To appear in Proc. of IEEE European Conference on Mobile Robots (ECMR), 2017, Paris, France.

Synthesized semantic views for mobile robot localization

Johannes Pöschmann, Peer Neubert, Stefan Schubert and Peter Protzel[1]

***Abstract*— Localizing a mobile robot in a given map is a crucial task for autonomy. We present an approach to localize a robot equipped with a camera in a known 2D or 3D geometrical map that is augmented with semantic information (e.g., a floor plan with semantic labels). The approach uses semantic information to mediate between the visual information from the camera and the geometrical information in the map. Moreover, semantic information is robust to appearance changes like lighting conditions. Instead of solely relying on salient semantic landmarks (i.e., "things" like doors) we also exploit "stuff"-like semantic classes such as wall and floor. The presented localization approach builds upon the idea of computing a semantic segmentation of an incoming camera image using a Convolutional Neural Network and subsequent matching to semantic views synthesized from a map. We give details about the algorithmic approach on how to semantically segment images, synthesize images from the semantic 2D or 3D map, the matching between images from both sources, and the integration in Monte Carlo localization. Further, we provide a set of proof-of-concept experiments and evaluate the influence of the selected set of semantic classes. To work towards the usage of hand-drawn sketches as input map, we also evaluate the robustness of the presented approach to map distortions.**

I. INTRODUCTION

The ability to recover and maintain knowledge about the own position in the world is essential for mobile robots. The usage of a priori known maps can significantly facilitate this task, reducing the SLAM problem to a pure localization. Moreover, if we can use cameras for this localization task, the expected sensor costs are small and we can also transfer this capability to other mobile sensor devices than robots. But where do the maps for visual localization come from? Instead of mapping the environment with cameras in advance, in this paper, we work towards exploiting other sources of maps: known 2D floor plans and hand-drawn top-view sketches of the environment. To allow for comparison with the actual camera view, the maps are augmented with semantic information about walls, doors, and other objects.

The way we use the robot's current camera image and the map information for localization is partially inspired by the navigation system of desert ants presented by Möller [1]. As climatic conditions doesn't allow the usage of pheromone traces, desert ants developed a vision-based navigation approach: Before they leave their nest's location, a snapshot (home view) of the environment is taken. After an ant has finished the foraging, it has to find its way back to the nest. Therefore, it starts to compare the home view with *synthetic* views: distorted current views that are obtained by transforming the current image into possible motion directions – the most similar image pair then indicates the correct homing direction. Details on a technical realization of the *ant algorithm* on mobile robots are provided by Möller [2].

In the here presented approach to localization using the given quite different modalities (a 2D plan and a visual sensor), we exploit recent advances in Deep Learning to assign pixel-wise *semantic* information to images. Semantic information serves as an intermediate layer between a current camera image and a geometrical map in order to facilitate the matching between both modalities. Semantic labels represent the world in a rather abstract manner; this prevents a camera-map matching algorithm from being sensitive to appearance changes caused by lighting or rotation (e.g., swivel chair). For a 2D map, semantic information about walls, floor, windows, etc. can be automatically extracted from construction plans or feasibly drawn into 2D floor plans by humans. Humans can also easily integrate additional semantic classes like furniture or plants. Starting from this 2D map, we create a semantic 3D map using knowledge about the occurring classes. Localization is done in a Monte Carlo manner using synthesized semantic views from this map. A future use case would be a human operator sketching the path for a robot's delivery task on a tablet and adding some semantic information, encoding information like: "Follow that way and go left after the third tree".

In this paper, we

- present an approach to localization in given 2D semantic maps using synthesized semantic images and ConvNet-based semantic segmentations from a robot's camera
- describe the implementation in a Monte Carlo localization system
- provide proof-of-concept experiments as well as an evaluation of influences of chosen semantic classes and robustness to geometrical map errors

II. RELATED WORK

The task of localizing camera images within a given map by image synthesis can be addressed in different ways. Our previous work [3] uses the geometrical information of a map for a depth image synthesis at desired positions with a subsequent comparison to a current greyscale camera view. Instead of using distance information of a map directly, Wolcott and Eustice [4] use a map enhanced with intensity information to synthesize greyscale-similar images for a subsequent comparison to the current view, and the approach of Pascoe et al. [5] builds upon a map with colour information for image synthesis. Caselitz et al. [6] show an alternative approach for the matching of a given map with a current camera image: Instead of using the map to

[1]The authors are with Chemnitz University of Technology, Germany firstname.lastname@etit.tu-chemnitz.de

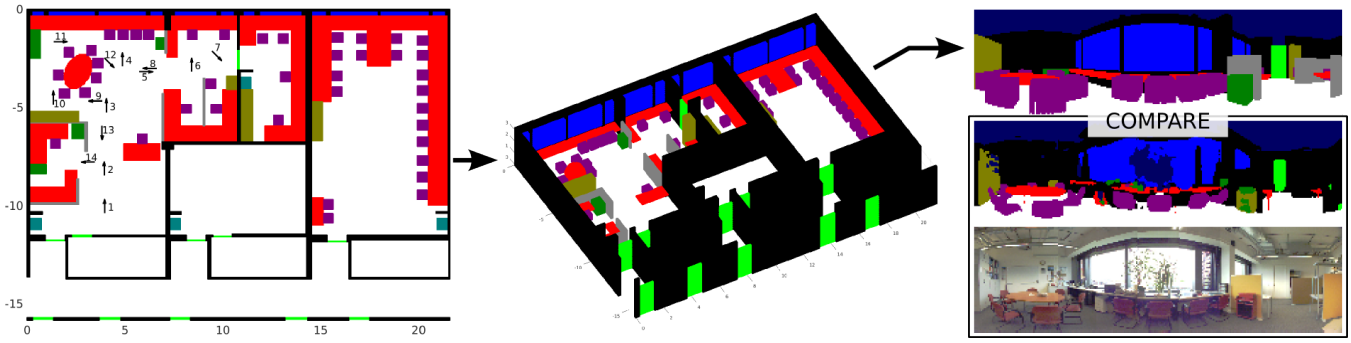


Fig. 1. Left: Accurately hand-drawn 2D semantic top-view map of an indoor environment, consisting of 10 different object classes. Annotated are the 14 ground truth positions of the robot during our real world experiments. This map is used to generate a 3D semantic pointcloud (centre), in which 2D semantic panorama images are synthesized (top right). Bottom right: Camera images are fed into a CNN for image segmentation, resulting in a semantic panorama image of a real world scene. Comparison of both semantic panoramas yields a image similarity and enables localization.

generate synthesized images, they used the camera image stream to reconstruct the environment's geometrical structure for a subsequent point cloud matching.

Another way of localization within a given map is to exploit the semantic information of the environment. Kuiper's spatial semantic hierarchy paradigm [7] uses semantic information as a topological map. Vasudevan et al. [8] add semantic information to observed objects (doors and household objects) in order to build a hierarchical topological-semantic map, with which they can annotate places (e.g., office, corridor). Another approach is landmark-based localization. Atanasov et al. [9] and Anati et al. [10] use object detectors to recognize landmarks, which are annotated in the given map, followed by particle filter localization. Our work is closest related to [10]. Through the application of a soft object detector, they generate a heatmap with semantic information about the occurring objects for a set of panoramic camera images and compare them with the expected heatmap at each particle location. In contrast, we propose a camera-based semantic localization algorithm which does not rely on specific, distinctive object classes, but rather works with non-expressive classes like wall and floor. Furthermore, we evaluate the robustness of our systems towards distortions of the given semantic map. This opens a variety of use cases like localization within a human-made sketch of the environment.

III. ALGORITHMIC APPROACH

Fig. 1 illustrates the involved steps. Starting from a 2D floor plan with semantic labels, a 3D map is created. For an assumed camera pose, we can synthesize a label image. Given a camera image taken from the real robot pose, we can compute a second label image using a semantic segmentation algorithm. By comparing the two label images, we can evaluate the assumed robot pose. This approach directly integrates in the well-know Monte Carlo localization (e.g., see [11] for an introduction): each sample provides an assumed robot pose that can be evaluated using the above approach.

A. Synthesizing images from a given 2D semantic map

The task is to localize a robot within a 2D semantic map. The map can be automatically generated (e.g. from floor plans), augmented by humans (e.g., with the semantic labels) or completely hand-drawn. In the later presented experiments, a simple, colour-coded 2D map of the environment is used (each class has a different colour). Given a 2D floor plan, such a map can be easily created by a human with some basic image editing software. All objects occurring in the real world can be freely drawn within the map. Each object class (e.g., wall, floor, table) is represented by a different colour and is associated with a class-label and a minimum and maximum height. With this information, the 2D semantic map can be converted into a 3D semantic point cloud. A 2D semantic map and the corresponding 3D point cloud of an indoor environment are shown in Fig. 1.

To synthesize an image given this 3D point cloud (with associated semantic class labels) and a requested camera pose, we follow the straight forward approach described in [3]: Each 3D point's distance is projected onto a unit sphere centred at the requested camera pose. The azimuthal and polar angles are discretized to the target image resolution. By keeping only the class label of the point with minimal distance for each direction, this spherical grid corresponds to the synthetic label image; pixels without projected 3D points are set to NaN values. In contrast to the representation used in [10], this allows for seeing multiple objects on the vertical image axis. Again, see Fig. 1 for an example.

This preliminary approach is easy to implement but its runtime is linear in the number of 3D points. Presumably, the runtime could be improved by the usage of computer graphics techniques including ray tracing algorithms and efficient data structures like k-d trees or octrees.

B. Semantic image segmentation

To be able to compare synthesized semantic images with real world images, we need to associate semantic information to each pixel in an image. This is a well known task, called semantic segmentation, for which the currently best performing methods build upon Deep Learning techniques. We use the "Pyramid Scene Parsing Network (PSPNet)" [12], which bases on a Convolutional Neural Network and achieved first rank in ImageNet scene parsing challenge 2016. We used an out of the box CNN, trained on the ADE20k dataset [13], which includes 150 object classes,

containing both indoor and outdoor class instances. Given an image as input, the PSPNet associates each pixel with one of the 150 object classes. Afterwards, the resulting semantically segmented images are transformed into spherical coordinates in order to correspond to the synthesized-semantic-image shape and thus enabling a holistic image comparison between both images (see Fig. 1).

C. Holistic image comparison

We decide to use a holistic image comparison instead of landmark based approaches, since we also want to exploit often occurring, less-expressive classes like walls or floor ("stuff" in contrast to "things" [14]) for localization. Consequently, a key component of our approach is an expressive similarity metric for a holistic comparison of semantic images. Therefore, a weight matrix is used that scores each assignment of pixels of the 150 ADE20k classes used by the PSPNet to pixels with classes present in our 2D semantic map. Currently, the values of the weight matrix are set empirically based on the following three key concepts:

1) Soft class matching: Full similarity between matching classes (e.g., desk and table) results in a score of 1, whereas partial similarity between related classes (e.g., chair and sofa) results in a score between 0 and 1.
2) Weighted class significance: Often occurring classes (e.g., wall or floor) are not expressive, leading to a smaller score between 0 and 1.
3) Class matching penalty: Overlapping of two contradicting classes leads to a score between -1 and 0. Penalization of, e.g., overlapping floors and walls results in a precise distance measurement to nearby walls.

Synthesized and segmented images are compared pixel-wise with the following score function:

$$similarity = \frac{\sum_{pixels} w_{i,j}}{\sum_{pixels}} \tag{1}$$

where w_{ij} is the weight/score for the linkage between class i from a segmented image and class j from a synthesized image. With a similarity of 1, the two images are identical. Negative similarities are set to 0.

D. Monte Carlo localization

Algorithm 1 gives details on how the described image synthesis and comparison approach integrates in Monte Carlo localization. The main loop in line 3 processes each image and applies the described image synthesis and comparison. A semantic segmentation J_t is computed for the current camera image I_t (line 4). We use two different strategies to generate a particle heading direction.

1) **Odometry heading**: Either we sample for each particle (the loop starting in line 6) odometry and compare a synthesized semantic image S with the semantic segmentation J_t using equation 1 (line 17).
2) **Visual compass**: Or we apply for each particle an image comparison between J_t and multiple S with different directions and use the direction with highest image similarity as the particles direction (similar to a visual compass).

Algorithm 1: Semantic Monte Carlo localization

Data: Semantic 3D map M, image sequence $I_{1:n}$, odometry measures $U_{1:n}$
Result: Particle set P, each $p_j \in P$ is a robot pose

```
   // initialize particle set
1  P ←initParticles()
2  S_cache ← prepareSynthImageCache(M)
   // process each image
3  for t = 1 : n do
4      J_t ← computeSemanticSegmentation(I_t)
5      E_cache ← prepareSimilarityEvaluationCache(M)
6      foreach p_j ∈ P do
           // sample motion from odometry
7          p_j^u ← applyOdometry(p_j, U_t)
           // query cache for existing similarity
              evaluation E of this pose
8          if isInCache(E_cache, p_j^u) then
9              E ← queryCache(E_cache, p_j^u)
10         else
               // query cache for existing
                  synthesized image
11             if isInCache(S_cache, p_j^u) then
12                 S ← queryCache(S_cache, p_j^u)
13             else
14                 S ← synthesizeImage(M, p_j^u)
15                 S_cache ←insertInCache(S_cache, S, p_j^u)
16             end
               // compare images
17             E ← compareImages(J_t, S)
18             E_cache ←insertInCache(E_cache, p_j^u, E)
19         end
           // obtain sampling weight from E
20         w_j ← getWeight(E)
21     end
       // importance sampling
22     P ← importanceSampling(w, P)
23 end
```

Both strategies are described in detail in [3] and are evaluated in section IV. The resulting similarity is used to compute a weight w_j for the particle resampling in line 22. For resampling, we use the low variance sampler from [11].

A major bottleneck for runtime could be the image synthesis (particularly since we use the straight forward approach described in section III-A). To keep runtime feasible, the above algorithmic listing includes two caching stages to exploit the property that many particles are very close to each other: S_{cache} stores all already synthesized images, its lifetime corresponds to the lifetime of the map. The second cache E_{cache} holds the comparison result for a synthesized image and a current camera image. It is cleared with each new camera image. To effectively use these caches, we have to discretize the set of possible poses. Dependent on the application, the synthesized images could also be computed offline in advance. In our experiments, we precompute synthesized images at a spatial resolution of 10 cm in x and y direction and 3° in angular direction.

IV. EXPERIMENTAL SETUP

In order to evaluate the robustness of our algorithm against map distortions and the selected semantic classes, we applied our semantic Monte Carlo localization approach in a set of proof-of-concept experiments.

A. Experimental setup

An indoor **scene** with two labs, some smaller rooms and a corridor serves as experimental environment. Building upon an existing floor plan, we created a metrically exact semantic 2D map and manually augmented it with a set of basic object classes. With the idea of a hand-drawn map in mind, we constrained the number of classes in our semantic map to 10: wall, dividing wall, floor, door, window, table, chair, cupboard, sideboard, and sink. The resulting semantic map is shown in Fig. 1. For our proof-of-concept experiments, we collected camera images, odometry and ground truth pose data of our mobile robot at 14 different positions (see Fig. 1).

For **data acquisition**, we used a skid-steering mobile robot equipped with a *uEye UI-1240ML-C-HQ* RGB camera mounted on a pan-tilt unit; details on our mobile robot can be found in [15]. By using a wide-angle lens, the camera captures images with 76° vertical and 95° horizontal field of view. Since the pan-tilt unit can be moved in the range of approx. -90° to $+90^\circ$, we were able to capture multiple images at one location to enhance the field of view. For each robot position, 7 camera images are semantically segmented, transformed into spherical coordinates, and stitched into a single 268° panoramic image. A high number of camera images is used to avoid distortions at the image borders of raw images (without rectification).

The choice of the **weight matrix** in the holistic image matching is crucial. As the classes wall and floor appear more often in an indoor environment, the score for a correct match was reduced from 1 to 0.25, whereas, more expressive classes have higher impact on the similarity score. A match between wall/dividing wall and floor is scored with -1. Thereby the image similarity receives a huge penalty if the edges between floor and wall do not match in both images, leading to a localization with correct distances to nearby walls. Furthermore, we apply soft class matching as ADE20k contains many similar classes, e.g., chair, armchair, seat, swivel chair, stool, bench, sofa, etc.. All these classes need to be linked to our general class chair, with a matching score between 0 and 1. For our proof-of-concept experiments, the matching scores (and thereby the weight matrix w) are chosen empirically, following the design guidelines presented in section III-C. As a baseline, we use an identity matrix, in which all scores between classes are either 0 or 1. Complete weight matrices are available upon request.

We conduct the following **experiments**:

1) Local and global localization. We run all setups on two tasks: (1) "local" localization with known initial position and 500 particles in the particle filter and (2) "global" localization with unknown start position and 2000 particles. We choose a high number of particles for the known initial position case, since we need to apply high uncertainty (σ=0.5 m) to our robot's odometry due to the severe map distortions (see Fig. 2).
2) Visual compass vs. odometry heading.
3) Scaling of the semantic map with factors ranging between 1.1 and 1.5 in one direction and 0.91 and 0.67 in the other direction (see Fig. 2).
4) Shear of the semantic map in both directions with a factor between 0.1 and 0.5 (see Fig. 2).
5) Variation of classes in the map with three cases: (1) construction basic: containing wall, dividing wall and floor; (2) construction extended: containing case (1) and door and window; (3) furniture: containing table, chair, cupboard, sideboard and sink.
6) Variation of panoramic field of view with following steps: 50°, 100°, 150°, 200° and 268°.

We conduct 10 runs of each experiment to account for the stochastic nature of Monte Carlo localization. We use the Euclidean distance to the ground truth position for evaluation metric and compare against plain odometry measures of our mobile robot.

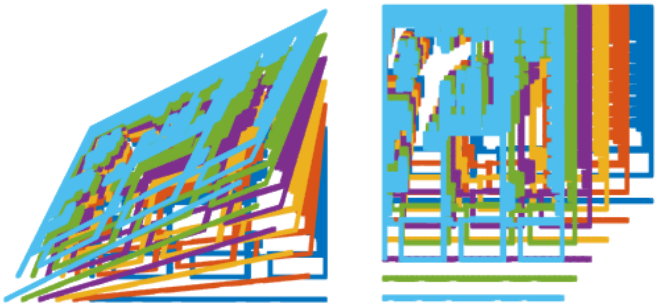
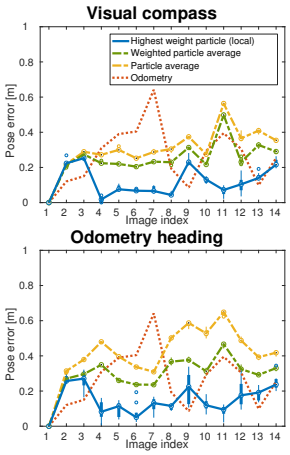


Fig. 2. Left: Original 2D semantic map (see Fig. 1) is shown dark blue. The five overlaid images depict a map sheared with factors ranging between 0.1 and 0.5. Right: The same for scaled maps with factors ranging from 0.91 to 0.67 in x direction and 1.1 to 1.5 in y direction.

B. Experimental results

Evaluation results of the presented approach using the **visual compass and odometry heading** strategy are shown in Fig. 3 (left). Both strategies result in lower maximum and average pose error than the plain odometry. The visual compass based strategy achieves high localization accuracies with distances to ground truth positions constantly below 30 cm, at some locations close to zero. Since our holistic image comparison depends on a reasonable alignment between synthesized image S and segmented image J_t, the multiple sampled directions in visual compass result in more precise and repeatable localizations. Comparison of average particle distance and average weighted distance shows, that our image similarity metric is feasible, since particles with a higher weight are closer to ground truth positions than particles with a lower weight. In some (not shown) experiments the odometry heading strategy provided better results, presumably due to visual aliasing that can occur for the visual compass method. In summary, the visual compass strategy achieves the overall better results and is therefore used in all following experimental results.

The influence of **map scaling** (see Fig. 2) is shown in the first row of Fig. 4. As expected, errors rise with growing scaling factors. Up to a scaling by factors 1.40 & 0.71, a reliable localization, with considerably smaller errors than robot's odometry, is possible. Global localization is achieved within the first three images for all cases.



Experiment	Identity matrix		Weighted matrix		
	highest weight	min dist. to GT	highest weight	min dist. to GT	odometry
Unmodified semantic map	0.160 m	(0.023 m)	**0.120 m**	(0.020 m)	0.260 m
Scaling by 1.10 & 0.91	0.267 m	(0.034 m)	**0.176 m**	(0.033 m)	0.488 m
Scaling by 1.20 & 0.83	3.036 m	(0.078 m)	**0.363 m**	(0.052 m)	1.052 m
Scaling by 1.30 & 0.77	6.836 m	(0.671 m)	**0.569 m**	(0.073 m)	1.625 m
Scaling by 1.40 & 0.71	8.376 m	(1.565 m)	**0.686 m**	(0.097 m)	2.197 m
Scaling by 1.50 & 0.67	8.352 m	(1.991 m)	**1.176 m**	(0.122 m)	2.768 m
Shearing by 0.10	0.287 m	(0.028 m)	**0.165 m**	(0.026 m)	0.632 m
Shearing by 0.20	1.396 m	(0.039 m)	**0.391 m**	(0.042 m)	1.213 m
Shearing by 0.30	2.956 m	(0.060 m)	**0.534 m**	(0.056 m)	1.808 m
Shearing by 0.40	3.441 m	(0.152 m)	**0.790 m**	(0.079 m)	2.407 m
Shearing by 0.50	4.063 m	(0.388 m)	**1.234 m**	(0.118 m)	3.006 m
200° field of view	0.160 m	(0.024 m)	**0.126 m**	(0.023 m)	0.260 m
150° field of view	0.852 m	(0.169 m)	**0.189 m**	(0.023 m)	0.260 m
100° field of view	0.969 m	(0.210 m)	0.284 m	(0.026 m)	**0.260 m**
50° field of view	1.712 m	(0.082 m)	0.367 m	(0.051 m)	**0.260 m**
Classes construction ext.	**0.174 m**	(0.024 m)	0.186 m	(0.024 m)	0.260 m
Classes construction basic	1.956 m	(0.025 m)	**0.252 m**	(0.027 m)	0.260 m
Classes furniture	2.674 m	(1.069 m)	0.678 m	(0.266 m)	**0.260 m**

Fig. 3. Left: Results for particle heading strategies visual compass and odometry heading (described in III-D) applied to the unmodified semantic map. Each blue (green, yellow) curve shows the median result of the (x,y) pose error of the particle with the highest weight (weighted average particle distance, average particle distance) over ten runs and is augmented with boxplots for each image of the sequence (25th and 75th percentiles, whisker scale is 1.5, outliers are shown as circles). Right: Comparison of highest weight particle, minimal distance particle and odometry by means of mean Euclidean distance to ground truth positions over ten runs under usage of the visual compass method for local localization.

The influence of **map shearing** (see Fig. 2) is shown in the second row of Fig. 4. Our semantic Monte Carlo localization achieves small localization errors and high repeatability up to a shearing with factor 0.4. A shearing factor of 0.5 leads to severe localization error fluctuations between single runs. Furthermore, localization gets lost in the last three images. This is caused by ambiguous image similarity results for image 12, leading to different resulting paths. Nevertheless, global localization is achieved for all cases within first three images. Summarizing, our semantic Monte Carlo localization achieves remarkably good localization results for map shearing with factors up to 0.4.

The influence of **horizontal field of view** is shown in the third row of Fig. 4. With only $50°$ field of view, a wide fluctuation between localization errors are observable, since information provided by the image is very limited. Still, our semantic Monte Carlo localization algorithm is able to localize from an unknown initial position, as in all other cases. With more than $150°$ the image contains enough information for a reliable localization. Summarizing, a large field of view panoramic image is crucial for reliable localization results.

The influence of **classes within the semantic map** is shown in the fourth row of Fig. 4. Localization with only furniture classes did not work in this experiment: Without walls, the synthesized images contain a lot of objects from adjacent rooms, which are not present in the real camera images (since there are walls, of course). Furthermore a lot of information is missing in scenes, for which our robot mainly observes walls and floor. On the other hand, walls, dividing walls and floor ("construction basic") offer enough information for a coarse but reliable localization in the margin of our robot's odometry. With additional information from classes window and door ("construction extended"), localization accuracies rise as expected. Summarizing, our semantic Monte Carlo localization depends on the classes wall and floor, which offer enough information for a rough localization and prevent confusing views into adjacent rooms. Additional classes facilitate a precise localization.

The achieved localization results are summarized in Fig. 3 (right). The baseline approach, an identity matrix for image comparison, is sensitive for map perturbations, thus reliable results are only achieved within slightly modified maps. The minimal particle distance shows, that the ground truth position is not lost in most cases, but the particles rather diverged into multiple clusters. Sometimes the highest weight particle is located in a cluster far from the ground truth position because the identity matrix leads to ambiguous similarity measurements. Therefore, the choice of the weight matrix is essential for our semantic Monte Carlo localization, since it considerably improves robustness against map perturbations.

V. Discussion

In this paper, we presented a semantic Monte Carlo localization algorithm based on holistic image comparison of a synthesized map view and the result of a semantic image segmentation. A set of prove-of-concept experiments showed its ability to localize a mobile robot, equipped only with a RGB camera and odometry sensors, within a given 2D semantic map. Furthermore, we applied a wide range of perturbations to the given semantic map, in order to investigate our system's robustness. Experimental results are promising, since reliable local and global localization is achieved in a wide range of map deformations. Additionally, our semantic Monte Carlo localization algorithm does not depend on distinctive classes or landmarks, but rather on common classes like walls and floor, with which it achieves a rough localization. Also, our system depends on a large field of view panoramic image.

Currently, a key part of our localization algorithm, holistic image comparison, is handcrafted and uses empirically chosen parameters (e.g. weight matrix w). Presumably, our system could greatly benefit from better image similarity measurements, e.g., using (supervised) learning of class

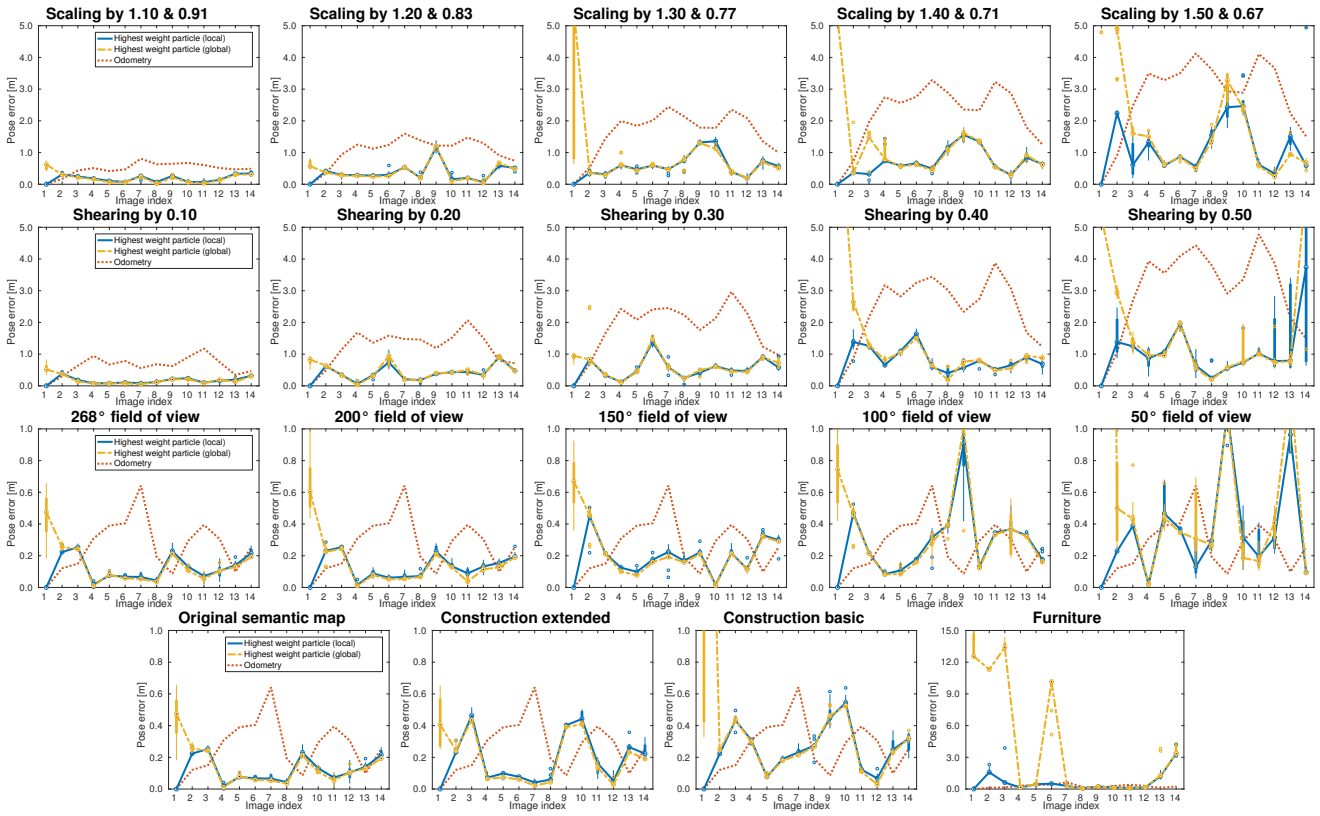


Fig. 4. First row: Results for synthetic scaling of the original semantic map (see Fig. 2). Second row: Results for synthetic shearing of the map (see Fig. 2). Third row: Results for different horizontal field of views. Fourth row: Results for variation of classes in the semantic map with three cases (see IV-A for more information). Each orange curve shows the (x,y) pose error of the robot's odometry. Each blue respectively yellow curve shows the median result of the (x,y) pose error of the particle with the highest weight over ten runs and is augmented with boxplots for each image of the sequence (25th and 75th percentiles, whisker scale is 1.5, outliers are shown as circles) in the case of local respectively global localization.

weights and mutual influences. Further, the evaluation of the different strategies for heading direction estimation (visual compass or odometry) motivates the development of a combined approach as direction for future work.

Based on the achieved results, our future work aims at applying our semantic Monte Carlo localization algorithm to real world scenarios. One promising use case is localization in truly hand drawn sketches, since our localization algorithm is robust against map deformations, enabling a variety of applications in human robot interaction, e.g. navigation of autonomous transportation robots in a construction site based on a sketch of the surroundings. Another reasonable use case is localization within a building based on fully automatic preprocessed floor plans.

References

[1] R. Möller, "A model of ant navigation based on visual prediction," *Journal of Theoretical Biology*, vol. 305, pp. 118 – 130, 2012.

[2] ——, "Local visual homing by warping of two-dimensional images," *Robotics and Autonomous Systems*, vol. 57, no. 1, pp. 87 – 101, 2009.

[3] S. Schubert, P. Neubert, and P. Protzel, "Towards camera based navigation in 3d maps by synthesizing depth images," in *Proc. of. Towards Autonomous Robotic Systems (TAROS)*, 2017.

[4] R. W. Wolcott and R. M. Eustice, "Visual localization within lidar maps for automated urban driving," in *2014 IEEE/RSJ Int. Conference on Intelligent Robots and Systems*, Sept 2014, pp. 176–183.

[5] G. Pascoe, W. Maddern, A. D. Stewart, and P. Newman, "Farlap: Fast robust localisation using appearance priors," in *2015 IEEE Int. Conf. on Robotics and Automation (ICRA)*, May 2015.

[6] T. Caselitz, B. Steder, M. Ruhnke, and W. Burgard, "Monocular camera localization in 3d lidar maps," in *2016 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS)*, Oct 2016, pp. 1926–1931.

[7] B. Kuipers, "The spatial semantic hierarchy," *Artificial Intelligence*, vol. 119, no. 1, pp. 191 – 233, 2000.

[8] S. Vasudevan, S. Gchter, V. Nguyen, and R. Siegwart, "Cognitive maps for mobile robotsan object based approach," *Robotics and Autonomous Systems*, vol. 55, no. 5, pp. 359 – 371, 2007.

[9] N. A. Atanasov, M. Zhu, K. Daniilidis, and G. Pappas, "Semantic localization via the matrix permanent," in *Robotics: Science and Systems (RSS)*, July 2014. [Online]. Available: http://terraswarm.org/pubs/305.html

[10] R. Anati, D. Scaramuzza, K. G. Derpanis, and K. Daniilidis, "Robot localization using soft object detection," *2012 IEEE International Conference on Robotics and Automation*, pp. 4992–4999, 2012.

[11] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005.

[12] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," *CoRR*, vol. abs/1612.01105, 2016.

[13] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, and A. Torralba, "Scene parsing through ade20k dataset," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[14] D. A. Forsyth, J. Malik, M. M. Fleck, H. Greenspan, T. K. Leung, S. Belongie, C. Carson, and C. Bregler, "Finding pictures of objects in large collections of images," in *Proceedings of the International Workshop on Object Representation in Computer Vision II*, ser. ECCV '96. London, UK, UK: Springer-Verlag, 1996, pp. 335–360. [Online]. Available: http://dl.acm.org/citation.cfm?id=645306.648641

[15] S. Lange, D. Wunschel, S. Schubert, T. Pfeifer, P. Weissig, A. Uhlig, M. Truschzinski, and P. Protzel, "Two autonomous robots for the dlr spacebot cup - lessons learned from 60 minutes on the moon," in *Proc. of ISR: International Symposium on Robotics*, 2016.