Noname manuscript No.
(will be inserted by the editor)

The BehaviorScope Framework for Enabling Ambient Assisted Living

Athanasios Bamis · Dimitrios Lymberopoulos · Thiago Teixeira · Andreas Savvides

the date of receipt and acceptance should be inserted later

Abstract The in-house monitoring of elders using intelligent sensors is a very desirable service that has the potential of increasing autonomy and independence while minimizing the risks of living alone. Because of this promise, the efforts of building such systems have been spanning for decades, but there is still a lot of room for improvement. Driven by the recent technology advances in many of the required components, in this paper we present a scalable framework for detailed behavior interpretation of elders. We report on our early deployment experiences and present our current progress in three main components: sensors, middleware and behavior interpretation mechanisms that aim to make effective monitoring and assistive services a reality.

Keywords assisted living framework · activity recognition · activity learning · assisted living interfaces

1 Introduction

Monitoring people activities and providing automated services that improve safety and quality of life is a very attractive proposition for elders living alone. Although the problem was considered for many years, it has recently begun to become more relevant for two main reasons. First, many studies together with the rising costs of healthcare point out that the caring of elders that live alone at home is about to become a challenge in the next few years [7, 15]. The second, and more positive development is that communication, sensing and processing technologies are rapidly maturing to the point that make automated services for elders living alone possible both in terms of cost and technology.

A. Bamis, T. Teixeira and A. Savvides
ENALAB, Yale University, 51 Prospect St. Room 000, New Haven, CT 06511, USA
Tel.: +120-34-320042
Fax.: +120-34-320593
E-mail: firstname.lastname@yale.edu

D. Lymberopoulos
Networked Embedded Computing Lab, Microsoft Research
E-mail: dlymper@microsoft.com

From a technology perspective, the majority of components required to build such systems are becoming readily available. Many systems under development both in academia [8, 14, 16, 20] and industry [4], as well as some commercial systems [2, 6] are already capable to provide essential monitoring services (for a survey of current state-of-the-art see [5, 7]). What is mostly missing is experience and systematic knowledge to intelligently assemble the components in to robust architectures and practical, deployable systems. In addition, most of these systems focus on collecting and presenting simple statistics, often using intrusive sensors (e.g., wearable devices), requiring, thus, the involvement of healthcare providers and stakeholders in the system loop.

The BehaviorScope project at Yale [19] is investigating these challenges by trying to build a functional system that can autonomously understand behaviors with enough detail to provide meaningful services. The goal of the project is to design an extensible architecture that can use a wide variety of sensors to interpret human activity, dynamically generate activity models and use them to generate alarms, reports, triggers and to answer queries. In this paper we provide an overview of the architecture of the system under development, and report on the main components that our research is trying to address. Section 2 provides an overview of our system requirements, section 3 outlines our system architecture and section 4 explores various methods we have considered for interpreting the data.

2 Overview

The provision of services requires a set of sensors to be deployed inside a home to observe the inhabitants, interpret the observations and provide meaningful responses. Depending on their condition, one can anticipate that the home inhabitants would be willing to subject themselves to a certain level of observation (i.e., give up some of their privacy) in exchange for services. The goal of our architecture is to provide a versatile system that can accommodate this at different levels, from very simple to very detailed observation, according to individual needs. The initial form of the system is intended for elders that live alone, and are fairly independent. In this case, the role of the system would be to eliminate certain risk factors that could otherwise be avoided by resorting to institutionalization. In its simplest form, such a system would offer a wide variety of services:

– *Queries* - the system should be able to answer queries such as: where is the person, is that person getting enough sleep, is the person out of the house beyond the expected time?
– *Alarms and triggers* - notify stakeholder when the person returns/leaves the house, notify when the person wakes up/goes to bed.
– *Detect anomalies* - By observing and learning routines (e.g., daily, weekly, monthly), the system can provide notifications when an unusual deviation from the routine happens.
– *Recognize specific behaviors* - By allowing the programming of specific behavior recognition libraries into the system, one can tailor the system to provide customized observations and actions for each house. This for example would help tailor the same system to people suffering with cognitive decline and people who are frail and run the risk of falling or getting stuck somewhere (e.g., bed, toilet).
– *Actuate* - Take action when certain events (or combinations of events) are detected.

The users of the system should be able to configure the above properties to adapt it to their individual needs by programming custom triggers, defining custom queries for future use and specifying what actions should be taken when a specific behavior is detected. Moreover, for detecting routine behaviors and timing parameters, the system should be able to use a generic specification as a starting point and automatically "cast" itself to the home and the activity patterns of individuals when it is actually deployed.

The above requirements create a new set of challenges involving sensing and data interpretation, and call for a middleware architecture that can support a heterogeneous set of devices and their tailored configuration for each home. Furthermore, for cost effectiveness and ease of installation, a practical system should provide the aforementioned services without requiring the exhaustive tagging of every item in the home with sensors. To make this possible, the *BehaviorScope* project seeks to build a rigorous understanding of what today's off-the-shelf sensors can do, what types of new sensors are required and how a heterogeneous set of such sensors can co-exist in the same framework to collect and interpret data.

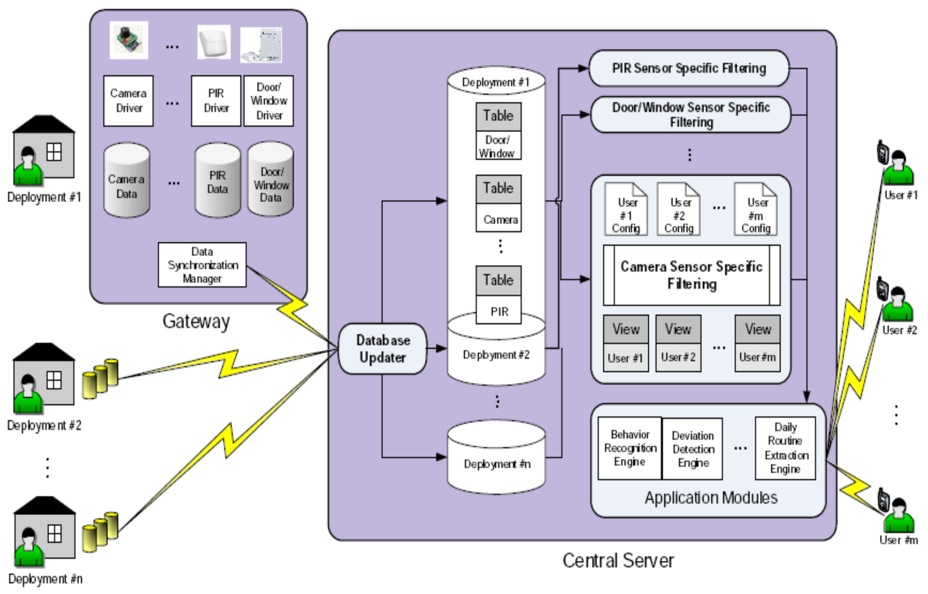


Fig. 1 General overview of the system architecture. By defining a modular architecture, and multiple levels of abstraction, we can achieve scalability and robustness.

An outline of our system-wide architecture is shown in Figure 1. A set of wireless sensors is placed at key locations to collect sufficient information for recognizing a person's activity profile around the house. The data collected by the sensors is forwarded to an intelligent gateway installed inside the house that processes and interprets the data by communicating with a central server. Caregivers and stakeholders can interact with the system via two main interfaces, a mobile phone interface and a web interface. The mobile phone is the main interface for communicating, daily summaries, alarms, triggers and queries. The web interface supports a more elaborate setup that allows the end-user to customize the behavior of the system to each home.

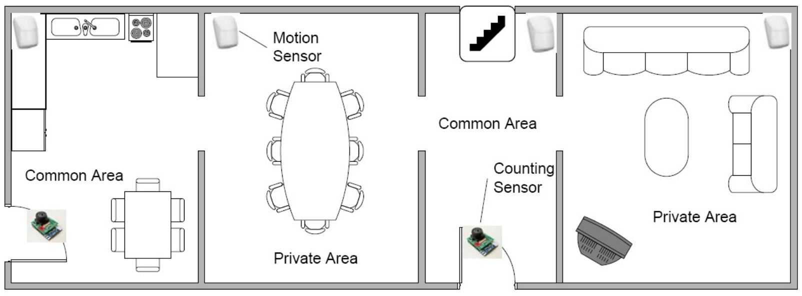


Fig. 2 The areas of a house are separated into common and private. Common areas can be monitored by "intrusive" sensors as cameras, whereas private areas can contain only "non-intrusive" sensors as PIR.

3 System Architecture

A key premise of the BehaviorScope infrastructure is the ability to jointly consider information from multiple sensor types to infer behavior from low-level data. Most of the sensors are off-the-shelf Passive Infrared (PIR), and door/window sensors, assisted by more powerful, motion discriminative sensors derived from cameras. The latter form of sensors is aiming to define a new sensing modality in which people locations and movements in the house can be sensed but no images can be produced. The home floor plan is divided into two types of areas, common and private (see Figure **??**). Counting sensors are only placed in the common areas of the home especially near the exits. PIR sensors and door/windows sensors can be placed anywhere in the house according to the specific monitoring needs. Although it would be possible to exhaustively cover the house with a large number of sensors, in this paper we consider the possibility of achieving similar or better activity inference with a smaller kit of sensors.

The main components of the system include an intelligent gateway (see Figure 3) able to collect data from a large number of sensors, process them and transmit them back to a central server. In the central server data can be stored, preprocessed in a number of different formats depending on the types of sensors and the information that needs to be extracted before it is passed to the application modules. In cases of increased privacy concerns, data processing can be done locally inside the gateway, and the results can be directly transmitted to the authorized end-users, with the central server responsible for only the authentication of the end-users, the configuration of the deployment and the system maintenance.

Thanks to modular design, the addition of new applications or sensors to the system does not interfere with its normal operation. In particular, to add a new type of sensors, cameras for example, the developers have to provide a "driver application" for the gateway, that will be able to collect data from the particular type of sensor network, and dump it into the gateway's database. The system will automatically take care of the data synchronization process with the central server. In addition, the developer

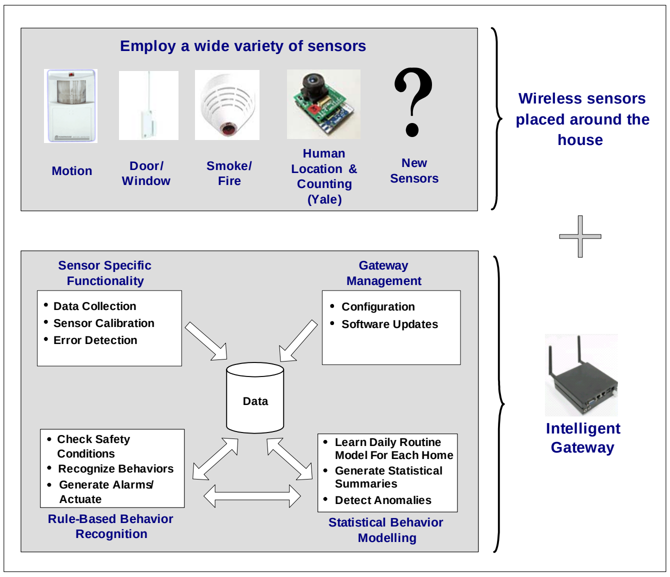

Employ a wide variety of sensors
Motion
Door/ Window
Smoke/ Fire
Human Location & Counting (Yale)
New Sensors
Wireless sensors placed around the house
Sensor Specific Functionality
• Data Collection
• Sensor Calibration
• Error Detection
Gateway Management
• Configuration
• Software Updates
Data
• Check Safety Conditions
• Recognize Behaviors
• Generate Alarms/ Actuate
Rule-Based Behavior Recognition
• Learn Daily Routine Model For Each Home
• Generate Statistical Summaries
• Detect Anomalies
Statistical Behavior Modelling
Intelligent Gateway

Fig. 3 A home sensor network kit consists of one or more sensor networks and an intelligent gateway that can manage the sensor networks, collect data, pre-process them and transmit them to the Central Server.

can add a number of preprocessing modules, depending on the type of "fundamental" information that needs to be extracted from the data. For example, in the case of a camera node we can define areas of interest and generate an event, whenever motion is detected inside that given area. The outputs of the preprocessing modules are added back to the database, and can be used by applications running either on the central server or locally by the end-users. Similarly, in order to add a new application on the central server, all we need to do is add a preprocessing module, that given possibly some configuration parameters from the users, will convert raw (or previously preprocessed) data into the proper format and it will then pass them to the application module.

3.1 Camera-Based Privacy Preserving Counting and Human Localization Sensors

The PIR and door sensors used in our system are off-the-shelf sensors readily available form different vendors. Although PIR sensors detect motion, they don't necessarily detect occupancy of an area inside the house. For instance if two people enter a room, and one person leaves, measurements from PIR sensors alone cannot easily determine that a person is still in the room if the person does not move. Moreover, most commercial sensors have very primitive MAC layers, primarily geared towards security alarm trigger applications. This does not always favor assisted living setups where readings from multiple sensors, and their relative timing have a meaning. A straightforward solution is to attach PIR sensors to off-the-shelf sensor nodes, but that would cancel their main advantages of low cost and increased battery lifetimes.

Because of these limitations of PIR sensors, and the need to count and track multiple people we are currently developing a new custom sensing modality that can localize and track people inside the house without requiring them to wear a tracking device. Although the sensor is derived from cameras, it directly aims at the development of a new camera chip that can localize, count and track people without providing any image information to the rest of the system.

Due to the extensive amount of processing that is generally required for computer vision tasks, camera nodes architectures in the literature have typically followed one of two approaches: (1) using the fastest low-power CPU available; (2) adding specialized processing components that are capable of a high degree of parallelism, such as DSPs or CPLDs. In our research, we take a third route, by making fundamental changes to the underlying image sensors themselves. The typical image sensor outputs a serialized array of pixel intensity values. This array contains raw data that must be heavily processed before any desirable information can be gathered. What is more, only after the image is processed can one know whether or not the captured scene is interesting. The result is that many uninteresting frames end up being captured and processed before ultimately being discarded, resulting in a large waste of resources.

Our platform, on the other hand, is built with biologically-inspired Address-Event (AE) imagers in mind [17]. Instead of outputting arrays of pixel intensity, these imagers asynchronously output an *address* (in pixel coordinates) every time an *event* is detected. Events can be any measurable phenomenon. In the case of the imagers we use, an event is triggered every time a pixel senses motion (an above-threshold change in intensity). The power of address-event lies in three separate properties: Processing occurs at the pixel level, freeing the controlling CPU from complex imaging tasks; AE sensors do not discretize time into "frames", which allows for precise measurements and provides privacy; AE sensors are typically ultra-low-power.

In our current platform, we emulate the address-event imager in software. The emulated parameters are used to guide our custom hardware imager design. Since our algorithms are written for address-event input, once a hardware AE design is fabricated it can directly substitute for the emulated version. The sensor nodes in our deployment use Intel iMote2 sensor nodes coupled with a custom camera board. The purpose of the nodes is to find and track the people in their field-of-view, communicating the detected coordinates back to their base.

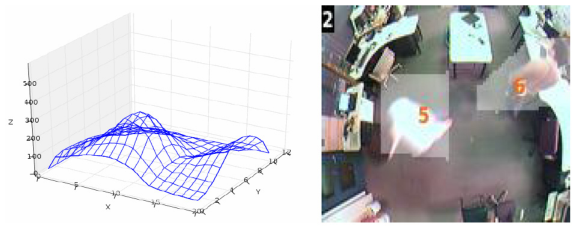


Fig. 4 Multiple people counting and tracking can be achieved by estimating a motion histogram and detecting its peaks.

The software on the sensor nodes detects humans based on size and motion by constructing a motion histogram [18]. The histogram utilizes person-sized bins to compute a density estimation of possible human locations. This is done by dividing the image into partially-overlapping person-sized areas, and counting the number of above-threshold motion pixels that lie within each area. These counts are organized as bins in a two-dimensional histogram, and the local maxima are computed to locate the histogram peaks. Each peak indicates the likely location of a moving person, as seen in Figure 4.

3.2 Intelligent Gateway

Our gateway architecture consists of four main categories of software modules, shown in Figure 3. The first type of modules are sensor specific and consist of the drivers for receiving the sensed data from the network, removing or correcting erroneous measurements, detecting malfunctions of the sensors and, generally, managing the correct operation of the deployment. The collected data is stored in a local database, which is incrementally (i.e., only new data) transmitted to the central server, by a synchronization module. Other modules in this category include modules for receiving software updates, modules for checking the correct functionality of the gateway, modules performing authentication, modules allowing the remote configuration of the gateway parameters and, generally, any module that isn't sensor specific or concerns data processing. When it comes to data processing, there are two categories of software modules. The first category involves software modules that collect statistics, learn from the collected data and possibly respond on significant deviations, whereas the second category includes modules that try to detect certain behaviors and patterns inside the network and possibly take certain actions as a response.

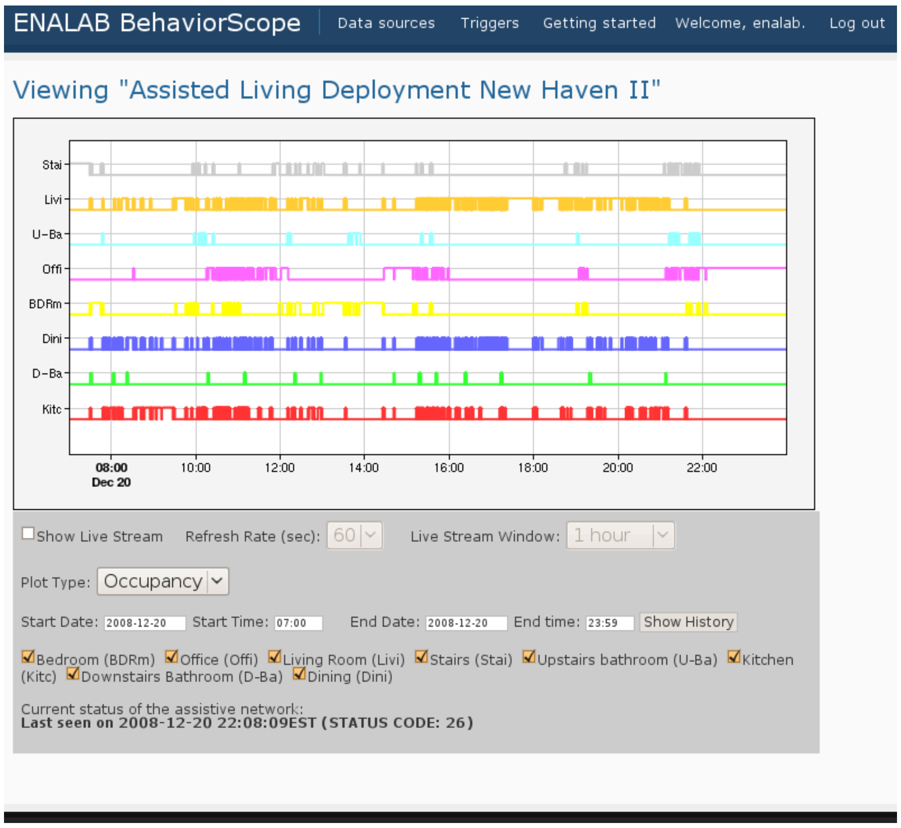
3.3 Central Server

Besides the system management modules (e.g., gateway software updating module) and the module that updates the database with the incoming data from the deployments, the central server contains two more main categories of software. The first one has to do with preprocessing and conditioning of the incoming data, and depends on the type of the sensors and the requirements of the end users. In the case of complex sensors, such as cameras, the data collected from every type of sensor can be processed in order to extract some features that can directly be used by the users, or be given as input to one or more applications.

The central server stores the data in a separate database for each deployment and incrementally preprocesses the data according to sensor types and the required information that needs to be extracted (i.e., according to the data processing module that we wish to use). The new data is then passed to a sensor specific module, which using user-specified and statistically learned configuration parameters create user-specific views for each user and each possible data processing module that is available for the given deployment and sensor. These views are subsequently accessed by the application modules located inside the server, which will generate a number of results, or by custom applications designed by the users (and located outside of the server).

3.4 The BehaviorScope Web Portal

To be of use, every assisted living environment must provide both a synchronous interface for real-time monitoring of the persons of interest and an asynchronous interface



Viewing "Assisted Living Deployment New Haven II"

☐ Show Live Stream Refresh Rate (sec): 60 Live Stream Window: 1 hour

Plot Type: Occupancy

Start Date: 2008-12-20 Start Time: 07:00 End Date: 2008-12-20 End time: 23:59

✓ Bedroom (BDRm) ✓ Office (Offi) ✓ Living Room (Livi) ✓ Stairs (Stai) ✓ Upstairs bathroom (U-Ba) ✓ Kitchen (Kitc) ✓ Downstairs Bathroom (D-Ba) ✓ Dining (Dini)

Current status of the assistive network:
Last seen on 2008-12-20 22:08:09EST (STATUS CODE: 26)

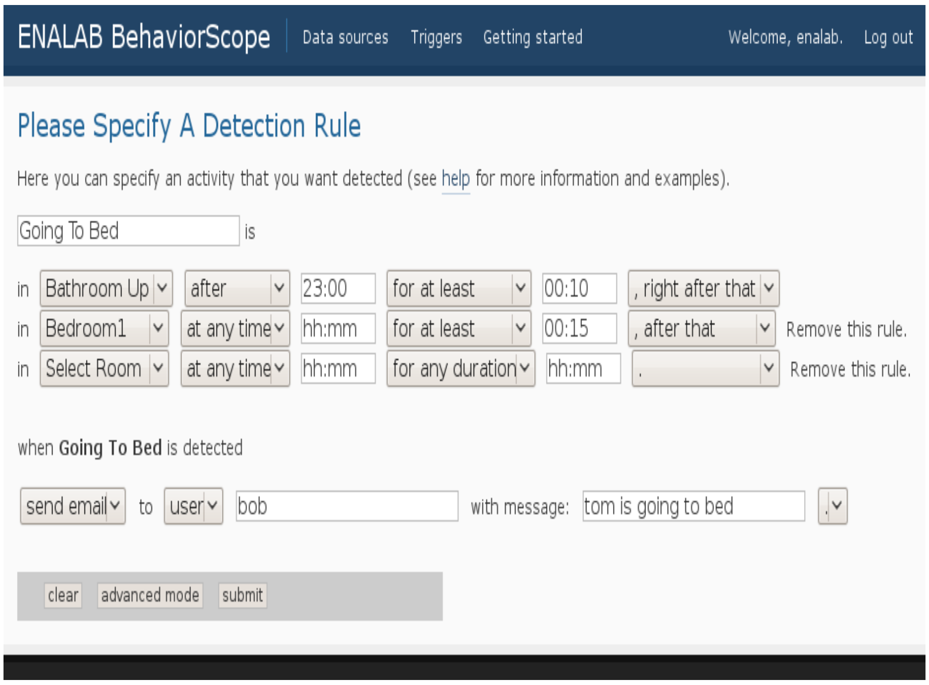
Fig. 5 Occupancy information for the rooms of an assisted living deployment. This visualization type can be used for easily inspecting the durations spend in each room for a given time interval.

that can be used to communicate notifications, updates and most importantly alerts in cases of emergency. Since the interface needs to be accessible to a large and diverse set of users the best option is to implement it as a web service, where people can login securely, access all the information they need and configure the types of notifications, updates and alerts that they require.

This set of users can include besides the monitored person and one or more stakeholders, caregivers, emergency personell, persons that reside close to the monitored person (and can be of assistance in cases of emergency), researchers, social workers and others. Each of these groups has in general different requirements from the assisted living environment, and needs to access different types of information. Moreover, the monitored person will usually be willing to sacrifice different amounts of privacy in exchange for services provided by each group. To what is more, information meant for different groups of users will usually involve different levels of anonymization.

The interface for an assisted living deployment should provide many different types of representation of the collected data to accommodate the needs and technology competence of different users, it should provide different types of statistics, and it should allow programming different types of notifications, alerts and statistical summaries.

These services should be provided by both a graphical environment and using natural language. Moreover, the interface should provide both online access to the incoming data and access to the history of the particular monitored person. Configuration and customization options are essential to allow the users to share information maintaining full control to the type and amount of information that different users can access.



Please Specify A Detection Rule

Here you can specify an activity that you want detected (see help for more information and examples).

Going To Bed is

in	Bathroom Up	after	23:00	for at least	00:10	, right after that	
in	Bedroom1	at any time	hh:mm	for at least	00:15	, after that	Remove this rule.
in	Select Room	at any time	hh:mm	for any duration	hh:mm	.	Remove this rule.

when **Going To Bed** is detected

send email to user bob with message: tom is going to bed .

Fig. 6 The advanced user interface provided by STFL (STFL GUI) can be used by the users of the BehaviorScope system to define their own rules and sequences of rules and constraints to actuate upon the detection of events of interest.

The BehaviorScope web portal (BScopeWeb) [1] provides most of this functionality by allowing to the users to register datasources of different types (currently: motion sensors, localization sensors based on cameras, rfids, GPS-enabled phones, and simple sensors as door/window, panic, tamper or temperature/humidity sensors), and share the information provided by the datasources with other users of the system. Among others the BScopeWeb provides different forms of visualization of the data (see for example Figure 5), different types of statistics and advanced interfaces for the customization of a deployment and the definition of notifications and alerts (see Figure 6).

Every datasource is characterized by a serial number, which identifies a datasource uniquely and allows web based recording and access to the data. Every deployed sensor network appears as a different datasource on the portal and a gateway can support transparently different types of networks using different serial numbers. In addition a user is allowed to define a new datasource using some primitive data types offered by the system and re-use the existing data sharing, visualization, statistics extraction and data interpretation mechanisms.

A cell phone interface (based on SMS and email) provides a subset of this functionality and is mainly used for communicating alerts, high-level statistics and enabling the user to perform simple queries.

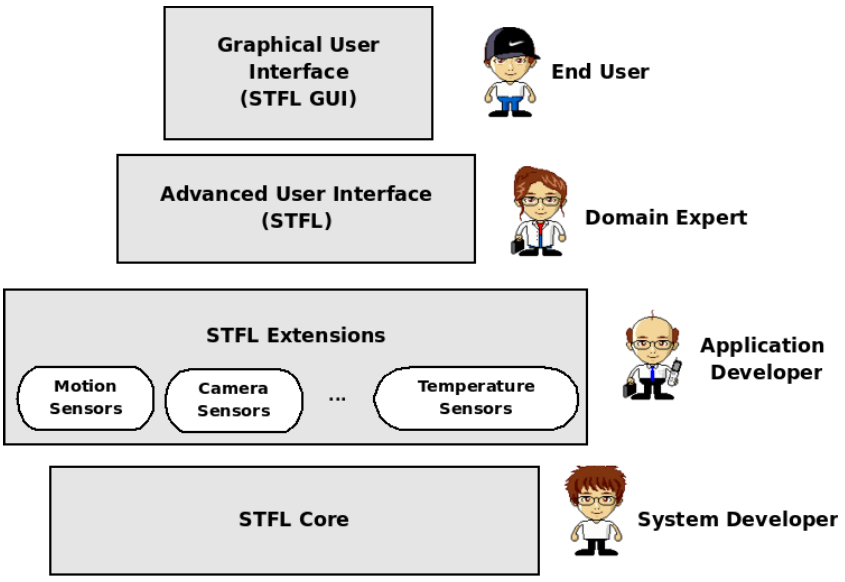


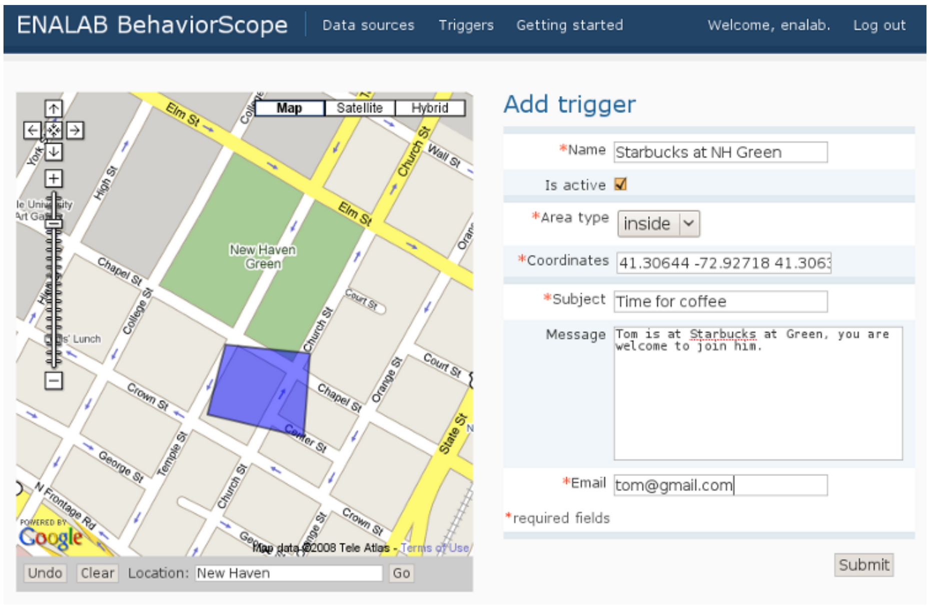
Fig. 7 STFL provides four different layers that aim to users of different expertise. The two top layers can be used by the end-users of an assisted living environment to define their own custom triggers, as well as by the owners of an assisted living deployment in order to control the access rights for other users and group of users.

3.5 A SpatioTemporal Filtering Language (STFL) for Enabling Actuation

In order for any assistive living environment to be of value to its users, it must provide at least a minimum set of functionality. In its simplest form, it must provide the means for communicating events of interest to its users. The most common methods for communicating updates, notifications and alarms to the users of the system are email messages and SMS. Other options include phone calls, updating visual notifications (e.g., turning on lights, changing the displayed image on a digital picture frame) or triggering audio alarms. Recipients of the alarms and the notifications can include the monitored person itself (e.g., “remember to take your medication”), stakeholders, caregivers and emergency personnel.

Besides these simple forms of actuation, the user is also allowed to programmaticaly create new types of “virtual datasources” that are generated from higher-level semantics extracted from the data. These datasources, among others, enable fine-grained control to the access rights of different users and groups of users to the data of the monitored person. This becomes possible, by allowing the owner of a datasource to define simple rules and sequences of rules and constraints using a *SpatioTemporal Filtering Language (STFL)* [3].

STFL is a close to natural language that consists of four different layers (see Figure **??**), each aiming at users of different expertise. The main assumption of STFL is that high-level activities can usually be decomposed into simple rules or Finite State Machines (FSMs) describing the activity in terms of sequences of locations and specific temporal characteristics. The two top layers can be used from within the BehaviorScope web portal, whereas the two bottom layers aim mostly at the developers of the system. In particular, the "Advanced User Interface" (STFL) provides a programming language close to natural language, which can be used to describe rules and sequences of rules and constraints, whereas the "Graphical User Interface" (STFL GUI) provides a graphical interface (see Figure 6), through which users can set simple triggers.



Add trigger

*Name	Starbucks at NH Green
Is active	✓
*Area type	inside
*Coordinates	41.30644 -72.92718 41.306
*Subject	Time for coffee
Message	Tom is at Starbucks at Green, you are welcome to join him.
*Email	tom@gmail.com

*required fields

Location: New Haven

Fig. 8 The BehaviorScope web portal allows users to specify locations of interest on a map and generate a message whenever the monitored person, carrying a GPS-enabled phone running the mobile client, enters or leaves these areas.

3.6 Beyond In-House Monitoring Using GPS-Enabled Phones

In many cases, extending the monitoring of a person outside of the house can both increase safety and provide valuable information about the condition and the routine of the person. The BehaviorScope system allows users to record their position using GPS-enabled phones [21]. Our initial deployment in an urban setting heavily relies on the client application running on mobile phones. The deployed mobile client (currently supports GPS-enabled Blackberry phones) can be downloaded from the BehaviorScope web portal and supports several features than enable it to be an active contributor to the overall system architecture.

Since the goal of this application is to make it easy to stay safe and secure anywhere, at anytime, the application needs to efficiently manage its power consumption and make its state known to the server at all times. The application informs the server of its status on power up and shutting down, loss of GPS signal, and feature usage. To conserve power, local processing, intelligent sampling and other sensors such as accelerometers need to be exploited. Our prototype experiences have shown that reading the GPS alone can take a noticeable toll on the phones battery lifetime. Such excessive power consumption could be reduced by utilizing accelerometer sensors and context inferred from the behavior monitoring applications to intelligently manage the GPS sampling and communication frequency. The BlackBerry smart phones used in our prototype deployment do not have accelerometers but other phones such as the Nokia N95 and iPhone already have them. We anticipate that more phone models will have them in the future.

The mobile client contains a basic tracking feature that users can select to turn on to allow their location to be sent to a central server. The user can choose to have this feature on 24-hours a day, or just during select commutes or times of day. Ideally, the application will be running continuously so as to collect as much information and allow as much personalization as possible. This location information is then accessible securely from any computer or mobile device that has access to the internet via the BehaviorScope web portal.

A “Virtual Escort” feature is an integral part of the mobile client. It allows users to have an escort when the user cannot find anyone else to commute with. This feature provides a cost-effective and time-efficient solution to staying safe and gives the user access to a programmable PANIC button that can let interested parties know there’s trouble and exactly where the user is. The web interface also allows users to set up triggers that inform family and friends via SMS or e-mail alert when the user is leaving or entering a pre-defined space. This automates the process of checking a user’s location by having an automated message sent out according to the pre-specified preferences. These triggers can be simple conditions about geographic locations (see Figure 8) or more advanced rules and sequences of rules and constraints described using STFL.

If the user defines certain areas as being associated with specific activities, the system engine can write automatic digests of a user’s day to send to friends and families. This would allow a user to automatically “keep-in-touch” with the monitored person, even when the persons are very busy and have no time to call or e-mail.

4 Data Processing & Interpretation

Depending on the granularity of the data and the application of interest to the end-user, the system can provide a wide range of statistical information. In the following subsections, we will use data collected from online deployments in two different homes, consisting mainly from PIR and door/windows sensors. *Deployment A* has been continuously monitoring an elder person living alone in the USA for more than 7 months, whereas *Deployment B* monitors an elder couple and their adult son in Cyprus for the past 4 months. In both deployments camera sensors are located near the exits of the house and are used only for counting the number of persons present in it.

The following subsections first discuss the statistics we can extract from motion-only information generated from PIR sensors in the BehaviorScope deployments. The discussion is separated into two cases, the case where we have a single person living in the house and the case where we have more than one persons living in the house. Afterwards, we are going to shortly discuss a method for detecting significant deviations from the "normal" living pattern of a person or a house, and in the following two sections we are going to discuss how our system can automatically generate a high level model of the daily living pattern of a person, as well as how it can be programmed to detect specific behaviors.

4.1 Motion Statistics

The lowest level of information we can extract from a motion sensor is a time-stamped notification of when motion was detected. Although PIR motion measurements are not sufficient to determine occupancy (i.e., wether a person is in a certain room or not), they can provide information about people movement inside the house. This information provides an indication of the room occupancy patterns inside the house.

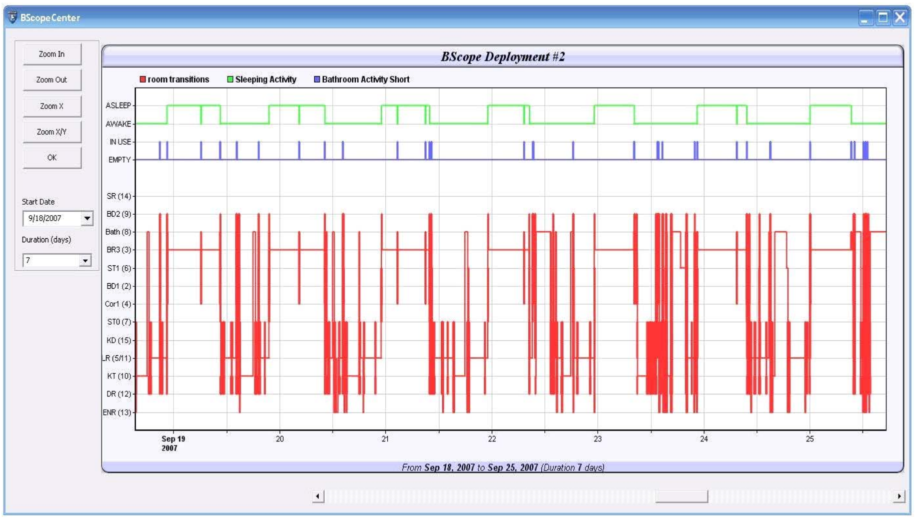


Fig. 9 *Room transitions and detection of sleeping activity and bathroom usage (using rule based triggers) of the monitored person of Deployment A (person living alone) for a period of one week.*

4.1.1 Single Person Case

In the case where only a single person is in the house, time-stamped motion sensor measurements capture the room-to-room transitions of the person. This information on its own can reveal the activity profile of a person and the level of periodicity of a person's daily routine. Figure 9 shows the room transition profile of the elder in *Deployment A* over the period of one week (September 19-25, 2007). The sequences reveal that the person has a very consistent daily pattern, and with a few basic rules and statistics we can extract basic activities and sleep patterns [9].

Moreover, using simple rules and collected statistics (e.g., average sleep duration) we can detect very simple activities, as for example night sleep. Night sleep in the deployment of Figure 9 can be inferred when we detect motion in the third bedroom of the house (BR3), after 11pm, and followed by lack of motion for at least approximately 30 minutes.

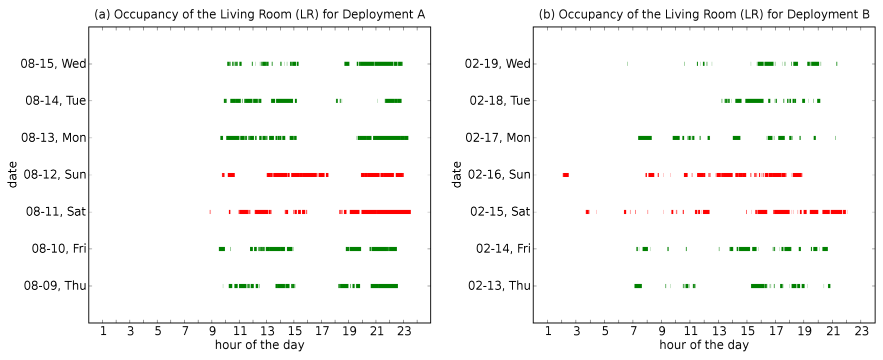


Fig. 10 Occupancy of the living room inferred from motion information for (a) *Deployment A* (single person living alone) and (b) *Deployment B* (multiple people).

Assuming that a person is not moving between two consecutive motion notifications, we can, additionally, provide occupancy statistics for a given location. Figure 10.(a) shows the occupancy of the living room of *Deployment A* (person living alone) for the duration of a week (August 9-15, 2007). From this figure it is easy to observe that the person spends significant amounts of time in the living room, and usually around the same time of the day. In particular, we can see that the person will always spend time in the living room (watching TV), late in the evening (before going to bed), as well as during most of the morning and noon, until she goes to work around 3:30pm. It is easy to observe, that during the weekend this pattern changes significantly, since for example the person will spend more time in the living room and will, also, spend time in the living room between 3:30-6pm, something which can't happen during a normal weekday when the person is at work.

4.1.2 Multiple People Case

In the case of multiple people living in a house, motion sensor data loses its sequence properties and cannot reveal the daily patterns of one person in specific. Since, the sequence property is essential for inferring occupancy information, we can see in Figure 10.(b) that there is no clear occupancy pattern for the living room of *Deployment B* (multiple people in the house). The data however still provides some useful information on the usage profile of each room in the house.

To provide meaningful statistics comparing the utilization of the rooms of the house, we first need to define a common representation of the "quantity of motion" for a given area and time window, that we are interested. Hence, we define a new metric for the mobility of a person at a given location and time window, called *relative mobility level*.

The relative mobility level is essentially the normalized amount of motion in a specific area and time window with respect to a given time period and a given area of interest.

More formally, given the minimum time duration t_{min} (e.g., 15 minutes) for which we are interested we can define a set of *time bins* t_{b_i} over a given time period T (e.g., a day) as $t_{b_i} = [\frac{T}{i \cdot t_{min}}, \frac{T}{(i+1)t_{min}})$. Similarly, for a given set of sensors S we can define a set of *space bins* s_j as the union of space covered by one or more members of S (e.g., the area covered by sensors with ids 5 and 11, which is the living room area for *Deployment A*). If we take all the pairs of space and time bins, we define as relative mobility level of each such pair the total number of motion notifications that we received in the particular space bin and the particular time bin over the total number of motion notifications we received for all the period T and all the sensors in S.

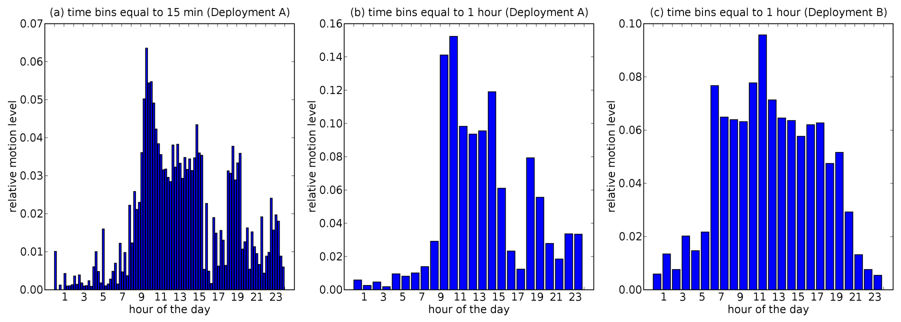


Fig. 11 Relative mobility levels for the entire house and (a) *Deployment A* for time bins equal to 15 minutes, (b) *Deployment A* for time bins equal to 1 hour, (c) *Deployment B* for time bins equal to 1 hour.

For example, Figure 11.(a) shows the average relative mobility level of an elder person living alone in 15-minute intervals. In this case S is defined to be the entire house (i.e., all sensors) and the *time bins* are selected to be 15-minute durations during the course of the day. Consequently, every bar in the graph indicates the average "quantity of motion" at the given 15-minute window during the day for the entire house. Apparently, depending on the information that the end-users or specific applications require, we can have different types of resolution. For instance, in Figure 11(b) shows the average relative mobility level for the same deployment, but with hour-long *time bins*.

From Figures 11.(a) and 11.(b) it is easy to extract useful information for the daily living pattern of the monitored person. From the plots it is easy to infer when the person is sleeping or is out of the house by combining measurements with other context information such as the time of the day or the last known location of the person inside the house. In deployment A, it is easy to observe that the person is going to bed some time between 11:30pm and 12:30am, and wakes up some time between 8:30am and 9:30am, since the detected "amount" of motion suddenly increases. Moreover, it is easy to observe that the person consistently goes out of the house some time after 3:30pm and returns some time after 5:00pm and before 6:00pm. Spikes that appear while the person is absent or during the night (when the person is sleeping), are mainly attributed to sensing errors. For instance, you can see a spike at around 5:30am in the morning, which is caused by a misconfigured motion sensor that triggers whenever it detects light changes (in the particular case, sun rising). The statistical information provided by the *relative mobility level* can be used in order to provide time windows, where interesting events occur or specify the required timeouts for detecting certain events, based on user-defined rules. Apparently, selecting different time or space resolutions can be useful for the detection of different types of events.

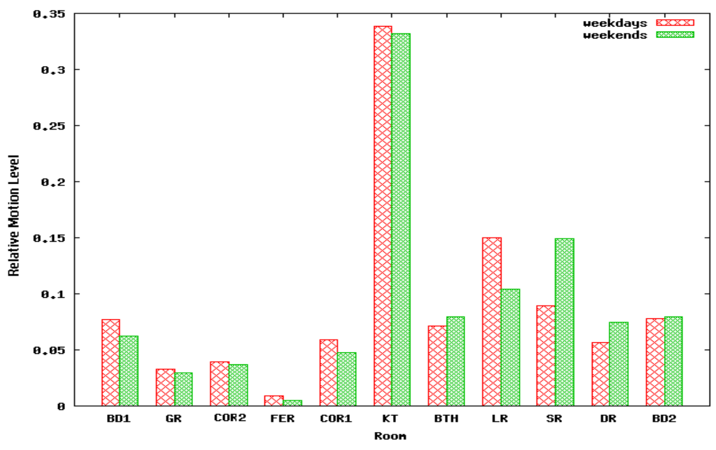


Fig. 12 Average daily relative mobility level for every room of Deployment B. Weekdays are separated from weekend days in order to demonstrate that they follow different occupancy patterns.

Figure 11.(c) shows a similar plot for the house of *Deployment B*, where the motion pattern is significantly different from that of *Deployment A*, and doesn't provide as much information as that of Figure 11.(b). Figure 12 plots the average *relative mobility level* of the rooms of the house of *Deployment B*, which is an indication of their utilization. We separate weekdays from weekends, in order to make some interesting observations obvious. In particular we can see that for most of the basic rooms the motion pattern remains approximately the same. However, we can see that the utilization of *Bedroom 1* ("$BD1$" in the figure) decreases during the weekend. This happens due to the fact that *Bedroom 1* is used by a young adult, who on a Saturday night will spend most of his night out of the house. On the contrary, the utilization of *Bedroom 2* ("$BD2$"), which is used by two elders (who don't work) remains the same. Similarly, we can see that during the weekend the family spends significantly more time hanging out in the *Sun Room* ("SR") of the house and, also, more time in the *Dining Room* ("DR") having lunch and dinner.

4.2 Detecting Deviations

To detect deviations in the living pattern of a person or the house, we have first to model the motion activity pattern in a way that will enable us to find regularities, thus defining a notion of "normal". Apparently, every person or house (in the case of multiple people) has its own pattern, which changes over time. and is also dependent on many macroscopic parameters, as for example the time of the year or, in the case of the house, the current set of people living inside it. Of course, we can try to detect deviations in many different time windows, but for the following discussion we will limit ourselves to detecting "deviating days". Besides a daily pattern, a person or a house can have patterns in many different time resolutions both larger and smaller. For example, most people have a certain wake-up routine and a house has a yearly usage pattern, that is, the utilization of the rooms changes depending on the season of the year. This becomes apparent even by simple observation of Figures 10, and 12, where it can be seen that the normal pattern of a weekday presents several differences from the pattern of a weekend day, both when we focus on the pattern of a person and when we focus on the pattern of a house.

To learn the daily pattern of a person, ideally, we need to discover what remains invariant, possibly adapting over time to the new parameters. Our intuition is that a person will spend approximately similar amounts of time in a place over the course of a "normal" day and will produce proportionally equal amounts of motion information. By modeling the daily motion pattern of a person using a vector, with each field of the vector indicating the relative mobility level at a given place of the house during a given time window, we can use the distance of the vectors as an indication of how different two days are. We expect that the distance between "normal" days will be relatively small in comparison to "deviating" days. Thus, we can define as "deviating days" any vectors who are outliers. In order to detect outliers a clustering algorithm, such as k-means, can be used.

4.3 Extracting Activity Models

Moving to a different category of information, instead of just trying to collect statistical information or identify variations of the daily living pattern of a person, we can try to detect repeating patterns, and based on them create a model of the daily habits of the person. To accomplish this, first, we model the sensor network as a spatiotemporal symbol generator that is triggered by the monitored person as he moves over space and time. Based on our network model, we formulate the problem of finding the daily activity model of a person as the problem of finding the most probable, network-level, sequences of node-level, sensing features, namely location, time and duration. By simple observation of Figure 9 it is, already, easy to observe that that the daily activity of the person under observation has regular recurring patterns.

While the statistical representation of the raw sensing data and its variation over time can provide valuable information about the monitored person, it fails to provide an in-depth analysis about the person's daily living habits. As the person moves inside the house, a sequence of detected sensing features is produced over time. These features might encode spatial information, such as the rooms /areas the person visits or the objects with which she interacts, as well as temporal information, such as the exact time and duration of these features. The sequence of these recorded sensing features over the course of a day represent the monitored person's daily activity signature. Using this stream of symbols, we formulate the problem of human activity modeling as a spatiotemporal pattern-matching problem on top of the sequence of recorded sensing features and solve it using an exhaustive search algorithm [9].

To automatically discover the sequence of sensing features that frequently appear in a collection of daily signatures we use an exhaustive search algorithm that is based on

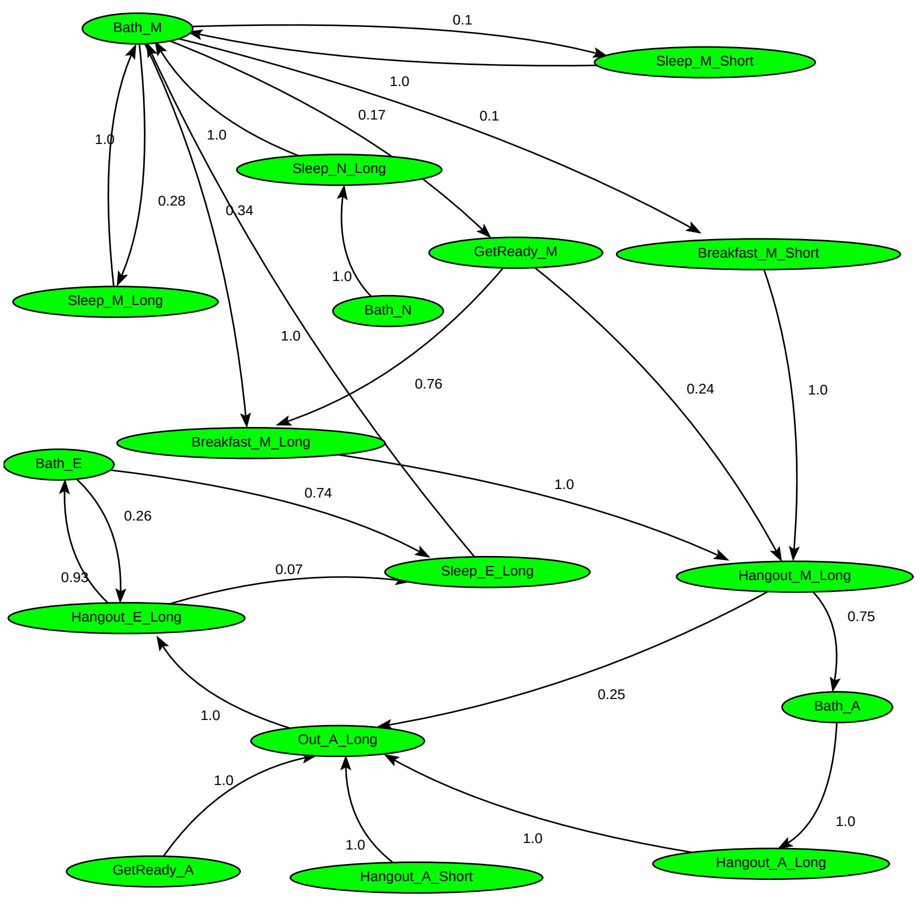


Fig. 13 Simplified daily living model of the monitored person of Deployment A.

the *a-priori* principle: *any subsequence of a frequent sequence has to also be frequent* [9]. Given this, we have employed an exhaustive, yet very efficient, search algorithm that automatically discovers the most frequent sequences of sensing features. Initially all the frequent sequences of size 1 are discovered. Then, using the set of frequent sequences of size 1 as our starting point we identify the most frequent sequences of size 2 and the algorithm continues iteratively until no frequent pattern is finally discovered.

At the end, the most frequent sequences of features of different sizes have been identified. Since these sequences represent the monitored person's frequent activities, when combined, they can be used to build the daily living model of the monitored person. For instance, Figure 13 shows the daily living activity model that was extracted out of 30 days of recorded data of an elder person living alone in *Deployment A*. In this case, the basic sensing features recorded were very primitive activities such as sleeping, having breakfast etc.

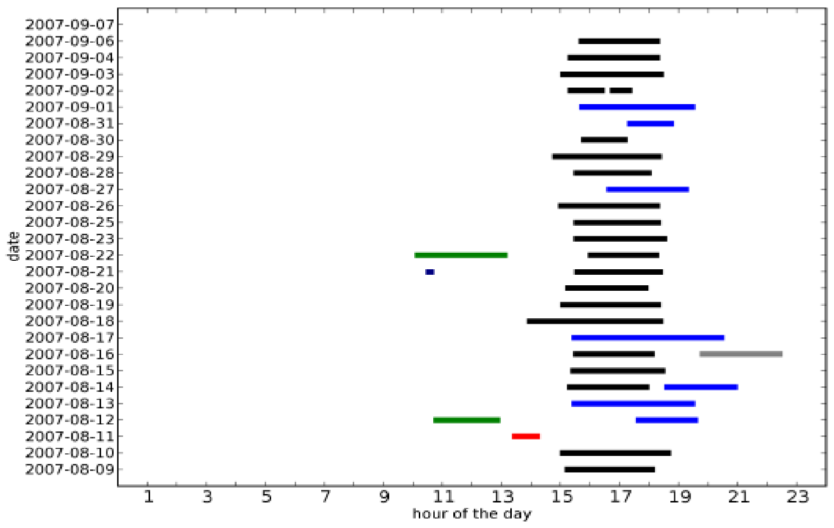


Fig. 14 The different instances of the "Out" activity (i.e., the person being outside of the house) of the person of Deployment A for the period of approximately 1 month. With the same color appear clusters with instances with approximately the same temporal characteristics (i.e., start time and duration). In the plot the y-axis depicts different dates and the x-axis depicts absolute time for a given date. Thus the start of a line indicates the start of an instance of the "Out" event and the end of the line symbolizes the end of the event.

4.4 Learning Temporal Characteristics

Although reasoning with sequences of events and locations can reveal many informations about a persons routine, it lacks a significant component characterizing every human activity, namely time. The stream of symbols generated by the sensor network contain a temporal dimension, which can be used to improve our knowledge and increase the accuracy of our models. In particular, every event and activity is associated with two temporal parameters; the start time (e.g., sleep started at 11pm) of the event or the activity and its duration (e.g., sleep lasted 8 hours).

The main challenge in extracting temporal characteristics lies on the fact that time and duration of a sensed event, are continuous variables that can take any value. To consider them in a model, these quantities need to be appropriately discretize. In doing so however, one needs to consider the fact that temporal characteristics may differ in two ways. Temporal variations within one particular event's time, and temporal variations across different event types. Even worse, these characteristics for a given sensing event might completely change over time in a given sensor network deployment or even across different network deployments. Because of this, extracting a set of discrete time and duration parameters that best describe a sensing event across different event instances is not trivial.

Thus, a data driven approach that is able to automatically discover the temporal properties of the sensed events assuming no a-priori information about the event or its source is needed. The goal of this process is to provide an answer to the question: "when and for how long does this event type take place?".

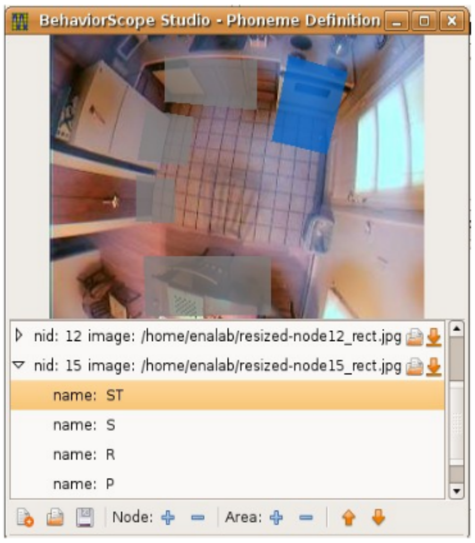
The BehaviorScope framework, provides a method for extracting the temporal characteristics of an event or an activity by formulating the problem as a clustering problem of 2-dimensional vectors and proposing a new metric and a new agglomerative clustering algorithm for grouping together events and activities with very similar temporal characteristics [10]. Every vector contains the start time and the duration of an event, which essentially correspond to a line in the plane (see Figure 14). A distance metric called pairwise-density is used to characterize the degree of overlap between two lines or equivalently how similar the temporal properties described by two vectors are. Finally, an agglomerative clustering algorithm attempts to maximize the overall average pairwise-density of the clusters, without making any assumptions about the number of the clusters.

The output of this algorithm (see Figure 14) are clusters of the instances of an event or an activity with very similar temporal characteristics, along with a metric indicating the confidence for the particular cluster. Using the latter metric we can identify relatively invariant activities of the person (i.e., activities that will repeat at approximately the same time and for the same duration every day) and improve our model by defining new symbols that incorporate time. As an example, Figure 14 shows when the monitored person of *Deployment A* leaves the house. From this figure we can identify two main clusters, the one starting at approximately 15:15 and lasting for about 175 minutes and one, significantly less probable, starting at about 16:35 lasting for about 190 minutes.

4.5 Rule-Based Activity Inference

In addition to the automatically extracted activity models, our project has also developed a behavior interpretation system with which users are able to describe activities as a collection of probabilistic rules with spatial and temporal characteristics expressed in high level script form. Each activity description has well-defined inputs and outputs enabling the creation of a library of activity components that can be connected together into hierarchies to provide even more complex interpretations. The power of such a framework comes from the hierarchical organization of reasoning. This allows the use of simple timestamped, localized sensor measurements to reason about more macroscopic behaviors taking place in space and time.

The main idea is that human behaviors are sequences of very primitive actions that take place over space and time. Different activities can be described by simply combining these primitive actions over time in different ways. A multimodal wireless sensor network monitoring a person's location and interaction with different objects over space and time provides a stream of basic sensing features for identifying these primitive human actions. The proposed method suggests to parse the sequence of detected sensing features into higher level human behaviors in a hierarchical bottom-up processing model that is similar to natural language processing. The set of recorded features becomes the human activity alphabet. In the same sense that we combine letters to form words, we combine these features to define primitive actions; similarly, as words are combined to form sentences, sequences of primitive actions are combined to describe basic human activities; and so on from sentences to paragraphs, paragraphs to stories, we combine human activities over space and time to define macroscale human behaviors.



nid: 12 image: /home/enalab/resized-node12_rect.jpg

nid: 15 image: /home/enalab/resized-node15_rect.jpg

name: ST

name: S

name: R

name: P

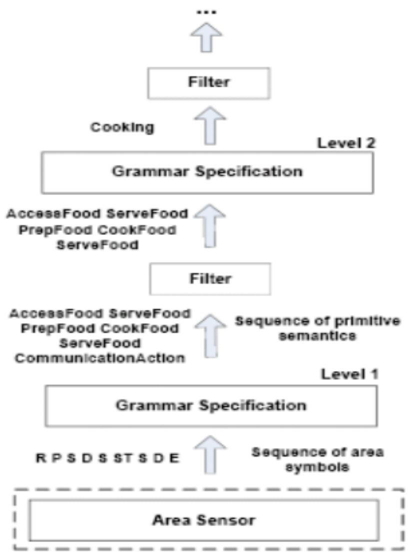

...
Filter
Cooking
Level 2
Grammar Specification
AccessFood ServeFood PrepFood CookFood ServeFood
Filter
AccessFood ServeFood PrepFood CookFood ServeFood CommunicationAction
Sequence of primitive semantics
Level 1
Grammar Specification
R P S D S ST S D E
Sequence of area symbols
Area Sensor

Fig. 15 To detect an activity the user must (a) specify a model of the areas and events of interest, and then (b) specify a hierarchy of Probabilistic Context Free Grammars (PCFG) that specify the activity that needs to be detected.

The basic interpretation blocks in this hierarchy are Probabilistic Context-Free Grammars (PCFGs) [11, 12] that can be either specified by the user or even automatically extracted from the collected data as shown in [9]. Through a simple high-level interface, users provide a collection of probabilistic rules that form a PCFG. This set of rules specifies one or more activities by enforcing a syntax on the recorded input stream of sensing features (for an example see Figure 15). This syntax takes into account spatial characteristics (detected sensing features and their sequences over time) as well as temporal characteristics. A flexible time abstraction layer we have designed and implemented [13], enables users to associate time information to the recorded sensing features on a per-grammar and on a per-feature basis allowing the definition of grammar specific spatiotemporal features. By parsing these sequences of spatiotemporal features, activity recognition at different levels of spatial and temporal granularity is achieved.

The grammar hierarchy interpretation framework has already been used in several home network deployments to automatically interpret the recorded stream of data and provide meaningful activity summaries [12, 13]. Its interpretation power has also been demonstrated by the successful detection of complex activities, such as the cooking activity [12].

5 Conclusions & Future Work

This paper described our up-to-date progress on a scalable system for monitoring elder activities in assisted living. Our problem consideration and deployment experiences have shown encouraging signs that fine-grained monitoring for providing services will be possible in the near future. To achieve that one needs high precision sensors for localizing people, preferably without requiring them to wear sensors. Furthermore, we have discovered that there is a lack of synergy between learned and predefined models. Our work up-to-date has demonstrated that the two model types are complementary, and in order to deploy an effective system the two models should work together in close coordination. This and the development of intelligent motion discriminative sensors will become the focus of our future work.

Acknowledgements This work was partially funded by the National Science Foundation under projects CNS 0626802 and CNS 0751513. Any opinions, findings and conclusions or recommendation expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation (NSF).

References

1. Enalab. the behaviorscope web (bscopeweb) portal, http://bscope.eng.yale.edu, 2008.
2. Living independently group inc. the quietcare system. http://www.quietcare.com, 2008.
3. A. Bamis and A. Savvides. Stfl: A spatiotemporal filtering language with applications in assisted living. In *under submission*, 2008.
4. S. Consolvo, P. Roessler, and B. Shelton. The CareNet Display: Lessons Learned from an In Home Evaluation of an Ambient Display. *UbiComp 2004: Ubiquitous Computing: 6th International Conference, Nottingham, UK, September 7-10, 2004: Proceedings*, 2004.
5. D. Cook and S. Das. How smart are our environments? An updated look at the state of the art. *Pervasive and Mobile Computing*, 3(2):53–73, 2007.
6. EATON's Home Heartbeat. http://www.homeheartbeat.com.
7. European Commission's 6th Framework Programme: Service-oriented Programmable Smart Environments for Older Europeans (SOPRANO).
8. N. M. Gil, N. A. Hine, J. L. Arnott, J. Hanson, R. G. Curry, T. Amaral, and D. Osipovic. Data visualisation and data mining technology for supporting care for older people. In *Assets '07: Proceedings of the 9th international ACM SIGACCESS conference on Computers and accessibility*, pages 139–146, New York, NY, USA, 2007. ACM.
9. D. Lymberopoulos, A. Bamis, and A. Savvides. Extracting spatiotemporal human activity patterns in assisted living using a home sensor network (under review). *Sensor, Mesh and Ad Hoc Communications and Networks, 2008. SECON '08. 5th Annual IEEE Communications Society Conference on*, 2008.
10. D. Lymberopoulos, A. Bamis, and A. Savvides. A methodology for extracting temporal properties from sensor network data streams. In *under submission*, 2008.
11. D. Lymberopoulos, A. S. Ogale, A. Savvides, and Y. Aloimonos. A sensory grammar for inferring behaviors in sensor networks. In *IPSN '06: Proceedings of the fifth international conference on Information processing in sensor networks*, pages 251–259, New York, NY, USA, 2006. ACM.
12. D. Lymberopoulos, T. Teixeira, and A. Savvides. Detecting patterns for assisted living using sensor networks: A case study. *Sensor Technologies and Applications, 2007. SensorComm 2007. International Conference on*, pages 590–596, 14-20 Oct. 2007.
13. D. Lymberopoulos, T. Teixeira, and A. Savvides. Macroscopic human behavior interpretation using distributed sensor networks. *To appear in Proceedings of IEEE*, 2008.
14. D. Malan, T. Fulford-Jones, M. Welsh, and S. Moulton. CodeBlue: An Ad Hoc Sensor Network Infrastructure for Emergency Medical Care. *International Workshop on Wearable and Implantable Body Sensor Networks*, 2004.
15. M. Pollack. Intelligent Technology for the Aging Population. *AI Magazine*, 26(2):9–24, 2005.
16. E. Tapia, S. Intille, and K. Larson. Activity Recognition in the Home Using Simple and Ubiquitous Sensors. *Pervasive Computing: Second International Conference, Pervasive 2004, Linz/Vienna, Austria, April 18-23, 2004: Proceedings*, 2004.
17. T. Teixeira, E. Culurciello, J. H. Park, D. Lymberopoulos, A. Barton-Sweeney, and A. Savvides. Address-event imagers for sensor networks: evaluation and modeling. In *IPSN '06: Proceedings of the fifth international conference on Information processing in sensor networks*, pages 458–466, New York, NY, USA, 2006. ACM.

18. T. Teixeira and A. Savvides. Lightweight people counting and localizing in indoor spaces using camera sensor nodes. *Distributed Smart Cameras, 2007. ICDSC '07. First ACM/IEEE International Conference on*, pages 36–43, 25-28 Sept. 2007.
19. The BehaviorScope project. http://www.eng.yale.edu/enalab/behaviorscope.htm.
20. A. Wood, G. Virone, T. Doan, Q. Cao, L. Selavo, Y. Wu, L. Fang, Z. He, S. Lin, and J. Stankovic. ALARM-NET: Wireless Sensor Networks for Assisted-Living and Residential Monitoring. *University of Virginia Computer Science Department Technical Report*, 2006.
21. A. S. Yu, A. Bamis, D. Lymberopoulos, T. Teixeira, and A. Savvides. Personalized awareness and safety with mobile phones as sources and sinks. In *International Workshop on Urban, Community, and Social Applications of Networked Sensing Systems (UrbanSense08)*, 2008.