Automatic Generation of Topological 2D Indoor Maps for Real-Time Map-Based Filtering

Martin Schäfer, Christian Knapp and Samarjit Chakraborty
martin.schaefer@tum.de, knapp_christian@mytum.de, samarjit@tum.de
Institute for Real-Time Computer Systems,TU-Munich, Germany.

Abstract—**Personal location information is regarded as the most important contextual information transmitted in ubiquitous systems. Many pedestrian indoor localization systems rely on map-matching to constrain sensor errors. The maps required for computer aided localization and tracking need to incorporate a semantic structure. Such maps are not available on a larger scale and therefore most groups working on localization solutions manually create the required maps for specific testing scenarios. To provide a solution for map generation on a larger scale, we propose a map generation toolkit that parses standard CAD-plans, to automatically generate topological maps for indoor environments. We propose a heuristic parser that separates superfluous data from the information depicting semantic building entities, e.g. rooms and doors. In our experiments approximately 95% of all structures were detected successfully. After the extraction we transform the distilled building information into an object-based building model designed for the application of fast particle-filter-based map-matching algorithms. An exemplary filter implementation demonstrates that the model is sufficiently optimized to achieve pedestrian tracking and localization in real-time.**

Keywords: Map Generation, Map Matching, Indoor Localization

I. Introduction

Every personal indoor positioning system needs an underlying map as reference. An absolute position is of no use without the relation to the surrounding building. Furthermore many pedestrian indoor localization and tracking concepts rely to a certain extent on map-based filtering algorithms to bound drift and noise induced errors. These algorithms are most commonly based on particle filters. The users trajectory is described by a set of particles. The particle distribution models the measured trajectory as well as the errors of the measurement systems. To limit the accumulation of measurement errors, the set is filtered using map induced constraints. In other words: if a particle traverses a wall, its weight is reduced to zero and the particle is deleted from the filter set. To filter the trajectories of all particles, a topological map structure with fast look-up times is imperative. In contrast to semantically enhanced outdoor maps, e.g used for car navigation, information on indoor environments is usually available in form of architectural plans in CAD files. These files contain all types of building information like walls, stairs, windows and appliances. However, their structure is focused on human readability and does not contain information that permits a computer to distinguish the different object types. Since a semantic classification of obstacles and accessible areas is necessary for map-based indoor localization and tracking, most research groups in this domain have resorted to manual construction of suitable plans [1], [2], [3]. While this approach is valid to demonstrate the performance of a filtering concept,the application of the proposed algorithms in global or urban scenarios usually remains infeasible, due to the shortage of appropriate map information. As also stated in [4], we argue that the lack of maps that are suited for both, visualization and map-based filtering, is one of the main obstacles for the mass-market deployment of indoor positioning systems. Our work therefore aims to provide the missing link between architectural maps in CAD format and semantic mapping information. We present a parser that analyses standard CAD files to distill topological map information. This information is used to create an object-based map optimized for localization and tracking applications. To illustrate the model's properties, we additionally propose an adapted map-matching algorithm and analyze its performance.
The remainder of this paper is structured as follows:
In section II we briefly cite related work. Section III describes the parsing algorithms necessary to extract information from the CAD files. Section IV introduces the object-based map and the proposed map-matching algorithm and section V concludes the paper.

II. Related Work

As CAD plans do not provide any topological information, several standards on building models have been formulated to extend the geometric information of CAD plans with semantic content. The most prominent among them are the Industry Foundation Classes (IFC) [5] and the CityGML framework [6]. Both offer detailed descriptions of internal structures including semantic links, but they are still focused on visualization in the architectural domain. In [7] a variation of the CityGML model is described. The model named BIGML provides a semantic map suited for location-based services. In [8] the refinement of digitalized floor-plans with user generated data is proposed. Map users are encouraged to indicate appliance and furniture positions to improve the accuracy of a map. However, the basic floor-plan must still be derived manually from CAD files. To the best of our knowledge none of the mentioned frameworks provides automatic map-creation from standard 2D CAD files and none of the maps is designed to provide map-based filtering.

III. Parsing CAD Files

Since we aim at providing a general map extraction tool, our focus lies on analyzing CAD files in the Drawing Interchange Format (DXF) published by AutoDesk [9]. DXF is an open specification specially designed to provide an interchange format between proprietary CAD formats of commercially available CAD tools. Since almost every CAD file can be converted into a DXF file, authorities wanting to provide indoor navigation for their facility can easily generate the required format from their building plans.

The CAD data encoded in DXF files consist of several unconnected lines, arcs and poly-lines spread across several drawing layers. Lines depicting doors are typically grouped in one or two layers. The outlines of rooms are often grouped together with labels, pillars and other line information and spread over several layers. Additionally the desired entities are not distinguished and the lines delimiting physical rooms are often superimposed with drawing information concerning floor, ceiling and accessory labels. For the scope of this paper, we assume the CAD data to be structured as one file per each floor of a building. The current parser is restricted to the extraction of 2D structures like rooms and doors. Although this is a shortcoming, the impact on the suitability for map-matching is only minor. The main purpose of map-based filters is to bound noise and drift of a 2D position and the associated heading. The measurement of the altitude is generally much more accurate and does not necessarily require sophisticated filtering.

A. Data Pre-Processing

Before the parser analyses the CAD data, the user has to indicate the floor depicted by the opened file and mark the most promising of the visualized drawing-layers. CAD designs are never completely error free. Furthermore, conversion from proprietary formats into the DXF format can introduce additional conversion errors like numerical inaccuracies or logical errors. To take these circumstances into account, the proposed extraction algorithms use error tolerant calculation and not an exact algebra. A point is defined as a normal 2D vector $p \in \mathbb{R}^2$. For brevity, the vector connecting two points p_1, p_2 is written as p_{12}. A line is defined as an ordered set of two points $\mathbb{L} := \{(p_1, p_2) | \; p_1, p_2 \in \mathbb{R}^2\}$. On these types we define the following operations:

a) Adjacent Points: Two points are adjacent iff their distance is smaller than a predefined threshold ϵ. $\forall p_1, p_2 \in \mathbb{R}^2 : p_1 \; adj \; p_2 \Leftrightarrow | \; p_{12}| < \epsilon$

b) Point adjacent to Line: A point p is adjacent (adj) to a line L iff one of L's points is approximately equal to p. $\forall p \in \mathbb{R}^2, \forall L \in \mathbb{L} : p \; adj \; L \Leftrightarrow \{\exists! q \in L | \; p \approx q\}$

c) Adjacent lines: Two lines L_1, L_2 are adjacent (adj) iff L_1 contains one point adjacent to L_2. $\forall L_1, L_2 \in \mathbb{L} : L_1 \; adj \; L_2 \Leftrightarrow \{\exists! q \in L_1 | \; q \; adj \; L_2\}$

To reduce the amount of redundant data the following pre-processing steps are automatically performed on each superset of layers selected for one of the extraction routines:

- If several layers contain information required in the same extraction routine, these layers are merged into one superset.
- Lines that are exact or flipped duplicates are removed from the superset.
- Consecutive lines that are parallel and have no other neighbours, i.e that could also be formulated as one single line without loosing information, are concatenated.

B. Extracting Doors

A door is normally encoded as a line depicting the open door and an attached arc or poly-line depicting its opening path (Fig. 1). The opening path usually touches the wall of a room indicating the position of the closed door. The origin of the arc indicates the inge. The information important for map-matching is the actual doorstep. To isolate this information, we use the following steps.

- Find arcs and poly-lines that roughly depict a quarter circle with a radius $\approx$ the length of a door step.
- For each identified quarter circle, search for lines that are almost orthogonal an end of arc and *adjacent* to the end and the center of the arc. If found save as candidate L_i

As depicted in Fig. 1 up to two lines can be identified per door. If one line is found this usually depicts the open door. In some sources two lines per door are depicted, one for the open door the other one representing the actual doorstep. The following approach isolates the desired doorsteps.

- For all arcs with only one orthogonal line L_1: Construct a second line L_2 between the arc's opposing end and its center.
- For each arc: Find a wall line superimposing $L_i, i \in 1, 2$. if found, mark L_i as doorstep candidate.
- If only one line is valid, this is the doorstep.
- If two or no lines are valid, mark the arc for further investigation.

C. Extracting Rooms

For room extraction a more sophisticated approach was necessary, since the variation of rooms representations used in CAD files is significantly higher. For instance in the available CAD files we encountered problems like incomplete room boundaries, lines delimiting several rooms and lines depicting only half a wall. We therefore propose an iterative algorithm that isolates closed and almost closed line-sequences that are candidates for rooms. These candidates are stored and

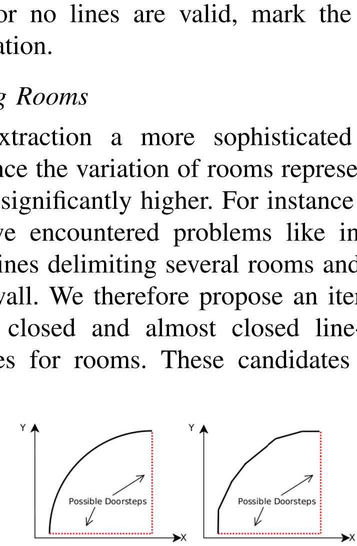

Fig. 1. Examples of Alternative Door Representations, often only one of the dotted lines (the open door) is present in the CAD data

filtered for duplicates and erroneous detections using several additional constraints. The algorithm is structured into 4 stages (a,b,c,d). Fig. 2 gives an example for each stage for better understanding.

(a) Iterate all lines L_i for the generated superset and search for lines adjacent to each L_i's **Endpoint** (Not their starting point). The lines found are called successors of L_i.
(b) For each successor L_j build a poly-line P_{ij} consisting of L_i and L_j. Make a copy of the superset $\mathbb{L}$ containing all lines dedicated to room extraction, remove line L_i and its successors from the copy forming the new set $\mathbb{L}_i$.
(c) Grow each poly-line by finding new successors in the reduced set $\mathbb{L}_i$ and building new poly-lines $P_{ijk...}$. The set of unused Lines is reduced for each split ($\mathbb{L}_{ij...}$).
(d) The the propagation of a poly-line is complete, when a line forming a closed polygon with the starting point is found. This polygon is inserted into the list of room candidates. If a polygon with another point is found, it is discarded. The propagation is canceled if no adjacent lines can be found. This criterion leads to robust termination because the set of possible lines is reduced with each step.

When the algorithm has finished iterating over all lines, some polygons have been detected mutliple times and are redundant. This is due to the fact that several lines belonging to the same room all lead to the generation of a duplicated room. In addition some falsely detected rooms can occur. For instance, closed polygons depicting a large pillar can be detected as a room, and overlapping polygons can be created, due to erroneously parsed annotation lines. We therefore use the following post-processing rules to clean the list from errors and complete the maps raw data.

- Delete all polygons with surfaces below a_{th} (usually $1 - 2m^2$)
- Delete all polygons that have no edge superimposing a door or a stair (Every Room has an entry).
- Delete all polygons intersecting two other polygons.
- Delete all polygons that are inside bigger ones.

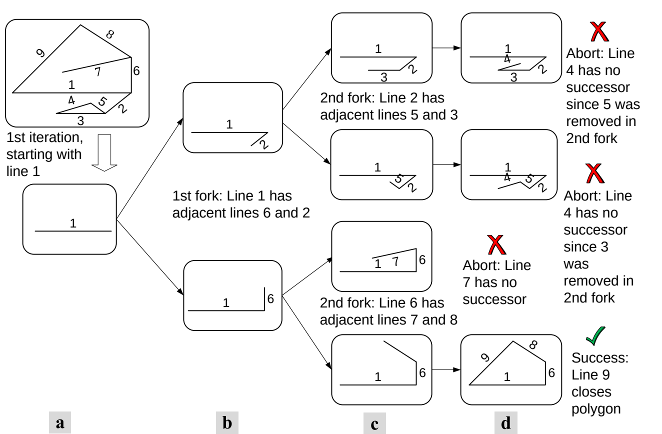

Fig. 2. Schematic Description of the Room Extraction Process

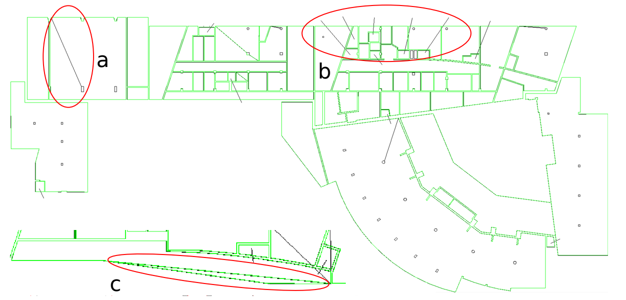

Fig. 3. Extraction Examples

Fig. 3 shows a section of an analyzed floor containing several problematic regions. The extracted rooms are shown in green while the raw lines are depicted in black. Region (a) and region (b) are examples for the error correction functionality of the proposed algorithm. Although several intersecting lines perturb the plan, the rooms are correctly identified. Region (c) on the other hand is an example for the algorithm's limitations. It is unclear, which of the lower lines depicts the room's boundary.

D. Parsing Results

To evaluate the average detection quality, we compare original building plans with the extracted representations. Table I lists the results for one exemplary university building.

Floor	Doors	Detected	Ratio	Rooms	Detected	Ratio
-1	98	98	100%	73	69	94.5%
0	80	79	97.5%	59	56	94.9%
1	76	76	100%	57	57	100%
2	73	73	100%	54	51	94.4%
3	68	68	100%	50	50	100%
4	70	70	100%	48	46	95.8%
5	50	47	94%	33	32	96.7%
Average:			98.8%			96.6%

TABLE I
EXTRACTION RESULTS BUILDING 1

The door detection algorithm is observed to be quite robust. For the majority of analyzed CAD files 100% of the depicted doors were detected. No false positives were found. The majority of the remaining outliers are caused by inaccurate sources, e.g. a door being literally drawn beside its room. The detection ratio obtained for rooms is similar on average but the algorithm is not as robust due to the different drawing alternatives. However such failures can be corrected via a user interaction interface that visualizes the differences between the raw data and the extracted information. The user can thus identify inaccuracies and remedy them with an integrated drawing GUI.

IV. MAP STRUCTURE

When all chosen layers have been analyzed and post-processed, the extracted information is used to create the semantic map model shown in Fig. 4. The model is similar to the one proposed in [7] and contains the physical entities *Building*, *Floor*, *Room* and *Door*. The map is constructed bottom-up. First the rooms are created. The lines of a room's outline are compared to the room's doors and both are used to create an edge vector that is later used in the filtering

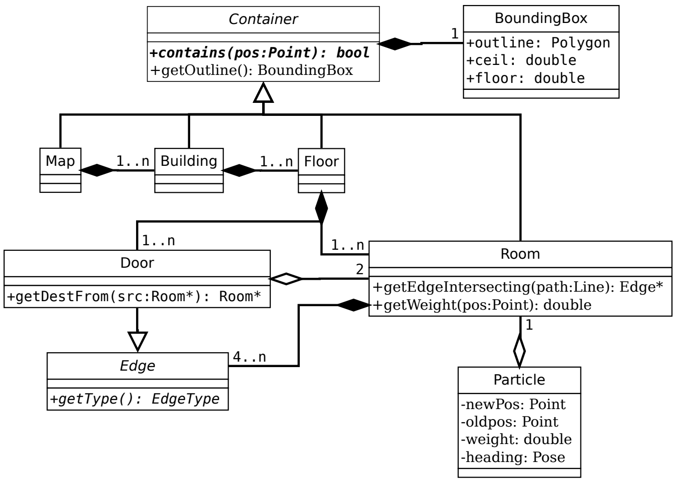

Fig. 4. UML-Model of the Semantic Map Including Particle Representation

algorithm. When a door is inserted, the room is added to the door object, since the door will serve as gateway during the filtering process. When all rooms of one floor are parsed, the *BoundingBox* of the floor is calculated as conjunction of all rooms' outlines. Equally a building's *BoundingBox* is determined, using the floors' outlines and their altitude values.

A. Filtering Algorithm

For localization and tracking, we implement a map-matching particle filter, which basically consists of the well-known phases re-sampling, propagation and correction (see e.g. [10] or [11] for details on particle filters). Similar to the approaches used in [2] and [12], we use measurements from a 3D Accelerometer, 2 2D Gyrometers and a 3D magnetic field sensor as input for the propagation phase of the filter and incorporate the map constraints in the correction phase. The detailed structure of the filter's re-sampling and propagation phase is out of scope of this work and does not differ fundamentally from the cited works. What differs is the retrieval of legal paths necessary for every particle during the filters correction phase. As depicted in Fig. 4, we define a particle incorporating its old and new position, its heading, its weight and a pointer to its room. Using this type and the proposed map structure the filter's correction algorithm can be formulated as follows:

Map-based particle filter correction phase

```
foreach(p in particleSet){
  path = p.newPos - p.oldPos;
  *edge = p.room->getEdgeIntersecting(path);
  if (edge==null) p.weight = p.weight;
  else if (edge->getType()== Wall) p.weight = 0;
  else if (edge->getType()== Door){
    *door = edge;
    p.room = door->getDestFrom(p.room);
    p.weight = p.weight * p.room->getWeight(newPos);
  }
}
```

Since no search iterations across adjacent entities are necessary, the complexity of the described algorithm only depends on the current polygon's complexity. An upper bound can be given as $O(max(n_{poly_i}))|i = [1..rooms]$ where n indicates the number of edges of polygon $poly_i$. Since most room-outlines consist of relatively few edges, the iterations are usually short. Our experiments have shown that the presented particle filter is capable of performing tracking and localization with sufficient speed to provide real-time position updates. As a testing scenario, we deploy 45400 particles on an office-floor that covers $1050m^2$ to perform localization. Run on a 1,6 GHz Mobile-CPU, the entire particle filtering process, including re-sampling, propagation and correction, terminates in less than $150ms$, hence significantly faster than required to provide real-time position updates for pedestrian movement.

V. Conclusion and Future Work

In this paper paper we have presented a novel parser that automatically distills semantic building information from architectural CAD floor plans. Since these plans are available for the majority of public and official buildings, our work provides a practical low-cost solution to create topological maps of large-scale urban environments. The created map-model is designed to allow the application of efficient particle-filter-based map-matching algorithms. Our exemplary implementation has proven that real-time indoor tracking and localization, based on the proposed model, are easily achievable. Our next steps are the extension of the parser towards the automatic detection of stairs and elevators. If this is achieved, we will be able to construct accurate 3D building representations. Furthermore, we plan to investigate the possibilities of weighted maps that incorporate different likelihoods for different rooms. Those likelihood values could, for instance, reflect the usage of certain rooms or structural changes not incorporated in the often slightly dated CAD-files. Finally we plan to extend the map-model towards a navigateable map.

References

[1] Widyawan, M. Klepal, and D. Pesch, "A bayesian approach for rf-based indoor localisation," in *4th International Symposium on Wireless Communication Systems, ISWCS*, 2007.

[2] O. Woodman and R. Harle, "Pedestrian localisation for indoor environments," in *ACM International Conference on Ubiquitous Computing (UbiComp)*, 2008.

[3] S. Beauregard, Widyawan, and M. Klepal, "Indoor PDR performance enhancement using minimal map information and particle filters," in *Position Location and Navigation System Conference IEEE/ION PLANS*, 2008.

[4] L. Wirola, T. Laine, and J. Syrja Andrinne, "Mass-market requirements for indoor positioning and indoor navigation," in *International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, 2010.

[5] *The Industry Foundation Classes (IFC) specification*, Buildingsmart International Std., Rev. IFC2x3-TC1, 2007.

[6] *OpenGIS City Geography Markup Language (CityGML) Encoding Standard*, Open Geospatial Consortium Inc. Std.

[7] M. Kessel, P. Ruppel, and F. Gschwandtner, "BIGML: A location model with individual waypoint graphs for indoor location-based services," *PIK - Praxis der Informationsverarbeitung und Kommunikation*, 2010.

[8] A. Rice and O. Woodman, "Crowd-sourcing world models with openroommap," in *8th IEEE International Conference on Pervasive Computing and Communications Workshops (PERCOM Workshops)*, 2010.

[9] Autodesk, *DXF Reference*, AutoCAD Std., Rev. v.u.26.1.01, 2012.

[10] I. M. Rekleitis, "A particle filter tutorial for mobile robot localization," Centre for Intelligent Machines, McGill University, Tech. Rep. TR-CIM-04-02, 2004.

[11] B. Ristic, S. Arulampalam, and N. Gordon, *Beyond the Kalman Filter: Particle Filters for Tracking Applications*. Artech House, 2004.

[12] B. Krach and P. Robertson, "Integration of foot-mounted inertial sensors into a bayesian location estimation framework," in *5th Workshop on Positioning, Navigation and Communication, (WPNC)*, 2008.