A Framework for Bearing-Only Sparse Semantic Self-Localization for Visually Impaired People

Irem Uygur, Renato Miyagusuku, Sarthak Pathak, Alessandro Moro,
Atsushi Yamashita, and Hajime Asama

***Abstract*— Self-localization in indoor environments is a critical issue for visually impaired people. Most localization approaches use low-level features and metric information as input. This can result in insufficient output for visually impaired people since humans understand their surroundings from high-level semantic cues. They need to be provided their location with respect to the objects in their surroundings. Thus, in this work, we develop a novel framework that uses semantic information directly for localization, which can also be used to inform the user about his surroundings. The developed framework directly uses sparse semantic information such as the existence of doors, windows, tables, etc. directly within the sensor model and localizes the user within a 2D semantic map. It does not make use of any distance information to each semantic landmark, which is usually quite difficult to obtain. Nor does it require any kind of data association - the objects need not be uniquely identified. Hence, it can be implemented with simple sensors like a camera, with object detection software. For our framework, one of the most popular game engines, Unity was chosen to create a realistic office environment, consisting of necessary office items and an agent with a wide-angle camera representing the user. Experimentally, we show that this semantic localization method is an efficient way to make use of sparse semantic information for locating a person.**

I. INTRODUCTION

According to the World Health Organization (WHO), 253 million people are visually impaired, of whom 36 million are blind[1]. One of the problems visually impaired people face is self-localization. The self-localization problem can be divided into two categories: outdoor localization and indoor localization. Outdoor localization has been greatly achieved by GPS. On the other hand, GPS does not work well inside of a building. Such environments remain a problem for blind people.

Indoor localization has been investigated and shaped by researchers mostly for robots with the help of different sensors. Robots generally need accuracy in metric information with respect to metric landmarks [1]. Therefore, traditional indoor localization and mapping systems generally use range sensors and metric maps [2]. Even though this information is useful for robots, it is insufficient for visually impaired people. According to [3], visually impaired and blind people make a mental map of an environment and find the relative positions of semantic landmarks such as doors, tables, etc. useful. Such information, i.e. the distribution of objects around the visually impaired person can be useful for self-localization as well as navigation/interaction with these objects. In recent years, advancements in neural networks in computer vision have made real-time acquisition of this type of high-level information about semantic landmarks possible via cameras [4] . This change led to the computation of a semantic map [5], mostly to just represent the environment, on top of metric maps in Vision-based Semantic Localization (VSLAM). A semantic map generally consists of detected objects in an environment. Such systems are prone to have a drift in scale and pose, loop closure and global consistency problems. Here, a more direct way to locate a person inside of a 2D floor map is proposed by using semantic information directly in the sensor model. Unfortunately, most approaches that use landmarks for detection require some kind of information about the distance and data association. For example, in case of WiFi-based localization [6]. Semantic landmarks can best be detected by a camera. However, distance information is difficult to obtain. Also, data association is difficult as it would require tracking every semantic landmark uniquely. Instead, in this work, we show that distance information and data association are not necessary. The only inputs to our system are the object type of the semantic landmarks, their bearing angles, their detection scores, and a 2D floor map showing object positions (doors, tables, windows, etc.) which is generally available for indoor environments. Moreover, these landmarks are not continuous but located in the environment sparsely. Thus, the aim of this research is to investigate the feasibility of using bearing-only, sparse semantic information for localization, in order to help visually impaired people.

In this study, x, y, θ pose information on a 2D annotated floor map is aimed to be retrieved by using bearing angle and semantic data in a sensor model (Fig. 1). The system includes semantic information from a spherical camera to acquire a wide field-of-view, an annotated 2D floor map which has additional information about certain objects on it and a particle filter for localization. Object type, detection score, and bearing angle for each object are considered by the sensor model in order to take advantage of the semantic information. Distance information is not included as a part of the sensor model for localization as it is difficult to obtain. On the other hand, bearing angle can be very descriptive if the types of the objects are known since it provides information about their distribution. Using the proposed system, feedback about the user's location as well as the relative positions to a big object such as windows, doors etc. could be provided. In order to

I. Uygur, R. Miyagusuku, S. Pathak, A. Moro, A. Yamashita, and H. Asama are with the Department of Precision Engineering, The University of Tokyo. Contact: uygur@robot.t.u-tokyo.ac.jp

[1]World Health Organization,http://www.who.int/news-room/fact-sheets/detail/blindness-and-visual-impairment, October 2017.

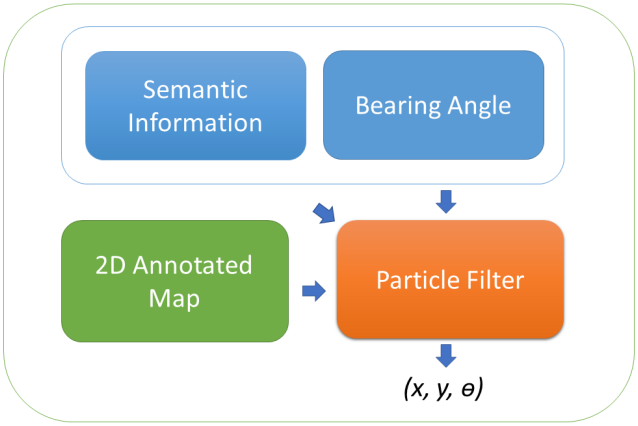

Fig. 1. Overview of the localization system. Inputs of the particle filter are sparse semantic information, bearing angle to the objects obtained by a spherical camera and an annotated 2D map. The output is the pose (x, y, θ) of the agent on a 2D floor map.

evaluate and design the prepared sensor model, we developed a simulation environment. The goal is to acquire an error that will be manageable for a person.

The simulation has been implemented in the Unity cross-platform game engine.

Unity[2] is a cross-platform game engine created by Unity Technologies, which also offers rendering solutions to different fields. It can simulate a virtual camera and provide information about the location of various objects inside the view of the camera. In this work, a realistic office environment was created by using Unity, in order to test the algorithm.

II. Previous Work

There is a lot of different work with different sensors for the guidance of visually impaired people. When it comes to indoor localization, the most natural way for a human is to locate himself on a 2D floor map. This idea inspired [7] to include a 2D floor map into their localization algorithm. The developed method uses a 2D floor map to lift it up to a third dimension to create a scale-free point cloud for comparison in different geometrical criteria. The problems with this method are the assumption of a third dimension (Manhattan world), unnecessary and heavy computation of creating a 3D point cloud of an entire floor, and heavy computation during the matching methodology. In [8] the authors developed a semantic navigation and obstacle avoidance system for visually impaired and blind people by using a floor map and an RGB-D sensor, while another work [9] with a 2D floor map, an RGB-D sensor, and an object-specific likelihood maps for each object type showed the use of semantic information instead of range measurements. The authors combined the traditional rangefinder perceptual model with a semantic model to locate a robot while using dense semantic information. However, dense semantic information requires semantic segmentation of images and is difficult to implement. Instead, in this work, we show that dense semantic information is not necessary to locate a person. In this work, a bearing only sparse semantic localization method using a spherical camera has been proposed to locate a person on an annotated 2D map, without data association or distance measurement.

[2]Unity, https://unity3d.com/

III. Methodology

A. System Setting

a) Problem Setting: The system setting consists of a 2D map of a 3D environment, semantic information, and a particle filter. The object classes to be used in the sensor model are limited to tables, windows, and doors in the current setting. There is no data association i.e. these objects are not uniquely identified. For example, a table on the map and in a sensor reading just has the tag "table". They are annotated on the map. The agent with a spherical camera obtains a large field-of-view so that semantic information can be obtained from every direction. The input has a bearing angle to an object of one of the annotated object classes. Information required from the camera are bearing angles to objects, the types of the objects, and detection scores.

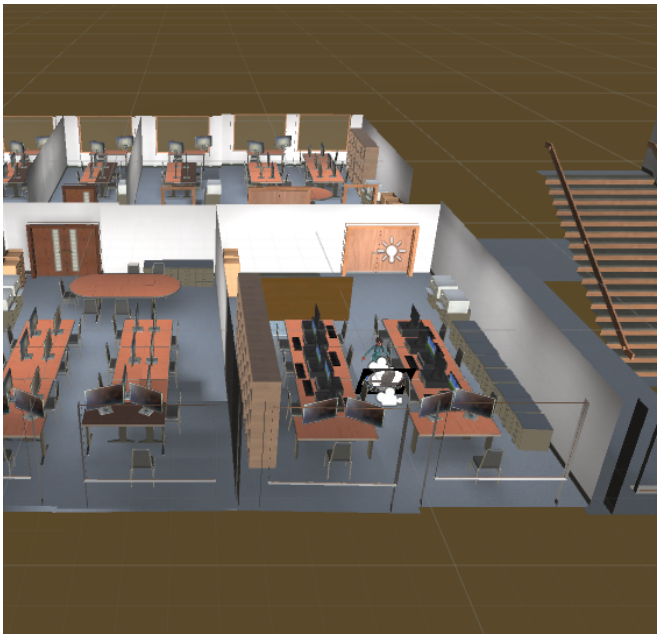

Fig. 2. An overview of the simulation environment.

B. Framework

A typical indoor environment consists of the basic components of a room such as doors, windows, tables, etc.. Each of these objects is modeled as game objects in Unity's development environment. Game objects can be assigned many properties such as texture, transformation etc. and can be controlled by scripts. An office environment was designed by using objects typically found in offices i.e. desks, chairs, shelves, monitors, printers etc. An overview of the simulation environment can be seen in Fig. 2. An agent, which is also

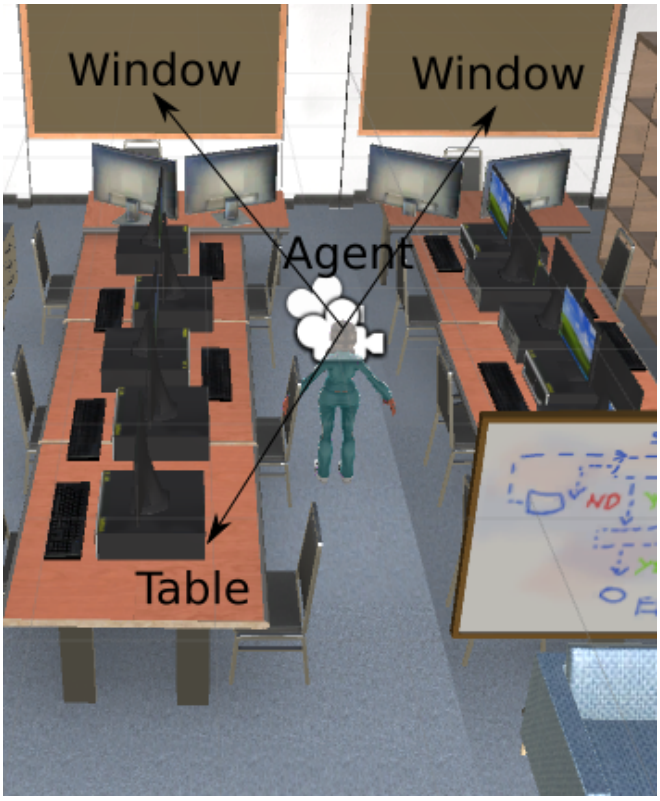

Fig. 3. An agent with a spherical camera attached in the simulation.

another game object, was assigned and controlled by a script to move around in the designed environment easily. The agent can be seen in (Fig. 3), along with a representation of the kind of information used for localization i.e. object types, bearing angles to the object center, and detection scores. Physical simulations of the game objects are provided by Unity's built-in physics engines. In order to get a view of the agent's surroundings, a camera is attached to the agent. As the agent navigates, the attached camera follows it. Unity supports rendering images to cube maps. Therefore, a 360 deg view can be captured during the navigation. For localization, a particle filter has been implemented in ROS environment. The communication between Unity and ROS is done via UDP. Different threads run for the particle filter and the simulation data. The system overview can be seen from Fig. 4.

For localization, different localization algorithms can be employed that use bearing-only information, such as Kalman Filters [10], which can even be performed in 3D maps [11], as well as Monte Carlo Localization (MCL) algorithms. Monte Carlo Localization [12] approach has been used in this project. In Monte Carlo localization, robots with rangefinders generally use range and bearing angle information to compare against a pre-built distance map, while vision-based methods generally use information from the camera to compare against a pre-built image map. In vision-based methods generally, low-level features like points, lines, and corners are used.

MCL has three stages: prediction, correction, and re-sampling. At the prediction step, particles are propagated according to odometry information. Each particle is a pose hypothesis. Afterward, at the correction stage, these particles are assigned weights proportional to how much they match the current measurement. Since localization is assumed to be a Dynamic Bayesian Network, only the latest measurement and odometry is used. Finally, at the resampling stage, particles are resampled according to the assigned weights, making less likely particles be replaced by more likely ones.

In our method, MCL requires a map $\mathbf{M}$ that stores Cartesian Coordinates (x,y) of annotated objects from different predetermined object classes c. The map $\mathbf{M}$ is defined as $\mathbf{M}$= $\{x_n, y_n, c_n\}_{n=1}^{N}$ where N is the number of objects in the map. MCL also requires odometry $\mathbf{u}$ and sensor measurements $\mathbf{Z}$, which consists of detected object class c, bearing angle α to the center of the object and detection score w. The measurement update $\mathbf{Z}$ is defined as $\mathbf{Z}$= $\{(c_k, \alpha_k, w_k)\}_{k=1}^{K}$, where K is the number of objects in each sensor reading. Using this information, MCL will estimate agent state $\mathbf{s} = (x,y,\theta)$ with bearing θ.

The sensor model used at the correction step is defined as $P(\mathbf{Z}|\mathbf{s},\mathbf{M})$. Using MCL we compute the posterior $P(\mathbf{s}_t|\mathbf{Z}_t,\mathbf{u}_t)$ for time step t as

$$\begin{aligned} &P(\mathbf{s}_t|\mathbf{Z}_t,\mathbf{u}_t,\mathbf{M}) = \\ &P(\mathbf{Z}_t|\mathbf{s}_t,M)P(\mathbf{s}_t|\mathbf{u}_t,\mathbf{s}_{t-1})P(\mathbf{s}_{t-1}|\mathbf{Z}_{t-1},\mathbf{u}_{t-1},\mathbf{M}) \end{aligned} \tag{1}$$

For computing our sensor model we group objects belonging to the same class as $\mathbf{O}_i = \{\mathbf{Z}_k : c_k = c_i \forall k\}$ and compute the likelihood of an observation as

$$\begin{aligned} &P(\{\mathbf{O}_1,...,\mathbf{O}_C\}|\mathbf{s},\mathbf{M}) = \\ &\prod_{i=1}^{C} P(\{\mathbf{O}\}_i|\mathbf{s},\mathbf{M}) \end{aligned} \tag{2}$$

where C is the number of object classes.

Including different object classes into a sensor model allows introducing different weights to each class. The object classes are assumed to be independent. The measurement probabilities are weighted by $P(\mathbf{z}|\mathbf{s},\mathbf{M})^{1/size(\mathbf{O}_i)}$. We use the inverse of the number of objects of the same class present, in order to prevent the dominance of classes with a large number of objects. Another reason for this is to be able to define different standard deviations for different objects because, as explained in [9], some objects can be more descriptive than the others. Since the acquisition of semantic information can be provided by methods which can ensure detection with a level of certainty for each object type, the confidence of the detection is included as a part of the sensor model to reduce the effect of unreliable detections.

$$\begin{aligned} &P(\mathbf{z}^k|\mathbf{s},\mathbf{M}) = \\ &max(P(\mathbf{z}^k|\mathbf{s},\mathbf{M}^n)\forall \mathbf{M}^n \in \mathbf{M}|\mathbf{M}_c^n = \mathbf{z}_c^k) \end{aligned} \tag{3}$$

The particle weights are updated according to the maximum likelihood correspondence. Therefore, since the objects

TABLE I

ERRORS COMPARING DIFFERENT PERCEPTION MODELS (FIELDS-OF-VIEW)

Errors			
Errors	180 deg	270 deg	360 deg
Mean (m)	0.21069	0.04796	0.03316
Std (m)	0.23469	0.03102	0.01944
Mean (deg)	0.042	0.008	0.004
Std (deg)	0.047	0.007	0.003

in the map and objects in the sensor data are not uniquely identified, for each observation data, the most likely landmarks are selected and accepted to be true. This requires a distinction among objects or a certainty of the pose, which is difficult to obtain. However, knowing object types provides the necessary distinction, especially if the number of elements in an object class is low. In the next section, the proposed localization method is evaluated under various conditions of practical consideration such as change of field-of-view, occluded view, and missing object detections.

IV. EXPERIMENTS

For the experiments, information about objects and the agent were obtained from Unity with added artificial noise. The objects in the map or in the observation did not have unique identification codes. They only had an object category they belong. For our experiments, the included object categories for the perception model were tables, windows, and doors. The bearing angle was calculated according to the center of each object with an added artificial noise. A bounding box of an object was used for calculating the center for the bearing angle[3] calculation. The detection scores were set to maximum for this experiment since at this stage since the data source is certain. All experiments were conducted with 10000 particles and the effects of the field-of-view, occlusions, and missing detections were evaluated.

a) Different Perception Models: In order to decide the appropriate field-of-view, three perception models were implemented for the experiments. Cameras with 180 deg, 270 deg and 360 deg field-of-view were used respectively during the experiments. Each of them used semantic information and bearing information. The room was very symmetric. In fact, the only object that broke the symmetry was the door. All of the models were run simultaneously with random initialization in position and orientation. The generated trajectories can be seen at Fig. 5. The errors are shown in Fig. 6 and Table I. As expected the model with a 360 deg field-of-view converged faster and gave better results than others, while the worst performance belonged to the model with 180 deg camera. However, all results are accurate enough to locate a person.

[3]In real life, this parameter is expected to have a lot of noise, due to jittery detection of bounding boxes.

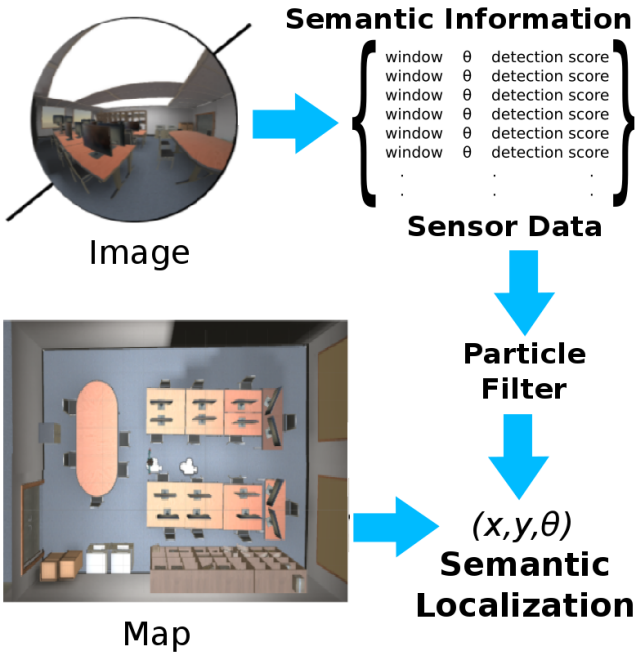

Fig. 4. The system overview. Images from a spherical camera, which is attached to an agent, are used to obtain sensor information. The Unity map of the environment is used in particle filter for the computation of the location of the agent.

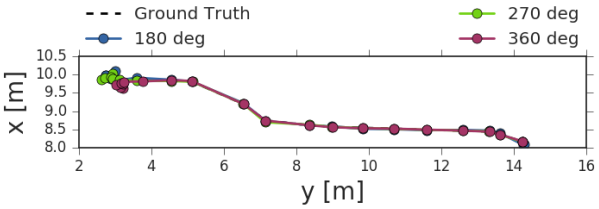

Fig. 5. Generated trajectories of the different perception models (fields-of-view). The perception models with the 180 deg camera, 270 deg camera, 360 deg camera are represented in blue, red, and green colors respectively. The ground truth is given by the black dashed line.

b) Effect of Occlusion: An indoor space can be very plain without any descriptive features to characterize the location. It also can be cluttered by many objects and people, which can block the detection. A localization system using a camera has to be robust against those problems because when a camera is used as a sensor, it is very easy to lose information from something simple like facing to a wall for a long time. In order to evaluate the effect of this, the three perception models with 180 deg, 270 deg, 360 deg of field-of-view cameras were compared in a scenario where the view of the agent got blocked twice due to facing a corner. The impact of lost information was also observed respectively. It should be noted that in this experiment half of the time the most of the view was blocked for cameras with

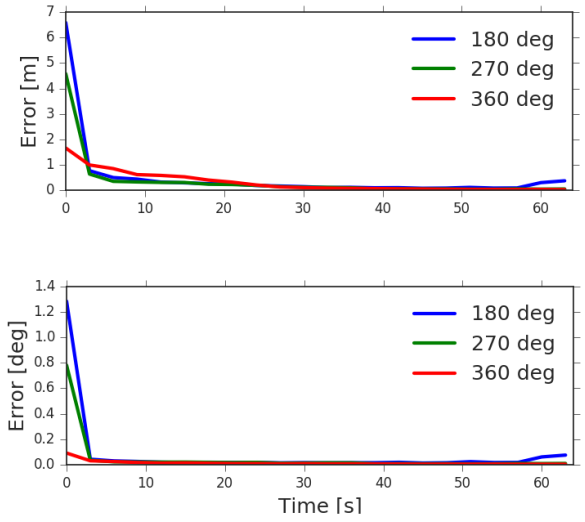

Fig. 6. Errors comparing the different perception models (fields-of-view). The first graph shows the 2D position errors in meters. The second graph shows the error in angle in degrees.The perception models with the 180 deg camera, 270 deg camera, 360 deg camera are represented in blue, green and red colors respectively. The ground truth is given by the black dashed line.

smaller fields-of-view. As expected, cameras with 180 deg and 270 deg and could not recover from the loss of information, whereas the agent with 360 deg camera had 5 cm, 0.03 deg mean error and 4 cm, 0.04 deg standard deviation. The trajectory is shown in Fig. 7. The results can be seen in Fig. 8 and Table II. In this work, the followed approach used maximum likelihood for object matching without any data association among objects. Even though it gave good results with a 360 deg camera, with smaller field-of-view the system could not recover from resulting wrong matches. High symmetry in the environment and the lack of data association, i.e. individual object identification creates wrong matches. This result shows the necessity of keeping field-of-view as large as possible.

c) Missing Detections: Real systems that acquire semantic information can be expected to miss detections once in a while. Hence, an experiment was conducted with missing detections to evaluate the effect of this phenomenon. When the loss of detection was experienced at a smaller scale, where at each step the agent lost a random amount of the detection (up to entire detection), instead of being blocked for a big amount for a long time, all of the models still performed accurately. The comparison of the cameras and the average trajectories are given in Fig. 10, Table III and Fig. 9 respectively.

V. Discussions

One of the concerns about using high-level semantic features is the input quality. Since in the experiments semantic information was directly obtained from the simulation, detection errors were not a major issue. However, when

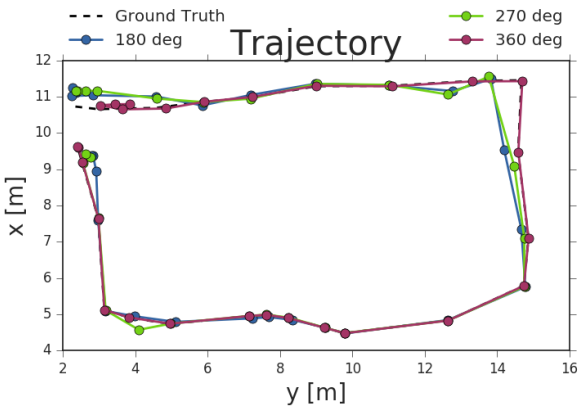

Fig. 7. Generated trajectories of the experiment with missing detections. The perception models with the 180 deg camera, 270 deg camera, and 360 deg camera are represented in blue, green, and red colors respectively. The ground truth is given by the black dashed line.

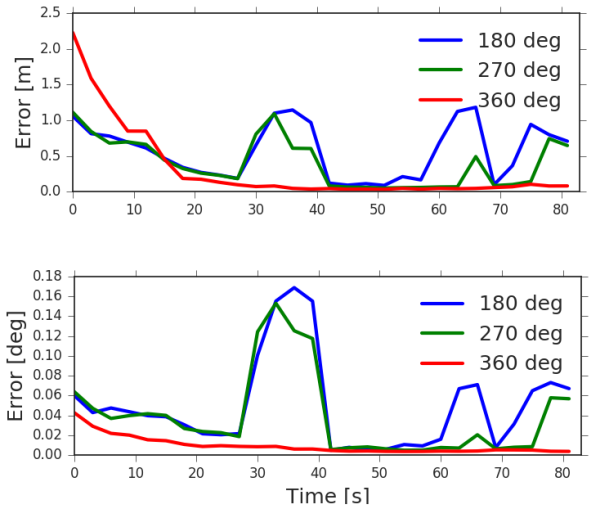

Fig. 8. Errors of the experiment with blocked view. The first graph shows the 2D position errors in meters. The second graph shows the error in angle in degrees.The perception models with the 180 deg camera, 270 deg camera, and 360 deg camera are represented in blue, green, and red colors respectively. The ground truth is given by the black dashed line.

TABLE II
Errors comparing different perception models (fields-of-view) with occluded view

Errors (Occlusion)			
Errors	180 deg	270 deg	360 deg
Mean (m)	0.57748	0.22748	0.05762
Std (m)	0.61999	0.41747	0.04620
Mean (deg)	0.038	0.0170	0.004
Std (deg)	0.048	0.0278	0.0031

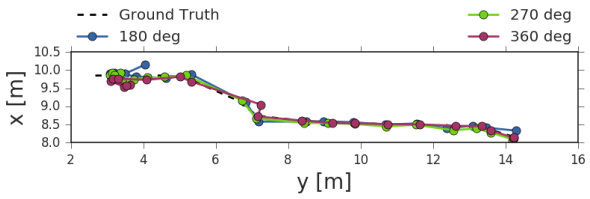

Fig. 9. Generated trajectories of experiment with missing detections. The perception models with the 180 deg camera, 270 deg camera, 360 deg camera are represented in blue, green and red colors respectively. The ground truth is given by the black dashed line.

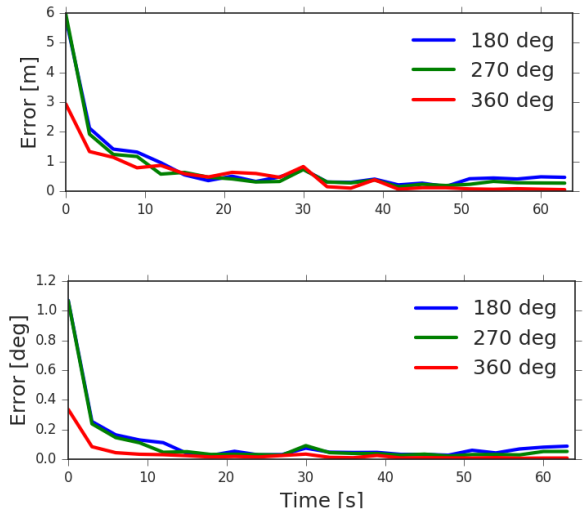

Fig. 10. Errors of the experiment with missing detections. The first graph shows the 2D position errors in meters. The second graph shows the error in angle in degrees.The perception models with the 180 deg camera, 270 deg camera, 360 deg camera are represented in blue, green and red colors respectively. The ground truth is given by the black dashed line.

this information is obtained from a neural network like [4], false positives might cause a higher uncertainty in measurement quality. In addition, the jitter of the bounding box i.e. the bearing angle is expected to create a bigger error. Even though artificial noise was added, the noise in a real-life implementation is expected to affect the system more seriously.

Since the approach depends on the number of objects and their distribution, the system is constrained by the amount of available information, which might create a problem in a bigger map due to symmetries, or if the system is expanded to a navigation method due to insufficient corrections.

Finally, the necessity of having a large field-of-view was proven in the second experiment.

VI. CONCLUSIONS

This research aimed to obtain the 2D pose of an agent on an annotated map by including bearing angle and sparse semantic information without any data association or distance measurement using a particle filter so that a visually impaired person can be located and informed about his surroundings at the same time. A simulation was created by using the Unity game engine to provide the necessary realistic and flexible environment. The framework was evaluated with different fields-of-view, under different conditions such as missing detections and occlusions. Sufficient localization accuracies were achieved, and the benefits of using a large field-of-view were brought to light. The next step is to train a CNN like [4] to obtain real-life semantic information. The final aim of the study is to test the system in real life.

TABLE III

ERRORS COMPARING DIFFERENT PERCEPTION MODELS (FIELDS-OF-VIEW) WITH MISSING DETECTIONS

Errors (Missing Detections)			
Errors	180 deg	270 deg	360 deg
Mean (m)	0.44887	0.28283	0.07261
Std (m)	0.48477	0.42585	0.16332
Mean (deg)	0.067	0391	0.007
Std (deg)	0.072	0.060	0.018

REFERENCES

[1] T. Goto, S. Pathak, Y. Ji, H. Fujii, A. Yamashita, and H. Asama, "Line-based global localization of a spherical camera in manhattan worlds," in *Proceedings of the 2018 IEEE International Conference on Robotics and Automation*. IEEE, 2018, pp. 2296–2303.

[2] L. Zhang and B. K. Ghosh, "Line segment based map building and localization using 2d laser rangefinder," in *Robotics and Automation, 2000. Proceedings. ICRA'00. IEEE International Conference on*, vol. 3. IEEE, 2000, pp. 2538–2543.

[3] R. Tscharn, T. Außenhofer, D. Reisler, and J. Hurtienne, "Turn left after the heater: Landmark navigation for visually impaired users," in *Proceedings of the 18th International ACM SIGACCESS Conference on Computers and Accessibility*. ACM, 2016, pp. 295–296.

[4] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the 2016 IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[5] A. Nüchter and J. Hertzberg, "Towards semantic maps for mobile robots," *Robotics and Autonomous Systems*, vol. 56, no. 11, pp. 915–926, 2008.

[6] R. Miyagusuku, A. Yamashita, and H. Asama, "Precise and accurate wireless signal strength mappings using gaussian processes and path loss models," *Robotics and Autonomous Systems*, vol. 103, pp. 134–150, 2018.

[7] H. Chu, D. Ki Kim, and T. Chen, "You are here: Mimicking the human thinking process in reading floor-plans," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 2210–2218.

[8] B. Li, J. P. Munoz, X. Rong, J. Xiao, Y. Tian, and A. Arditi, "Isana: wearable context-aware indoor assistive navigation with obstacle avoidance for the blind," in *European Conference on Computer Vision*. Springer, 2016, pp. 448–462.

[9] O. Mendez Maldonado, S. Hadfield, N. Pugeault, and R. Bowden, "Sedar–semantic detection and ranging: Humans can localise without lidar, can robots?" in *Proceedings of the 2018 IEEE International Conference on Robotics and Automation*, 2018.

[10] M. Deans and M. Hebert, "Experimental comparison of techniques for localization and mapping using a bearing-only sensor," in *Experimental Robotics VII*. Springer, 2001, pp. 395–404.

[11] T. Lemaire, S. Lacroix, and J. Sola, "A practical 3d bearing-only slam algorithm," in *Intelligent Robots and Systems, 2005.(IROS 2005). 2005 IEEE/RSJ International Conference on*. IEEE, 2005, pp. 2449–2454.

[12] F. Dellaert, D. Fox, W. Burgard, and S. Thrun, "Monte carlo localization for mobile robots," in *Robotics and Automation, 1999. Proceedings. 1999 IEEE International Conference on*, vol. 2. IEEE, 1999, pp. 1322–1328.