Accurate Estimation of Indoor Travel Times – Learned Unsupervised from Position Traces

Thor S. Prentow
Aarhus University
prentow@cs.au.dk

Henrik Blunck
Aarhus University
blunck@cs.au.dk

Mikkel B. Kjærgaard
Aarhus University
mikkelbk@cs.au.dk

Allan Stisen
Aarhus University
allans@cs.au.dk

Kaj Grønbæk
Aarhus University
kgronbak@cs.au.dk

ABSTRACT

The ability to accurately estimate indoor travel times is crucial for enabling improvements within application areas such as indoor navigation, logistics for mobile workers, and facility management. In this paper, we study the challenges inherent in indoor travel time estimation, and we propose the InTraTime method for accurately estimating indoor travel times via mining of historical and real-time indoor position traces. The method learns during operation both travel routes, travel times and their respective likelihood—both for routes traveled as well as for sub-routes thereof. InTraTime allows to specify temporal and other query parameters, such as time-of-day, day-of-week or the identity of the traveling individual. As input the method is designed to take generic position traces and is thus interoperable with a variety of indoor positioning systems. The method's advantages include a minimal-effort setup and self-improving operations due to unsupervised learning—as it is able to adapt implicitly to factors influencing indoor travel times such as elevators, rotating doors or changes in building layout. We evaluate and compare the proposed InTraTime method to indoor adaptions of travel time estimation methods for outdoor navigation. Our extensive evaluation uses datasets collected in real-world hospital work environments. InTraTime is deployed at a hospital as an online system, demonstrating that it learns automatically and in real-time travel times as position traces are collected within the building complex. Results indicate that InTraTime is superior with respect to metrics such as deployment cost, maintenance cost and estimation accuracy, yielding an average deviation from actual travel times of 11.7 %. This accuracy was achieved despite using a minimal-effort setup and a low-accuracy positioning system. Furthermore, we evaluated InTraTime also when using in place of the simple positioning system an almost twice as accurate alternative system. The results show that improvements in the positioning accuracy will further improve the travel time estimation, but only slightly, thus also confirming InTraTime's low requirements on the underlying positioning system.

MOBIQUITOUS 2014, December 02-05, London, Great Britain

DOI 10.4108/icst.mobiquitous.2014.258020

Categories and Subject Descriptors

H.4.2 [**Information Systems Applications**]: Types of Systems—*Logistics*

General Terms

Algorithms, Experimentation, Measurement, Performance

Keywords

Travel Time Estimation, Trajectory Analysis

1. INTRODUCTION

Estimation of travel times for indoor settings has many potential applications, such as expected arrival times for indoor navigation, task scheduling for logistics for mobile workers, airport information systems, and indoor traffic bottleneck detection for facility management. Today's solutions are limited to statically estimating travel times solely via estimating walking distances. This traditional methodology fails to address four main challenges, which are further elaborated on in Section 3: $\mathcal{C}1$) distance estimation requires laborious, heavy-weight modeling of the covered buildings to be accurate [12]; $\mathcal{C}2$) the travel time estimation may still be inaccurate if the route, obstacles or speed of the respective target is not realistically modeled; $\mathcal{C}3$) changes in the covered environments will require a manual remodeling to adapt the travel time estimates; $\mathcal{C}4$) indoor travel allows for a comparatively large freedom of movement: the segments along which users travel are primarily locations, such as entrance halls, that have a spatial extent, and that allow for many ways to traverse it and for many locations to traverse to; this complexity needs to be modeled appropriately, and it stands in contrast to car travel, which primarily occurs on simple road segments. While methods are available for computing outdoor travel time estimates reliably [3, 19, 6, 2] they cannot easily be applied to indoor settings, as they fail to address the challenges $\mathcal{C}1$-$\mathcal{C}4$. Furthermore, no methods specifically for indoor travel time estimation which address these challenges exists.

In this paper, we propose and evaluate a novel approach for indoor travel time estimation to overcome the challenges mentioned above. To this end, we present the InTraTime method that provides such estimates—given a start and destination

pair in a given building complex— via mining of historical and real-time position traces in an unsupervised manner. The method overcomes the above challenges $\mathcal{C}1$-$\mathcal{C}4$, as it i) does not require extensive building modeling, ii) does not rely on realism of models of travel speeds, iii) automatically detects changes in expected travel times via incorporation of data collected in real-time and iv) models the multiple ways in which locations with a spatial extent are traversed. As InTraTime's estimations are based on empirical data, the method implicitly learns the impact on travel time of complex environment and route elements, such as elevators, stairs, doors, or other obstacles. The method learns during operation in an unsupervised manner travel routes and their likelihood — both for complete routes as well as for the route parts these are comprised of. As additional benefit the InTraTime method also learns the time-wise shortest paths through the building among locations. Finally, the method allows easily for further adjusting, e.g. for personalized travel time estimates via mining primarily position traces from the current target in question. In the evaluated InTraTime deployment, position trace data is gathered using the multiple sensor capabilities of modern smartphones, including Wi-Fi for indoor positioning and accelerometers to detect motion to filter out stationary sequences of collected position traces.

The contributions of this paper are twofold. Firstly, for providing accurate indoor travel time estimations with minimal setup and maintenance costs we present a self-learning method which mines historical and real-time position traces from mobile devices. The resulting method InTraTime has low infrastructural requirements: only an indoor positioning system of low (or higher if available) accuracy is required. Secondly, we evaluate the proposed InTraTime method by datasets consisting of position traces and travel times from both a university department building complex as well as a large hospital complex. For comparing the InTraTime results with competing methods, we developed two additional methods as adaptions of known methods for outdoor travel time estimation [19]. We show that InTraTime is superior with regards to setup costs, maintenance costs and accuracy, and that travel times are predicted accurately with an average absolute error of 15 seconds which corresponds to a average deviation of 11.7 % from actual travel times. These results are achieved with no prior information on building model layout provided to InTraTime, and with a simple positioning system that yields a comparatively low mean positioning accuracy of $30m$. We show that when this positioning accuracy is improved, the travel time estimates improve accordingly. In addition, we deploy and evaluate InTraTime as an online system which receives position traces from targets carrying smartphones while moving around a building.

The results obtained suggest that InTraTime is an easy means to provide accurate travel time estimates and that it can benefit applications which yet rely on travel time estimation based on walking distances. Specifically, InTraTime can provide up-to-date current travel time estimates, or estimates for, e.g., specified times of day, through querying a webservice. Ideal application scenarios benefiting from InTraTime are thus characterized by that i) emperical travel data is easily obtainable, ii) accurate travel time estimates are essential, but may depend on current contexts, and iii) querying InTraTime, including the specifying of origin and destination, is done with no or little overhead for the user, i.e either implicity, or highly intuitively. Such scenarios include, among others, automatic scheduling and selection of work tasks in hospitals, and real-time individual time-of-arrival estimates for indoor travels in e.g. airports or large shopping malls.

2. RELATED WORK

The problem of travel time estimation has been well studied for outdoor settings, where travel time estimation is useful in areas such as car-navigation and bus arrival prediction. The Easytracker system detects bus routes and computes travel times and arrival times for buses on these routes based on collected GPS-traces [3]. Pfoser et. al. produce dynamic travel time information based on GPS traces from fleet management applications [19]. Janecek et al.[6] describe methods for estimating travel times on a highway based on cellular mobility data. Several similar systems have been developed, providing travel time estimates or traffic condition information based on position traces from GPS, GSM, or static road-sensors [22, 20, 18, 21, 4]. Balan et al. [2] relies on historical data for travel time estimation, but in contrast to InTraTime, they use only historical travels on the very same complete route for which an estimate is needed. Common for the above approaches is that they cannot directly be applied for low-effort indoor travel time estimation, as they assume either precise positioning in the form of GPS traces, or that a road network model is available. Another line of work tries to automate the construction of indoor location models and walking distances at various detail levels, which then can provide the indoor equivalent of road network maps [11]. One example is the system presented by Du et al. [5] that based on a depth camera constructs indoor location models. However, such systems still require a human in the loop and an authorized person to walk around the area to build an indoor model, and this step will take a considerable amount of time, e.g., to cover a large hospital. Furthermore, the constructed model will be prone to the same issues as manually constructed location models in regards to estimating travel times. Another example is presented by Alzantot et al. [1], based on using inertial sensors; however, also in this case they do not consider the estimation of travel times and depend on activity recognition which has issues in regards to generalizability across users [16].

For indoor applications GPS-positioning is largely unreliable [10]. Instead a large range of indoor positioning methods may provide positioning data to the method proposed here. These cover several different types of mediums and technologies as covered by LaMarca et. al [14]: radio-based, e.g., WiFi, Bluetooth, RFID or ultra-wide-band technologies, sound, e.g, hearable or ultrasound and light, e.g., infrared light. Furthermore, recent developments also couple such technologies via sensor fusion with the input from on-target sensors measuring heading and movement.

3. CHALLENGES FOR PRECISE INDOOR TRAVEL TIME ESTIMATION

As outlined in Section 1, several challenges $\mathcal{C}1$-$\mathcal{C}4$, not sufficiently addressed by existing approaches, must be overcome for computing and providing precise indoor travel time estimates on basis of real-world data. This section further studies these challenges and motivate why precise travel time estimates are needed in real-world indoor applications.

Challenge $\mathcal{C}1$: Existing methods fall short w.r.t. challenge $\mathcal{C}1$, i.e., they depend on laborious, heavy-weight modeling of the covered buildings. These methods deduce travel time estimates from calculating walking distances—for which a building model is needed; the only alternative being time-consuming manual measurement of distances throughout the building. For indoor building complexes digital maps are often either unavailable, non-digital, or do not contain the necessary information to easily or automatically determine walkable paths. Manually measur-

ing walking distances, either by map or by foot, does not scale well and becomes unfeasible for large building complexes. This motivates methods for precise travel time estimation without requiring knowledge of travel distances.

Challenge $\mathcal{C}2$: To illustrate challenge $\mathcal{C}2$, i.e. that travel time estimation may still be inaccurate even when suitable building models are provided, Figure 1 depicts travel times collected from four persons walking various routes through a building containing common obstacles such as stairs and doors. The data for specific persons show that the effective walking speed varies significantly: E.g. for person 1, the standard deviation in walking speed is $0.14m/s$ from a mean speed of $1.19m/s$. Extrapolating to the distance covered through, e.g., a hospital orderly's whole work shift which yields ca. 6 hours of transporting patients and equipment, this deviation implies that the actual time to cover it with speeds within the standard deviation ranges from 5:22 hours to 6:48 hours, a difference of 85 minutes. By accounting for all speeds falling within a 95% confidence interval, this difference further increases to 144 minutes. This difference in walking speed at various locations for a person, here termed *area-specific speed-variance*, is caused by different obstacles such as stairs, doors and the layout of the building which can cause both increases and decreases in walking speed. Further differences may arise from temporal obstacles, such as doors locked after-hours. Thus, precise travel time computations must take into account observable variances in speed across different paths, situations, etc. Furthermore, as the figure as well as further studies suggest [17], the overall walking speed of a person is strongly affected by factors such as personal preference, fitness and age. Indeed, we see in the figure an increase in the standard deviation of walking speeds to $0.15m/s$ when including all persons. We term this factor the *person-specific speed-variance*, as it for a person will affect travel times independently of the specific location. In order to be precise, travel time computations must take into account both the area-specific and person-specific speed-variance.

Challenge $\mathcal{C}3$: The third challenge $\mathcal{C}3$ captures that changes frequently occur in all of the aforementioned factors and that those changes may influence travel times. On a long-term scale buildings may be modified, e.g., new buildings are added, hallways are changed or doors are removed or added. When using travel time estimates based on static distance estimates, these would have to be manually updated in such cases. This causes a large maintenance overhead to keep these estimates up to date. Factors in the person-specific speed-variance changes over time, e.g., preferred walking speed changes with age or fitness level. On a shorter timescale people may, e.g., obtain a temporary injury causing them to move slower. Factors in area-specific speed-variance similarly change both on a small and large timescale, e.g., escalators that are shut off at night, doors that are locked or elevators that break down. In order to automatically take all these changes into account without a large maintenance overhead, real-time position traces must be taken into account so that up-to-date travel time information can be extracted.

Challenge $\mathcal{C}4$: The fourth challenge $\mathcal{C}4$) constitutes the appropriate modeling of the comparatively complex freedom of movement in indoor environments. This complexity can be addressed by modeling the spatial extent of a room and the several ways (and corresponding travel times) for traversing it. For building-model-based travel time estimation this can be addressed by calculating the (shortest) travel distances for each entry/exit pair within the room. We will detail in Section 4.2 how to model this freedom of movement for basing travel estimates not on building modeling but on recorded historical data.

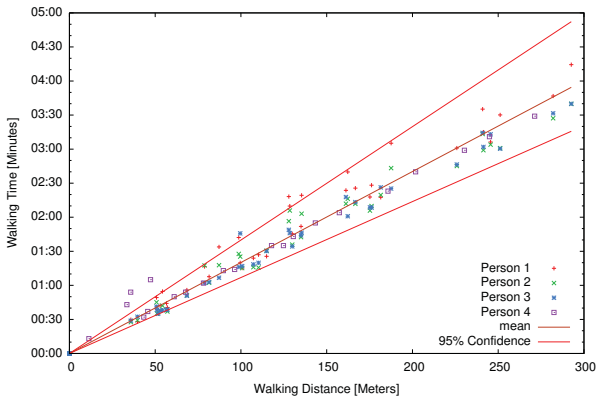

Figure 1: Empirically determined travel times

To illustrate the need for accurate travel time estimates we discuss in the following a task scheduling problem example situated in a simplified hospital containing just four locations: operation room, patient ward, intensive ward and maintenance room. Three types of task are to be scheduled for and performed by two orderlies: T1: moving patients from patient ward to operation ward and back again (after surgery); T2: performing maintenance jobs, of 30 minutes each, at the maintenance room. T3: turning over immobile patients in the intensive ward—which takes little time but requires two orderlies. Figure 2(top) pictures a corresponding example schedule, created using a simple travel time estimation method based only on distance between the wards (here assumed to be 1.2km respectively), and assuming identical walking speed (resulting in 15min estimated travel time each). In the simple schedule, all T1 (resp. T2) tasks are assigned to orderly O1 (resp. O2). The cooperative tasks of type T3 are assigned to both O1 and O2. Figure 2(top) shows the scheduling as intended: no one's time is wasted waiting for patients or staff members, and the operation ward is utilized at all times. In contrast, Figure 2 (bottom) shows factual waiting periods (in red) when area-specific and person-specific speed-variances apply within the limits given in Figure 1 as follows: sv1) The travels from/to patient ward to/from operation room and intensive ward require the opening of doors and the use of elevators, leading to a 3 minute increase in travel time to 18 minutes each. sv2) O2 has a faster than average walking speed, a person-specific speed-variance, leading to 12 instead of 15 minutes travel time per 1.2km (i.e., between maintenance room and intensive ward. Various waiting times result: for (initially) O2, due to O1 arriving later than scheduled at the intensive ward; similarly, as O1 arrives late with the patient, surgery is delayed, causing waiting time for everyone involved, e.g., patients, doctors and nurses. Over the course of the shift the total waiting time, just for O1, adds up to 2.4 hours. Similarly, the operation room is not fully utilized, and surgeries are delayed—the last one by 90 minutes. Such scheduling flaws (which in the real-world easily becomes more complex and harder to alleviate than in this simplified example) as well as the resulting waste of time could be prevented by more accurate travel time estimation methods which take into account both person-specific and area-specific speed-variance, and which allow for a more accurate and adaptive scheduling.

4. INTRATIME - A METHOD FOR TRAVEL TIME ESTIMATION

In this section, we present the elements of the InTraTime method for accurately estimating travel times in indoor building

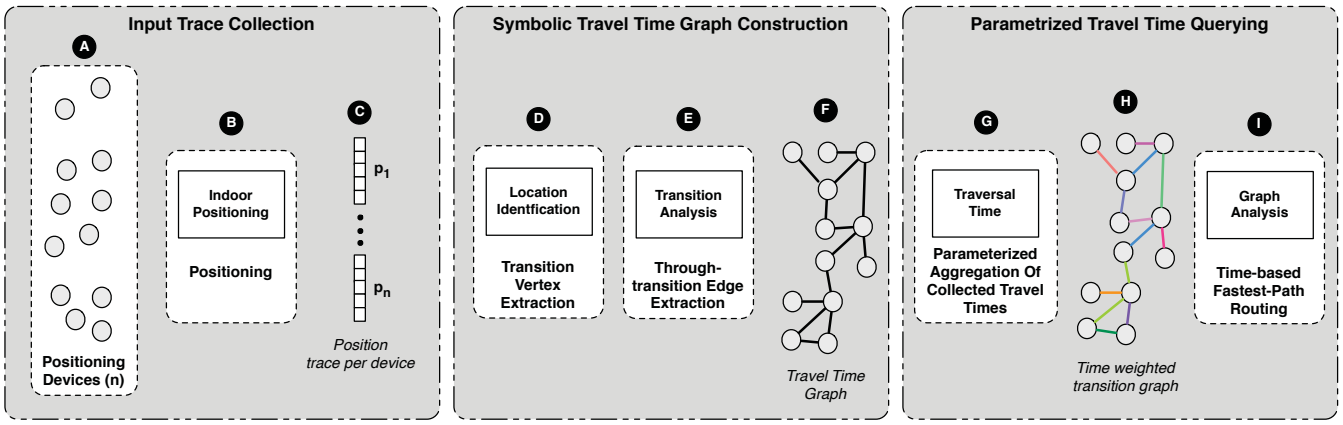

Figure 3: Overview of the InTraTime system for travel time estimation.

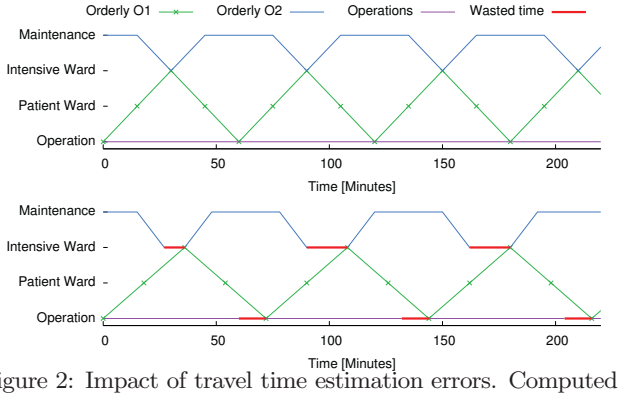

Figure 2: Impact of travel time estimation errors. Computed schedule (top), and actual schedule(bottom).

complexes via unsupervised learning from historical position traces. The InTraTime method consists of four main elements:

- Online updating from real-time position traces
- Symbolic travel time graph construction, independent of building model or location
- Parametrized aggregation of collected travel times
- Time-based fastest path routing

Through the combination of these elements, the InTraTime method is able to overcome the challenges presented in section 3, by producing accurate travel time estimates with no prior knowledge of building models and by automatically incoorporating factors and changes which may influence travel times. To provide an illustrative overview of the InTraTime method, Figure 3 pictures the flow of computation, separated into three areas: 1) input trace collection, 2) symbolic travel time graph construction, and 3) parametrized travel time querying. As part of input trace collection, the target entities move around within the space (A) while being positioned and tracked by a tracking component (B) which provides InTraTime with a position trace for each entity (C). The symbolic travel time graph $\mathcal{G}_T$ is meanwhile constructed and incrementally expanded and improved from the traces as follows: The position traces are analysed to i) discover symbolic locations that are then added as vertices to $\mathcal{G}_T$ (D), and to ii) detect possible direct and through-transitions that are then added as edges of $\mathcal{G}_T$ (E). The resulting symbolic travel time graph $\mathcal{G}_T$ (F) can then be queried for parametrized travel time estimates as required by applications. In order to perform these queries, weights are first computed for each edge in $\mathcal{G}_T$ as the weighed median of the collected traversal times (G). The weights are computed according to parameters, e.g. focusing on a specific time of day or a specific individual, resulting in a time-weighed travel time graph (H). The weighing of edges with traversal times allows the total parametrized travel time for the queried start, end pair of locations to be estimated by time-wise shortest path routing for the obtained weighted travel graph (I).

To help intuition regarding the graph structure underlying InTraTime, Figure 4 shows for the investigated hospital complex the complete travel time graph for a week's position data gathered from ten tracked targets. Furthermore, in Figure 5 is visualized a calculated fastest path (associated with the predicted travel time) for each the hospital and the university department. The nodes in the respective routes symbolize the symbolic locations, which the fastest route passes through. The nodes themselves are placed at the positions of the Wi-Fi access points associated with the respective locations.

4.1 Input Trace Collection

To fulfill InTraTime's task of providing travel time estimates we assume that the respective indoor space is covered by a suitable location model. This model can either be a symbolic one—in which locations are identified by semantic names—and/or a coordinate-based one [12]. A location model in the simplest form may be a bidirectional mapping between Wi-Fi access-point identifiers and room names. A semantic location model may be built simultaneously with the collection of position data for travel time estimation, using e.g. automated or user-driven methods [8, 9]. The location model is employed to identify the vertices in the symbolic travel time graph $\mathcal{G}_T$ which represent the coordinates or location identifiers of start and end locations

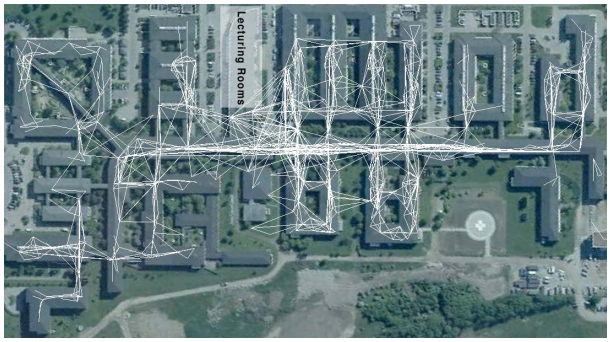

Figure 4: Travel graph constructed from collected position traces at hospital complex. Map: Google, Aerodata International Surveys

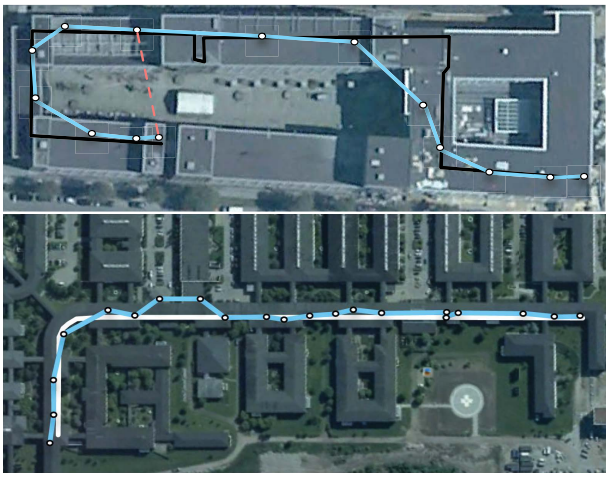

Figure 5: Visualization of routes corresponding to calculated fastest travel times, showing traversed locations (white, at corresponding access point locations), and location transitions (blue). University department (top), hospital complex (bot). Map: Google, Aerodata International Surveys

of travels. Thus, we require the location model, and also the utilized positioning system, to cover only such locations.

4.1.1 Online Updating from Real-time Position Traces

The input position traces for InTraTime consist of a sequence of timestamped position estimates for each given tracked mobile entity. Each position estimate consists of a timestamp T, an entity id I and a position P — either resembling coordinates or a symbolic location. Each given input position trace for a given device u is simplified in an online fashion into a sequence of *location intervals*, where each location interval consists of consecutive estimates yielding the same position. The sequence which is passed on to the next step thus has the form $[(T_0^{start},T_0^{end},u,P_0),(T_1^{start},T_1^{end},u,P_1),...]$ where $\forall i: P_i \neq P_{i+1}$.

4.1.2 Jitter Removal

This step addresses noise in the form of the erroneously reported rapid transitions back and forth between two locations. Such erroneous reporting is a common result of inaccuracies occurring in various positioning systems, e.g., when the target is moving close to the border between two locations. Such noise may impact the travel time estimates heavily, as it—if not addressed—suggests rapid travel transitions between adjacent locations in the historical data. We found that mitigating such noise sources worked well via simply discarding from a given position trace the 'back and forth' location changes which occurred during a time span smaller than an (empirically determined) threshold.

4.2 Symbolic Travel Time Graph Construction

As position traces are collected and preprocessed they are used for construction of the symbolic travel time graph. The graph is updated in an online fashion as the new position data is collected.

4.2.1 Travel Time Graph Representation

Generally, for routing applications a *transition graph* $G(V,E)$, is used where vertices in V correspond to locations of the covered space, and where each edge $e=(v_1,v_2)\in E$ resembles a connection to be used by travelers for direct transitioning between the locations represented by two vertices v_1 and v_2. Travel times for a travel, given start and end location v_s and v_e, can then

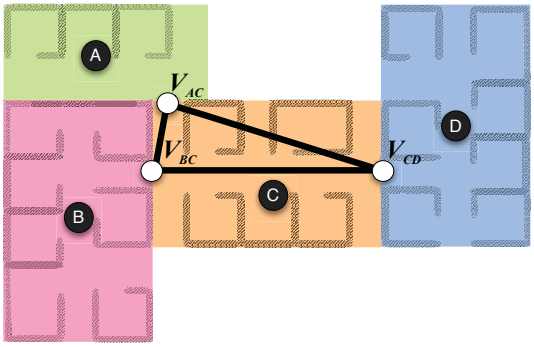

Figure 6: Travel time graph, with edges representing through-transitions

be estimated based on travel times for travels along paths in G from v_s to v_e. For the purpose of InTraTime however, we choose to employ a modified variant of the transition graph, which we term *travel graph* to represent *through-transitions* of the form $t=(v_1,v_2,v_3)$, denoting the indirect, 1-link transitions between two locations v_1 and v_3 through a third one v_2. In Figure 6 the travel graph $\mathcal{G}_T$ for an example space of 4 locations are given. We use this, more complex, travel graph instead of the transition graph, to address challenge $\mathcal{C}4$: The chosen graph modeling addresses the freedom of movement within rooms, specifically the different ways, i.e. entry-exit-pairs for traversing it and the corresponding travel times.

The vertices in $\mathcal{G}_T$ are the edges of the transition graph G, i.e. the set of all direct transitions between locations in the modeled space. The edges in $\mathcal{G}_T$ are then the through-transitions in the modeled space. Thus, for instance a recorded transition from A through C to B provides three items of information used in constructing the travel graph represented in Figure 6: 1) A is connected to C, 2) C is connected to B, and 3) B has been reached from A through C in x seconds. From this information the two vertices V_{AC}, V_{BC} as well as the edge between them can be constructed. Addressing the challenge $\mathcal{C}4$, the travel time graph allows now to estimate a travel time for a through-transition $t=(v_1,v_2,v_3)$ through a location v_2 more accurately than when using the transition graph—because it takes into consideration also the used entry and exit. This can be seen exemplary in Figure 6, where, e.g., through-transitions (A,C,B) and A,C,D use different exits to leave C, and thus also different paths and associated travel times for traversing the location C.

This choice to model and associate travel data to through- instead of to direct transitions is also highly beneficial in the case that a rather coarse positioning system is utilized for recording travel data: For direct transitions (A,C) and (C,D) precise start and end positions and times in A,C, and D, respectively may not be known, if those locations are atomic—as is the case for the simple positioning scheme described above: taking a direct transition is detected as an instantaneous event and thus travel times cannot be associated with this transition. For a through-transition, e.g., $t=(A,C,B)$, on the other hand, the travel time through C can be defined as the duration of the time interval for which C is reported by the positioning system as the traveler's location along his given position trace.

It follows that a modeling scheme that estimates travel times for individual through-transitions will allow for more accurate travel estimates than possible with modeling only direct transitions [12]. Furthermore, using a graph-based approach that models travel paths as sequences of location transitions has advantages over, e.g., using only end-to-end travel times: Thereby,

estimating a travel time for given start and end location does not require prior travel between those exact endpoints; instead the estimate may be computed by combining subroutes of other previously recorded routes. This allows for much greater utilization of the collected data and thus allows for producing travel time estimates between a large amount of locations from only a sparse amount of data.

4.2.2 Transition Vertex Extraction

At the arrival of each new location interval within a position trace for a given tracked entity, the interval is processed together with its two preceding location intervals, as illustrated here for the chronologically estimated locations A, C and B: (T_A^{start},T_A^{end},u,A),(T_C^{start},T_C^{end},u,C),(T_B^{start},T_B^{end},u,B). From the arrival of the newest location estimate B it follows that C is connected to B. Thus, the vertex V_{CB} is added to the set of vertices E of $\mathcal{G}_T$, if it does not already exist. In addition, we create for each location P two vertices, $V_{P-start}$, V_{P-end}, which are used for representing start, respectively end, points of travel events for which travel time estimates are to be calculated via traversal analyses of the travel time graph.

4.2.3 Through-transition Edge Extraction

With each incoming location interval, as part of a position trace, the resulting triple of the three most recent location intervals, constitutes travel time information about the completed traversal of a location. Every edge in the travel time graph represents such a possible transition through a location; e.g., an edge in $\mathcal{G}_T$ from V_{AC} to V_{CB} constitutes that it is possible to move through C when going from A to B. Each edge e holds the set $\mathcal{T}_e$ of historically recorded time spans for traversing it—one for each travel incident in which the edge has been traversed. E.g., receiving the three location intervals for locations A, C and B implies that when moving from A through C to B, the time spent between leaving A and arriving at B is $T_{ABC}=T_B^{start}-T_A^{end}$. We therefore create in the travel time graph an edge between vertex V_{AC} and V_{CB}, if one does not already exist, and add T_{ACB} to $\mathcal{T}_e$. Note, that edges in $\mathcal{G}_T$ are directed. This allows us to reflect in $\mathcal{G}_T$ differences in travel time estimates, e.g., for staircases, and for in practice solely one-directional transitions, e.g., through emergency exits. Note also, that an additional motivation for using a travel time graph, yielding through-transitions as edges, is that a transitivity of direct transitions does not hold in general: While in Figure 6 location C is reachable from A, and D is reachable from C, and D is transitively reachable from A, the latter transitivity would not hold in the following case: the symbolic location C may contain — inconveniently — a dividing, non-penetrable wall, which separates it into two areas, which are not directly reachable from another. To reflect such a case, the change in the travel time graph would be the omittance of the through-transitions (V_{AC},V_{CD}) and (V_{BC},V_{CD}). Note also, that the existence of such divided symbolic locations is not far fetched in areas of low access point density. Finally, additional edges in $\mathcal{G}_T$ are created from and to each potential travel start and destination location to the graph nodes corresponding to direct transitions from that location: E.g., for location A a start node $V_{A-start}$ and end node V_{A-end} are created, and edges connecting them to node V_{AC}.

4.3 Parametrized Travel Time Queries

The constructed travel graph is queried as required by applications in order to compute travel time estimates. These queries consist of a start and an end location, as well as optional parameters such as time of day, day of week, or the specific person for which travel time estimates should be computed.

4.3.1 Parametrized aggregation of collected travel times

For each edge e in $\mathcal{G}_T$ we calculate a weight w_e, representing the estimated travel time for traversing that edge, i.e. the represented through-transition. w_e is computed as an aggregate of the set $\mathcal{T}_e$ of recorded traversal time spans (together with the individual recorded traversing position traces) that we associate with the through-transition, i.e. with the edge e. As aggregation function we chose the median in favor of, e.g., the mean because of its the better resilience of the former to outliers. Outliers, specifically those containing unusually large time spans, may occur frequently, when, e.g., a travel event is interrupted for unknown reasons and the tracked device is standing still for a long time in a location, while the system assumes uninterrupted travel, i.e., it to be moving.

InTraTime also supports parametrized travel time queries, where parameters may specify, e.g. a time of day, or to which extent more recently recorded travel instances should be weighed higher than older ones. To support such queries we use in place of the basic median the weighed median [15] of the traversal times. Weights are assigned using a weight function $w(t)$, given a traversal time t annotated with additional information such as time of collection and the entity for which it was collected. Individual weight functions can be combined and be used individually to weigh according to given query parameters, e.g., the recentness of travel times, the traversal times collected on specific week days, or traversal times computed for a specific persons or person groups. We choose a weighing of measurements over simply filtering away those not matching specific parameters because the latter approach relies heavily on the—often overly optimistic—assumption that sufficent travel data matches the exact query parameters. In reality though data may not be available for some parts of a route for specific parameters—as it is unlikely that all of a building complex has been traversed by travels fulfilling all (of potentially many and restrictive) given query parameters.

4.3.2 Time-based Fastest-path Routing

In order to calculate a travel time estimate from a start location, represented by vertex $V_{P-start}$ in the weighed travel time graph, to a destination location, represented by V_{P-end}, it is assumed that the shortest route in regards to time will be taken. Thus, given the weighed travel time graph, in which the weight of each edge represents a time estimate for traversing that edge, travel time estimates can be computed by finding the shortest path between $V_{P-start}$ and V_{P-end} in relation to the graphs. This can be performed using standard shortest path graph algorithms, e.g., by using Dijkstra's algorithm for each query.

If the system is asked to output not only an estimated travel time, but also to deliver the suggested fastest route (which is associated with the produced travel time estimate), both outputs can be calculated by the same algorithms, and the latter can be presented to the user, e.g., in the form of a sequence of locations.

5. EVALUATION

This section evaluates InTraTime in regards to features provided as well as accuracy, comparing it also with alternatives paradigms for travel time estimation.

5.1 Dataset Collection

For the evaluation of methods we chose a simplistic and coarse positioning scheme, which utilizes solely existing Wi-Fi infrastructures: this scheme assumes only a mapping of Wi-Fi access point IDs and the elements of the location model. The current location of a target is then reported as the location associated with the access point which is currently received strongest by the target device. We determined the accuracy of this scheme experimentally to be approximately $30m$ in reported positions, through comparison to manually collected ground truth positions. By using this simplistic positioning scheme we can validate that the methods do not make further assumptions on the used positioning system and its accuracy. As the access points are placed with a mean distance of 14 meter, this approach is roughly equivalent to snapping positions to a irregular grid with a mean grid size of 14 meter. If alternatively a very fine-grained or continuous positioning method is used, snapping the position estimates to a similar sized grid should provide for similar results.

The data collection experiments were conducted at a medium-sized university department, covering 5250 m^2 per floor for in total four floors. An example route at the university is depicted in Figure 5 (a). Data was collected by three male participants using a Google Nexus 4 smartphone and walking along predefined routes both during and after office hours for providing realistic conditions. The phones ran sensor-logging software to log timestamped Wi-Fi signal strengths with a sampling rate of approximately 0.5 Hz. Ground truth was manually annotated at checkpoints for assessing the methods' accuracy in travel time estimation. Additionally data was labelled with the id of the collector. The nine routes at the university department had between 3-7 checkpoints each, totaling a 41 checkpoints. Each route had at least one change of floor level via stairs except for one floor change via elevator. The nine routes were chosen such that they covered regularly used routes spanning several connected buildings of the complex.

5.2 Overall Features of IntraTime

In the following, we describe the features of IntraTime via comparing it to alternative paradigms for travel time estimation.

5.2.1 Alternative Methods for Travel Time Estimation

To this end, we compare specifically against two methods, termed *EucTime* and *BModelTime*, as example instances of such alternative paradigms, and which in contrast to InTraTime estimate travel times based on estimating travel distances.

EucTime estimates travel time as the time t for travelling the euclidian distance d between start and end location with a speed s: $t = \frac{d}{s}$. Thus, EucTime serves as simple baseline method for travel time estimation, which illustrates which accuracy levels to expect when only very limited spatial information is used. The speed s can be assumed as, e.g., an historical average speed given by position traces. EucTime requires, in contrast to InTraTime, that the locations are given in coordinates from which euclidean distances can be computed, e.g., GPS or UTM coordinates. No further knowledge of building models or historical data is assumed.

BModelTime is designed as an indoor adaption of a method used commonly for outdoor travel time estimation, as described in Section 2: Travel time estimates are computed by predicting a route using an available road map by assuming that, e.g., the shortest path is used. In addition a speed is assumed, e.g., the speed limit of the roads used. Thus, BModelTime requires a digital building model containing the walkable paths, resembling the indoor equivalent of an outdoor road map. Then, as for EucTime, BModel deduces travel times by assuming the speed of the target, here along the calculated shortest path. BModelTime thus takes the layout and topology of a building complex into account when estimating travel times.

5.2.2 Feature Comparison

We compare the three described methods with regards to features and accuracy, as summarized in Table 1. BModelTime and EucTime are only able to adjust to the overall speed of a person, while InTraTime may also adjust to e.g. personal changes in speed for specific places in routes. In addition InTraTime is as the only method able to adjust to factors which are specific for e.g. time of day, such as doors which are locked at specific times. Due to the reliance on models of building layout, the setup and maintenance costs of BModelTime are high, while both EucTime and InTraTime can be deployed and maintained with only very litte effort, as they both assume only that an indoor positioning system is in place. However, as EucTime requires locations to be known in coordinates and not just symbolic locations, setup cost may be higher than for InTraTime.

With regards to the building layout of the deployment environment, InTraTime is self-learning in that it automatically detects the spatial connections between locations. BModelTime on the other hand requires a building model to be available, while EucTime does not take the building layout into account. However, as described in Section 3, precise digital building models are both hard to produce and costly to maintain, as this requires significant manual effort. Thus, in contrast to InTraTime BModelTime comes with a high deployment and maintenance cost. Furthermore, it cannot easily adapt to, e.g., sudden or temporary changes in building layouts, such as broken elevators or doors which are locked at night.

Table 1 also lists the accuracy of the individual methods given as deviation in percent from the actual recorded travel times as recorded by test persons in the university experiment. Despite InTraTime's obliviousness of the building layout it shows an error of 10.2 %. and thus an accuracy superior to EucTime (60 %) and comparable to BModelTime(8 %), even though BModelTime has full knowledge of the building layout.

5.3 Estimation Accuracy

In the following, we detail the evaluations of the methods' accuracy. In Figure 7 (top) for each of the 9 routes in the university department actual travel times are shown in green (as averages over several walks of the routes), and travel distance is shown in blue.The respective estimation errors of the methods are depicted with arrows — as the length of an arrow that points to the estimated travel time. The respective travel times and estimations shown are averaged over the three participants. In the case of InTraTime for each participant the historical data used for estimation was solely from recordings of the other two participants, in order to ensure proper separation between training and test data. Note, that for each arrow, solely in this plot, its length is given as the average over the magnitudes of the three estimation errors, and its orientation according to a majority vote among their signs. This evaluation setup insofar matches the use scenario that travel times are to be estimated for a person for which no historical data can yet be utilized, which would be expected to impact the results for InTraTime negatively, as no personalized data is available. The errors of BModelTime and EucTime have been computed using an overall walking speed as computed from

System	Self-learning	Adjusts to changes	Person-alizable	Adaptable to e.g. day-time	Setup & Main-tenance cost	Measured avg accuracy, when distance modeling	
						unavailable	available
BModelTime	✗	✗	(✓)	✗	High	✗	8 %
EucTime	✗	✗	(✓)	✗	Low	60 %	
InTraTime	✓	✓	✓	✓	Low	10 %	

Table 1: Feature and performance summary for InTraTime, in comparison to alternative paradigms for travel time estimation.

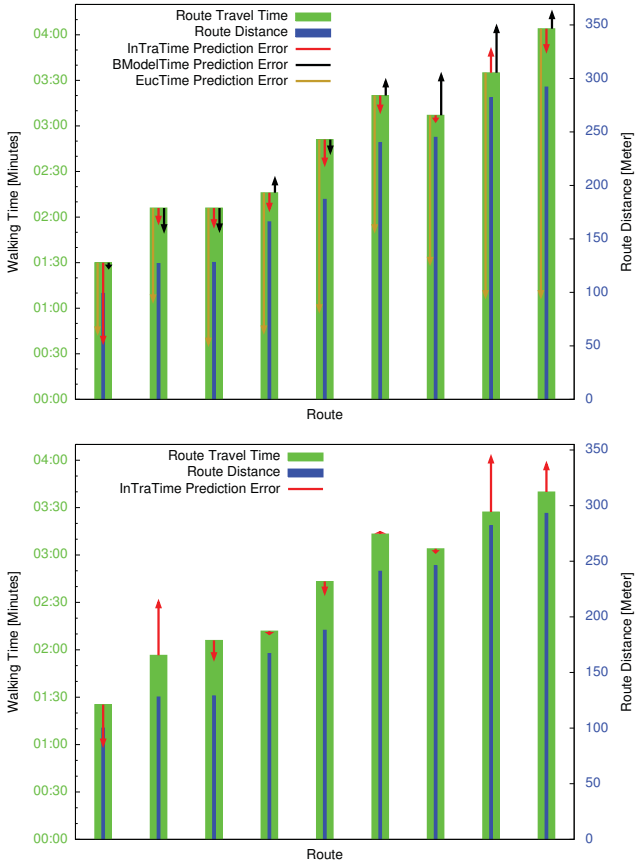

Figure 7: Distance, measured travel time travel time estimate error per route; based on all data (top) and on data solely from other routes (bottom).

position traces. The error values given in Table 1 are computed as the mean relative error of the routes for each method.

5.3.1 *Sources of Inaccuracy*

Overall, the figure indicates that InTraTime's estimation accuracy is more than comparable with the heavy-weight building model-based estimation approach of BModelTime — even though the latter has full knowledge about the route length. The evaluation though shows some outliers: Specifically for one of the three participants, the InTraTime estimate for the shortest of the routes was much too low—by about 50 percent. The reasons being, as made visible in Figure 5 (top) is an edge (in red) indicating erroneously a possible transition between two locations which cannot be reached directly from each other within the building complex. This false edge is likely caused by inaccuracies in the positioning system. As a result, the InTraTime system has concluded that a direct edge exists that connects the start and end node of the dotted line in the figure. Protecting against such error sources can be achieved by enhancing the filtering technique, described in Section 4.1.2; or by extending the aggregation technique, described in Section 4.3.1.

5.3.2 *Influence of Positioning Error*

In order to evaluate the influence from errors in position estimates on the travel time estimates computed by InTraTime, we additionally tested the centroid lateration approach for Wi-Fi positioning[13]. In this approach, the position of a device is computed as the weighted mean location of the access points which are received, using the received signal strength as weight. To keep the position estimates within buildings, the estimates are further snapped to the location of the closest access point. We evaluated the centroid lateration approach to have a mean accuracy of $15m$ in the university building, twice as accurate as the strongest access point approach. Figure 8 shows the resulting travel time estimates for both positioning methods. The figure shows that the travel time estimates computed based

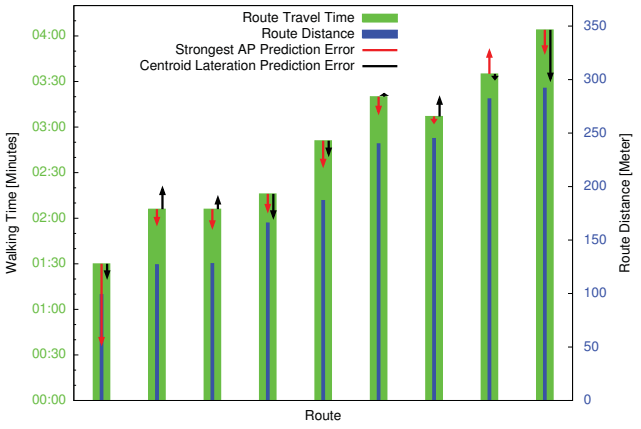

Figure 8: Travel time estimates computed on the basis of respectively strongest access point and centroid lateration positioning.

on positions from the centroid lateration method are comparable to those computed based on the strongest access point method. The slight improvement in mean error to 9.0% suggests that while InTraTime is able to function with low-accuracy positioning, it is also capable of utilizing increased accuracy.

5.3.3 *Performance for Sub-Routes-Only Data*

As it might happen for InTraTime that data from complete travels between a route's start and end is not available we evaluated also travel time estimation for a route on basis of historical data from only partial segments of the respective route. To this end, the presented evaluation for InTraTime has been repeated– but with prior exclusion of any historical measurements for travelling the complete route r for which estimates are to be calculated. The obtained results show only little loss in accuracy of 0.9 percentage point to a relative error in estimated travel time of 11.1 percent. This is mostly due to a structural bias, though: For most routes, the exclusion of end-to-end data increases the estimated travel time—on average by 9 seconds. This increase is mostly due to that from the collected historical segments, a direct route can in some cases not be assembled, and thus only a longer route and travel time is calculated. Note that this bias is diminishing with increasing amount and diversity of historical data.

5.3.4 *Automatic Adjustment to Changes*

In order to evaluate the ability of InTraTime to automatically adjust to changes in travel times through a building, we performed an evaluation where travel times were computed for evening travels through the university building. At the department off-hours, all external and internal doors are locked and require swiping a keycard and entering a PIN-code in order to open, which slows down travels in the building significantly. Figure 9 shows the results of this data collection and the estimated travel times. The columns show the actual travel times measured during respectively the evening and the day-time for the nine specified routes. The arrows again point to the estimated values, and thus show the error in travel time estimates for day and evening time. The day-time estimates have been computed with a weight function providing high weights to travel times collected at day-time (before 16:30 where doors are locked), while the opposite has been done for the evening estimates. The mean relative error of the evening estimates is 9.7% and thus comparable to those computed for day-time only data as described previously. The figures show that, even within these errors, the travel time estimates are adjusted for the longer travel times during the evening, and that the evening-weighed estimates are closer to the actual travel times than the day-weighed estimates. The same approach can be used for adjusting to unexpectedly occuring changes, by e.g. computing weights to data based on the time since collection.

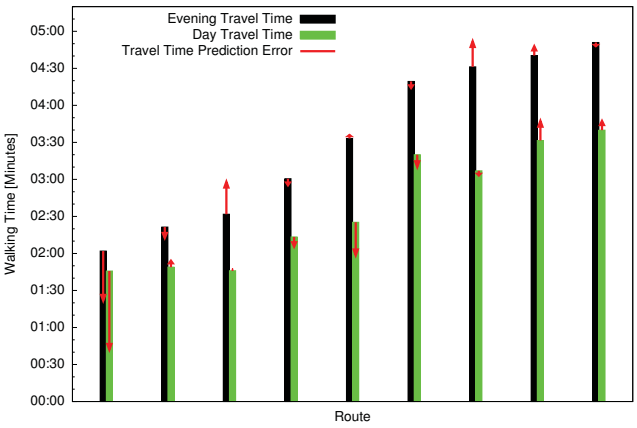

Figure 9: Actual and estimated travel times for travels during day-time and during evening hours, respectively.

5.4 Robustness and Scalability

We argue that the InTraTime method scales to complex buildings by using an additional dataset collected at the large hospital. This dataset was collected by giving non-clinical staff members at the hospital a smartphone with logging software. The staff carried the phone for a week during their normal work at the hospital. The work consisted of tasks spread across the hospital, such as transportation of goods and patients. Due to the nature of the collection method full ground truth with regards to travel times is not available. However, it is still relevant to consider if the travel time graph constructed by InTraTime from the data is sensible, e.g., contains only correct nodes and edges corresponding to possible travel paths. To do this we compare the travel time graph with the actual building layout, as depicted in Figure 4. From the figure we make the observation that in all the buildings that we know that the staff visits we have identified nodes. Buildings that we know that the chosen group of staff do not visit includes lecture rooms for students and visitors as highlighted in the figure.

The scalability of the computational effort was evaluated using a standard laptop PC with an Intel iCore i7 2.8GHz processor and 8GB RAM. Two input sets were considered: the one-week dataset from the hospital, and the smaller dataset from the university department. The hospital dataset revealed 560 distinct locations, 7810 direct transitions, i.e. nodes, and 47199 through-transitions, i.e. edges, in the travel time graph. In comparison, the university department data contained 60 distinct locations, 256 nodes, and 547 edges in its travel time graph. Computing travel time estimates for all pairs of end-locations took 44.17 seconds for the hospital data and 0.13 seconds for the university department data. Overall, the results as well as selective investigations into the data gave evidence for the following hypotheses: First, for all intents and purposes the number of edges in the travel time graph seems to be rather near-linear than quadratic in the number of nodes; the same seems to hold for the number of nodes with respect to the number of locations. Both respective ratios, though, are higher in the bigger dataset than in the smaller dataset. We hypothesize that this is not entirely due to the size of the dataset or the corresponding indoor space, but also due to the lower density of access points—resulting in a larger border area of the symbolic locations, and thus in a higher number of neighboring locations. Secondly, the running times per travel time query (resp. per graph edge) are about a factor 4 (resp. 5) higher for the bigger dataset than for the smaller one—suggesting that large data size comes with additional (i.e. super-linear) growth in running time, given the standard laptop PC setup utilized.

5.5 Online Operations Deployment

To further argue for the applicability, ease of deployment, and the fast learning of InTraTime in a real-world setting, we have implemented and deployed InTraTime as an online system which collects data and provides travel time estimates in real-time.

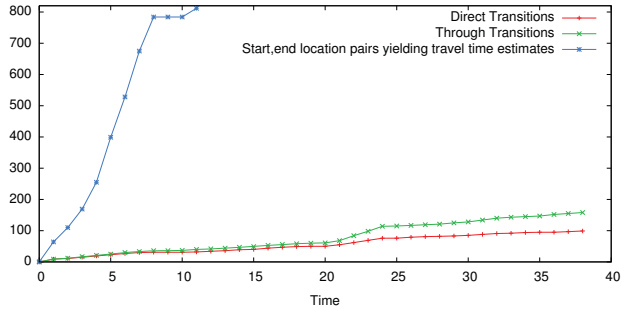

Figure 10: Increase over time of the number of transition vertices, through-edges and start,end location pairs for which travel estimates are computable.

For a respective evaluation, we instructed people supplied with LG Nexus 4 smartphones to move around the covered building for an hour. Figure 10 shows that the number of direct and through-transitions grow virtually linearly and they seem to still grow at the end of the test period. The coverage of InTraTime in terms of the start, end location pairs for which it can provide travel time estimates grows exponentially at first, but after ca. 30 minutes the increase is only linearly—correlating with that a convergence is approached where most locations are discovered, and where increasingly many pairs have at least one path connecting them in the travel time graph. This trend also shows in the query performance across the hour of data gathering: While at first the query times increase as the travel time graph grows, query time remains virtually constant ca. two thirds into the gathering period.

5.6 Extending InTraTime

Several extensions to InTraTime may be relevant to consider depending on the specific application and deployment setup. To improve system accuracy and robustness, one could accept a higher cost with regards to set up or maintenance, allowing for an integration with a heavy-weight building model. Alternatively, it remains to further explore to which extend accuracy can be improved by providing InTraTime with a more robust, accurate and fine-grained positioning system—at the added cost of installing and maintaining such a more advanced solution. In order to exploit more advanced positioning systems as well as an indoor space partition of granularity higher than room level, adding extra processing steps may be beneficial: e.g. pre-segmenting common trajectories and areas [7], to avoid an explosion of the travel time graph $\mathcal{G}_T$ in terms of number of nodes and edges. It needs to be evaluated how this would impact the accuracy and scalability results, e.g., by evaluating with more regular and finer grid structures than the irregular one, yielding an average grid cell size of $14m$, currently used. In addition the system's scope of operations could potentially be extended to mixed indoor/outdoor environments, by incorporating a fusion of position trace information from multiple additional sources such as GPS.

6. CONCLUSIONS

In this paper, we outlined and addressed fundamental challenges within travel time estimation in indoor scenarios. Furthermore, we presented the InTraTime method for accurate and unsupervised estimation of indoor travel times by mining of collected indoor position traces. We demonstrated that the system was able to accurately and timely produce travel times solely via mining collected data and thus with minimal set up costs. The system was evaluated for two building complexes, a university department and a large-scale hospital, which showed that the system learns quickly sensible travel graphs from which travel time estimates can be computed efficiently. We evaluate the system using datasets collected in the two mentioned real-world environments with regards to the accuracy of its predictions, and results show an average absolute estimation error of 15 seconds, which corresponds to a mean deviation of 11.7 % from actual travel times. We also compare InTraTime with two other travel time estimation methods with regards to accuracy and provided features. In a discussion of the promising results we suggest also further variants to address scenario-specific requirements and priorities in the outlined application domains of logistics for mobile workers, indoor navigation and facility management. There also remain challenges for future research, e.g., further analysis of different parameters impacting indoor travel times, e.g, crowdedness, or elevator operations in high-rise buildings.

7. REFERENCES

[1] M. Alzantot and M. Youssef. Crowdinside: automatic construction of indoor floorplans. In *SIGSPATIAL/GIS*, pages 99–108. ACM, 2012.

[2] R. K. Balan, K. X. Nguyen, and L. Jiang. Real-time trip information service for a large taxi fleet. In *Proc. MobiSys 2011*, pages 99–112. ACM, 2011.

[3] J. Biagioni, T. Gerlich, T. Merrifield, and J. Eriksson. Easytracker: automatic transit tracking, mapping, and arrival time prediction using smartphones. In *SenSys*, pages 68–81. ACM, 2011.

[4] P. S. Castro, D. Zhang, and S. Li. Urban Traffic Modelling and Prediction Using Large Scale Taxi GPS Traces. pages 57–72, 2012.

[5] H. Du, P. Henry, X. Ren, M. Cheng, D. B. Goldman, S. M. Seitz, and D. Fox. Interactive 3d modeling of indoor environments with a consumer depth camera. In *UbiComp*, pages 75–84. ACM, 2011.

[6] A. Janecek and K. A. Hummel. Cellular Data Meet Vehicular Traffic Theory : Location Area Updates and Cell Transitions for Travel Time Estimation. In *Ubicomp*, 2012.

[7] Y. Jiang, X. Pan, K. Li, Q. Lv, R. P. Dick, M. Hannigan, and L. Shang. Ariel: Automatic wi-fi based room fingerprinting for indoor localization. In *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, UbiComp '12, pages 441–450, New York, NY, USA, 2012. ACM.

[8] D. H. Kim, K. Han, and D. Estrin. Employing user feedback for semantic location services. UbiComp '11, pages 217–226, New York, NY, USA, 2011. ACM.

[9] D. H. Kim, Y. Kim, D. Estrin, and M. B. Srivastava. Sensloc: Sensing everyday places and paths using less energy. SenSys '10, pages 43–56, New York, NY, USA, 2010. ACM.

[10] M. Kjærgaard, H. Blunck, T. Godsk, T. Toftkjær, D. Christensen, and K. Grønbæk. Indoor positioning using gps revisited. In *Pervasive Computing*, volume 6030 of *Lecture Notes in Computer Science*, pages 38–56. 2010.

[11] M. B. Kjærgaard, G. Treu, P. Ruppel, and A. Küpper. Efficient indoor proximity and separation detection for location fingerprinting. MOBILWARE '08, pages 1:1–1:8, 2007.

[12] M. B. Kjærgaard, G. Treu, P. Ruppel, and A. Küpper. Efficient Indoor Proximity and Separation Detection for Location Fingerprinting. In *MOBILWARE*, page 1. ACM, 2008.

[13] A. LaMarca, Y. Chawathe, S. Consolvo, J. Hightower, I. Smith, J. Scott, T. Sohn, J. Howard, J. Hughes, F. Potter, et al. Place lab: Device positioning using radio beacons in the wild. In *Pervasive computing*, pages 116–133. Springer, 2005.

[14] A. LaMarca and E. de Lara. *Location Systems: An Introduction to the Technology Behind Location Awareness*. Synthesis Lectures on Mobile and Pervasive Computing. 2008.

[15] C. E. Leiserson, R. L. Rivest, C. Stein, and T. H. Cormen. *Introduction to algorithms*. The MIT press, 2001.

[16] P. Lukowicz, S. Pentland, and A. Ferscha. From context awareness to socially aware computing. *IEEE Pervasive Computing*, 11(1):32–41, 2012.

[17] A. Millonig and G. Gartner. Identifying motion and interest patterns of shoppers for developing personalised wayfinding tools. *Journal of Location Based Services*, 5(1):3–21, 2011.

[18] A. B. M. Musa and J. Eriksson. Tracking unmodified smartphones using wi-fi monitors. In *SenSys*, pages 281–294, 2012.

[19] D. Pfoser, S. Brakatsoulas, P. Brosch, M. Umlauft, N. Tryfona, and G. Tsironis. Dynamic travel time provision for road networks. In *GIS*. ACM Press, 2008.

[20] R. Sen, A. Maurya, B. Raman, R. Mehta, R. Kalyanaraman, N. Vankadhara, S. Roy, and P. Sharma. Kyun queue: a sensor network system to monitor road traffic queues. In *SenSys*, pages 127–140. ACM, 2012.

[21] A. Thiagarajan, J. Biagioni, T. Gerlich, and J. Eriksson. Cooperative transit tracking using smart-phones. In *SenSys*, pages 85–98. ACM, 2010.

[22] L. Vanajakshi. Travel time prediction under heterogeneous traffic conditions using global positioning system data from buses. *Intelligent Transport Systems, IET*, 3(1):1–9, 2009.