coefficients of the linear regression models corresponding to each of the landmarks should be similar since the similar physical processes are in play at each landmark. We developed the hierarchical model M2 with the following conditional densities:

$$X \sim \mathrm{uniform}(0, \mathrm{length}),$$
$$Y \sim \mathrm{uniform}(0, \mathrm{width}),$$
$$S_i \sim N(b_{i0} + b_{i1} \log D_i, \tau), \quad i = 1, 2, 3, 4, 5,$$
$$b_{i0} \sim N(b_0, \tau_{b0}), \quad i = 1, 2, 3, 4, 5,$$
$$b_{i1} \sim N(b_1, \tau_{b1}), \quad i = 1, 2, 3, 4, 5,$$
$$b_0 \sim N(0, 0.001),$$
$$b_1 \sim N(0, 0.001),$$
$$\tau_{b0} \sim \mathrm{gamma}(0.001, 0.001),$$
$$\tau_{b1} \sim \mathrm{gamma}(0.001, 0.001).$$

The initial parameters (b_{i0}, b_{i1}, b_0, b_1, τ_{b0}, τ_{b1}) of the model are unknown, and the training data is used to adjust the specific parameters of the model according to the relationships encoded in the network. Through MCMC simulation, BN returns the sampling distribution of the possible location of X and Y as the localization result.

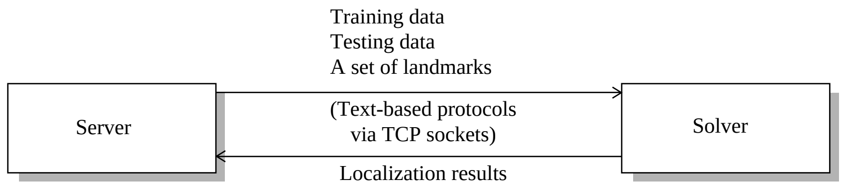
Both M1 and M2 need to use the training data to predict the parameters in BNs and inference the (X, Y) location of the transmitter. Moreover, we introduced a special model (Madigan *et al.*, 2005a) which provides a zero-profiling approach for location estimation and aims to significantly reduce the labor-intensive training step.

5.3 Communication interface

The server and solvers communicate using a simple text-based protocol over TCP sockets. In order to perform localization, the server needs to send the following information to the solver, as shown in Figure 5:

- *Landmark positions*. Positions of a set of landmarks in the region of interest.

Figure 5 Communication between the server and a solver



- *Training data.* The training data consists of the fingerprints collected at known locations along with their coordinates, similar to a signal map of the region. The training data is either collected off-line, or generated from stationary transmitters, or constructed from previous localization results. It can be retrieved from the database or from files stored on the server. This way both online algorithms and those that require off-line site surveys are supported.
- *Testing data.* The testing data is a list of fingerprints for the unknown wireless devices that need to be localized, marked by their layer-2 addresses. The list can include all observed transmitters or only a subset.
- *Requested return type.* The localization request can specify one of the following types of results: points, ellipses, or tiles. If a basic localization algorithm only supports areas (e.g. ellipses or tiles), it is the solver's responsibility to transform these results into a simpler format if requested. For example, selecting the tile with median *X*- and *Y*-coordinate is a simple method of converting a tile-based result to a point-based one. This is consistent with the principle of having most of the algorithmic computation in the solver and using the server only for coordination and control.

6. Database

The database holds the hard state of the system, and is in charge of maintaining all data that must survive a crash and recovery. Specifically, it stores location information for transmitters, the fingerprints used to generate that location information, and the landmarks associated with each RSS value of a fingerprint. Our implementation uses the postgreSQL database.

6.1 Schema description

Table I presents the database schema that implements our abstract data model. The table columns follow naturally from the abstractions described in Section 2.

The regions table stores the localization regions defined by the system. A region's unique identifier and human-readable name are stored in the region-id and region-name columns, respectively. The start-stamp and end-stamp columns records the lifetime of a localization region.

Table I GRAIL database schema; primary keys are underlined

Table name	Schema
Region	(region-id, region-name, start-stamp, end-stamp)
Landmarks	(device-id, name, region-id, *x*, *y*, *z*, start-stamp, end-stamp)
Transmitters	(device-id, type, description)
Packets	(device-id, t-stamp-msec, rssi, channel, protocol, deviceType, header)
Histograms	(device-id, start-stamp, end-stamp, rssi, count)
Fingerprints	(device-id, lm-id, region-id, start-stamp, end-stamp, rssi)
Loc-ellipses	(device-id, region-id, start-stamp, end-stamp, *x*, *y*, *z*, plusminusx, plusminusy, plusminusz)
Loc-tiles	(device-id, region-id, start-stamp, end-stamp, cellsize, cells)

The landmarks table stores the landmarks used by the system, and the regions in which they are defined. A landmark's unique identifier (e.g. layer-2 address), description of its position in physical space, defining region, and *X*-, *Y*-, and *Z*-coordinates within the region are stored in the device-id, name, region-id, *x*, *y*, and *z* columns, respectively. The start- and end-stamp columns records the time duration that a landmark resides at one position.

The transmitters table stores all the transmitters scanned in the environment. A transmitter's unique identifier (e.g. layer-2 address), type (e.g. stationary or mobile), and human-readable description are stored in the device-id, type, and description columns, respectively. The description column can be used to describe the position of the transmitter, or could contain information regarding any special characteristics that the device may have.

The packets table stores information about all packets forwarded by the landmarks while performing trace operations, and can be used for offline analysis. A packet is represented as a transmitter ID, timestamp, observed RSS value, radio frequency channel, link-layer protocol, device type (e.g. 802.11), and the first *n* bytes of the packet header. These values are stored in the device-id, *t*-stamp-msec, rssi, channel, protocol, deviceType, and header columns, respectively.

Similar to the packets table, the histograms table stores information about aggregated packet data, as described in Section 3. Histograms are represented by a transmitter ID, beginning and ending timestamps, RSSI value, and the number of packets observed with that RSSI value, and are stored in the device-id, start-stamp, end-stamp, rssi, and count columns, respectively.

The fingerprints table stores the computed RSS samples, in the rssi column, for each transmitter-landmark pair, identified by the device- and lm-id, respectively. The start- and end-stamp columns denote the timestamps of the oldest and youngest packets used to generate the fingerprint, respectively.

The loc-ellipses and loc-tiles tables are used for storing ellipse- and tile-based locations, respectively. A location is linked to a network transmitter within a region by the device- and region-id columnns, respectively. For ellipse-based locations, the center of the ellipse is stored as a 3-tuple of the *X*-, *Y*-, and *Z*-coordinates in the *x*, *y*, and *z* columns, and the columns plusminusx, plusminusy, and plusminusz represent the uncertainty range for each dimension. Points are represented as ellipses with uncertainty values of 0. A tile-based location is represented by the length of the side of a tile, and a list of tiles containing the possible position of the device, which are stored in the cellsize and cells columns, respectively.

6.2 Retrieving tracks

The database and the web server components are directly linked together, as shown in Figure 1 which allows the web server to issue simple SQL statements in order to retrieve location data for presentation to the user. Displaying device tracks, the estimated locations of a device over some finite range of time, is easily performed by making simple SQL queries on the database's loc-ellipses and loc-tiles tables.

It is important to note that a wide range of more complex queries can be made on the current localization database in order to extract additional information. For example, a network administrator can compose queries to monitor the coverage map of landmarks on a floor or locate a troublesome device that is utilizing a high percentage of network bandwidth. This would involve only a few extra queries on the fingerprints and landmarks table. In a similar fashion, a whole suite of applications could be built on top of the core tracking functions provided by the GRAIL system.

7. Web server

The Apache web server is used as GRAIL's external interface. Using a web interface has the key advantages that any client can connect to the service. Clients could be either human browsing web pages, or programs calling HTTP methods to extract the data. The programming interface outputs data as comma-separated value (CSV) files, while the human interface methods output both CSV and displays the results graphically.

Using a web server as the system interface also allows GRAIL to leverage the web's existing security and authentication methods, for example, directories and methods accessed only through secure HTTP. The current system provides a simple username/password structure to access the entire dataset. More sophisticated mappings of users to views of system data remains as future work.

The web-based GUI is a part of the web server and comprised of two parts: displaying location data and controlling the localization parameters. Figure 6 shows an example session using our GUI, where the left side of the screen depicts the configurable system parameters, and the right side is a graphical representation of a region, a research laboratory, being used for localization. The shadowed ellipses are the estimated positions of two transmitters.

The localization control interface allows users to specify localization parameters in terms of changes in time, RSS, or number of samples, and allows a user to initiate localization manually. The current interface does not allow the end-users to control the landmarks or the core server functionality. We conjecture this low-level administration and configuration is best left to the system administrators.

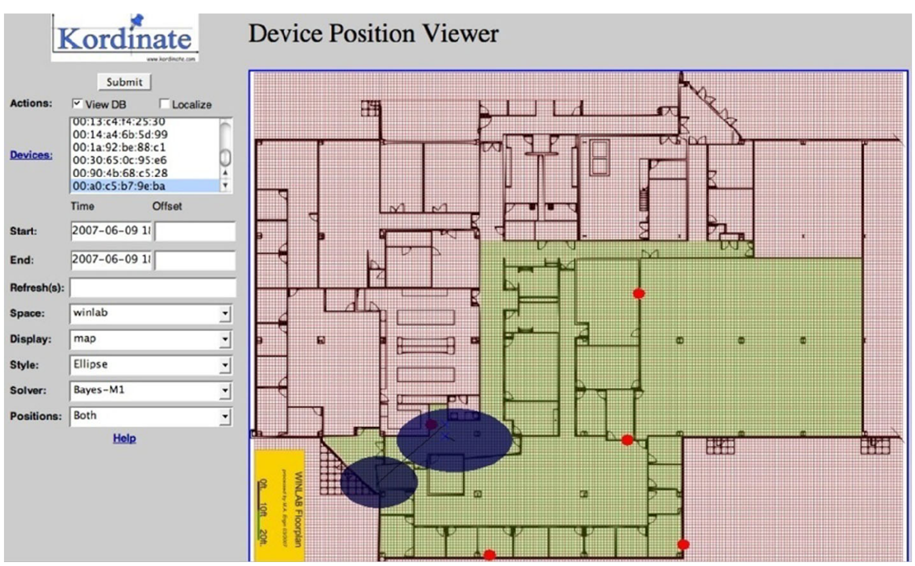
8. Deployment experience

We deployed the GRAIL system in a few office building environments including the Computer Science Department at Rutgers University, the Wireless Information Network Laboratory at Rutgers University, the Computer Science Department at Lafayette College, as well as in exhibit halls as part of conferences and trade shows. In these environments, the GRAIL system was used extensively to track various devices ranging from laptops, WiFi phones, ZigBee sensor nodes, and to Roll Call tags.

We found that the issues that concerned us the most were in the areas of security, privacy, localization performance, and state management:

- *Security.* Our initial assumption that all systems behind the web server would be in a "flat" IP address space was incorrect. Rather, in a typical deployment the landmarks are behind network address translation (NAT) boxes. We found that, from the system security point of view, our approach of using a centralized server has major advantages over distributed or hop-by-hop implementations because the server is the only component that needs a visible IP address to the other components. This makes our system more deployable because the other system components can be placed behind NAT firewalls, which is quite common in real environments.
- *Privacy.* The main question that people asked about our localization system is what should constitute "fair use" of the localization results? The challenge of privacy in

Figure 6 A screen shot of GRAIL web-based GUI



Notes: The left hand side contains configurable system parameters, and the right hand side is a graphical depiction of a research laboratory being use for localization

Sensor Review
Volume 28 · Number 2 · 2008 · 115–124

wireless localization is sharing the right information with the right people. A suite of access control policies can be developed towards this end. In GRAIL, currently we use a simple approach to only restrict the access of data to the authorized network administrators.

- *Localization performance*. Based on our experiences in office deployment and trade shows, we found that the average latency in localization ranges from a few seconds up to a minute. It varies depending on the solver model and the number of devices that need to be positioned. We experimented with positioning up to 70-100 devices simultaneously. Further, signal coverage can be a significant issue during deployment. We found that landmark placement is critical for achieving good localization performance (Chen *et al.*, 2006). To achieve a better tracking capability, we also noticed that a denser deployment of landmarks is needed in practice. Unlike previous results (Elnahrawy *et al.*, 2004), we found that seven to eight landmarks are needed in order to cover a floor of 16,000 f^2, as compared to four to five landmarks reported previously. On the other hand, we found that less training is needed for some of our solver models. Another key challenge we found is that the localization algorithms need to be robust to missing data. They should produce reasonable location estimation even when only one landmark sees a packet, e.g. a circle around a single landmark is better than no estimate. Finally, we found that it is easy to incorporate new radio technology into GRAIL. For instance, incorporating Roll-Call into GRAIL took only two days.
- *State management*. We used the database to store all the server activities and the positioning results. We found that it is hard to draw lines between the data that is considered as either soft state or hard state. We had to alter the original schema to add time-stamps to landmarks and the regions of localization, which helped sorting out different instances of localization scenarios when the landmarks were moved over the floor or when the region of interest changed in size. Moreover, as a future extension, annotating positioning results with the solver algorithm, which performed localization calculation, in the database can be useful for further data analysis.

9. Related systems

Indoor localization has attracted an immense volume of research since the introduction of pervasive computing. Although it is impossible to cover this broad base of work here, the majority of these works have focused on algorithms and modalities, as opposed to general purpose frameworks.

Many different modalities have been investigated, including RSS (Bahl and Padmanabhan, 2000; Thrun *et al.*, 1998), TOA (Priyantha *et al.*, 2000), and AOA (Niculescu and Nath, 2004; Elnahrawy *et al.*, 2007). Algorithmic work also spans a wide range, including basic matching (Bahl and Padmanabhan, 2000), probabilistic approaches (Thrun *et al.*, 1998), and machine learning algorithms such as BNs (Madigan *et al.*, 2005b) and neural networks (Battiti *et al.*, 2002). These works typically focus on the algorithms' localization accuracy and computational costs. Typically, these works conducted off-line experiments using data collected off-line rather than being implemented as part of a running system. Although such proposed algorithms can be easily integrated into our GRAIL system as solver modules, they are not directly related to the design of our GRAIL system. Instead, in the rest of this section we compare GRAIL to a few running indoor localization systems.

Ekahau, Inc. (2003) is a commercialized localization system using WiFi (802.11) networks. The system bootstraps by site surveying the targeted building and localization is thus achieved through comparison of the RSS set from the mobile device with the survey data.

The key architectural difference with GRAIL is that Ekahau is client-based, while GRAIL is infrastructure-based. That is, while Ekahau does not deploy additional landmarks or base-stations, it needs software components to be installed on the WiFi devices to be localized. Our experience has been that installing extra landmarks is easier and cheaper when the organization controlling the infrastructure wants to provide a localization service. On the other hand, if users want to localize their devices and do not have the support of the organization, client-based solutions are better. The value of a client or infrastructure system depends more on who wants the service (network administrators or users), because of the separation of control and responsibility over devices: administrators control the radio channel but users control their devices.

Ekahau's system also requires that a site survey be performed by a mobile device during the deployment, and whenever the radio environment changes. We found performing site surveys tedious, error prone, and that they needed to be repeated every few months. For this reason, GRAIL is built to allow continuous monitoring as well as support algorithms that do not require a site survey. Finally, Ekahau only supports the 802.11 radio type, while GRAIL is designed to support any packet-based radio networks, such as WiFi and ZigBee networks.

Cricket (Priyantha *et al.*, 2000) and Active Bat (Ward *et al.*, 1997) systems use both RF radio and ultrasound. The localization is performed through TOA and trilateration. We can build ultrasound sniffers and directly integrate the Active Bat system into our GRAIL infrastructure. The cricket system can be similarly adopted if we reverse the role of transmitters and receivers from its original design. However, we will not be able to keep the privacy feature after adaption, i.e. location information is not only available to the particular device being localized.

LEASE (Krishnan *et al.*, 2004) system has a very similar data acquisition component as GRAIL. It uses sniffers to scan the WiFi channels, which are equivalent to our landmarks. It also uses stationary emitters with known locations to continuously sample the environment. This feature is readily supported in GRAIL. GRAIL offers much more than LEASE because it contains the additional server, solver, and database modules to enable a more general purpose and flexible system, as opposed to supporting only a single algorithm.

Place Lab (Lamarca *et al.*, 2005) works both indoors and outdoors and focuses on improving the coverage of localization services in general. In the running system, to automate the localization step, each mobile device will also need to install specific software to query beacon location databases. Place Lab demonstrated an achieved median error of 20-30 m. This is usually unacceptable for indoor applications. However, we may use Place Lab as an intermediate step to cover the outdoor areas between buildings equipped with GRAIL, similar to BGP versus RIP. Such a combination can help us easily achieve high coverage and scalability.

Finally, many outdoor localization systems such as GPS do not work indoors and are thus not comparable to GRAIL.

Similar to Place Lab, they can also be used to integrate buildings equipped with GRAIL. However, additional hardware may be needed to accommodate GPS integration.

10. Conclusions

In this work, we present GRAIL, a general purpose localization system. The main objective of GRAIL is to position arbitrary wireless transmitting devices in an indoor environment. The current system can position 802.11, 802.15.4 and Roll-Call devices. In contrast to a specialized localization system, GRAIL is designed to support multiple physical modalities and a diverse set of localization algorithms. A key contribution of this system is its universal approach: it can integrate different hardware and software capabilities within a single localization framework.

Moreover, we found that a centralized solution has critical advantages that are often overlooked in the literature. First, it makes cleaning and summarizing the traffic observations much easier. Second, it enables a variety of additional services, such as attack detection and tracking, to utilize the same underlying localization system. Finally, we believe that centralization makes enforcing contracts and privacy policies tractable. However, we will leave open the issues of privacy contracts and policy enforcement as future work.

The deployment of such a system in academic and research environments will allow researchers to explore issues beyond algorithms and simulation tools. It would make it possible to conduct higher-level integrated research investigation including privacy studies, security services, and policy enforcement. For instance, we utilized the GRAIL system to conduct research on detecting and localizing identity-based spoofing attacks in wireless networks (Chen *et al.*, 2007a, b). Further, the practical usage of such an approach is significant because it can be applied to a broad array of applications such as monitoring, tracking, routing, and security services.

References

Bahl, P. and Padmanabhan, V.N. (2000), “RADAR: an in-building RF-based user location and tracking system”, *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM)*, pp. 775-84.

Battiti, R., Brunato, M. and Villani, A. (2002), “Statistical learning theory for location fingerprinting inwireless LANs”, Technical Report DIT-02-086, University of Trento, Informatica e Telecomunicazioni, Trento.

Chen, Y., Francisco, J., Trappe, W. and Martin, R.P. (2006), “A practical approach to landmark deployment for indoor localization”, *Proceedings of the Third Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*.

Chen, Y., Moore, R.S., Elnahrawy, E., Kleisouris, K., Francisco, J., Li, X., Chandrasekaran, G., Turgut, B. and Martin, R.P. (2007a), “GRAIL: description and characterization of a real time location service”, Technical Report DCS-TR-619, Rutgers University, Piscataway, NJ, September.

Chen, Y., Trappe, W. and Martin, R.P. (2007b), “Detecting and localizing wireless spoofing attacks”, *Proceedings of the Third Annual IEEE Communications Society Conference on Sensor, Mesh and Ad Hoc Communications and Networks (SECON)*.

Ekahau, Inc. (2003), “The Ekahau positioning engine 2.1”, Whitepaper, available at: www.ekahau.com

Elnahrawy, E., Li, X. and Martin, R.P. (2004), “The limits of localization using signal strength: a comparative study”, *Proceedings of the First IEEE International Conference on Sensor and Ad hoc Communications and Networks (SECON 2004). 406-414*, pp. 406-14.

Elnahrawy, E., Austin-Francisco, J. and Martin, R.P. (2007), “Adding angle of arrival modality to basic rss location management techniques”, paper presented at IEEE International Symposium on Wireless Pervasive Computing (ISWPCÕ07), Puerto Rico.

Kleisouris, K. and Martin, R.P. (2006), “Reducing the computational cost of Bayesian indoor positioning systems”, *Proceedings of the Third IEEE International Conference on Sensor and Ad hoc Communications and Networks (SECON 2006)*.

Krishnan, P., Krishnakumar, A.S., Ju, W., Mallows, C. and Ganu, S. (2004), “A system for LEASE: location estimation assisted by stationary emitters for indoor RF wireless networks”, *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM)*.

Lamarca, A., Chawathe, Y., Consolvo, S., Hightower, J., Smith, I., Scott, J., Sohn, T., Howard, J., Hughes, J., Potter, F., Tabert, J., Powledge, P., Borriello, G. and Schilit, B. (2005), “PLACE lab: device positioning using radio beacons in the wild”, *Proceedings of the 3rd International Conference on Pervasive Computing (Pervasive 2005)*.

Madigan, D., Elnahrawy, E., Martin, R., Ju, W., Krishnan, P. and Krishnakumar, A.S. (2005a), “Bayesian indoor positioning systems”, *Proceedings of the IEEE International Conference on Computer Communications (INFOCOM)*, pp. 324-31.

Madigan, D., Elnahrawy, E., Martin, R.P., Ju, W.-H., Krishnan, P. and Krishnakumar, A.S. (2005b), “Bayesian indoor positioning systems”, paper presented at the 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM’05). Miami, FL.

Niculescu, D. and Nath, B. (2003), “DV based positioning in ad hoc networks”, *Kluwer Journal of Telecommunication Systems*, Vol. 13 No. 6.

Niculescu, D. and Nath, B. (2004), “VOR base stations for indoor 802.11 positioning”, *Proceedings of the Annual ACM International Conference on Mobile Computing and Networking (MOBICOM)*, pp. 2926-31.

Priyantha, N., Chakraborty, A. and Balakrishnan, H. (2000), “The cricket location-support system”, *Proceedings of the ACM International Conference on Mobile Computing and Networking (MobiCom)*, pp. 32-43.

Thrun, S., Burgard, W. and Fox, D. (1998), “A probabilistic approach to concurrent mapping and localization for mobile robots”, *Machine Learning*, Vol. 31 Nos 1/3, pp. 29-53.

Ward, A., Jones, A. and Hopper, A. (1997), “A new location technique for the active office”, available at: citeseer.ist.psu.edu/ward97new.html.

Corresponding author

Yingying Chen can be contacted at: yingying.chen@stevens.edu