APPEARANCE BASED SELF LOCALIZATION AND NAVIGATION USING PLACE MEMORY

by

Esen Yel

B.S., Electrical and Electronics Engineering, Boğaziçi University, 2014

Submitted to the Institute for Graduate Studies in
Science and Engineering in partial fulfillment of
the requirements for the degree of
Master of Science

Graduate Program in Electrical and Electronics Engineering
Boğaziçi University
2016

APPEARANCE BASED SELF LOCALIZATION AND NAVIGATION USING PLACE MEMORY

APPROVED BY:

Prof. H. Işıl Bozma ..... (Thesis Supervisor)

Prof. Yağmur Denizhan .....

Prof. Hakan Temeltaş .....

DATE OF APPROVAL: 20.07.2016

ACKNOWLEDGEMENTS

First of all, I would like to thank my supervisor, Prof. H. Işıl Bozma for all of her support and ideas during this work. She introduced me to the research topic, taught how to conduct research and present an academic work, which was an irreplaceable experience for me.

I also should show gratitude to my thesis committee members, Prof. Yağmur Denizhan and Prof. Hakan Temeltaş for their valuable ideas.

I want to thank Dr. Hakan Karaoğuz, a former member of ISL, for all of his ideas on my research topic. I want to thank members of ISL, namely Mahmut Demir, Deniz Şenel, Gökçe Erdem, Çağatay Odabaşı, Berkan Höke and Kadir Cumali for their support and assistance during this work. Additionally I want to thank to Halil Samed Çıldır for the technical support and I would like to express special thanks to Kadir Türksoy for his extensive assistance on this work.

This work has been supported in part by TUBITAK EEEAG-115E380.

Lastly, I should thank to all of my friends and my family: my father, my sister Başak and her husband Özgür for their support. Everything would be much harder without them.

I would like to dedicate this thesis to the most wonderful person I have ever known, my mother, who passed away three years ago. I couldn't have even imagined being who I am right now without her. She was my brightening star and the luck of my life, and will always stay like this.

ABSTRACT

APPEARANCE BASED SELF LOCALIZATION AND NAVIGATION USING PLACE MEMORY

This thesis is concerned with appearance based self-localization and navigation. First, we assume a place-aware robot. Such a robot can recognize a place (i.e. a particular room etc.) after having learned it based on a collection of appearances. A novel approach to self-localization is presented in this framework. In this approach, all the respective learned appearances contribute simultaneously to the location estimation - assuming them to be tagged with their respective locations during learning. With the extensive set of experiments, it is shown that the proposed approach can be used as a solution to mobile robot self-localization. Secondly, we introduce a novel approach in which the robot couples place awareness and self-localization. The robot first recognizes the place and then uses the retrieved appearances from its place memory. Experimental results demonstrate the advantages of coupling place awareness and self-localization. Lastly, the proposed approach is considered in appearance-based navigation.

ÖZET

YER HAFIZASI KULLANARAK GÖRÜNTÜ BAZLI OTO-KONUMLANDIRMA VE YÖNGÜDÜM

Bu tezde görüntü odaklı oto-konumlandırma ve yöngüdüm ile ilgilenilmiştir. İlk olarak, robotun yer farkında olduğu varsayılmıştır. Bu tip bir robot kendi yerini (mesela bulunduğu odayı vs.), o yere oraya ait görüntüler toplamı şeklinde öğrendikten sonra tanıyabilir. Bu çerçevede, oto-konumlandırma için özgün bir yaklaşım geliştirilmiştir. Bu yaklaşımda, robotun konum kestirimine öğrenilmiş ve ilgili konum bilgileri ile öğrenme esnasında etiketlenmiş görüntülerin tamamı eş zamanlı olarak katkıda bulunurlar. Yapılan kapsamlı deneyler ile önerilen yaklaşımın oto konumlandırma problemine çözüm olarak kullanılabileceği gösterilmiştir. İkinci olarak yer farkındalık ile oto-konumlandırmayı birleştiren bir yaklaşım açıklanmıştır. Burada robot öncelikle olduğu yeri tanır ve yer hafızasındaki tanıdığı yere ait görüntüleri kullanır. Deneysel sonuçlar yer farkındalık ile oto-konumlanırmayı birleştirmenin avantajlarını ortaya koymuştur. Son olarak önerilen yaklaşım görüntü odaklı yöngüdüm probleminde düşünülmüştür.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS . . . iii
ABSTRACT . . . iv
ÖZET . . . v
LIST OF FIGURES . . . viii
LIST OF TABLES . . . xi
LIST OF SYMBOLS . . . xii
LIST OF ACRONYMS/ABBREVIATIONS . . . xiv
1. INTRODUCTION . . . 1
1.1. Contributions . . . 2
1.2. Outline . . . 2
2. APPEARANCE BASED SELF-LOCALIZATION . . . 4
2.1. Related Literature . . . 4
2.2. General Approach . . . 5
2.3. Appearance Based Localization . . . 6
2.3.1. Self-Localization . . . 6
2.3.2. Considering All Appearances . . . 8
2.4. Experimental Results . . . 10
2.4.1. Grid-Based Learned Appearances . . . 10
2.4.2. Learned Appearances from Curvilinear Path . . . 15
3. COUPLING PLACE AWARENESS AND SELF-LOCALIZATION . . . 21
3.1. Place Awareness . . . 22
3.2. Place Verification . . . 23
3.3. Experiments . . . 24
3.3.1. Experiments with COLD Dataset . . . 24
3.3.2. Experiments with Turtlebot . . . 31
4. APPEARANCE-BASED NAVIGATION . . . 39
4.1. Coupling Navigation & Self-Localization . . . 40
4.2. Navigation Experiments . . . 43

5. CONCLUSION 47
REFERENCES 49
APPENDIX A: BUBBLE SPACE 53
APPENDIX B: PLACE DETECTION MODIFICATION 57
APPENDIX C: TURTLEBOT ROBOT 58
C.1. Hardware 58
C.2. Software 60
C.3. User's Guide 60
C.3.1. Powering the robot 60
C.3.2. Remote Connection 60
C.3.3. Operating the Robot 61
APPENDIX D: SOFTWARE USER GUIDE 62
D.1. Grid Based Self-Localization 62
D.2. Coupling of Place Awareness and Self-Localization 64
D.3. Navigation 67

LIST OF FIGURES

Figure 2.1. Self-localization. 7

Figure 2.2. Learned locations. 11

Figure 2.3. Sample appearances during learning and testing. 12

Figure 2.4. 1m × 1m grid-based learning of a 6m × 7m place and localization results. 13

Figure 2.5. Variation of μ_e (× symbol) and σ_e^2 (vertical lines) with respect to translational offset of test locations. 15

Figure 2.6. Places in Freiburg lab. 16

Figure 2.7. Localization results when the testing and learning is done along similar path. 17

Figure 2.8. Localization results when the testing and learning is done along different path. 18

Figure 3.1. Sample visual data from Freiburg Lab. 26

Figure 3.2. Learned places (shown in red) in first-time visits. 26

Figure 3.3. Detected places (shown in red) in second-time visits. 27

Figure 3.4. κ values of all possible recognized places. 29

Figure 3.5. Sample appearance from Saarbrucken Lab. 29

Figure 3.6. Learned places (shown in red) in first-time visits. 30

Figure 3.7. Detected places (shown in red) in testing stage. 31

Figure 3.8. Sample appearance. 31

Figure 3.9. Learned places (shown in red) in first-time visits in Ljubljana site. 33

Figure 3.10. Detected places in second-time visits in Ljubljana site. 34

Figure 3.11. Sample images taken from turtlebot with V360 camera in three different places 36

Figure 3.12. Detected places (shown in red) in first-time visits. 37

Figure 3.13. Detected places (shown in red) in second-time visits. 38

Figure 4.1. Appearance-based navigation. 40

Figure 4.2. Navigation based on estimated location, heading and goal using appearance knowledge. 41

Figure 4.3. Navigation experiments setup 42

Figure 4.4. Navigation paths 46

Figure A.1. Sample views and corresponding bubble surfaces. 55

Figure C.1. Turtlebot robot. 58

Figure C.2. Wiring diagram of turtlebot. 59

Figure D.1. Main window of ISLFramework user interface. 62

Figure D.2. Kinect image acquisition dialog box. 63

Figure D.3. Image processing dialog box. 63

Figure D.4. Implementation of coupling place awareness and localization . . . 64

LIST OF TABLES

Table 2.1. Localization Results and Comparison. MSL - Maximum similarity location. NA - Not available. Seq: Sequentiality. 14

Table 2.2. Comparative localization performance with similar learning and test paths. 19

Table 2.3. Comparative localization performance with different learning and test paths. 20

Table 3.1. Place recognition and localization results in Freiburg site. 28

Table 3.2. Place recognition and localization results in Saarbrucken site. . . . 32

Table 3.3. Place recognition and localization results in Ljubljana site. 35

Table 3.4. Place recognition and localization results in Turtlebot experiments. 36

Table 3.5. Modified place recognition and localization results 37

Table 4.1. Goal errors for experiments varying in initial and goal positions. . 44

LIST OF SYMBOLS

B	Bubble space
$c \in R^2$	Robot's position
e	Localization error
f	Robot's viewing direction
f_1	Pan angle
f_2	Tilt angle
$\mathcal{F}$	Robot's viewing direction state space
$\hat{g}$	Estimation for goal location
$I(x)$	Bubble descriptor vector at base point x
m	Detected place index
M_p	Number of base points in place p
N_I	Bubble descriptor dimension
N_p	Number of heading directions in from a single base point
$p \in \mathcal{P}$	A learned place
$\hat{p} \in \mathcal{P}$	A recognized place
$\mathcal{P}$	Set of learned places
r_j	Radius of the smallest possible circle around base point x_j
r_{min}	Minimum r_j value in a place
w_j	Weight value associated with base point x_j
w	Weighting vector
$\alpha \in S$	Robot's heading
γ_j	Visual similarity measure
μ_e	Localization error mean
σ_e^2	Localization error variance
κ	Measure of confidence
τ_p	Place verification threshold
τ_d	Descriptor difference threshold

τ_l	Linear difference between the goal and current location threshold
τ_κ	Coherency threshold
τ_e	First point coherency threshold
ρ_j	Parameter associated with the denseness of base points
$\hat{c} \in R^2$	Estimation for robot's position
$J_p : R^2 \to R^{\geq 0}$	Energy function of place p
$\delta\alpha$	Heading increment
$\mathcal{X} \subseteq R^2 \times S^1$	Base space
$x = \begin{bmatrix} c^T & \alpha \end{bmatrix}^T \in \mathcal{X}$	Base point
$\delta\hat{\alpha}(t+\delta t)$	Estimated heading of the robot between time t and $t+\delta t$
$\delta_{c_p}(t+\delta t)$	Location change of the robot between time t and $t+\delta t$
$\delta_g(t+\delta t)$	Position difference between the robots position at time $t+\delta t$ and the goal point
$\phi(\hat{c}_p, \delta\hat{\alpha}, \hat{g})$	Angular and linear distance metric for navigation

LIST OF ACRONYMS/ABBREVIATIONS

COLD	COsy (Cognitive Systems for Cognitive Assistants) Localization Database
HOG	Histogram of Oriented Gradients
ISL	Intelligent Systems Laboratory
MSL	Maximum Similarity Location
RIF	Robust Invariant Feature
ROS	Robot Operating System
SIFT	Scale Invariant Feature Transform
SLAM	Simultaneous localization and mapping
TSC	Topological Spatial Cognition

1. INTRODUCTION

Robot localization is an integral problem in mobile robotics because a robot needs to know its position in order to perform many tasks like navigation or manipulation. Appearance plays a key role in localization - as odometric data may not be always available or reliable all the time. The robot estimates its position based on matching its current visual data with those of previously learned locations. This thesis considers appearance based self localization for place aware robots. First, a novel approach for self-localization problem is defined in which all the respective learned appearances contribute simultaneously to the location estimation - assuming them to be tagged with their respective locations during learning. This is achieved via optimizing an objective function in which each learned location affects the position estimate with a weight that depends on its similarity in the appearance space. Second, a novel approach is proposed in which the robot couples place awareness and self localization. A place aware robot is to be able to recognize a place (i.e. a particular room etc.) after having learned it based on a collection of appearances. This is achieved using the TSC model. The TSC model entails a long-term place memory and a set of processing modules as to enable a robot to build its knowledge of places and refer to this knowledge autonomously. As places refer to spatial regions, recognizing a place only coarsely designates its whereabouts. Due to perceptual aliasing, the robot may not always correctly recognize a place. Self-localization is activated if the robot recognizes its current whereabouts as one of places that it has learned - as retained in its place memory.

Lastly, a vision-guided navigation approach is also proposed within the scope of this work. In navigation approach, the objective of the problem is to enable the robot to navigate through a goal location based on the appearances it takes from its environment. For that purpose, robot constantly estimates its location using the proposed self localization approach and find its relative location and heading towards the goal location. Its linear and angular velocities are also updated continuously based on the angular and linear distance to the goal location to provide a smooth motion.

It is also checked visually if the robot reaches the goal location or not. The set of navigation experiments are also conducted to support the theory.

1.1. Contributions

The contributions of that work can be summarized as follows:

- Localization: With the proposed novel self-localization approach, all the learned related knowledge is used to localize the robot. Therefore, the localization error is improved and seems to be less affected by the coverage denseness of each place during learning. As the proposed approach is applicable even if the incoming appearance data is not of sequential nature, the self-localization can be used when starting a mission or if the incoming images are sparse or the robot is kidnapped.
- Coupling place-awareness with localization: Coupling place awareness with self-localization makes it possible to limit the search process to locations pertaining to one place, which makes the approach computationally more efficient. In addition, it is shown that the localization results can be used to compute a measure of confidence whether the place was correctly recognized or not.
- Vision-based navigation: With the vision-based navigation, the robot can navigate through the target location by just using the appearance data used in self-localization approach.

1.2. Outline

This thesis is organized as follows: In Chapter 2, the appearance based self localization is presented along with extensive experimental results conducted both on real robot and using the COLD benchmark. In Chapter 3, the coupling of place awareness with self-localization is explained along with experimental results. In Chapter 4, the appearance based navigation using self localization is proposed. This work concludes in Chapter 5 with a brief summary. For completeness, the mathematical formulation

of bubble space representation for omnidirectional images is presented in Appendix A. The modifications made in the previous place detection approach are explained in Appendix B. The description of Turtlebot robot is given in Appendix C. Finally, the installation and the usage of the developed software are explained in Appendix D.

2. APPEARANCE BASED SELF-LOCALIZATION

Robot localization is an integral problem in mobile robotics - since a robot needs to know its position in order to perform many tasks like navigation or manipulation. In this chapter, we focus on the appearance based self localization based with a place aware robot.

2.1. Related Literature

In related work in localization, it is observed that what is meant by 'localization' varies. In most work, localization refers to the exact position coordinates in some global reference system. Appearance plays a prevalent role in this since purely odometry based methods are prone to errors. The most common approach is to build a local map and then to align it with a previously learned global map. For example, 'metric SLAM' methods jointly maintain an estimate of robot's pose and a set of map landmarks using Kalman or particle filters [1]. Others simply combine map matching with odometric data and the known motion dynamics [2–6]. If an initial estimate of the robot's position is available, the problem is posed as a tracking problem based the appearance data in which the robot's position is updated probabilistically [3]. As all these methods require odometry, motion model and the sequentiality of appearance data in order to restrict the alignment or the search to physically plausible positions, they cannot be applied in cases where they are not available at all times.

Alternatively, appearance-based approaches have been motivated by biological systems like humans who can estimate their locations even if they do not have such knowledge. Here, the goal of localization is to find the closest appearance. The most common approach is based on optimizing a similarity measure that encodes a score for each possible position of the local map with respect to the known global map - using Extended Kalman filter [7], maximum likelihood estimate [8–10] or cosine distance and voting [11]. The scores are based on the distances from each of the features in the local

map to the closest feature in the global map. If the appearance data is of sequential nature or robot's motion model is known, then this knowledge is used to limit the search to highly likely locations [12,13]. The matching of local features is critical since it affects both localization error and time requirements. Approaches such as using a smaller set of landmarks have been proposed for this [14, 15]; however robustly detecting such entities is still an open problem. Alternatively, there are also localization approaches that do not consider maps. These are computationally simpler as there is no need for constructing a global or a local map or aligning the two. Rather, localization is done simply via comparing the incoming descriptors with those previously learned and finding the closest appearance with each location defined based on a single image [16], a set of views or panoramic images [17–19]. The search process may be simplified if the sequentiality of the appearance data can be assumed or localization is done only coarsely by defining the localization problem as finding a set of nearby locations. In all, since the best match is used, the localization error increases if the true location is not close to one of the learned positions. Thus, the environment needs to be densely represented in order to minimize the localization error. This in turn increases both memory and processing requirements considerably.

2.2. General Approach

In this work, we also consider appearance based self-localization without maps. We assume a place-aware robot that is to able to recognize a place (i.e. a particular room, etc.) after having learned it based on a collection of appearances - as previously presented in [20, 21]. Self-localization is defined as estimating its position in a recognized place. We also assume that learned appearances have been tagged with location data. The tagging needs to be done only during learning and can be obtained in each place locally using odometry, external guidance or even via gps. We introduce a novel approach in which all the appearances associated with this place contribute simultaneously to the location estimation. As such, it differs from related work in which only the best matching location is used. This is achieved via minimizing an objective function that is constructed as the weighted sum of all the learned positions. The weights encode

the similarity of appearances from the current location to be estimated to those from learned locations. The advantages of the proposed approach are two-fold: First, since all the learned related knowledge is used, the localization error is improved and seems to be less affected by the coverage denseness of each place during learning. Second, it is applicable even if the incoming appearance data is not of sequential nature - as may be the case when starting a mission, incoming appearances are sparse or the robot is kidnapped.

2.3. Appearance Based Localization

2.3.1. Self-Localization

Consider a robot in a particular place p. The place p is assumed to have been learned previously based on a collection of appearances from M_p locations $c_{pj} \in R^2$, $j = 0, \ldots, M_p - 1$ and N_p headings $\alpha_{pjk} \in S^1$, $k = 0, \ldots, N_p - 1$:

$$\alpha_{pjk} = \begin{cases} \alpha_{pj} \in S^1 & k = 0 \\ \alpha_{pj} + k\delta\alpha & 0 < k < N_p \end{cases} \tag{2.1}$$

We refer to a particular location-heading combination as a base point which can be shown by $x_{jk}(p) = \begin{bmatrix} c_{pj}^T & \alpha_{pjk} \end{bmatrix}^T$. The number of locations M_p should enable the robot to see this place with sufficient coverage. At the same time, the N_p parameter should enable the robot to observe the sufficient portion of its surroundings from a fixed location. Getting a 360° view each location alleviates perceptual aliasing to some extent. As such, there is a lesser chance for having appearances from two different locations being completely the same. If the camera's field of view covers 360° (as is the case with an omnidirectional camera), $N_p = 1$. The data collection could have been done either under guided operation or autonomously. The appearance data from each base point $x_{jk}(p)$ is then encoded by a N_I-dimensional descriptor $I(x_{jk}(p))$. The descriptors can be constructed using a variety of local or global features - as long as nearby appearances generate similar descriptors. They are tagged with their respective

base points $x_{jk}(p)$. This may be repeated over time in order to capture appearance-related changes in each place. In this case, the place knowledge is simply updated to include the new descriptors as well.

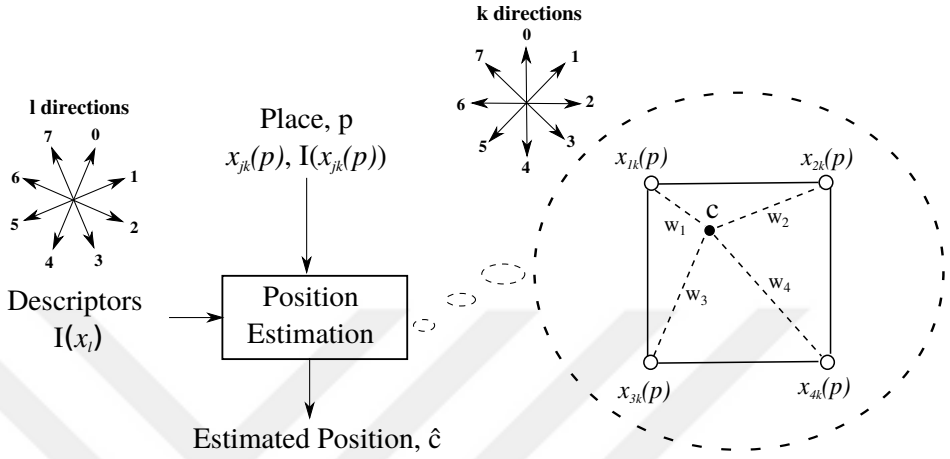


Figure 2.1. Self-localization. Each place p is learned as a collection of appearances and their locations. As the robot estimates its position, the knowledge from each learned location weights in depending on its visual similarity to the robot's current surroundings.

The place-aware robot is assumed to have correctly recognized the place p - based on the current appearance data as it relates to the learned appearance data [21]. However, it does not know its exact location $c \in R^2$ and the heading at that position as $\alpha \in S^1$ in this place p. The goal is to estimate $\hat{c} \in R^2$ as shown in Fig. 2.1. The self-localization approach consists of two stages: The first step of the proposed approach is to consider appearances from a full view of its surroundings as much as possible. For this, it looks around through a series of N_p viewing angles $\alpha_l \in S^1$, $l = 0, 1, \cdots, N_p - 1$ with $\delta\alpha = \frac{2\pi}{N_p}$ increments in order to get a 360° view:

$$\alpha_l = \begin{cases} \alpha & l = 0 \\ \alpha + l\delta\alpha & 0 < l < N_p \end{cases} \tag{2.2}$$

The corresponding base points are $x_l = \begin{bmatrix} c^T & \alpha_l \end{bmatrix}^T$. The appearance data associated with each base point x_l is encoded by a N_I-dimensional descriptor $I(x_l)$.

The second step is to estimate $\hat{c}$ via relating the descriptors $\{I(x_l)\}_{l=1,\ldots,N_p}$ with those associated with p. The estimation is based on solving an optimization problem

$$\hat{c} \in \arg\min_{c \in R^2} J_p(c) \tag{2.3}$$

where $J_p : R^2 \rightarrow R^{\geq 0}$ is a properly constructed objective function. The necessary condition to minimize J_p is as follows:

$$\nabla_c J_p(c)|_{\hat{c}} = 0 \tag{2.4}$$

As such, the construction of J_p becomes critical.

2.3.2. Considering All Appearances

The function J_p is constructed in a manner so that each learned base point c_{pj} in place p affects the position estimation by an amount proportional to the similarity with the associated descriptors with those of the unknown position c as weighted by a parameter $w_j \in R^{\geq 0}$:

$$J_p(c) = \sum_{j=0}^{M_p - 1} \frac{1}{2} w_j (c_{pj} - c)^T (c_{pj} - c) \tag{2.5}$$

This model has a spring-like analogy as shown in Figure 2.1. Consider the sample scenario where place p has been learned based on the base points from four locations $j = 1, \ldots, 4$ and eight headings $k = 1, \ldots, 8$ as shown in 2.1. All the thirty-two base points affect the estimated position of the robot with the weight associated with each base point (w_1, w_2, w_3, w_4 respectively). The weighting parameter encodes several factors that influence how individual known locations should be taken into account:

$$w_j = e^{-\frac{1}{\rho_j}\gamma_j^2} \tag{2.6}$$

- The value γ_j measures the similarity between N_p descriptors for known c_{pj} position and descriptors from unknown position. When the robot does not know its initial orientation, it selects the orientation which makes the similarity with the sensory data from other base points maximum:

$$\gamma_j = \min_{k=0}^{N_p-1} \sum_{l=0}^{N_p-1} \sqrt{(I(x_l) - I(x_{j(k\oplus l)}(p))^2} \tag{2.7}$$

  where $k \oplus l = (k + l) \mod N_p$. As the value of γ_j goes to 0, the similarity increases. Since the robot does not know its initial orientation, it turns around itself and save its data from each orientation. By definition of γ_j, the similarity is measured from the orientation of robot with sensory data most similar to the learned base points.
- The parameter $\rho_j > 0$ is proportional to the ratio of the radius of the largest circular area centered at each known location c_{pj} that is not overlapping with those of nearby known locations to radius of the largest circular area centered at a base point with the smallest circle around it. Hence, the weights of locations associated with relatively densely sampled regions in a place will be relatively smaller. As such, the algorithm will be applicable even in places that have been learned with non-even sampling of learning locations. So, a dense area of learning positions will not be weighted more than those from regions containing fewer learning positions. For evenly sampled locations ρ_j value is the same for each base point.

Using Equation 2.4 and the definition of J_p in Equation 2.5, with some manipulation, the location estimate $\hat{c}$ is derived to be equal to:

$$\hat{c} = \frac{1}{\sum_{j=0}^{M_p-1} w_j} \begin{bmatrix} c_0 & \dots & c_{M_p-1} \end{bmatrix} w \tag{2.8}$$

where the weighting vector $w \in R^{M_p}$ is constructed:

$$w = \begin{bmatrix} w_0 & \dots & w_{M_p-1} \end{bmatrix}^T \tag{2.9}$$

As such, similar to all the previous approaches, we expect the localization error $|\hat{c} - c|$ to be low only in cases in which the true location c lies inside a region that is covered by the learned locations. However, differing from them, it is expected to be less affected by the denseness of the coverage. Furthermore, its variance is expected to be smaller as all the appearances weight in the self-localization.

2.4. Experimental Results

We have conducted an extensive set of experiments. In all, the robot is assumed to learn and recognize the respective places based on previous work on topological place recognition [21, 22]. The descriptors are constructed using bubble space representation [22] due to its demonstrated advantages such as nearby appearances generating similar descriptors, being rotationally invariant, incorporating any number of observations while preserving their relative S^2 geometry. However, it should be emphasized that the proposed localization approach is in no way dependent on this particular choice and thus can be used with other suitable descriptors – as preferred. During learning, the descriptors are tagged with relative position information. In the tests, the localization error is defined as $e = |c - \hat{c}|$ with mean μ_e and variance σ_e^2. We also consider the accuracy A that measures the percentage of test points with localization error e less than the corresponding learning distance in that place.

2.4.1. Grid-Based Learned Appearances

The first set of experiments are done in 6m $\times$ 7m place (our lab). A Turtlebot robot equipped with Kinect sensor learns the place as a collection of appearances from a set of grid locations. The grid denseness is varied from 0.5m $\times$ 0.5m (denser) to sparser 3m $\times$ 3m. For example, with a 1m $\times$ 1m grid, there are 16 learning locations

as shown in Figure 2.2. At each location, the robot looks around from $N_p = 8$ different orientations - in order to have 360° coverage as Kinect's field of view is 57.8°. As such, the visual data from consecutive two orientations overlap slightly. Appearances from a sample 11th location are as shown in Figure 2.3(a). The descriptors are constructed using 5 filter responses, hue and depth and they are of dimension $N_I = 700$. The learning process is repeated at 2 different times with 2 month time interval between them to alleviate the effect of dynamic changes in the place.

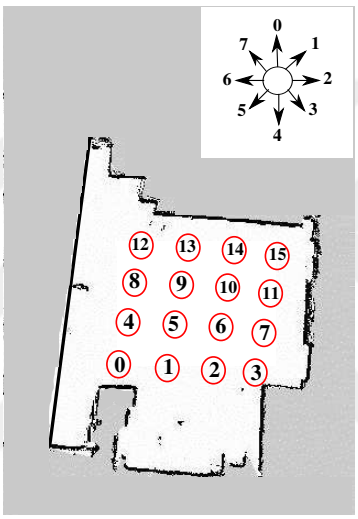
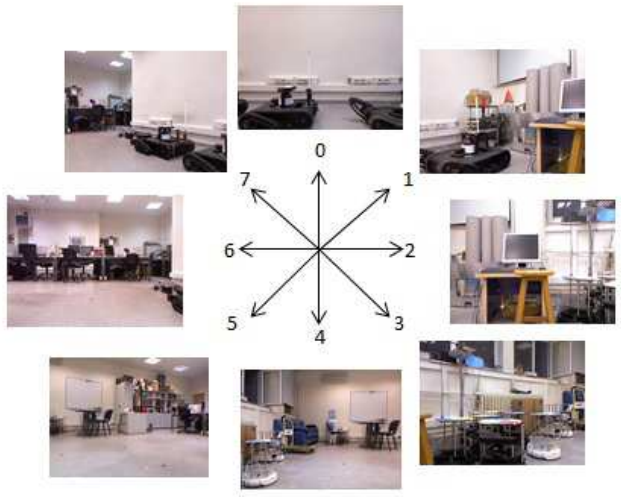
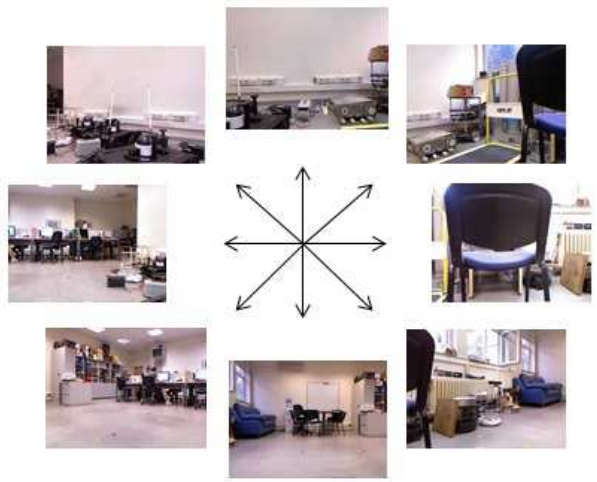


Figure 2.2. Learned locations.

The localization tests are conducted one month after learning. It is observed that appearances associated with the learned positions have changed to some extent - as shown for 11th position in Figure 2.3(b). The robot is tested in both learned and new positions. For example, with learning 1m × 1m grid, the robot is taken to 31 positions - 16 of which are at learned positions as seen in Figure 2.4(a). The estimated positions are as shown in Figure 2.4(b). It is observed that e at the learned positions is minimal despite the dynamic scene changes. In parallel, e goes up slightly in the new locations. With 1m × 1m learning, the mean localization error μ_e is 27.42 cm with a variance of σ_e^2 is 2.71cm. This suggests around 8% positioning uncertainty with respect to the $1m^2$ learning grid cell. We also study how performance is affected as test locations are distanced from learned locations. The tests are conducted with the 1m × 1m learning grid. The robot is moved to a set of test locations that are translated from each learned location with amounts varying from 10-50 cm in random directions. The results are as



(a) Learned appearances at 11th location.

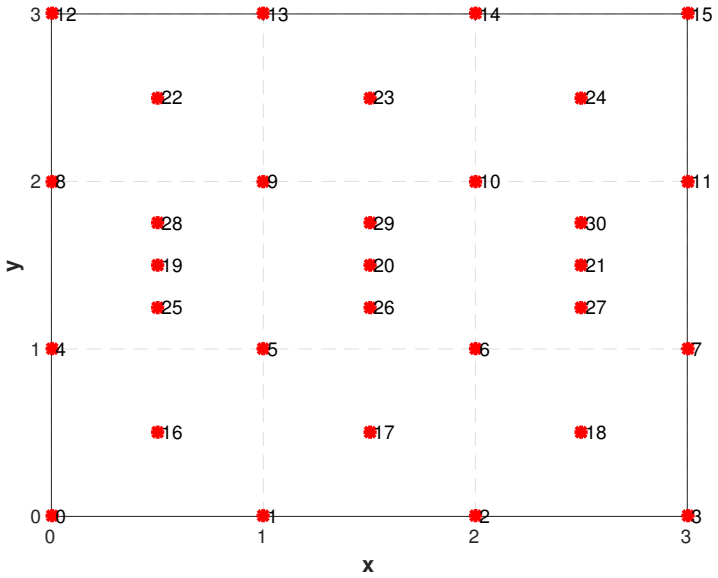


(b) Appearances from same location during testing.

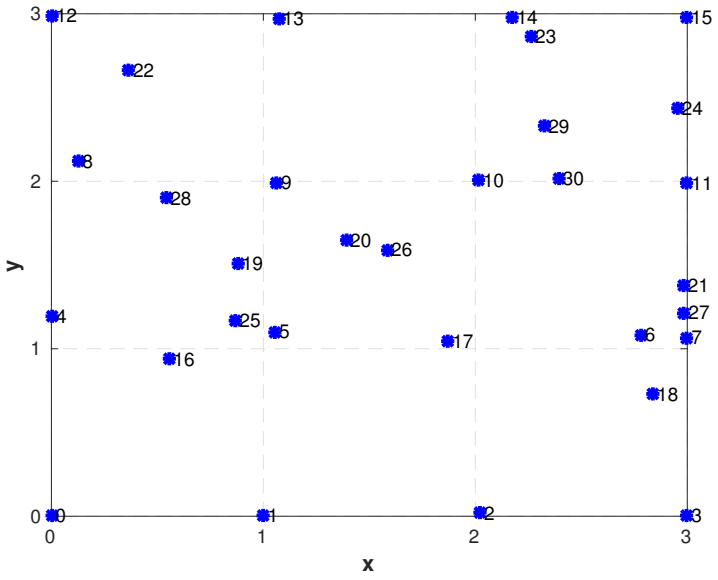
Figure 2.3. Sample appearances during learning and testing.

shown in Figure 2.5. As expected, e increases as translation amount increases. With 10cm translational offset, $\mu_e = 12$cm while with a 50cm translational offset, μ_e is 45cm.

The effect of learning grid denseness is also studied with results as shown in Table 2.1. As expected, both μ_e and σ_e^2 increase as learning grid dimension increases. For example, with 0.5m $\times$ 0.5m grid, $\mu_e = 24.36$cm with $\sigma_e^2 = 2.91$cm. With a grid size 3m $\times$ 3m, they increase to $\mu_e = 1.18$m and $\sigma_e^2 = 59.9$cm respectively. Interestingly,



(a) Test locations. Those at grid locations have been learned while others are new.



(b) Estimated locations (index as shown). The mean localization error is 27.42cm.

Figure 2.4. 1m × 1m grid-based learning of a 6m × 7m place and localization results.

the corresponding accuracy A ranges between 88.52-100% which indicates that the proximity between the estimated locations and their true values is in general less than the learning grid dimensions. We also compare our method with three other methods.

Table 2.1. Localization Results and Comparison. MSL - Maximum similarity location. NA - Not available. Seq: Sequentiality.

Method	Dataset	Learning		Seq	$\mu_e(cm)$	$\sigma_e^2(cm)$	$A(\%)$
		Grid Denseness	M_p				
Proposed	Custom	0.5m×0.5m	49	Not Used	24.36	2.91	88.52
		1m×1m	16		27.42	7.41	96.77
		1.5m×1.5m	9		56.89	18.71	96.77
		3m×3m	4		118.89	56.98	100
MSL	Custom	0.5m×0.5m	49	Not Used	27.40	9.66	80.33
		1m×1m	16		35.68	13.10	100
		1.5m×1.5m	9		76.43	40.05	87.10
		3m×3m	4		143.83	79.14	96.77
[2]	Custom	0.5m×0.5m	300	Not Used	40	NA	NA
[19]	Quorum (Custom)	0.4m× 0.4m	510	Used	50 (Four.Sig.)	NA	NA
					50-100 (HOG)		
					100-300 (Gist)		

The first is the maximum similarity location (MSL). In this case, it is observed that both μ_e and σ_e^2 are larger. This suggests that the proposed approach has improved localization error with less sensitivity to the learning grid dimensions. The results are also better as compared to those reported in [2] and [19]. In these work, only a fixed grid size is considered. In the former, localization error is $e = 40$cm with a learning grid 0.5m $\times$ 0.5m. In the latter, even though the sequentiality of appearance data is used, the best localization error is $e = 50$cm with a 0.4m $\times$ 0.4m learning grid.

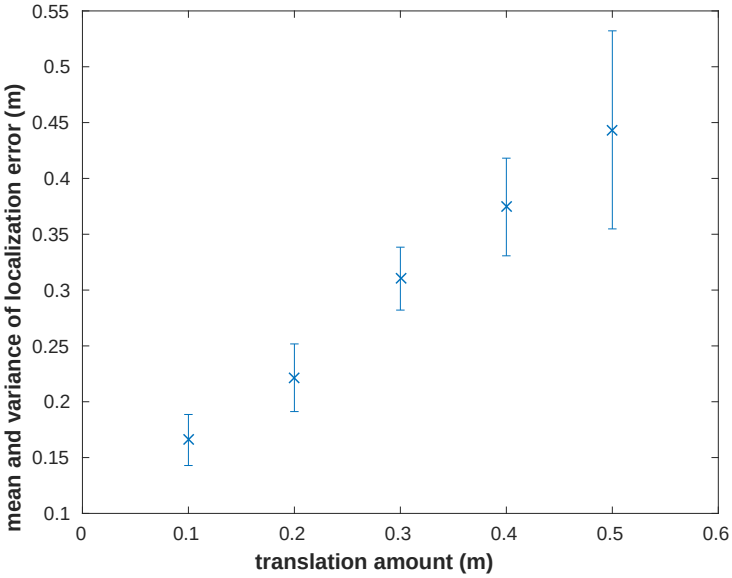


Figure 2.5. Variation of μ_e ($\times$ symbol) and σ_e^2 (vertical lines) with respect to translational offset of test locations.

2.4.2. Learned Appearances from Curvilinear Path

The second set of experiments are done with the learning of appearances from a set of locations along a curvilinear path - using omnidirectional image data taken from the extended path of Freiburg lab in the COLD dataset under night illumination condition [23]. There are 9 places as shown in Figure 2.6 with indexes as follows: printer area (1), corridor (2), kitchen (3), large office (4), two-persons office 1 (5), two persons-office 2 (6), one-person office (7), bathroom (8) and stairs area (9). We consider varying learning denseness in each place with the distances between learned locations ranging from around 7.34cm to 72.54cm in each place. The appearance at each location is learned as a $N_I = 600$ dimensional descriptor since there is no depth information.

We then consider two sets of test locations. The first set is from a path similar to that of learning. However, they differ from those of learned locations. A sample case for place 3 with learning distance 7.34cm is as seen in Figure 2.7(a). It is observed that the μ_e per place varies between 3.7cm to 6.6cm with variance ranging from almost

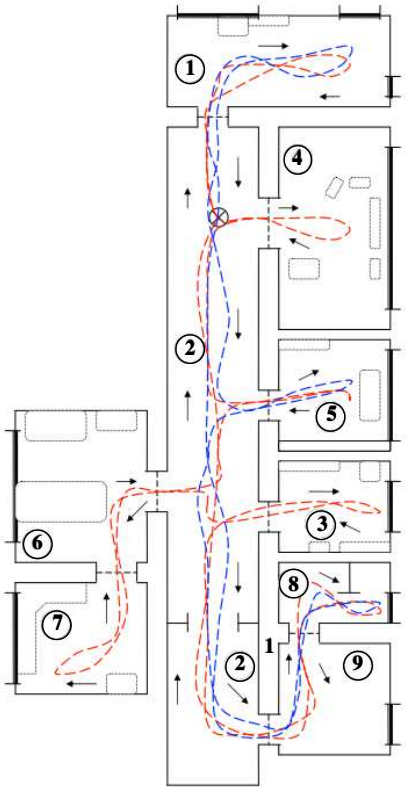
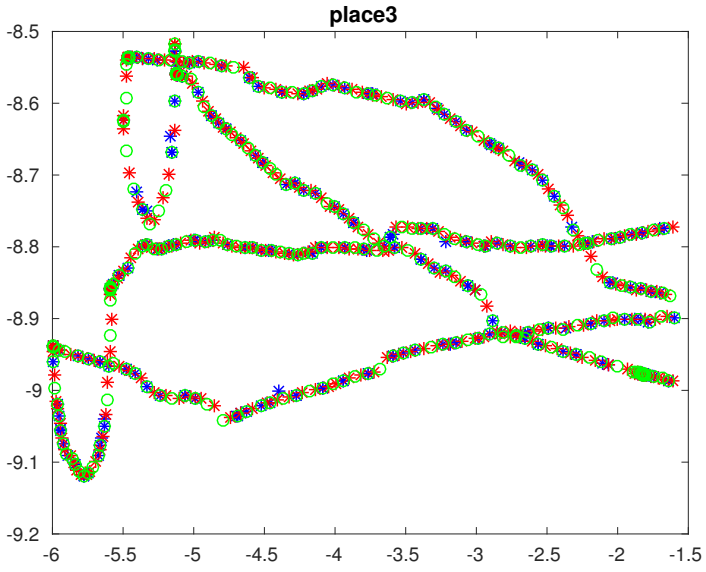


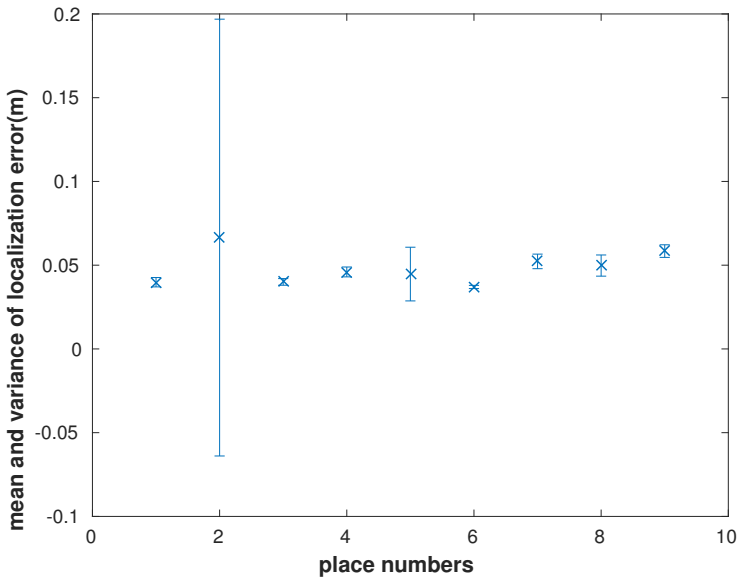
Figure 2.6. Places in Freiburg lab.

0 to 13cm. Its minimum occurs in place 6 while the worst is in place 2. This is quite expected as the corridor is a large place with similar appearances along its spatial. As such, the variance of the error is also the largest. In the second set, the test locations are from a completely different path as compared to that of learning - with possibly some intersections. This implies that the robot needs to localize itself even though it has not been to some of these locations actually. Again, for place 3, they are as shown in Figure 2.8(a). The results are shown in Figure 2.8(b). It is observed that overall the places, the mean localization error 41.77 cm with an average variance of 26.68 cm. This error is also higher as compared to those of previous experiments - even if the learning distance is considerably smaller. This can be attributed to the fact that these locations have not been covered during learning. As such, we cannot not expect the robot to localize reliably.

Next, we vary the mean learning distance from 7.34cm to 72.54cm and study its effect on localization performance - considering similar learning and test paths. The

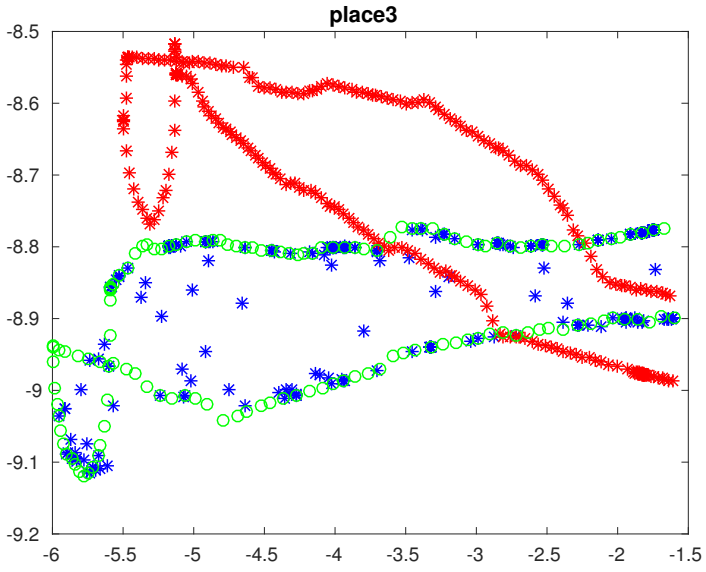


(a) Place 3:Learning (green) and test (red) locations are along a similar path. Estimated locations (blue) are also shown.

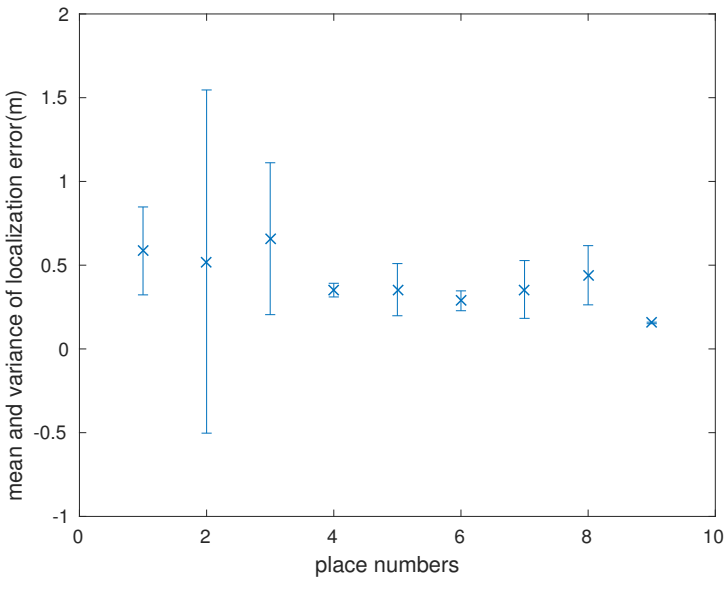


(b) Similar learning and test paths.

Figure 2.7. Localization results when the testing and learning is done along similar path.



(a) Place 3: Learning (green) and test (red) locations are along different paths. Estimated locations (blue) are also shown..



(b) Different learning and test paths.

Figure 2.8. Localization results when the testing and learning is done along different path.

Table 2.2. Comparative localization performance with similar learning and test paths.

Learning		Proposed			MSL		
Distance (cm)	M_p (#)	μ_e (cm)	σ_e^2 (cm)	A (%)	μ_e (cm)	σ_e^2 (cm)	A (%)
7.34	300	4.83	1.88	89.10	5.55	1.89	89.38
14.54	155	6.74	4.11	90.93	7.00	4.20	90.28
21.58	104	9.88	6.11	91.50	10.33	7.39	91.14
28.48	78	13.46	10.30	91.71	14.33	12.59	91.28
35.4	63	17.71	19.10	91.75	18.51	21.65	91.54
48.20	45	23.55	24.06	91.50	24.59	28.82	91.79
72.54	29	35.63	51.06	88.49	36.31	54.40	88.92

increase in the learning distance implies that the place is covered less densely. The results are presented in Table 2.2. It is observed that even though the localization error increases with greater learning distances, the accuracy remains to be around 90%. This indicates that in most cases, the position estimate $\hat{c}$ is within the learning distance range of the true position c. Let us note that recent work using the same dataset report accuracy to be in the range 87-95% - considering all locations with less than 1m localization error [10]. We also compare the localization performance with MSL. It is observed that both localization error μ_e and its variance σ_e^2 increase. In summary, the experimental results obtained from the experiments conducted with our robot and using COLD dataset both demonstrate that localization errors and their sensitivity against learning coverage denseness are both reduced.

Table 2.3. Comparative localization performance with different learning and test paths.

Learning		Proposed			MSL		
Distance (cm)	M_p (#)	μ_e (cm)	σ_e^2 (cm)	A (%)	μ_e (cm)	σ_e^2 (cm)	A (%)
7.34	160	41.19	26.11	5.02	42.81	30.16	5.47
14.57	80	42.59	26.88	20.10	44.29	31.80	20.50
21.73	54	48.20	42.62	36.35	49.64	47.63	35.24
28.72	40	48.40	40.03	52.20	50.43	46.51	52.12
35.93	32	51.95	49.47	59.99	53.34	55.09	59.44
49.18	23	54.29	66.42	70.19	56.14	78.75	70.52
74.86	15	69.65	88.22	72.40	71.58	96.94	72.89

3. COUPLING PLACE AWARENESS AND SELF-LOCALIZATION

In this chapter, we introduce a novel approach in which the robot couples place awareness and self-localization explained in Chapter 2. A place-aware robot is to able to recognize a place (i.e. a particular room, etc.) after having learned it based on a collection of appearances. We assume that the learned appearances are tagged with location data during learning. The tagging is done only during learning and can be obtained from odometric data in each place locally, by external guidance or even via gps. As places refer to spatial regions, recognizing a place only coarsely designates its whereabouts. Thus, it does not imply finding its relative coordinates which is in contrast to related work in which a place refers to a single location, so that place recognition implies localization as well. It should be noted that due to perceptual aliasing, the robot may not always correctly recognize a place. The proposed approach consists of two modules that operate in a coupled manner. The first module enables place-awareness. This is achieved using the TSC model. The TSC model entails a long-term place memory and a set of processing modules as to enable a robot to build its knowledge of places and refer to this knowledge autonomously. The second module is the localization. This is activated if the robot recognizes its current whereabouts as one of places that it has learned - as retained in its place memory. In this case, the robot retrieves all the learned appearances pertaining to this place from its place memory and uses them in its self-localization as explained in Chapter 2. This is achieved via minimizing an objective function that is constructed as the weighted sum of all the learned positions. The weights encode the similarity of appearances from the respective learned locations to those from the robot's current position. The advantages of the proposed approach are as follows: First, the search process is limited to locations pertaining to one place. Second, since all the learned related knowledge is used, the localization error is improved and seems to be less affected by the coverage denseness of each place during learning. Third, it is applicable even if the incoming appearance data is not of sequential nature - as may be the case when starting a mission, its incoming

appearances are sparse or it is kidnapped. Finally, the localization results are used to check whether the place was correctly recognized or not. It should be noted that, to the best of authors' knowledge, none of the previous work provides all these advantages simultaneously.

3.1. Place Awareness

Place awareness is enabled through the previously introduced topological spatial cognition (TSC) model. In this section, we briefly summarize the key characteristics of this model as they pertain to our work. For details, the reader is kindly referred to [21].

The first aspect pertains to places. A place refers to a spatial region observed to have common perceptual signatures and physical boundaries. Such a definition is thought to be similar to that of human's 'place' concept.

Second, the knowledge of places is retained in a part of the long-term spatial memory - referred to as place memory. The long-term place memory is a record of robot's past experiences and plays an integral role in storing and retrieving the place knowledge. We consider a hierarchically organized place memory [20]. The hierarchy is such that inner nodes correspond to clusters of places and terminal nodes correspond to distinct places. Such an organization allows the robot to associate with its learned place knowledge efficiently. Furthermore, the memory is ensured of having both storage and construction efficiency as well as being order-invariant.

The final aspect is related to the processing modules. The processing modules are comprised of:

- Detection - Deciding what constitutes a place and collecting the respective appearance

- Recognition - Associate the current place with previously learned knowledge as retained in the place memory
- Learning - Build and expand place memory for future referral.

Places are detected based on delineating the incoming stream of sensory data and identify individual spatial regions [24]. For omnidirectional images, the proposed place detection approach is modified as explained in Appendix B. Once a place is detected, the robot attempts to recognize it via relating to its long-term spatial memory. This is achieved by traversing the place memory appropriately in order to associate the detected place (if possible) with a previously learned one using an hybrid strategy . In case of recognition, the robot simply updates its memory. In case of no recognition, it invokes place learning in order to add the detected place. As such, the memory becomes self-evolving via adapting its hierarchical organization. The processing modules operate together to build place map knowledge in an unsupervised, incremental and organized manner.

3.2. Place Verification

Consider a detected place as defined by the appearance data collected from a set of base points x_j with $j \in D_m$ where D_m is the index set. Recall that the estimated location $\hat{c}_j \in R^2$ is computed based on computing a weight vector w_j as given by Equation 2.6 is constructed. The components of w_j can used in forming a measure of confidence κ for deciding whether the place was correctly recognized or not:

$$\kappa = \frac{1}{|D_m|} \sum_{j \in D_m} w_j^* \tag{3.1}$$

where w_j^* is the maximum weight associated with each location x_j in recognized place as:

$$w_j^* = \max_{k=0,\cdots,M_p-1} w_k(x_j) \tag{3.2}$$

Thus, the decision is based on comparing κ with a threshold $\tau_p > 0$.

$$\kappa \geq \tau_p \tag{3.3}$$

The value κ encodes the average of maximum weights associated with each location in the recognized place to estimate the location of base points in detected place. If the current place is correctly recognized, the weights of the individual base points of recognized place are expected to be higher, hence the overall place weight is expected to be high. If place weight is small, it is indication of the wrongly recognized place, which shows that the localization results can be used for verification of the place recognition results. The value of threshold τ_p is observed to depend on the camera and descriptors used in encoding the appearance data. It needs to be set appropriately - taking these two factors into consideration.

3.3. Experiments

The proposed approach is evaluated by conducting two different sets of experiments in different sites. The localization experiments are done after in the second-time visits of these sites - after they have been visited for the first time and learned accordingly. In both, the robot tags the learned appearances with their respective positions using local odometric data. Each appearance is encoded as a $N_I = 600$ dimensional descriptor. In the tests, the localization error is defined as $e = |c - \hat{c}|$ with mean μ_e and variance σ_e^2. We also consider the accuracy A that measures the percentage of test points with localization error e less than the corresponding learning distance in that place.

3.3.1. Experiments with COLD Dataset

The first set of experiments are conducted on the indoor data from COLD dataset benchmark. The COLD dataset consists of omnidirectional camera data from three different sites: Freiburg Lab (Fr), Saarbrucken Lab (Sa) and Ljubljana Lab (Lj).

A sample image taken from a point in the Fr site is shown in Figure 3.1. The five rooms inside this site with their indexes are as follows: corridor (1), printer area (2), two-persons office (3), stairs area (4) and bathroom (5). In this Fr site, the robot learns 14 places in the first time visits as seen in Figure 3.2. It is observed that there are many transition regions (shown in blue) - due to the uninformativeness or incoherency of the incoming appearance data. In the second-time visit, 19 places are detected as seen in Figure 3.3. The detected places are shown with m and their set is shown as D_m. It should be noted that, the detected place m in the first time visit may not be the same with the detected place m in the second time visit. The place recognition and localization results associated with these detected places in second-time visit are given in Table 3.1. It is observed that among detected 19 places, 16 of them are recognized from place memory constructed in the first-time visit of the robot. The ids (m) of the detected places in the second time visit are shown in the first column of Table 3.1. The true place of this detected place among learned places is shown by p in the second column. It can be seen that, for example for $m = 2$, p can be 2 or 3, because they are almost the same place in the environment, and the detected place m can be considered as correctly recognized if $\hat{p} = 2$ or $\hat{p} = 3$. The recognized place id resulting from place recognition approach is shown as $\hat{p}$ in the third column. If $p = \hat{p}$, it means that the place is correctly recognized. The $-$ sign in the second column means that there is no learned place in the first time visit that covers the area of the place m. The $-$ sign in the third column means that the detected place m isn't recognized. For the detected places in the second-time visit where the robot recognize as one of the learned places in the first-time tour, the step size value of that recognized place is given in the 5th column. Step size is the average value of the linear distance between two consecutive base points in the recognized place. The mean localization errors of the base points in the detected place is indicated in the last column of Table 3.1. As shown, most of the time, the robot is localized with mean localization error μ_e smaller than the step size of the recognized place.

As shown, in this site, all of the places are checked as correctly recognized. For all sites in COLD dataset, $\tau_p = 10^{-24}$ is used to determine if the places are recognized



Figure 3.1. Sample visual data from Freiburg Lab.

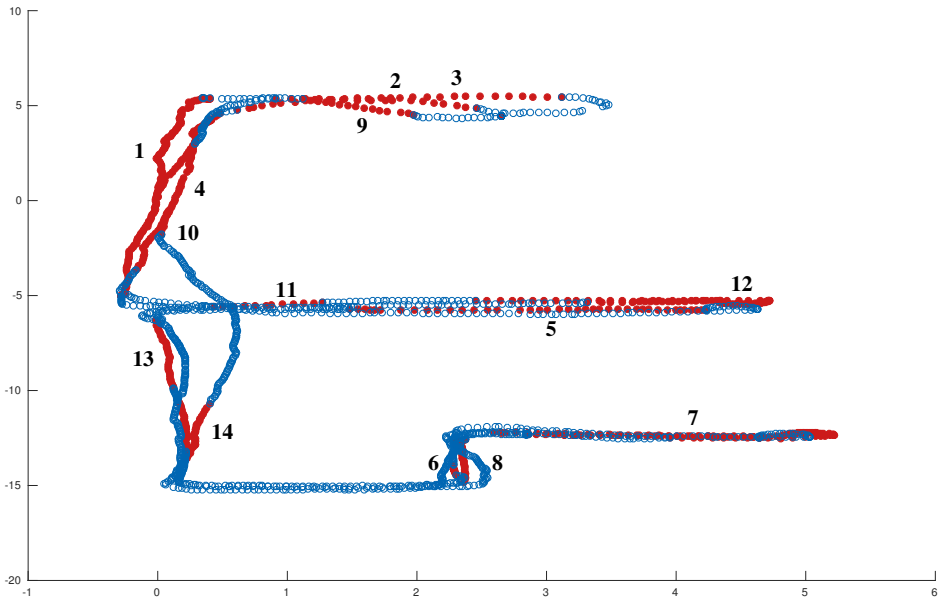


Figure 3.2. Learned places (shown in red) in first-time visits.

correctly or not. This value is selected depending on the type of the images used and the descriptors to represent these appearances. However, in some places like place 19, $m = 19$, the localization error is higher than the most although it is observed that the recognized place $p = 14$ is the one closest to that place, so it can be considered as correctly recognized. Despite this, the localization error is high because the base points of the recognized place $p = 14$ do not cover whole area where the base points of the detected place $m = 19$ are located. Therefore, the robot doesn't have knowledge in its memory about the base points from that uncovered area; therefore, it estimates the location of these base points based on only the available knowledge, which results in increase in localization error.

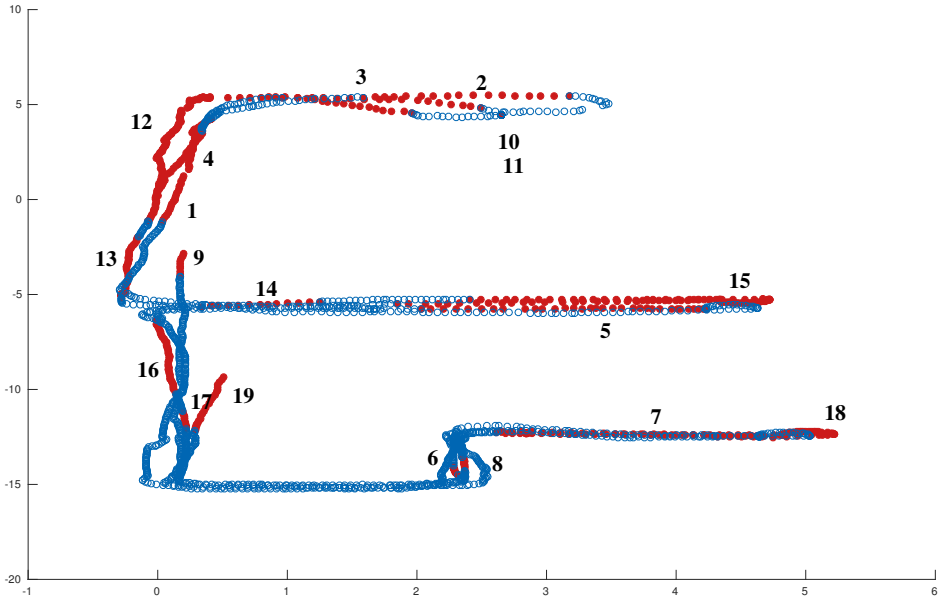


Figure 3.3. Detected places (shown in red) in second-time visits.

In this work, it is claimed that the κ values are indicator of the accuracy of place recognition results, namely, it means the place is correctly recognized if $\kappa > \tau_p$. The place recognition approach used in this paper gives correct results for this site, therefore, to testify our claim, we have run the localization approach with all possible learned places in the place memory and calculated κ values. The results are shown in Figure 3.4, where the x axis indicates the recognized places $\hat{p}$ and the y axis shows the places detected in the second tour. The κ values for each learned place is represented by a color which becomes darker as the value decreases. As shown, for each detected place, just one of the places has high κ (shown by the yellow square), κ values of others are all relatively very small, which indicates that, if one of the other places is recognized by the robot, κ value of this place will be so small that we can reach a conclusion of false recognition.

Similar experiments are also conducted with the visual data from Saarbrucken Lab. site. In this site, the environment has four room with indexes as follows: corridor (1), 2-persons office (2), printer area (3), and bathroom (4). A sample appearance taken in this site is shown in Figure 3.5. The localization experiments are done after

Table 3.1. Place recognition and localization results in Freiburg site.

m	p	$\hat{p}$	κ	Step Size	μ_e (m)
1	1 or 4	1	1.05×10^{-17}	0.247	0.377
2	2 or 3	2	2.61×10^{-19}	0.104	0.048
3	2 or 3	2	3.1×10^{-14}	0.115	0.053
4	4	4	9.32×10^{-8}	0.148	0.806
5	5	5	1.19×10^{-12}	0.084	0.165
6	6	6	4.66×10^{-12}	0.048	0.022
7	7	7	3.23×10^{-11}	0.075	0.0314
8	8	8	1.27×10^{-16}	0.080	0.030
9	–	–			
10	2	–			
11	2	2	2.00×10^{-14}	0.104	3.487×10^{-16}
12	1	–			
13	1	1	8.60×10^{-8}	0.102	0.035
14	11	11	3.33×10^{-15}	0.08	0.042
15	12	12	1.38×10^{-5}	0.0488	0.147
16	13	13	2.2×10^{-15}	0.0869	0.066
17	14	14	1.56×10^{-16}	0.151	0.107
18	7	7	2.47×10^{-9}	0.075	0.027
19	14	14	3.46×10^{-18}	0.151	0.609

in the second-time visits of these site - after they have been visited for the first time and learned accordingly. The detected places in the first-time visit are shown in Figure 3.6 whereas the detected places in second-time visit are shown in Figure 3.7. The temporal windows (blue regions in Figure 3.6 and Figure 3.7) are resulting from the uninformativeness of the images or incoherency between two consecutive appearances. The localization and place recognition results are given in Table 3.2. In this table id of the detected place in the second-time visit (m) is indicated in the first column and the learned place that m should be recognized as shown by p in the second column. The recognized place from place memory among the learned places in the first visit is

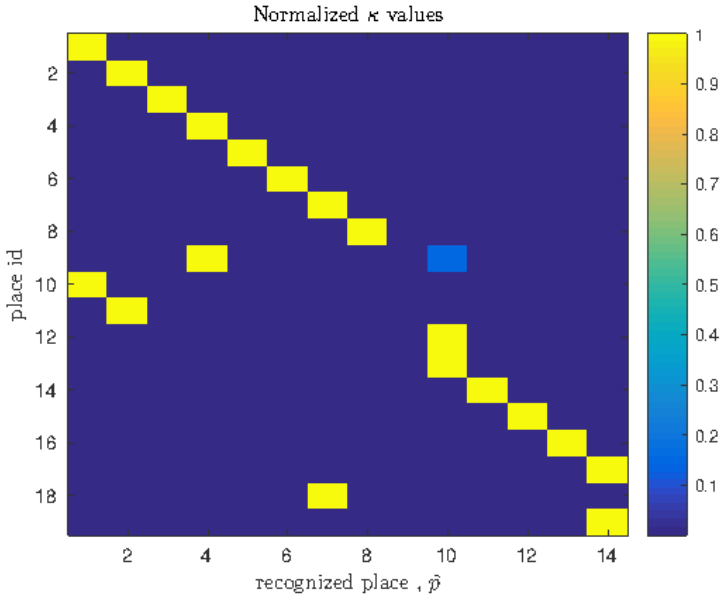


Figure 3.4. κ values of all possible recognized places.



Figure 3.5. Sample appearance from Saarbrucken Lab.

shown as $\hat{p}$ in the third column. If $p = \hat{p}$, it means the place is correctly recognized. κ values used to verify the place recognition results are given in the fifth column of the table and they are followed by the step size and mean localization error μ_e.

As shown in Table 3.2, among 15 detected places in the second-time visit, 9 of them are recognized among the places in the place memory constructed during the first-time visit and all of them are correctly recognized, i.e. for each of them $\kappa > \tau_p = 10^{-24}$. The results of the experiments conducted in this site are parallel to the ones acquired in Freiburg site. The localization error is usually smaller than the step size of the recognized place except for detected place $m = 9$ which is recognized as learned place $\hat{p} = 8$ from the first time visit. The reason why the localization error is higher is that,

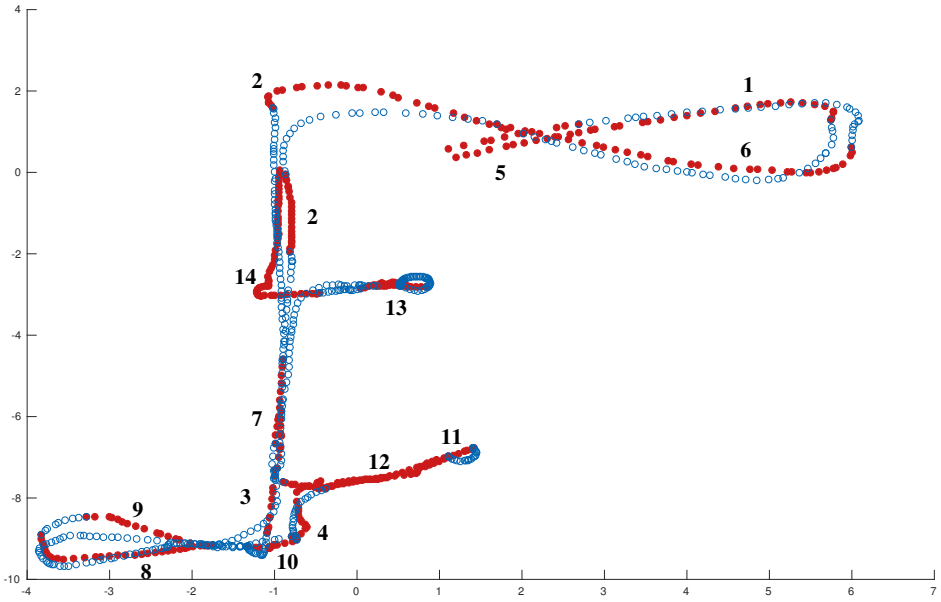


Figure 3.6. Learned places (shown in red) in first-time visits.

a part of the detected place $m = 9$ isn't covered by recognized place $\hat{p} = 8$, therefore, the localization error in the points from uncovered part increases the localization error mean.

Experiments similar to ones conducted in Freiburg and Saarbrucken site are performed in also Ljubljana Lab where a sample appearance taken from is shown in 3.8. There are 5 rooms in this site with indexes as follows: printer area (1), corridor (2), Lab (3), 2-persons office (4) and bathroom (5). Places learned in the first-time visits and detected places in the second-time visits are shown in Figure 3.9 and Figure 3.10 respectively. In this site, the quality and informativeness of the visual data is relatively lower as compared to that of the first two sites. In some appearances, the illumination is too low and illumination changes between two consecutive appearances can be high. As such, most part of the site remains uncovered during the first-time visit as shown in blue in Figure 3.9, which also affects the place recognition performance. As shown in Table 3.3, only 12 of the 23 detected places in second-time visit are recognized and all of them recognized correctly since they have $\kappa > \tau_p = 10^{-24}$. In these places, the localization error is again usually lower than the step size of the recognized place. The

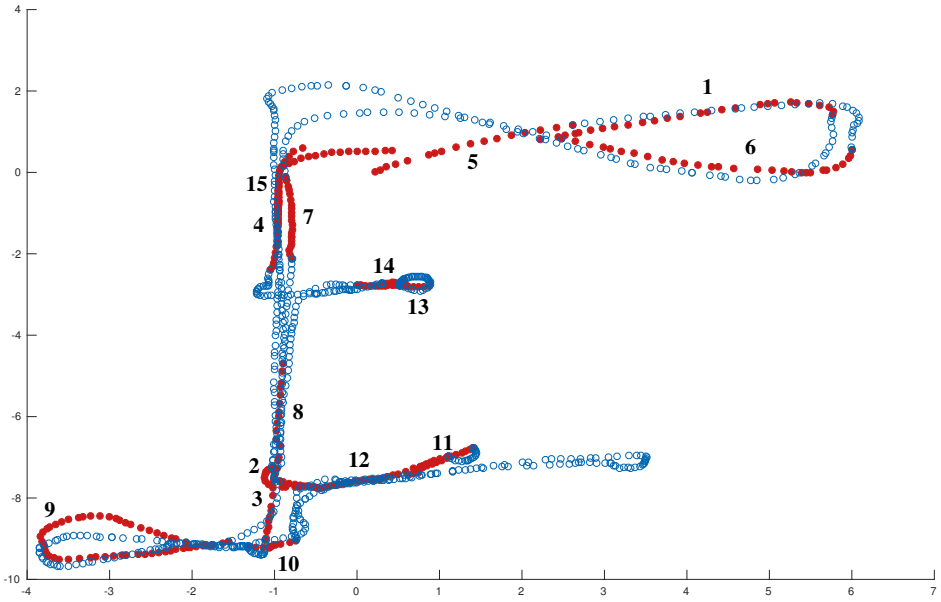


Figure 3.7. Detected places (shown in red) in testing stage.

reason of higher localization error mean is again originating from the coverage during the first-time visit.

3.3.2. Experiments with Turtlebot

The second set of experiments are conducted on the visual data taken from our Turtlebot robot with V360 omnidirectional camera mounted on. The data is collected from three places inside Square Building (three rooms: Intelligent Systems Lab. (ISL), corridor and the classroom). The corridor can be considered as two places due to

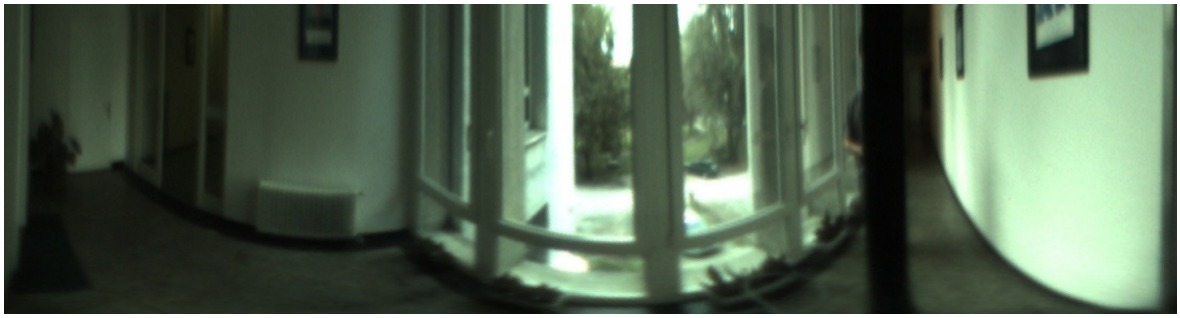


Figure 3.8. Sample appearance.

Table 3.2. Place recognition and localization results in Saarbrucken site.

m	p	$\hat{p}$	κ	Step Size	μ_e (m)
1	1	1	3.79×10^{-23}	0.190	0.098
2	3	3	9.90×10^{-14}	0.195	0.183
3	3	–			
4	14	–			
5	5	–			
6	6	6	1.00×10^{-20}	0.171	0.115
7	2	–			
8	7	–			
9	8 or 9	8	5.09×10^{-17}	0.085	0.322
10	10	10	7.97×10^{-16}	0.039	0.032
11	11	11	3.73×10^{-11}	0.060	0.02
12	12	12	3.36×10^{-12}	0.065	0.029
13	13	13	3.79×10^{-16}	0.038	0.095
14	13	13	1.81×10^{-15}	0.038	0.030
15	14	–			

significant appearance changes. The sample visual data taken from these three places are shown in Figure 3.11.

During first-time visit of the robot in the site, it visits base points in a grid-based scheme where the points are 50 cm and 60 cm apart from each other. In the second-time visit, the robot again navigates in a grid base scheme, but it visited different points lying in the area covered by the base points from first-time visit of the robot in the environment. The detected places in the first time visit of the robot are shown in Figure 3.12 and the detected places during the second-time visit are shown in Figure 3.13. As seen, during the first-time visit, the base points from the upper part of the corridor are not assigned to any place, therefore, they remained unlearned due to the uninformativeness of incoming data or incoherency between two consecutive appearances.

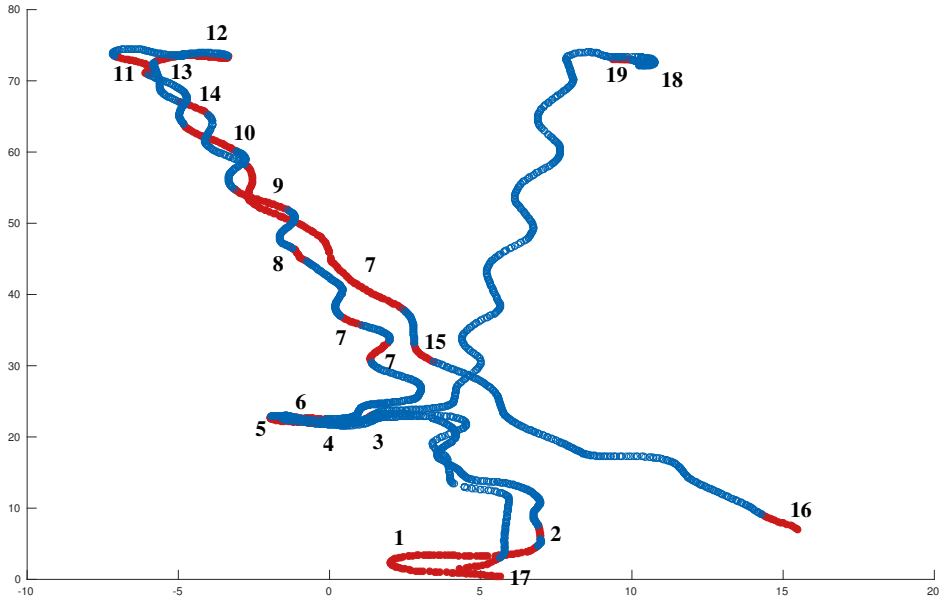


Figure 3.9. Learned places (shown in red) in first-time visits in Ljubljana site.

The localization results in conjunction with place recognition are given in Table 3.4. As shown, among five detected places in the second-time visit, just one of them is recognized from the place memory of the robot. The detected places $m = 2$ and $m = 3$ couldn't be recognized, because the part of the corridor where these places located are not covered in the first time tour, therefore, they are not located in the place memory. In this site, because different types of camera and images are used, the τ_p value is selected as 5×10^{-5}. With this value, it can be concluded that, the place $m = 4$ is recognized wrongly, which is actually the case. As shown in Figure 3.12, the recognized place $\hat{p} = 2$ covers the lower part of the corridor whereas $m = 4$ located in the upper part of the corridor, which also considerably increases the localization error in that place.

When it is looked at the localization error mean values (μ_e), it can be seen that these values are larger than the step size even if the place is considered as correctly recognized. This situation is originated from the place detection part. The detected places in the first and second visits do not overlap exactly. Therefore, some part of the detected places in the second tour may not be detected as place in the first tour. The

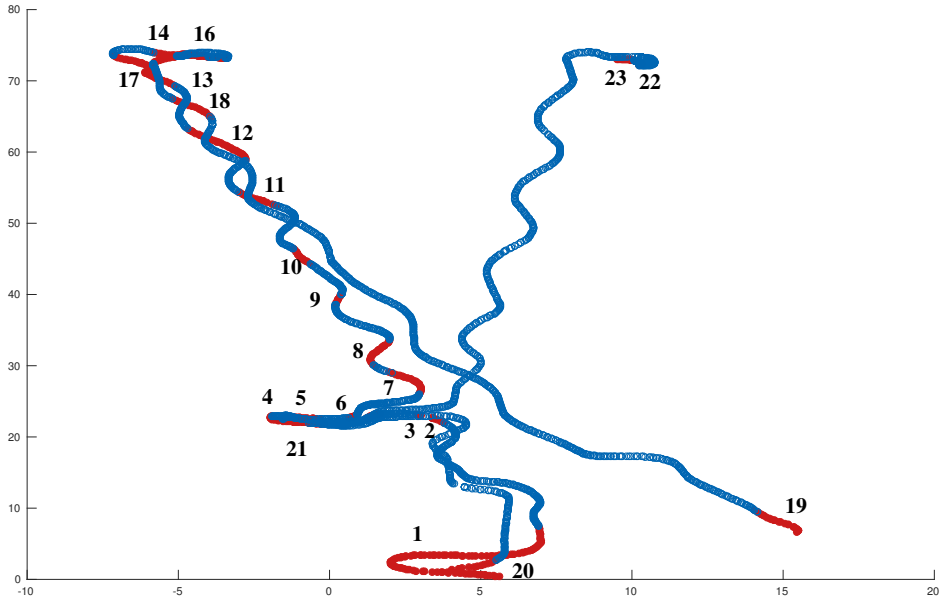


Figure 3.10. Detected places (shown in red) in second-time visits in Ljubljana site.

robot tries to estimate the locations of the base points from these parts based on the knowledge of detected places in the first tour, which in turn increases the localization error. If the detected places in the first and second tour overlap exactly and the place recognition works correct the localization errors acquired would become similar to the ones gotten in Section 2.4.1.

In order to find the uncovered parts of the detected places in the second tour, we have looked at the $\kappa_i = \max_{k=0,\cdots,M_p-1} w_k(x_j)$ values of individual base points of the detected places in the second tour if they are correctly recognized. If $\kappa_i < \tau_p$, we disregard these base points and do not estimate their locations. As such, the uncovered parts of the detected places are found and their locations are not estimated due to the lack of knowledge from that part of the place. After implementing this, the localization results become as shown in Table 3.5. As shown, the localization errors become smaller than the learning distance as expected. Because the place 4 ($m = 4$) is already wrongly recognized, we did not implement such a modification for that place.

Table 3.3. Place recognition and localization results in Ljubljana site.

m	p	$\hat{p}$	κ	Step Size	μ_e (m)
1	1	1	2.05×10^{-13}	0.157	0.704
2	–	–			
3	–	–			
4	5	5	6.82×10^{-14}	0.099	0.046
5	6	6	1.096×10^{-17}	0.118	0.0506
6	–	–			
7	–	–			
8	7	7	1.394×10^{-17}	0.334	0.098
9	–	–			
10	8	8	9.26×10^{-23}	0.155	0.100
11	9	–			
12	10	10	1.48×10^{-15}	0.160	0.487
13	11	11	2.42×10^{-19}	0.103	0.446
14	13	–			
15	12				
16	12	12	5.40×10^{-13}	0.044	0.029
17	13	13	6.03×10^{-19}	0.065	0.045
18	14	–			
19	16	–			
20	17	17	5.22×10^{-16}	0.181	0.086
21	5	–			
22	18	18	5.40×10^{-13}	0.044	0.029
23	19	19	6.03×10^{-19}	0.065	0.045



(a) Sample image from ISL



(b) Sample image from corridor



(c) Sample image from classroom

Figure 3.11. Sample images taken from turtlebot with V360 camera in three different places

Table 3.4. Place recognition and localization results in Turtlebot experiments.

m	p	$\hat{p}$	κ	Step Size	μ_e (m)
1	1	1	0.00359	0.517	1.241
2	–	–			
3	–	–			
4	–	2	3.147×10^{-5}	0.514	5.365
5	3	3	0.000319	0.656	1.181

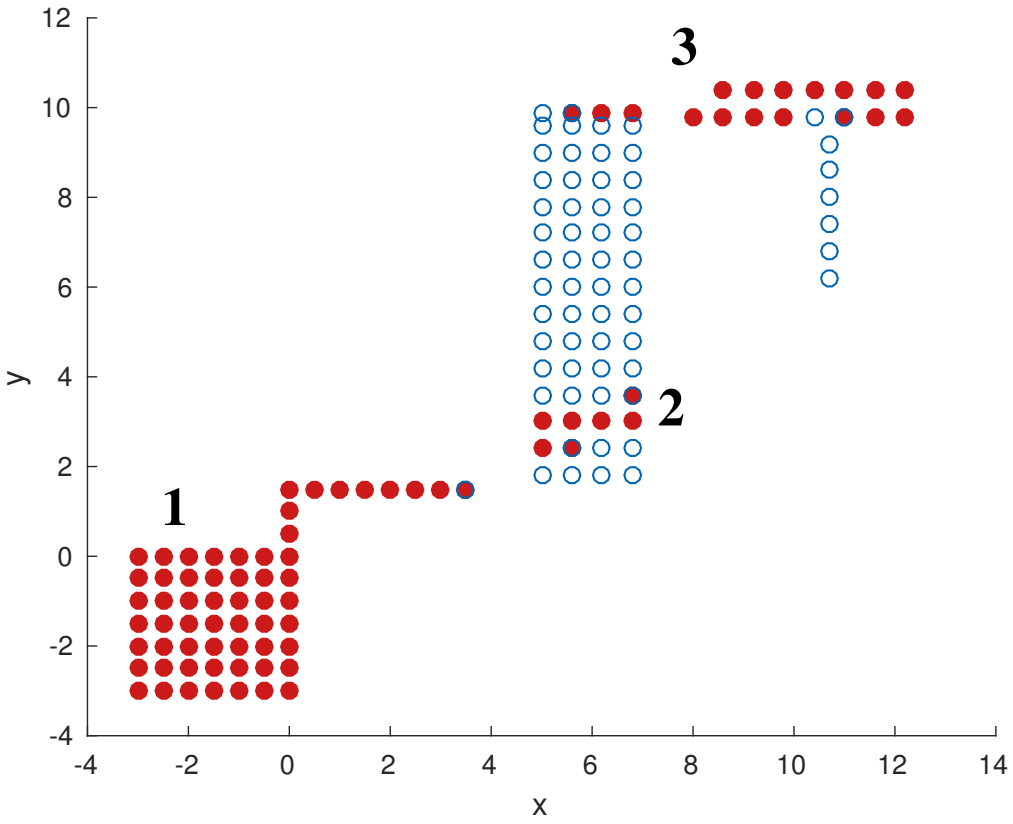


Figure 3.12. Detected places (shown in red) in first-time visits.

Table 3.5. Place recognition and localization results in Turtlebot experiments when the locations of base points from uncovered area are not estimated.

m	p	$\hat{p}$	μ_e (m)
1	1	1	0.3845
2	–	–	
3	–	–	
4	–	2	5.365
5	3	3	0.4368

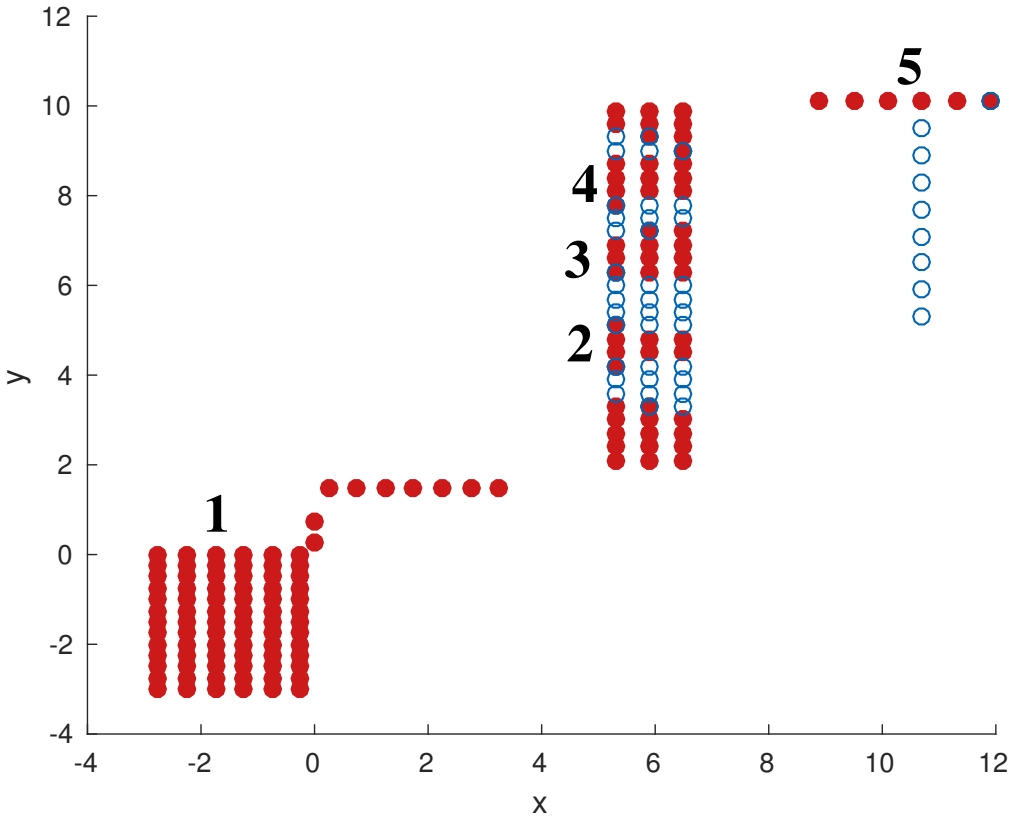


Figure 3.13. Detected places (shown in red) in second-time visits.

4. APPEARANCE-BASED NAVIGATION

Navigation is one of the fundamental capabilities of a mobile robot. In this chapter, appearance-based navigation is considered. In this problem, the robot is given a set of appearances that are associated with a particular (goal) location and is asked to navigate to that location - using only the appearance knowledge.

There has been extensive work on navigation. The proposed methods are categorized according to the type of data used. Metric approaches such as SLAM require precise position data [25]. As such, odometric data needs to be available and reliable at all times [26]. Otherwise reasoning with metric maps are computationally expensive and do not scale well. On the other hand, appearance-based or topological approaches use the associated appearances in driving the robot. The matching of the robot's current appearance-based knowledge to that previously learned based on appearances is used in localizing the robot and planning the motion accordingly. Appearance-based approaches are computationally more advantageous but they suffer from lack of resolution as discussed previously [27]. In hybrid approaches [28], navigation is based on the hybrid maps where the global information is topological and the local information is metric. Such approaches inherit the advantages of topological maps, however similar to the use of metric maps, self-localization becomes a critical component.

We also consider appearance-based navigation and propose novel hybrid approach - again assuming place-aware robots. This approach is based on the self-localization approach that has been presented in Chapter 2. As such, when in a certain place p, it is able to recognize it correctly as p and recollect the respective set of learned appearances that are tagged with their respective locations $c_{pi} \in R^2$, $i = 1, \cdots, M_p$. For example, a sample scenario is shown in Figure 4.1 in which the learned base points are shown with cross marks. The robot is also given a set of appearances that is known to be associated with the goal location $g \in R^2$ and is asked to go to the goal location g where the given appearance was taken. Note that as the odometric data is either

unavailable or unreliable and thus the robot cannot use it.

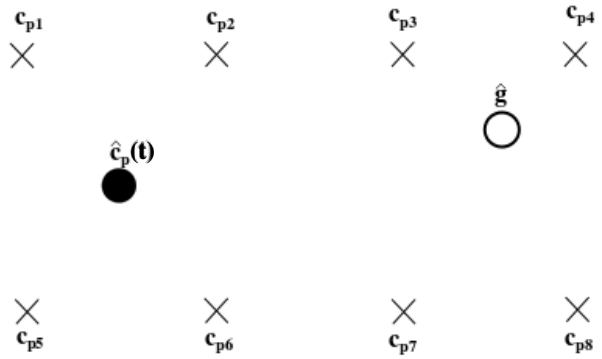


Figure 4.1. Appearance-based navigation. The robot knows the place based on appearances from a set of locations (as indicated by cross marks). It is now at a location (filled circle) and needs to navigate to a goal location (empty circle).

4.1. Coupling Navigation & Self-Localization

The proposed approach consists of the following stages: First, the robot estimates the goal location from the appearance data given to robot as $\hat{g}$ - using self-localization. Next, the robot considers the appearance data from its current location c and estimates its current location as $\hat{c}_p(t)$ - using self-localization. The vector difference between these two estimated locations is given by Equation 4.1:

$$\delta\hat{g}(t+\delta t) = \hat{g} - \hat{c}_p(t+\delta t) \tag{4.1}$$

Now, it needs to navigate from $\hat{c}_p$ to $\hat{g}$. However, as it does not know its heading $\alpha(t) \in S^1$, it needs to estimate $\hat{\alpha}(t) \in S^1$. For this, it moves in its current (albeit unknown) heading α, and reaches a new location $c_p(t+\delta t)$ as shown in Figure 4.2. The location of this new point is estimated as $\hat{c}_p(t+\delta t)$. Let $\delta\hat{c}_p(t+\delta t) \in R^2$ denote the vector from location $\hat{c}_p(t)$ and $\hat{c}_p(t+\delta t)$ as defined in Equation 4.2:

$$\delta\hat{c}_p(t+\delta t) = \hat{c}_p(t+\delta t) - \hat{c}_p(t) = \begin{bmatrix} \delta\hat{c}_{p,1}(t+\delta t) \\ \delta\hat{c}_{p,2}(t+\delta t) \end{bmatrix} \tag{4.2}$$

This direction is used to estimate $\hat{\alpha}(t)$ as given in Equation 4.3:

$$\hat{\alpha}(t) = \arctan 2(\frac{\delta\hat{c}_{p,2}(t+\delta t)}{\delta\hat{c}_{p,1}(t+\delta t)}) \tag{4.3}$$

In parallel, the heading $\delta\hat{\alpha}(t+\delta t)$ that will enable it to move towards the goal can be estimated as:

$$\delta\hat{\alpha}(t+\delta t) = \arccos\left[\frac{\delta\hat{c}_p(t+\delta t)^T\delta\hat{g}(t+\delta t)}{\|\delta\hat{c}_p(t+\delta t)\|\|\delta\hat{g}(t+\delta t)\|}\right] \tag{4.4}$$

The robot needs to turn by $\delta\hat{\alpha}(t+\delta t)$ amount and then move along the resulting heading by $|\hat{c}_p(t+\delta t) - \hat{g}|$ m in order to reach the goal. These two are then encoded by a navigation function as:

$$\varphi(\hat{c}_p, \delta\hat{\alpha}, \hat{g}) = \|\hat{c}_p - \hat{g}\|^2 + \delta\hat{\alpha}^T\delta\hat{\alpha} \tag{4.5}$$

The robot's motion dynamics is defined by the negative gradient of $\varphi(\hat{c}_p, \delta\hat{\alpha}, \hat{g})$ as:

$$\dot{c}_p = -D_{c_p}(\varphi) = -(\hat{c}_p - \hat{g}) \tag{4.6}$$

$$\dot{\alpha} = -D_{\alpha}(\varphi) = -(\delta\hat{\alpha}) \tag{4.7}$$

Both its linear and angular velocity get smaller as it approaches the goal location. As the robot moves incrementally from its true position $c_p(t)$ to $\hat{g}$, it corrects itself via repeating its self-localization. It stops navigating at time t_f either when the similarity

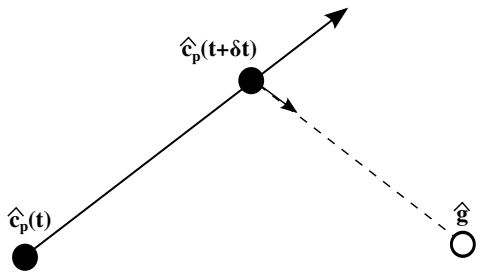


Figure 4.2. Navigation based on estimated location, heading and goal using appearance knowledge.

between its current appearance and the goal appearance is high enough as measured by comparing the difference between their respective descriptors with a similarity threshold τ_d as:

$$\|I(g(t_f) - I(c(t_f))\| < \tau_d \tag{4.8}$$

or when the distance between its estimated position and estimated goal position is small as determined via comparing it with a distance threshold τ_l

$$\delta\hat{g}(t_f) < \tau_l \tag{4.9}$$

The navigation parameter values depend on the learning denseness. In more densely learned places, these values should be smaller because the visual similarity between the current location and the learned ones is larger when the learned locations lie on the denser grid. The parameter τ_l value is selected as near to the mean localization error because that value is the uncertainty in the estimated location of the final position of the robot. The localization error is usually around half of the learning size.

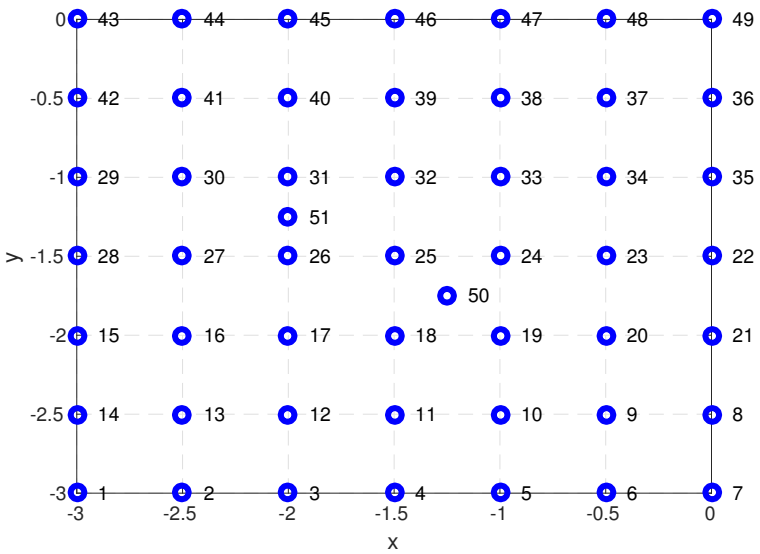


Figure 4.3. Navigation experiments setup. Initial and goal locations are indicated by their indices.

4.2. Navigation Experiments

In the experiments, the robot is assumed to correctly recognize its place and retrieve the associated appearances that are tagged with location knowledge. The place has been learned using a grid-based coverage with denseness 0.5m × 0.5m as shown in Figure 4.3. The robot repeats the self-localization procedure very δt times along its trajectory. The navigation parameters are set as $\tau_d = 0.7$ and $\tau_l = 0.3$ with the 0.5m × 0.5m grid learning distance. It should be noted that the τ_l parameter is selected near to the localization error with that grid distance. We define goal error δg as the error in attaining the goal position - namely difference between the 'true' goal location and the final location of the robot after it stops - namely $\delta g = \|g - c(t_f)\|$.

A set of 22 experiments that vary with respect to initial and target location are conducted. The robot's heading is also completely random - including headings that are in the opposite directions. The resulting navigation errors for each initial-goal position pair indicates by their respective indices are given in Table 4.1. The mean goal error δg is observed to be 0.465m with mean variance 0.052m. Both are reasonable as the mean self-location error is 0.28m. As the goal position deviates from the locations of learned appearances, the mean goal error increases to 0.60m. This is attributed to the fact that as self-localization errors are higher - as discussed in Chapter 2, the actual goal position differs from that it estimates. Furthermore, in some of the experiments (as in the cases of experiments 6, 12, 21 etc.), the goal error is higher. Closer analysis reveals dynamic scene changes as compared to learned appearances.

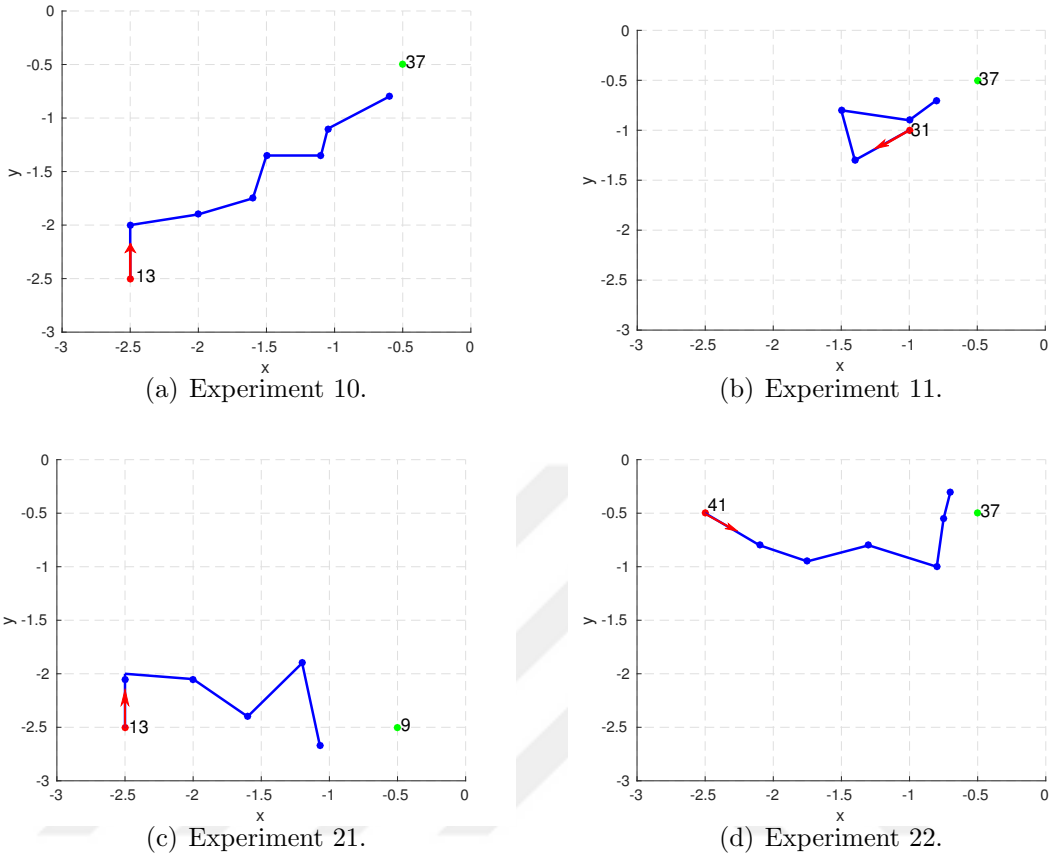
Furthermore, we also study the resulting navigation paths. Some samples are presented in Figure 4.4(a), 4.4(b), 4.4(c), 4.4(d). In these figures, the initial location of the robot is shown by red circle, initial heading is shown by read arrow while the goal location is shown by green circle. The initial linear velocity of the robot is set to 0.2m/sec. The paths are piecewise linear - as the robot estimates its location and heading once every δt times. This is because processing appearance data takes about 1.2 sec. The robot stops during this processing. Otherwise, continuing with its movement

Table 4.1. Goal errors for experiments varying in initial and goal positions.

No	$c(0)$ Index	g Index	δg (m)
1	13	25	0.52
2	9	25	0.13
3	37	25	0.23
4	29	25	0.42
5	26	37	0.20
6	19	37	0.88
7	17	37	0.5
8	29	37	0.26
9	10	37	0.20
10	13	37	0.52
11	31	37	0.35
12	19	13	0.77
13	38	13	0.34
14	25	13	0.46
15	28	13	0.29
16	41	13	0.76
17	15	50	0.64
18	37	50	0.59
19	19	51	0.43
20	38	51	0.77
21	13	9	0.59
22	41	37	0.27

would result in a higher self-localization error - since it would estimate its location based on appearance taken from a previous location. If these processes were faster, the navigation path would be much smoother. In experiment 10, the distance between the initial and goal location is 2.82m. While the initial and goal locations of the robot are among the learned locations (they are on the grid points), this is not the case for most of the locations on its path. While the robot' actual heading wanders from time to time, it reaches the goal location with reasonable error of $\delta g_e = 0.52m$. Next, we consider a different scenario where the robot's initial position gets closer to the goal with 0.71m distance as shown in Figure 4.4(b). Furthermore, the initial heading of the robot is in an opposite direction to the goal location. Even with that heading, the robot was able to adjust its heading after two iterations of localization and reached the goal location. It can be observed that, the robot reached the goal location with smaller error $\delta g_e = 0.35m$ and in a shorter time as expected.

We also consider the effect of dynamic changes in the environment as presented in experiments 21 and 22. In both, the distance between the initial and goal locations of the robot is 2m. The dynamic changes are observed to be due to furniture and lab equipment rearrangements. In experiment 21, the goal location is given to be in such a location. As seen in the Figure 4.4(c), the robot started the movement with initial heading that is not towards the goal location. At first, it corrected its heading towards the goal location, but when it reaches to the part with more dynamic changes, the robot's location estimates became more erroneous. Thus, the estimated heading error becomes larger. In addition, again because of the appearance changes, the robot considered its location as close to the goal location even if it didn't reach to a close location. It had navigation error $\delta g_e = 0.59m$. As such, both the navigation path and the final location of the robot became erroneous. In experiment 22, the goal is from a location where appearances have not changed. The robot is observed to follow a reasonable route and the goal error $\delta g = 0.27$ is nearly the same as localization error. The experimental results show that by coupling navigation with proposed self-localization approach, the robot is able to attain its goal with a goal attainment error that depends on the self-localization error. As this in turn depends on the learning



(a) Experiment 10. (b) Experiment 11.

(c) Experiment 21. (d) Experiment 22.

Figure 4.4. Navigation paths in sample experiments with varying initial location (red circle), initial heading (red vector) and goal location (green circle). Locations at which the robot estimates its location and heading are als shown (blue circles).

grid denseness, the robot can use the proposed algorithm in tasks in which the coarse attainment of the goal suffices.

5. CONCLUSION

Self-localization is one of the challenging and important tasks in mobile robotics due to its necessity for more complicated tasks like navigation and manipulation. In this thesis, an appearance based self-localization approach for place-aware robots is proposed. A place aware robot is capable of recognizing its place after it learned based on the collection of appearances. In the self-localization, all of these learned appearances and their corresponding locations simultaneously affect the location estimate of the robot. In order to limit the set of search in the localization, a novel approach that couples place awareness with localization is introduced. This results in a coarse-to-fine hierarchical reasoning. Place awareness is achieved using previously proposed TSC model where a long-term place memory and a set of processing modules are entailed to enable a robot to build its knowledge of places and refer to this knowledge autonomously. This is followed by self-localization that is activated once the robot recognizes its current whereabouts as one of places that it has learned from its place memory. In this case, the robot retrieves all the learned appearances pertaining to this place from its place memory and uses them to estimate its location. The advantages of this approach is that since the search process is limited to locations pertaining to one place, the localization error is reduced. As it doesn't require the sequentiality of the incoming data, it can be used in cases of kidnapped robots. Furthermore, this localization results can be used as a measure of confidence for the correctness of place recognition.

Finally, we consider self-localization in appearance based navigation. The aim of this approach is to make the robot navigate through a given goal location only by using appearance data. In order to achieve that goal, the robot continuously estimate its location using the proposed self-localization approach and compare it with the goal position. The motion dynamics of the robot also changes based on its relative position to the goal location. In the future, this work can be further expanded to improve the place recognition performance by correcting false place recognitions using

self-localization after detecting them.

REFERENCES

1. Durrant-Whyte, H. and T. Bailey, "Simultaneous localization and mapping: part I", *IEEE Robotics & Automation Magazine*, Vol. 13, No. 2, pp. 99–110, 2006.

2. Kröse, B. J., N. Vlassis, R. Bunschoten and Y. Motomura, "A probabilistic model for appearance-based robot localization", *Image and Vision Computing*, Vol. 19, No. 6, pp. 381–391, 2001.

3. Se, S., D. G. Lowe and J. J. Little, "Vision-based global localization and mapping for mobile robots", *IEEE Transactions on Robotics*, Vol. 21, No. 3, pp. 364–375, 2005.

4. Maddern, W., M. Milford and G. Wyeth, "CAT-SLAM: probabilistic localisation and mapping using a continuous appearance-based trajectory", *The International Journal of Robotics Research*, Vol. 31, No. 4, pp. 429–451, 2012.

5. Li, M., R. Lin, Z. Wang and Y. Hong, "Omnidirectional vision based mobile robot topological localization", *IEEE International Conference on Robotics and Biomimetics (ROBIO)*, pp. 1905–1910, 2013.

6. Engel, J., T. Schöps and D. Cremers, "LSD-SLAM: Large-scale direct monocular SLAM", *Computer Vision–ECCV 2014*, pp. 834–849, 2014.

7. Font, J. M. and J. A. Batlle, "Localization of a mobile robot with omnidirectional wheels using angular Kalman filtering and triangulation", *ASME 8th Biennial Conference on Engineering Systems Design and Analysis*, pp. 713–720, American Society of Mechanical Engineers, 2006.

8. Olson, C. F., "Probabilistic self-localization for mobile robots", *IEEE Transactions on Robotics and Automation*, Vol. 16, No. 1, pp. 55–66, 2000.

9. Lin, R., Z. Wang, R. Sun and L. Sun, "Vision-based mobile robot localization and mapping using the PLOT features", *International Conference on Mechatronics and Automation*, pp. 1921–1927, 2012.

10. Xiong, D., H. Lu, Z. Zeng and Z. Zheng, "Topological localization based on keyframes selection and vocabulary tree for mobile robots", *IEEE International Conference on Robotics and Biomimetics*, pp. 2505–2510, 2013.

11. Rady, S., A. Wagner and E. Badreddin, "Building efficient topological maps for mobile robot localization: An evaluation study on COLD benchmarking database", *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 542–547, 2010.

12. Cummins, M. and P. Newman, "FAB-MAP: Probabilistic localization and mapping in the space of appearance", *The International Journal of Robotics Research*, Vol. 27, No. 6, pp. 647–665, 2008.

13. Nerurkar, E., K. Wu and S. Roumeliotis, "C-KLAM: Constrained keyframe-based localization and mapping", *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3638–3643, 2014.

14. Hwang, S.-Y. and J.-B. Song, "Monocular vision-based global localization using position and orientation of ceiling features", *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3785–3790, 2013.

15. Atanasov, N., M. Zhu, K. Daniilidis and G. J. Pappas, "Localization from semantic observations via the matrix permanent", *The International Journal of Robotic Research*, Vol. 35, No. 1-3, pp. 73–99, 2016.

16. Blaer, P. and P. Allen, "Topological mobile robot localization using fast vision techniques", *IEEE International Conference on Robotics and Automation (ICRA)*, Vol. 1, pp. 1031–1036, 2002.

17. Kosecka, J. and F. Li, "Vision based topological Markov localization", *Proceedings IEEE International Conference on Robotics and Automation*, Vol. 2, pp. 1481–1486, 2004.

18. Ramisa, A., A. Tapus, D. Aldavert, R. Toledo and R. Lopez de Mantaras, "Robust vision-based robot localization using combinations of local feature region detectors", *Autonomous Robots*, Vol. 27, No. 4, pp. 373–385, 2009.

19. Payá, L., F. Amorós, L. Fernández and O. Reinoso, "Performance of global-appearance descriptors in map building and localization using omnidirectional vision", *Sensors*, Vol. 14, No. 2, pp. 3033–3064, 2014.

20. Erkent, O. and H. I. Bozma, "Long-Term topological place learning", *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5462–5467, 2015.

21. Karaoguz, H. and H. I. Bozma, "An Integrated Model of Autonomous Topological Spatial Cognition,", *Autonomous Robots (in press)*, 2015.

22. Erkent, O. and H. I. Bozma, "Bubble space and place representation in topological maps", *The International Journal of Robotic Research*, Vol. 32, No. 6, pp. 672–689, 2013.

23. Pronobis, A. and B. Caputo, "COLD: COsy Localization Database", *The International Journal of Robotics Research*, Vol. 28, No. 5, pp. 588–594, 2009.

24. Karaoguz, H. and H. Bozma, "Reliable topological place detection in bubble space", *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 697–702, 2014.

25. Estrada, C., J. Neira and J. D. Tardos, "Hierarchical SLAM: Real-Time Accurate Mapping of Large Environments", *IEEE Transactions on Robotics*, Vol. 21, No. 4, pp. 588–596, 2005.

26. Filliat, D., "Interactive learning of visual topological navigation", *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 248–254, 2008.

27. Varadarajan, K. M., "Topological mapping for robot navigation using affordance features", *6th International Conference on Automation, Robotics and Applications (ICARA)*, pp. 42–49, 2015.

28. Konolige, K., E. Marder-Eppstein and B. Marthi, "Navigation in hybrid metric-topological maps", *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3041–3047, 2011.

APPENDIX A: BUBBLE SPACE

In this section, bubble space representation is briefly summarized for completeness. Further details can be found in previous work [22]. Lets consider a robot positioned at a location $c \in R^2$ with heading $\alpha \in S^2$. The base point associated with this location and heading is shown as $x = x = \begin{bmatrix} c^T & \alpha \end{bmatrix}^T \in \mathcal{X}$ where $\mathcal{X}$ is the base space which is the set of all combinations of base point locations and headings. The bubble space $B = \mathcal{X} \times \mathcal{F}$ is an abstract representation of robots base along with its viewing directions (pan and tilt). The set of viewing directions (pan and tilt) is shown by $\mathcal{F} \subset S^2$ where each pan and tilt direction vector is shown as $f = \begin{bmatrix} f_1 & f_2 \end{bmatrix}^T$. Each point on bubble space b is defined as $b = \begin{bmatrix} x & f \end{bmatrix}^T$ where $x \in \mathcal{X}$ and $f \in \mathcal{F}$. Bubble surfaces $B_i(x,t) : Im(h(x)) \times R^{\geq 0} \rightarrow R^{\geq 0}$ are hypothetical spherical surfaces around the robot. It is defined as:

$$B_i(x,t) = \left\{ \begin{bmatrix} f \\ \rho_i(b,t) \end{bmatrix} | \forall f \in \mathcal{F} \text{ and } b = \begin{bmatrix} x & f \end{bmatrix}^T \right\} \tag{A.1}$$

where the function $\rho_i : B \times R^{\geq 0} \rightarrow R^{\geq 0}$ is a Riemannian metric that encodes the observed sensory values. The bubble surface is initialized with radius $\rho_0 \in R^{\geq 0}$. While the robot looks around its environment, for each viewing direction, a feature value $q_i(b,t) \geq 0$ is calculated and the bubble surface $B_i(x,t)$ is deformed at the corresponding viewing direction by an amount depends on the associated sensory information $q_i(b,t)$:

$$\rho_i(b,t^+) = q_i(b,t) \tag{A.2}$$

Bubble descriptors are vector representation of bubble surfaces and constructed by double Fourier series representation as follows:

$$\rho_i(b,t) = \sum_{h_1=0}^{H_1} \sum_{h_2=0}^{H_2} \lambda_{h_1,h_2} z^T_{x_i,h_1h_2}(t) e_{h_1h_2}(f) \tag{A.3}$$

For each (h_1, h_2) the vector $e_{h_1h_2}(f) \in R^4$ consists of an orthonormal set of trigonometric basis functions as:

$$e_{h_1h_2}(f) = \begin{bmatrix} \cos(h_1f_1)\cos(h_2f_2) \\ \sin(h_1f_1)\cos(h_2f_2) \\ \cos(h_1f_1)\sin(h_2f_2) \\ \sin(h_1f_1)\sin(h_2f_2) \end{bmatrix} \tag{A.4}$$

The corresponding vector $z_{x_i,h_1h_2}(t) \in R^4$ is defined as:

$$z_{x_i,h_1h_2}(t) = \frac{1}{\pi^2} \begin{bmatrix} \int_0^{2\pi}\int_0^{\pi} \rho_i(b,t)\cos(h_1f_1)\cos(h_2f_2)d_{f_1}d_{f_2} \\ \int_0^{2\pi}\int_0^{\pi} \rho_i(b,t)\sin(h_1f_1)\cos(h_2f_2)d_{f_1}d_{f_2} \\ \int_0^{2\pi}\int_0^{\pi} \rho_i(b,t)\cos(h_1f_1)\sin(h_2f_2)d_{f_1}d_{f_2} \\ \int_0^{2\pi}\int_0^{\pi} \rho_i(b,t)\sin(h_1f_1)\sin(h_2f_2)d_{f_1}d_{f_2} \end{bmatrix} \tag{A.5}$$

In Figure A.1 some sample scenes taken from ISL and COLD dataset are shown and their corresponding bubble surface representations are shown.

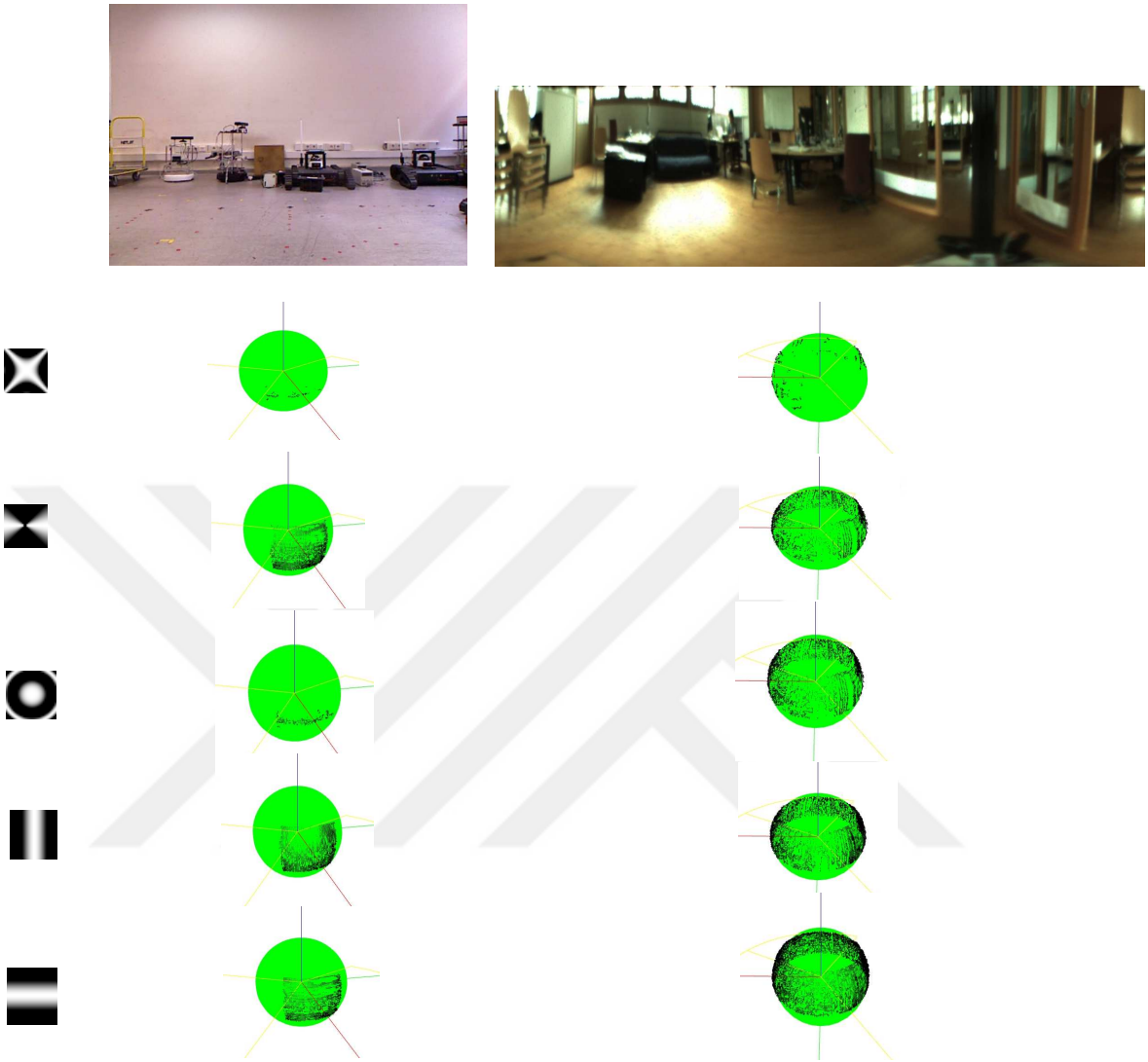


Figure A.1. Sample views and corresponding bubble surfaces.

A bubble descriptor $I(x,t) \in R^{N_I}$ is an N_I dimensional vector defined as:

$$I(x,t) = \left[\; I_{1,00}(x,t), \cdots, I_{N_v H_1 H_2}(x,t) \; \right]^T \tag{A.6}$$

where

$$I_{i,h_1 h_2}(x,t) = z^T_{x_i,h_1 h_2}(t) z_{x_i,h_1 h_2}(t) \tag{A.7}$$

Bubble descriptors are rotationally invariant.

APPENDIX B: PLACE DETECTION MODIFICATION

In this work, we have used the previously introduced Topological Spatial Cognition (TSC) model to provide the robot with place awareness. Within place awareness, the first step is detection - deciding what constitutes a place. In the previous work [21], a perspective camera is used as a visual sensor and the appearances from that camera is compared to detect the places. Place detection is based on the difference of visual descriptors of two consecutive appearances, simply, $|I(x_k) - I(x_{k-1})| < \tau_\kappa$ if the consecutive images are coherent. With the perspective camera, when the robot turns around itself, the appearance that the robot sees significantly changes although the robot doesn't move linearly. In such cases, a place is terminated. However, with omnidirectional sensor, the image descriptors doesn't change when the robot turns around itself because the bubble descriptors used are rotationally invariant. Therefore, the appearances do not change significantly when the robot follows a trajectory with small steps. In order to detect places with such sensors, another check parameters is introduced. The incoming data is started to be compared with not only the appearance from the previous base point, but also with the first base point in the detected. If $|I(x_k) - I(x_1)| < \tau_\kappa \tau_e$ and $|I(x_k) - I(x_{k-1})| < \tau_\kappa$ it means the corresponding base point is coherent and assigned to the detected place. The value of parameter τ_e depends on the descriptor of the appearances, because it affects the amount of change in the descriptor based on the physical deviation. In the experiments with COLD dataset benchmark, we used $\tau_\kappa = 0.38$ and $\tau_e = 5$, in real robot experiments we used $\tau_\kappa = 1.2$ and $\tau_e = 10$ because the difference between the descriptors from consecutive base points is larger in the real robot experiments due to the camera properties.

APPENDIX C: TURTLEBOT ROBOT

Turtlebot is a differential wheel mobile robot as shown in Figure C.1.

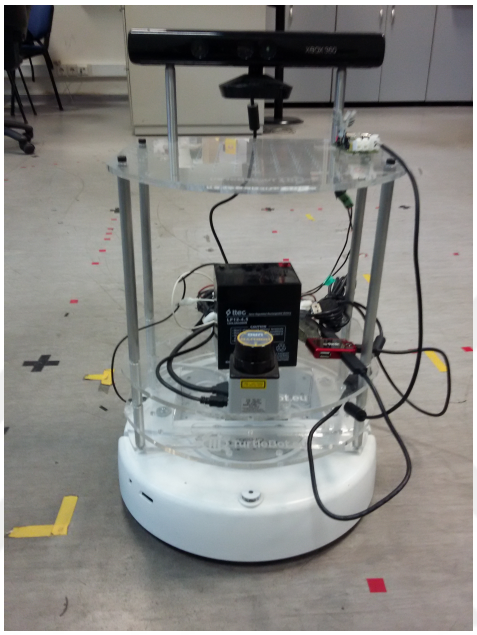


Figure C.1. Turtlebot robot.

C.1. Hardware

The technical specifications of the robot are as follows:

- Driving mechanism: Differential drive
- Diameter: 33.5 cm
- Height: 60 cm
- Maximum speed: 0.5 m/s
- Weight: 6 kg
- Wheel diameter: 63.5 mm
- Wheel Encoders: 360/32 degree, 2 cm accuracy
- Maximum Moving Time: 3.5 hours (with 3Ah battery fully charged and no payload)
- Communication with Robot Base: UART

- Battery for Hokuyo and Kinect: 12V 4.5AH
- DC to DC Converter for Hokuyo: Converting 12V to 5V

Its processing is done by Lenovo IdeaPad Yoga 11S Ultrabook (Intel Core i3 3229Y processor). Its sensing capabilities are as follows:

- Kinect (laser range scanner and camera)
- Hokuyo URG-04LX
- V 360 omni-directional camera
- Cliff sensors
- Bump sensor

The wiring diagram of Turtlebot is shown in Figure C.2.

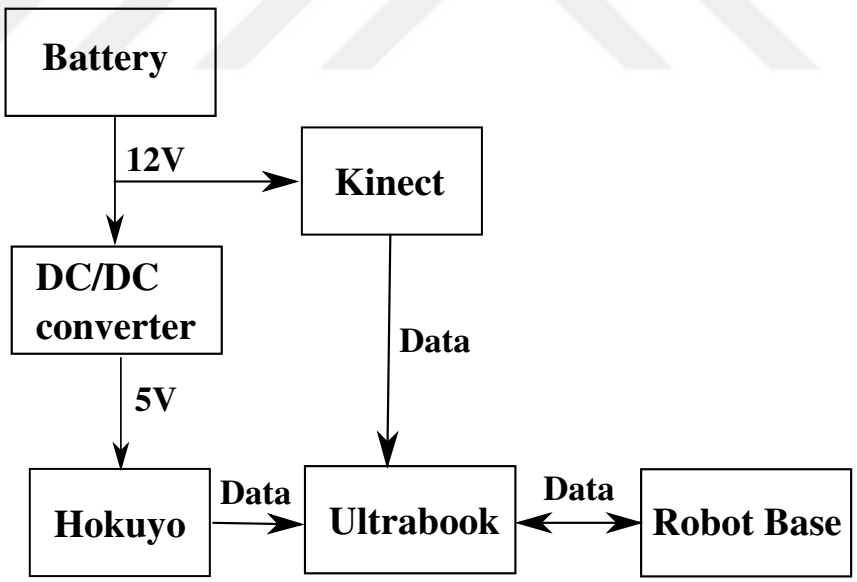


Figure C.2. Wiring diagram of turtlebot.

C.2. Software

The robot requires installation of the following software for operation:

- Operating System: Ubuntu 12.04 LTS
- ROS (Robot Operating System) Fuerte, Desktop Sharing, Remote Desktop Viewer

C.3. User's Guide

Before starting to use the robot, the wiring of the robot should be done according to the wiring diagram of the robot as shown in Figure C.2.

C.3.1. Powering the robot

The robot has its own integrated battery. It should be charged before using. Hokuyo and Kinect sensors require 5V and 12V power sources, which are provided with 12V battery. The Hokuyo sensor is connected to the battery via DC to DC converter and Kinect sensor is connected directly. The V360 omnidirectional camera also has its own battery. After powering on the robot base, the ultrabook and the sensors, the robot is ready for the operation.

C.3.2. Remote Connection

The robot can be controlled via remote connection. The following steps need to be followed:

- Desktop Sharing program should be working on the robot. It should also be noted that the robot should be connected to the network (either via WiFi or LAN connection).
- The remote computer connects to the robot via Remote Desktop Viewer program which is on the same network with the robot.

- The necessary packages can be run using remote computer.

C.3.3. Operating the Robot

In order to run the Turtlebot robot, first the wiring diagram of the robot shown in Figure C.2 is completed and the power is delivered to the robot as explained in section C.3.1. Then the remote connection is established as explained in Section C.3.2. Then using a computer running Ubuntu, the robot connection is done by completing the following steps:

- Open a new terminal and run `"roscore"`.
- In a new terminal tab, run:
  `"roslaunch turtlebot_bringup minimal.launch"`.
  Once the connection is done at the first time, the beep sound will come from the robot and the power led will be turned off.
- To connect the Kinect camera, in a new terminal tab run:
  `"roslaunch openni_launch openni.launch"`.
- To connect V360 camera, in a new terminal tab, run:
  `"rosrun camera2ros camera2ros_node"`.

APPENDIX D: SOFTWARE USER GUIDE

D.1. Grid Based Self-Localization

In proposed self-localization approach is supported by extensive set of experiments using Turtlebot robot. In order to test the approach, appearance data from Kinect sensor is collected in Intelligent Systems Laboratory using a previously developed software framework called ISLFramework. In order to collect data the following steps should be followed:

- Connect the robot and Kinect sensor as explained in C.3.3.
- In a new terminal tab, open ISLFramework by running:
  `"rosrun ISLFramework example"`.
- In the main window shown in Figure D.1, select OpenFile.

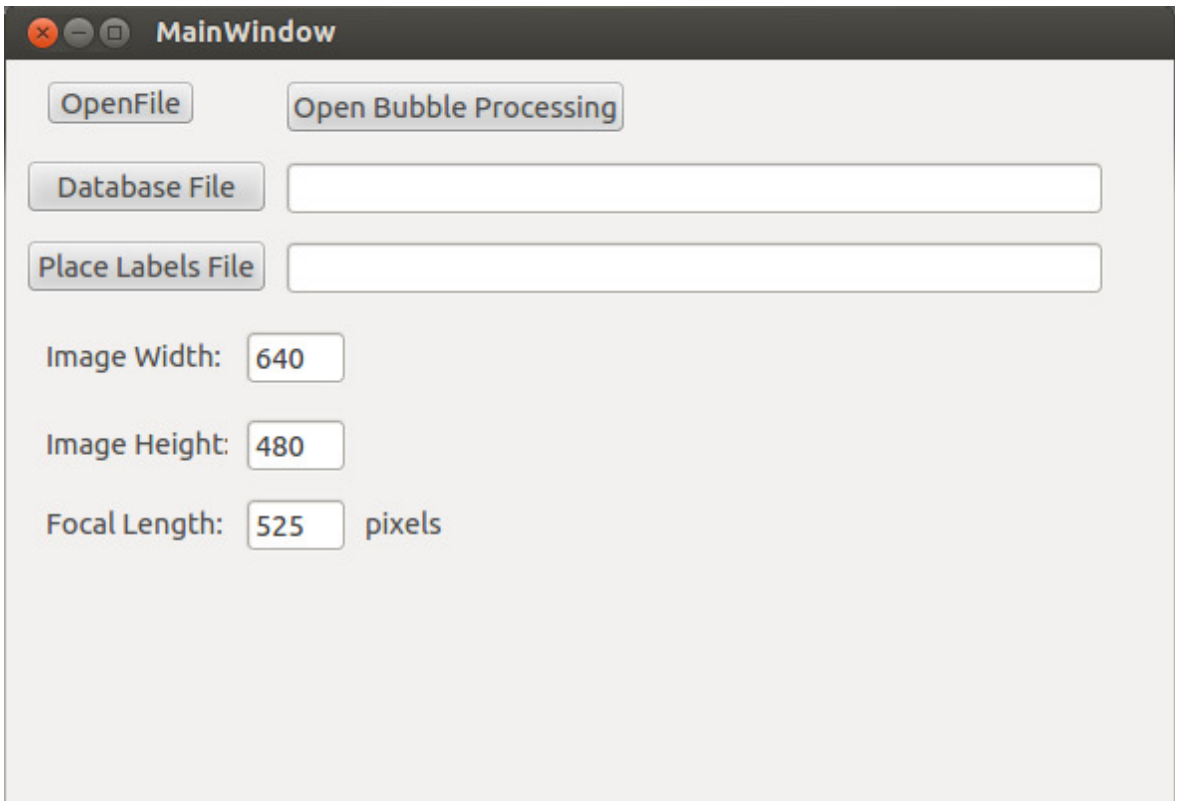


Figure D.1. Main window of ISLFramework user interface.

- In an opened Kinect dialog box shown in Figure D.2, by clicking start button, the RGB-D Kinect image can be saved.

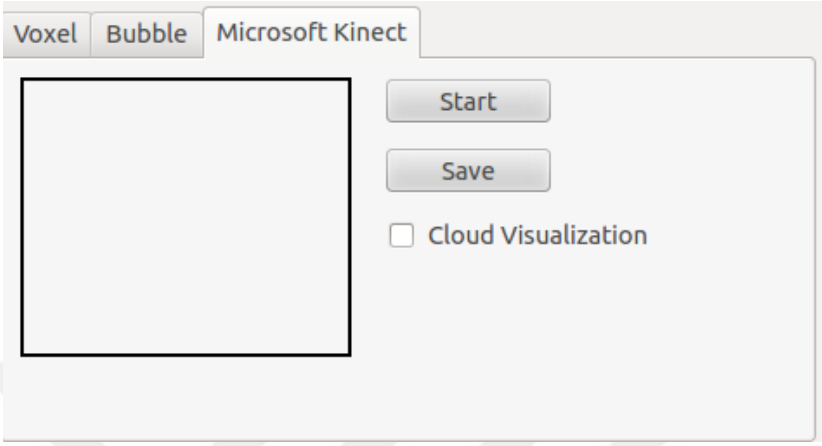


Figure D.2. Kinect image acquisition dialog box.

- Using these images the bubble descriptors can be constructed and saved in a database file using the Image Processing dialog box shown in Figure D.3 by selecting the necessary images and filter types.

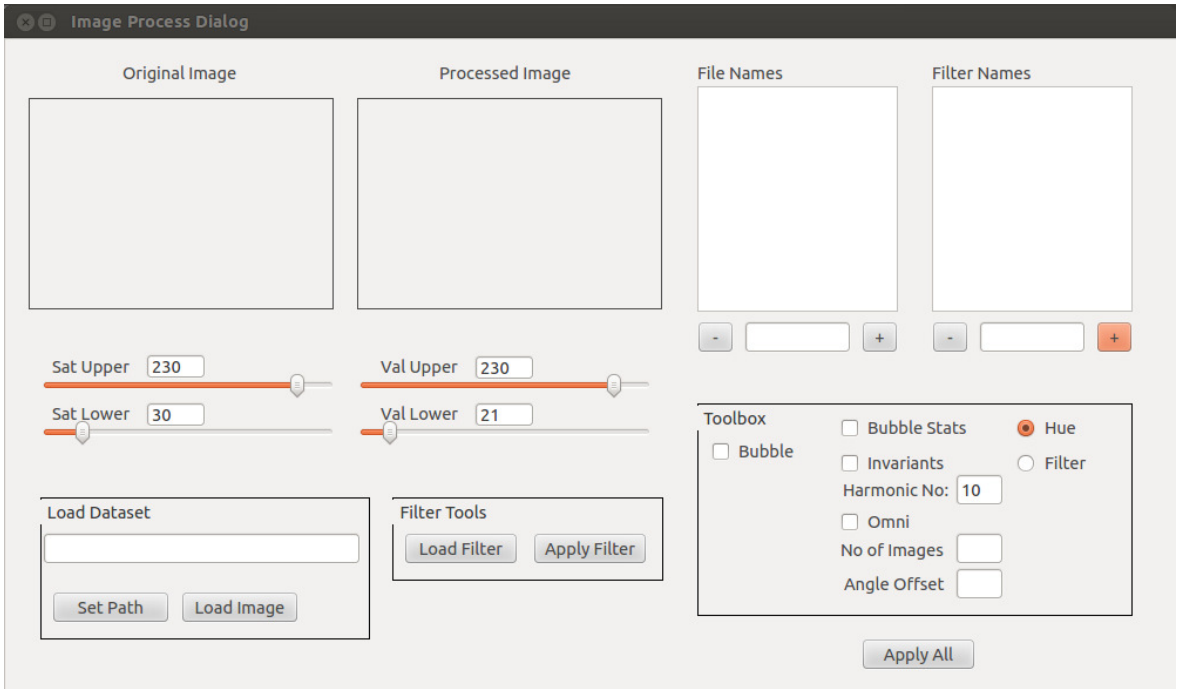


Figure D.3. Image processing dialog box.

- After constructing the database of bubble invariants and entering the their location information, the localization method can be run by running the following

command in a new terminal tab:

"roslaunch ISL_localization only_bubble_space.launch".

By this command, the robot will turn around itself and collect Kinect data, compare them with the ones from database and estimate the current location of the robot.

D.2. Coupling of Place Awareness and Self-Localization

The proposed approach of localization and coupling of localization with place awareness model is developed in the framework of Robot Operating System (ROS) which is an open-source robot meta-operating system. The place awareness is achieved by using previously developed TSC model [21] and it is coupled with the proposed localization model. The implementation of localization model is achieved via the design and development of a ROS node and it is integrated with the 2 developed ROS nodes for TSC model architecture. The coupling of these two architectures is shown in Figure D.4.

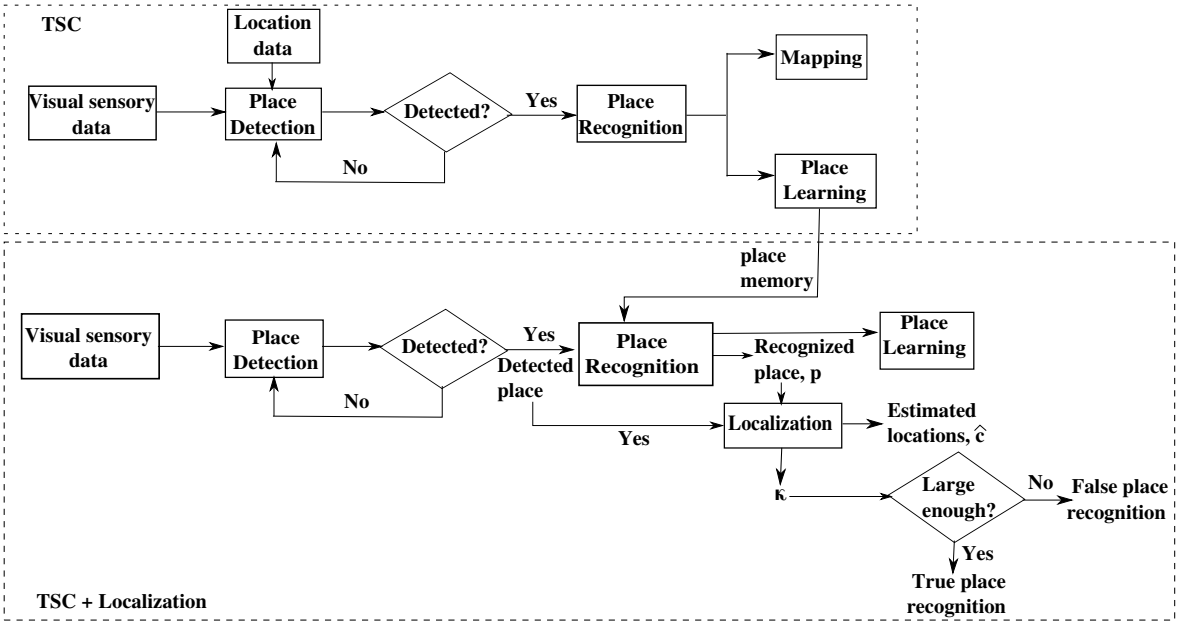


Figure D.4. Implementation of coupling place awareness and localization

The TSC model includes two ROS nodes to achieve place awareness. The first node is about sensory data representation and place detection. There should be empty

databases in the home folder to store the necessary knowledge. The place detection node subscribes to two messages, the Sensory Input which is image frame and the Node Control which controls the node's operation such as start, pause and shutdown. After the node is started, upon receiving the image frame, it transforms the frame into the internal representation and activates the place detection module. The image frame can be read from the sensors of the robot when the modules are working on robot or it can be read from a folder when the modules are working offline. As shown in the Figure D.4, the place detection module can give two possible results: either place detection in progress or a place is detected. Once the place is detected it advertises the detected place id and the knowledge associated with the detected place and transition regions are stored in the robot's knowledge base as follows:

- Base points: For each base point in the environment, their id, corresponding bubble descriptor, their status information and x and y coordinates of the location of the base point are stored. The status shows whether the base point is uninformative, incoherent or coherent. Location information is read from previously constructed text. This data is used for localization, either for estimating the locations of other base points, or as a ground truth for estimated location.
- Detected places: The id, mean descriptor and member base points of the detected place is stored.
- Transition regions: The id and start and end points of the transition regions between places are stored.

The place recognition node is responsible for initializing, updating and maintaining long term memory. It works simultaneously with place detection module and activated whenever a place detection event occurs. Once it gets the detected place id, it reads the detected place from detected places database and perform the necessary operations. If the place memory is empty, it initializes it with the newly detected place. If there is place memory is not empty, it activates recognition to decide if the detected place is previously visited or not. If the place is recognized, it advertises the recognized place id if it is not recognized it expands the place memory with the newly detected

place.

The localization model is responsible for estimating the locations of the base points in the detected place when the robot revisits the same environment. It works simultaneously with place detection and place recognition nodes and subscribes to them. It listen the topic of newly detected places from the place detection module and the topic of recognized place id from the place recognition module. The localization module is activated once it gets the recognized place id. When it gets the recognized id, it estimates the location of the detected place based on the location and descriptor knowledge of the base points of the recognized place. The estimated locations are advertised and the localization error is calculated by calculating the linear distance between estimated locations and their ground truth values read from the newly detected place knowledge. In the launcher file constructed for localization node, the path of the knowledge base file of the place memory constructed in the first visit should be specified.

In order to run these modules on robot the following steps should be followed:

(i) Opening the Turtlebot robot: Before opening the robot, the necessary wiring should be done and the robot should be powered as explained in C. After that the following commands should be run in a new terminal windows:
   - To activate ROS:

     `roscore`
   - To activate the robot:

     `roslaunch turtlebot_bringup minimal.launch`

(ii) To activate the omndirectional V360 camera on Turtlebot:

`rosrun camera2ros camera2ros_node`

(iii) To activate place detection node control:

`rostopic pub placeDetectionISL/nodecontrol std_msgs/Int16 1`

(iv) To activate place detection model for omnidirectional camera:

`roslaunch placeDetectionISL_omnidirectional placedetection.launch`

(v) To activate place recognition model:

`rosrun createBDSTISL createBDSTISLNode`

(vi) To activate the localization in the second visit:

`roslaunch localizationISL localization.launch`

D.3. Navigation

In the navigation approach, it is assumed that the robot has correctly recognized its place. Therefore the knowledge about the base points of its place should be provided in a database. In order to run the navigation model, the following commands should be run in separate terminal tabs:

- `"roscore"`
- To connect the robot:

  `"roslaunch turtlebot_bringup minimal.launch"`
- To connect the V360 omnidirectional camera:

  `"rosrun camera2ros camera2ros_node"`
- To run the navigation model:

  `"rosrun localizationISL navigation"`

After running these commands, the robot will start to navigate through the desired goal location.