HOW $ALBOT_1$ COMPUTES ITS ENDURING MAPS AND FINDS ITS WAY HOME

A THESIS SUBMITTED TO AUCKLAND UNIVERSITY OF TECHNOLOGY
IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Supervisors
Prof. Wai-Kiang Yeap Supervisor
Dr. Minh Nguyen Supervisor

April 2020

By
Wenwang Pang Candidate
School of Engineering, Computer and Mathematical Sciences

Copyright

Declaration

I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person nor material which to a substantial extent has been accepted for the qualification of any other degree or diploma of a university or other institution of higher learning.

Signature of candidate

Acknowledgements

I would like to express my special appreciation and thanks to my primary supervisor, Professor Wai Yeap; you have been a tremendous mentor for me. I would like to thank you for allowing me to join this fantastic lab and do such fascinating research. Your patience, motivation, and immense knowledge keep encouraging me and my research. I appreciate your time and ideas to make my Ph.D study a fascinating and a productive one.

I would also like to thank my secondary supervisor, Dr. Minh Nguyen, for his insightful comments and encouragement but also invaluable perspectives for my research whenever needed.

My sincere thanks to all CAIR mates: Peng Xia, Alvaro, Samantha and Darrel. And, to thank Bumjun Kim, a senior technician in the school, for helping me a lot whenever I came across technical problems. Special thanks to one of my best friends, Dr.Jing Ma; you have helped me a lot, both in my study and in my life.

I gratefully acknowledge the support from the school that made my Ph.D. study possible. My study was mostly supported by the School of Engineering, Computer and Mathematical Sciences of Auckland University of Technology which included my Doctoral Fees Scholarship. I would like to say that this research would not have been possible without all their assistance.

Last but not the least, I would like to thank my family. For my parents and my partner Yuan, their selfless love and support encouraged my every second in my Ph.D study. Thank you.

Abstract

This thesis continues the experimentation with Albot1, the second in the series of Albots, in search for a theory of cognitive mapping. Albots are robots created in the Centre for Artificial Intelligence Research (CAIR) Laboratory at AUT for investigating hard problems in cognitive sciences.

Using Albot1, Yeap and Hossain (2019) developed a theory of cognitive mapping that explains what kind of a global map is computed and why. However, they also leave open a puzzle: why couldn't one use the transient egocentric global map computed to detect that one is still moving in the same local space? Failure to do so would limit the usefulness of the theory itself. The goal of this thesis is to show how the theory could be extended to overcome this puzzlement.

To find a cognitively interesting solution, I continue to empower Albot1 to perform tasks that are deemed to be most relevant to cognitive mapping at this level. If a species were to compute a route map and an egocentric map as suggested in the theory, what is next? What would be computed by, say, a more advanced species and why? I began by empowering Albot1 to find its way home using the maps computed and in particular, could Albot1 discover the use of short-cuts? The first experimentation, consisting of nine experiments, leads to the use of a trace map for returning home. The latter is a global map of key points in the journey that can direct one to return home. It is generated directly from the route map and interestingly, one could also detect short-cuts using it. However, its use is limited as it is still a global map and therefore its accuracy

is an important factor to get one home.

The second experimentation, consisting of three experiments, focuses on studying the use of a more enduring map in cognitive mapping. What does it mean to have an enduring map? How is the process of computing such a map interacts with that of an egocentric map? What role does it play in cognitive mapping? These are the questions being investigated. The result is the realization that computing such a map is problematic and requires the use of a new set of mechanisms that are uncommon among species. It will also lead to computing a more precise and detailed global map.

Empirical researchers have shown the common use of geometry of a place to re-orient in a local environment among a variety of species. This suggests that computing the geometry of a place could be the first step towards computing a more enduring representation in cognitive mapping. The third experimentation, consisting of two experiments, investigated this possibility. Could computing the geometry of a place lead to a more enduring representation that could be used to find one's way in the environment? Could one discover short-cuts using these representations? The result leads to computing a useful place map from the route map. The former is an abstract representation that could help one to re-orient in a place and to find short-cuts home. It is not a precise map and computing it is straightforward, thereby overcoming the problems one faced earlier.

A major contribution of this work is the successful extension of Yeap and Hossain's (2019) theory, not only to overcome one of its severe limitations but also in ways that are cognitively interesting. The extension of the theory to compute either or both a trace map and/or a place map from the route map means that what is computed initially (i.e. the route map) is rich enough to support cognitive mapping. Both representations support the use of short-cuts to go home and they are computed directly from the route map using simple mechanisms that are found in many different species.

Publications

Contents

List of Tables

List of Figures

Chapter 1

Introduction

Since Tolman (1948) coined the term cognitive map, there has been much empirical studies investigating its use in different species in both natural and artificial (including simulated ones) environments. Major work that appeared since then include Lynch (1960 – who argued that there exists five basic elements in the mental map of city residents), O'Keefe & Nadel (1978 – who argued that the hippocampus is the site for computing a cognitive map and a noble prize was awarded for their work in 2014), and Cheng (1986 – who discovered that local geometry of a place plays an important role in helping one to orient in a local space).

Despite these significant work, and the enormous interests generated, some researchers have questioned the usefulness of the concept itself and have called for the idea to be abandoned, claiming there is little empirical evidence supporting it. For example, Benhamou (1996) and Bennett (1996) both argued that there are often simpler explanations to account for novel short-cutting behaviour. Mackintosh (2002), on the other hand, argued that the natural world provides us with a challenging environment to navigate in and different species would learn to use a variety of mechanisms to do so and not just that of a cognitive map. Finally, while humans might have a cognitive map, Wang and Spelke (2002) argued that it could come from our symbolic reasoning about

the environment and not that we have evolved a separate and unique mapping process. Humans, just like other animals, still navigate using some basic mechanisms.

Researchers interested in cognitive mapping often thought of it as a process whose primary goal is to compute a global metric map of the environment and use it to find one's way. As such, cognitive mapping is distinct from other known mechanisms that animals use to find their way. The latter includes path integration, matching of viewpoint-dependent representations, and a re-orientation system based on local geometry. However, Yeap (2011a) recently proposed a radical idea about cognitive mapping. He argued that cognitive mapping has evolved to take maps of local environments computed at the perceptual level (notably vision) as input and compute representations to maximise foraging efficiency and understanding of the environment for the individual species. For example, vision does not compute just what is out there but also where things are. Hence, remembering a view prior to moving forward would provide one with a map of the local environment that one is about to explore. With the map, one knows where things are in one's current local environment and where to run to if a predator suddenly appears. Cognitive mapping, viewed this way, is thus a fundamental part of nature's process.

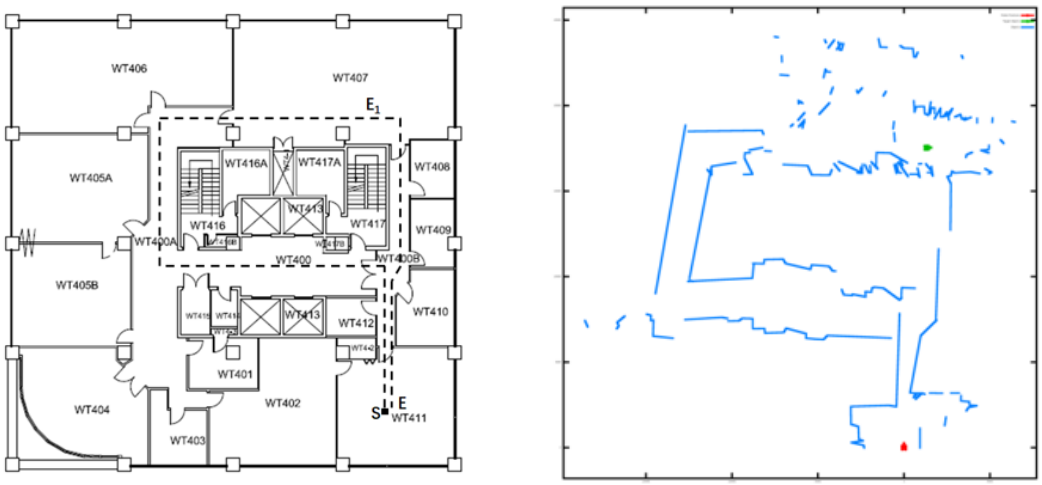
In developing his model, Yeap argued that such a view map is not updated with information from subsequent views. This is in direct contrast to the idea of integrating every successive view to compute a global map. Instead, objects in it are tracked in successive views which then allow one to localize oneself in the map, via a process of triangulation. If no objects are in view, one is deemed to have moved out of the current local map and the next view is remembered prior to exploring the new local environment. Using this model, cognitive mapping emerges as a strategy that takes one view at a time as one moves in and out of a local environment. This strategy is useful for the immediate task on hand: locate food, grasp it and know where you should run to next. Could such a strategy be extended to generate more complex cognitive mapping

behaviour observed in other species?

An important characterisation of cognitive mapping in higher species is that a global map that is inexact, incomplete and fragmented is computed. Could the basic strategy be extended to compute such a map? To investigate, Yeap (2011b) developed a robotic approach whereby one empowers a robot with the basic mapping process and designs algorithms for it to compute the desired representation (in this case, a global map) in ways that exhibit cognitive mapping behaviour observed. In contrast to traditional AI/robotics approach (Chown and Yeap, 2015), the goal is not to find the most efficient algorithms for computing the desired representations but rather some algorithms to show what could be computed and why. In short, the implementation allows us to "get inside the head to see what is happening" and if the robot's behaviour bear similarities to what is observed in other species, its process is a plausible model for cognitive mapping. Such a robot is referred to as an Albot (Yeap, 2011b).

Using this approach, Yeap and Hossain (referred to as Y&H) (2018) created Albot1, a pioneer-3DX mobile robot equipped with a 180-degree SICK laser ranger and an odometer, and empowered it with Yeap's (2011a) initial model of cognitive mapping. By allowing views in the route map to overlap, they show that one could compute a global metric map that is inexact (due to no error correction). Given that only some views are remembered, the map is also incomplete. Fig. 1.1 shows one such global map computed by Albot1. The learning process of Albot1 is not like what Tolman and Honzik (1930) tried to investigate using rats. In their theory, subjects were asked to learn the same environment couple of times, which was referred to as training, then tested whether those subjects computed the cognitive map in heads to help them to find hidden food. However, in this study, Albot1 is developed to study the process of cognitive mapping instead of a reinforced process. For each environment, Albot1 is instructed to learn it once, then uses what it learnt to form maps.

Such a global map bears two key interesting characteristics of a cognitive map i.e.



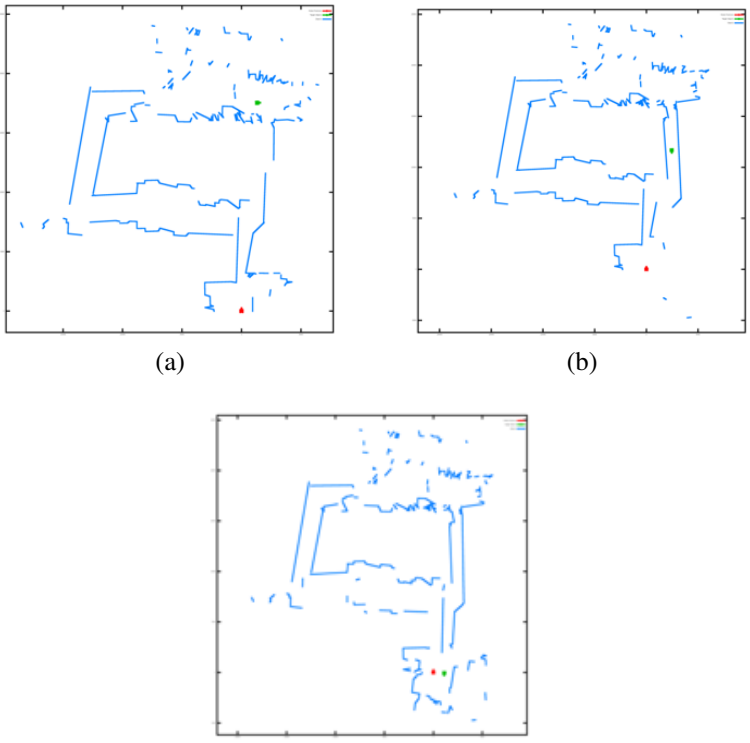
(a) (b)

Figure 1.1: (a) A typical test environment. (b) Albot1's global map of the environment at E1.

it is inexact and incomplete. If so, would computing it tell us more about cognitive mapping in general? To investigate, Y&H empowered Albot1 with this extended algorithm and performed three experiments to learn more about the process and the map computed. Their results showed that the process is modular and view-based and compute a series of maps, from the initial local map to a set of disjointed global maps. These maps have been proposed by empirical researchers in the past and they refer to them as route maps, survey maps, and fragmented maps. The process utilized mechanisms (such as path integration, triangulation, tracking of objects) that are available across species. Its performance degrades gracefully when faced with unexpected changes. It is thus an attractive model for cognitive mapping. However, despite being an attractive model for cognitive mapping, the global maps computed has an unexpected but intriguing shortcoming. It does not allow one to recognise that one is re-visiting a part of the environment just visited.

To illustrate this shortcoming, observe that if Albot1, in Fig. 1.1b, continues to move forward and reach its starting position, it will not be able to close the loop. However,

this latter step is needed only if one were to compute a complete and exact map. Given there is no such a cognitive map, Y&H argued that loops are not closed. They never do since this would involve correcting errors and hence, if incoming information overlaps with any part of the map, Y&H argue that that part of the map is simply deleted and replace with the incoming local map. Fig. 1.2 shows how Albot1's global map changes as it re-visits the environment.



(a) (b)

(c)

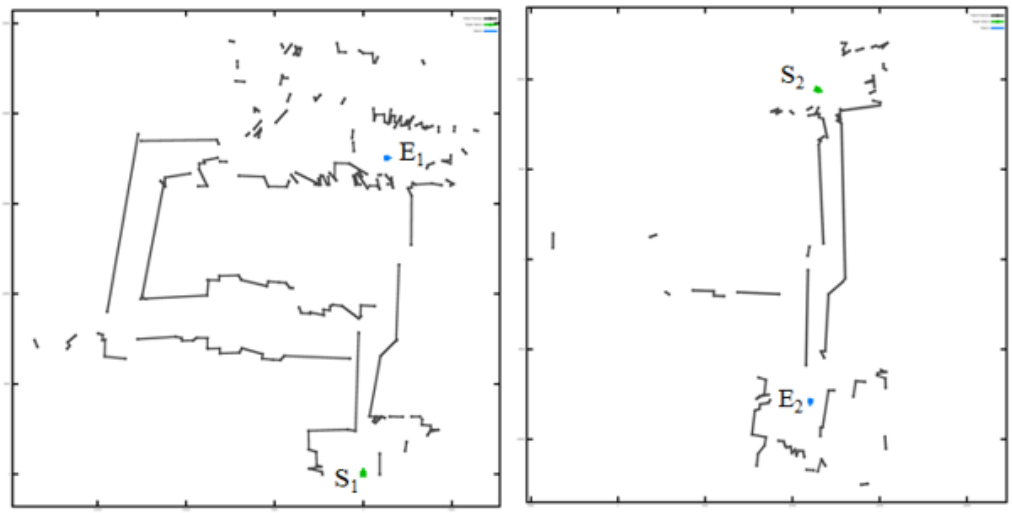
Figure 1.2: Albot1's egocentric map as it explores the environment in Figure 1.1

Such a small but significant change in the algorithm produces an interesting global map; one that has an unexpected "egocentric" nature. The latter is unlike what has often been debated among cognitive mapping researchers (see Yeap (2014) for a review) i.e. whether a cognitive map uses an egocentric or a non-egocentric frame of reference. The egocentric map computed using Y&H's algorithm refers not to where lies the centre of the co-ordinate system used but to where lies the centre of the map when reading the map. The centre provides a reference point for deciding how the map is interpreted. In an error free map, it does not matter since the distance between any two points on it is exactly what it says. In an egocentric map, the centre is the current local map (i.e. where one is) and from it, local maps that are not far behind it are less distorted than those that were entered a while back. Note that the centre of the map shifts when one moves from one local map to the next; thus giving it its egocentric nature. This is evident in Figure

1.2, whereby despite being seriously distorted when revisiting the starting position, the old information has been wiped away and the newly added description captures the spatial layout of that part of the environment well. What is distorted are the earlier parts of the map.

An egocentric map, as defined here, is transient since parts of it often get deleted. However, it is possible to save a copy of the egocentric map prior to any such deletion. The current egocentric map is then refreshed with the current local map, thereby creating a series of such global maps. In the above example, when Albot1 turns, its incoming local map overlaps a part of its egocentric map. Henceforth, its egocentric map is refreshed and one gets two egocentric maps when Albot1 reaches E2 (Figure 1.3). From now on, whenever such deletions occur, Albot1's egocentric map will be refreshed and start all over again. Note that such deletions can occur without necessarily travelling in a large circular route. It could just be walking down the corridor and up again or round and round in a single room. The corridor and room will be re-computed afresh and not recognized as a place re-visited. This is contrary to the believe that one computes a cognitive map to help one recognize places visited earlier. This poses a serious problem as a model of cognitive mapping because one does not expect, say, one to walk down the corridor, turn around and think that one is now in a novel environment.

The primary goal of this research is to find a cognitively plausible solution to extend Y&H's model overcoming this shortcoming. To develop such a solution, I will follow Y&H's approach. Albot1 will be empowered with new algorithms and experiments will be conducted to gain insights into cognitive mapping. Empirical researchers have discussed the idea that one computes first a transient map and then a more enduring map (Creem and Proffitt, 1998; Hartley and Burgess, 2005; Huttenlocher, Hedges and Duncan, 1991; McNamara, 2003; Mou, McNamara, Valiquette and Rump, 2004; Wang and Spelke, 2000). This empirical idea will form the basis for my investigation in seek of an answer: Can Albot1 compute a more enduring map and why? Studying this



(a) (b)

Figure 1.3: When Albot1 turns at E1, its existing egocentric map will start afresh building a new egocentric map. The previous one (left) could be kept to create a series of global maps computed.

key question is the motivation of this research, which can help us to understand the process of cognitive mapping in nature deeper inside instead of directly building up a comprehensive system.

The rest of the thesis is organised as follows. In chapter 2, I review existing literature on models of cognitive mapping. My review is not intended to be critical but rather to show the range of work done in this area. As we shall soon see, an area of empirical studies that is of particular interests is the use of geometry in cognitive mapping and a summary of some recent work will also be provided in the review. In Chapter 3, I began my investigation by empowering Albot1 to use its existing maps to navigate itself in its environment. In particular, could Albot1 find its way home? Could it do so using short-cuts? With its limited mapping process, one expects Albot1's ability to utilize its map would be limited too but by identifying these limits, it helps us to identify why and where in the process an enduring map is computed. In Chapter 4, I investigate how Albot1 could compute an enduring map and use it to find its way home. I studied

two strategies. The first is simply an attempt to compute an enduring map given the route map as input. This is an engineering approach. The second strategy is to adopt a mechanism that has been observed to be widely used among species and observe how a more enduring map could emerge in the cognitive mapping process. This mechanism is one that has been much debated among empirical researchers recently and involves the use of geometry for re-orientation in a place.

In chapter 5, I conclude with a discussion of how Y&H's model of cognitive mapping could be extended to include the use of geometry of a place to re-orient oneself in it. However, discussion of such extensions from an empirical standpoint is limited and mostly beyond the scope of this thesis. Future directions for developing the model will also be discussed.

Chapter 2

Background and Literature Review

There has always been a huge interests in spatial cognition in general and cognitive mapping in particular. Not surprisingly, the ability to find one's way in one's environment is the key to survival for natural species and a key skill for mobile robots. In section 2.1, some recent work in both areas (three psychological studies and three computational models) are reviewed to show the range and recent work done in this area of research. One recent and much discussed empirical research is the idea of using geometry of a local environment to re-orient oneself in it. This is an important idea which will be used to help solve the problem identified in Y&H's model of cognitive mapping. In section 2.2, some background of such work is discussed.

2.1 Recent work/models of Cognitive mapping

2.1.1 Wang and Spelke's (2000) model

In Wang and Spelke's (2000) work, they tested whether human navigation depends on enduring allocentric representations, or on momentary egocentric representations by using a disorientation paradigm. They suggested that the configuration error after

disorientation will reveal which representation is being used. Configuration error provides measurement of the relative accuracy of pointing to different objects and is defined as the standard deviation of all individual object's errors. Their hypothesis is as follows: Disorientation will have different effects on enduring allocentric and dynamic egocentric spatial representation of objects. If humans remember the locations of objects allocentrically, pointing-accuracy to objects after disorientation will be the same. In contrast, if humans remember the location of objects egocentrically, then disorientation configuration error will be increased. This is because, in allocentric representation, only one update (view's position) is necessary with respect to the reference system, whereas in egocentric representation, update is necessary for all individual objects with respect to view's new position.

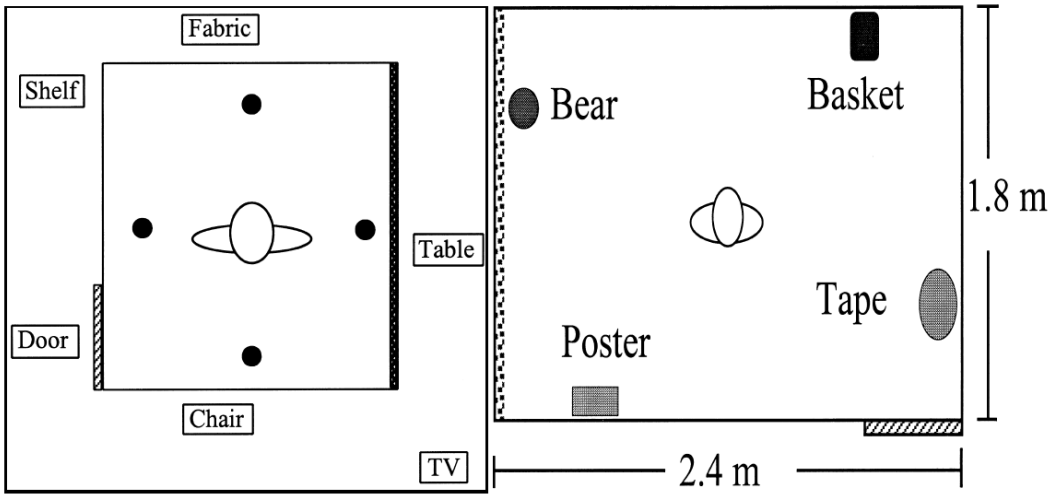


Figure 2.1: This figure shows two different experiments conducted by Wang and Spelke. The left diagram shows the layout of the test environment in the first environment. The right shows a rectangle environment placed different objects for pointing to each corner. (Reproduced from Figures 1 and 2, Wang and Spelke, 2000)

They tested their hypothesis using seven experiments under different conditions. For example, in one experiment, subjects learned locations of six objects situated around a chamber as shown in Figure 2.1. In the testing phase, subjects pointed to six

objects from the centre of the chamber with eyes-open (served as based line pointing), eyes-closed after brief rotation (served as orientation pointing), and eyes-closed after extensive rotation (served as disorientated pointing). They found that configuration error increases significantly after disorientation. In another experiment, subjects learned the location of objects and the corners of a rectangular room as shown in Figure 2.1. They found that, disorientation has no effect on pointing to the corners. From these findings, they argued that humans compute a dynamic egocentric representation of objects locations.

Based upon the above findings and a brief review of research on navigating animals, from ants to primates, Wang and Spelke (2000) concluded that even humans do not compute an allocentric map directly from perception. However, they do have allocentric maps in their head, but these are the results of their symbolic reasoning about their environment rather than being computed directly from their perception of the environment.

2.1.2 Buckley et al.'s (2019) model

In Buckley and his colleagues's work (2019), they proposed that both local and global representations are encoded for orientation tasks. To support their hypothesis, two experiments were conducted to test whether humans' ability to do so using an inside-outside paradigm.

Figure 2.2 (a) illustrates the first experiment arena. In this experiment, participants were trained to find a hidden goal that was located at the left corner inside a virtual kite-shaped arena. In such an arena, the participants will find the goal at a corner whereby the left wall is a long wall and the right wall is a short wall. Following training, participants were placed outside and given a single test trial on the outside of the same environment, and were asked to locate where the target signal was. In the second

experiment, Figure 2.2 (b), participants were asked to do the same task as in experiment 1. However, the key difference from the first experiment was that there were the same corners from outside as target corner trained inside the arena (e.g. the top-left corner in (b) was a right angle corner with a long left arm and short right arm, the corner located close to the environment had the same configuration observing from the outside perspective), but the spatial locations of them were different. In other word, participants could not simply use local representation of the space to locate where the target was. Therefore, Buckley et al. expected that participants would search these near corners, at least for some time but no more than the time of searching the correct corner. Namely, they would associate local and global representations to achieve the reorienting tasks.

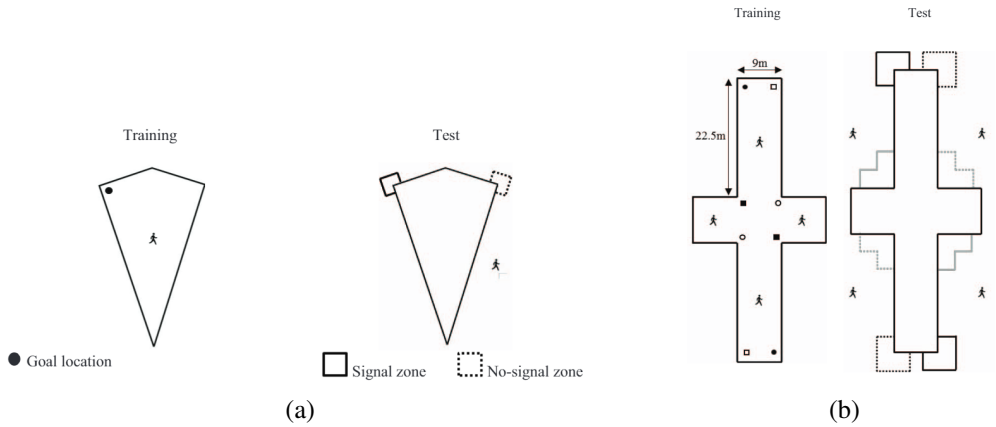


Figure 2.2: Buckley et al.'s experiments using two different types of arenas, kite-shaped and cross-shape. In (a), filled circles indicate the target location. In (b), filled and empty circles and squares indicate the target positions for four different groups. In both (a) and (b), squares indicates the search zone during testing. All participants were trained inside arenas and tested from outside of it (Reproduced from Figures 1 and 3, Buckley et al., 2019)

Result of the first experiments implied that participants displayed no preference for the signal over the no-signal zones during the test. This result was consistent with the idea that a global map of the shape of environment was encoded during training and reoriented based upon it. However, in the second experiment, participants did search

the for the wrong corners that had the same configuration for some times.

2.1.3 Hegarty et al.'s (2006) model

Using a route integration paradigm, Hegarty et al. (2006) attempted to test whether participants learn the names and locations of distinctive places along two separated routes. They then hypothesized that all forms of spatial layout learning will be produced by visual encoding abilities and attempted to show that a map-like representation is linked to the travelled routes.

Figure 2.3 shows the experiment conducted. Each participant was led to walk through two different floors of a building and pointed out the landmarks along the path. Participants learned the layout of a route through two floors of a building, along with the locations of 8 landmarks on that route. After traversing the route once, the participant was taken outside the building and back to the route for testing. In testing, participants were asked to estimate the distance and direction to two other landmarks that were not visible from its location, which including making estimation between landmarks on different floors. After completing the estimation, participants were asked to draw a sketch map of the route including locations of landmarks and ignored the vertical dimension

In the result of this experiment, they found that participants were not only internally computed for each individual floor, but also joined the two maps together as a whole global map. In other words, they internally computed a single global map to contain all these memorised detail with spatial relationship.

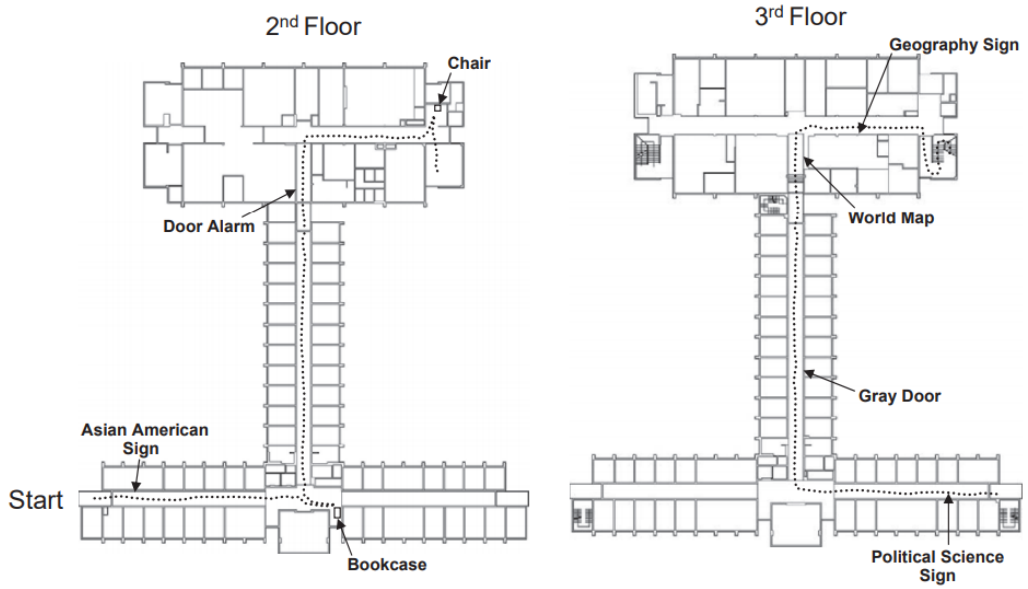


Figure 2.3: Maps of the environment learned from direct navigation, showing the locations of the landmarks. In these two maps, dotted lines indicate the path travelled by participants. All participants followed the same path that learnt the environment. Along the path, all 8 landmarks were pointed, which were labelled on the maps. (Reproduced from Figure 3, Hegarty et al., 2006)

2.2 Recent computational models

2.2.1 Zeng and Si's (2019) model

Zeng and Si (2019) implemented a novel model of cognitive mapping, referred to as a compact cognitive mapping system, which was inspired by neighbourhood cells and neighbourhood fields in neurology. Such a compact system utilised information-based criterion to determine whether rejecting redundant vertices (the pose of robot) or adding informative measurements to the map in order to control growth of the size of the global map and save the memory and computational capability.

Figure 2.4 illustrates how such a compact mapping system adding spare vertices through neighbourhood fields. Figure 2.4A and B firstly shows that, in the system, if the neighbourhood field does not meet the threshold to create a new vertex, vertex from e_{i+2} to $e_{i+n-1,i+n}$ are removed, and edges $e_{i+1,i+2}$ and$e_{i+1,i+3}$ and other intermediate edges

are merged into $e_{i+1,i+n}$. Namely, many consecutive vertices do not meet threshold requirements of creating as new vertices, it would remove these consecutive vertices and merge all these edges into one.

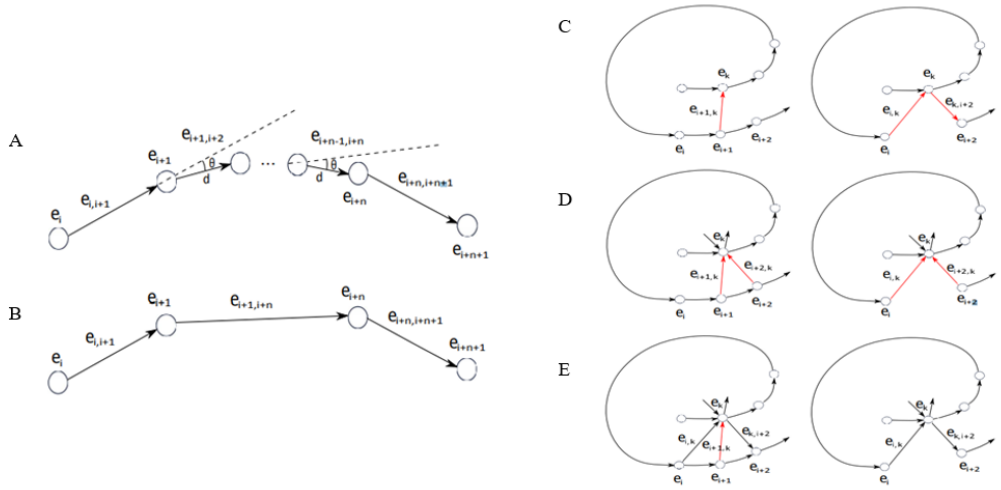


Figure 2.4: A and B show the adding of sparse sequential vertices and edges to both a. standard cognitive map and a compact cognitive map respectively. C – E show scene integration. When one revisits a familiar space, it removes the redundant vertices and edges. Red arrows indicate the loop closure edges. (Reproduced from Figures 2 and 4, Zeng & Si, 2019)

Furthermore, for dealing with loops in the map and achieving long-term mapping, scene integration plays an important role for reducing redundant information. When the robot revisits familiar image views, redundant vertices are not added, and the same scenes are integrated with the earlier scene stored in the map. In Figure 2.4, C, D and E show the scene integration in three different situations. For instance, if there is only one vertex (e_{i+1}) connecting another existing vertex (e_k) in the loop, it will be removed, and merge 3 edges (e_{i+1}, $e_{i+1,i+2}$ and $e_{i+1,k}$) into 2 edges ($e_{i,k}$, $e_{k,i+2}$). Moreover, if multiple vertices connect to the same vertex, as shown in Figure 2.4E, the redundant vertices and relevant edges are simply removed without merging.

In Figure 2.5, it illustrates the results of Zeng and Si's model applied on a rat-like

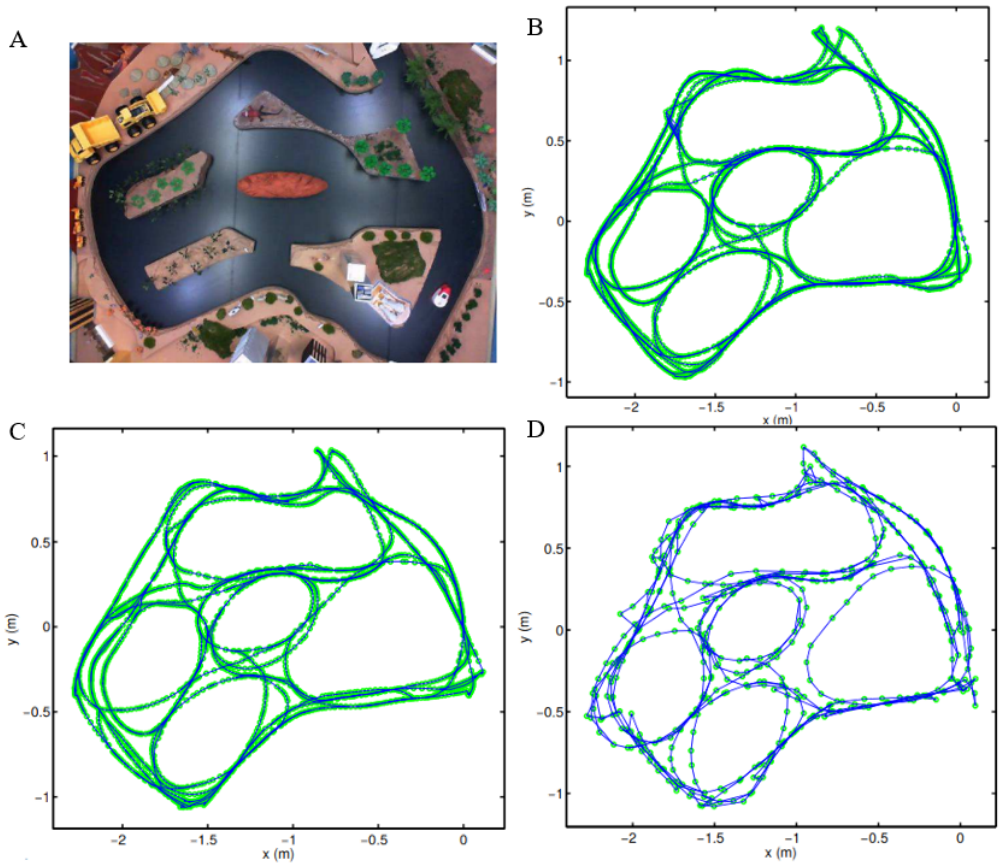


Figure 2.5: The compact cognitive mapping system was applied on rat-like maze environment. Green dots are vertices of cognitive map and blue thin lines are edges between connected vertices. A is the overhead view of the test environment. B shows the standard cognitive maps; C show the compact cognitive map with less updated vertices and edges with only scene integration. D shows the compact cognitive map updates much less vertices and edges and scene integration with full process. (Reproduced from Figure 6, Zeng and Si, 2019)

maze environment. The compact cognitive mapping system updated 497 vertices and 602 edges by using full steps of adding sparse vertices, edges and scene integration with good overall shape, which was much less than the other 2 systems (shown in Figure 2.5B and C). The results showed that Zeng and Si's approach was less theoretical, but pragmatic and efficient for cognitive mapping. They achieved sparsification of the cognitive map by introducing the concept of neighbourhood fields. The movement information with topographical orientation is applied to sparsify sequential vertices and

edges in the cognitive map.

2.2.2 Gupta et al.'s (2017) model

Gupta and his colleagues proposed a neural architecture for visual navigation in novel environments. Their approach was reminiscent of classical work in navigation that also involves building maps and then planning paths in these maps to reach desired target locations, referred to as Cognitive Mapper and Planner (CMP). Gupta et al. believed that this model enjoys the power of being able to learn behaviours from experiences instead of making the whole environment explicit first (e.g computing a global metric map first, then planning the path in it). Instead, the map computed using their model is dependent on how much the agent experiences the environment.

As the name of the system implies, it combined mapping and planning together. The mapper updates its multi-scale belief about the world based on the current observation then passes this belief to the planner as input. Finally, the planner outputs the actions to take.

The Mapper: it fuses information from input views (first-person images) as observed by one over time to produce a metric egocentric multi-scale belief about the world in a top-down view. At each time step, it maintains a cumulative estimate of the free space in the coordinate frame of the robot. A multi-channel 2D feature map (f_t shown in Figure 2.6) that metrically represents space in the top-down view of the world. In Figure 2.6, it also illustrates that the feature map f_t is estimated from the current image I_t, cumulative estimate from the previous time step f_{t-1} and egomotion between the last and this step e_t using an updating rule U.

The Planner: the planner is based on value iteration networks, which can be thought of as the value of each state iteratively recalculated at each iteration by taking the maximum over the values of its neighbours plus the reward of the transition to those

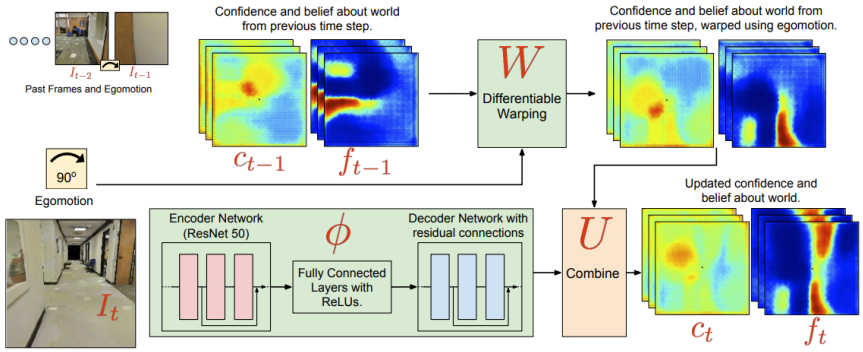


Figure 2.6: Architecture of the mapper. The mapper module processes first person image from the robot and integrates the observations into a latent memory, which corresponds to an egocentric map of the top-view of the environment. The mapping operation is not supervised explicitly – the mapper is free to write into memory whatever information is most useful for the further usage like planning. In addition to filling in obstacles, the mapper also stores confidence values in the map, which allows it to make probabilistic predictions about unobserved parts of the map by exploiting learned patterns. (Reproduced from Figure 2, Gupta et al., 2017)

neighbour states. Namely, the planner takes the egocentric multi-scale belief of the world output from mapper and uses values iteration expressed as convolutions to output a policy, which is the action to take at this step; it then passes the action to the agent.

Figure 2.7 shows four examples of computed maps and travelled trajectories of navigation tasks. A and B show two success cases. In these two experiments, agent was able to traverse large distances across multiple rooms to get to the target location, go around obstacles and quickly resolve that it needs to head to the next room. C and D show two failure cases. The problems were that when agent navigated around tight spaces (corner or pass through a door), it would lead to another shorter path or get stuck.

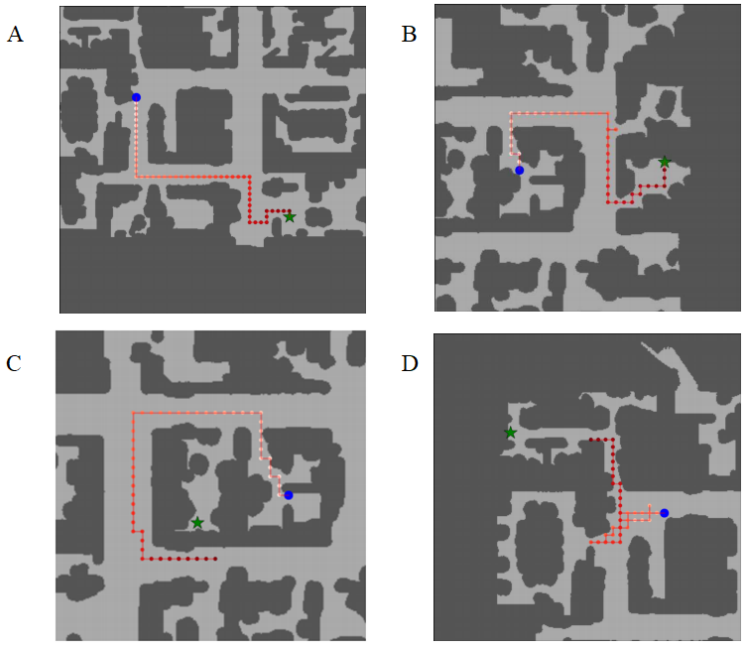


Figure 2.7: Four examples of computed maps using Gupta et al.'s system. In the map dark grey indicates occupied place and shadow grey indicates free space. In the maps, robot starts from the place of blue circle and targets to the position of green star. Red dotted lines indicate the path. (Reproduced from Figure 5, Gupta et al., 2017)

2.2.3 Brom et al.'s (2012) model

Brom et al. (2012) proposed a computational model of cognitive mapping, paying attention to the use of an egocentric (transient) and an allocentric (enduring) frame of reference. They assumed that the allocentric map contains only object-to-objects relations on it and these objects keep updating while one is moving. On the other hand, the egocentric map contains only self-to-objects relations. Based on such an assumption, they developed their model. In Figure 2.8, it illustrates how their approach works.

In Figure 2.8, it shows that in the system, there are three phases for computing maps: 1) sensory systems as perceptual phase; 2) working memory for computing transient map; 3) long-term memory for computing enduring maps. In egocentric map, both transient and enduring, they always remain the fresh objects and some earlier objects and the spatial relationship between these objects and the agent itself. On the contrary,

allocentric map only remains the spatial relationship between each object in map. Figure 2.9 illustrates how the process exactly works in both egocentric and allocentric frames.

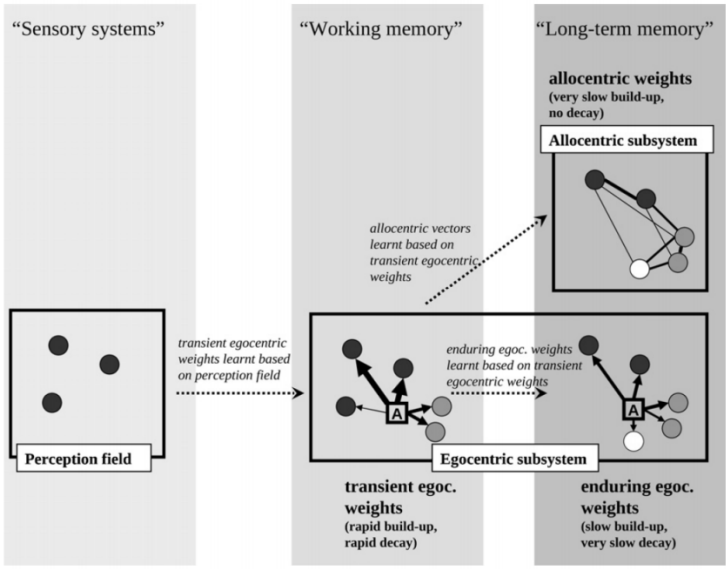


Figure 2.8: In Brom et al.'s DP-system, there are two separate subsystems, egocentric and allocentric subsystems, composing the whole cognitive mapping system. This figure illustrates how the process of their model works, from perception to spatial knowledge. Initial step is the sensory system; the agent perceives objects at this step. The next step is working memory; agent computes transient egocentric weights, which determines which objects are needed to update onto allocentric representation. Egocentric representation includes the location of the agent with respecting to objects, but allocentric only contains vectors between objects. (Reproduced from Figure 6, Brom et al., 2012)

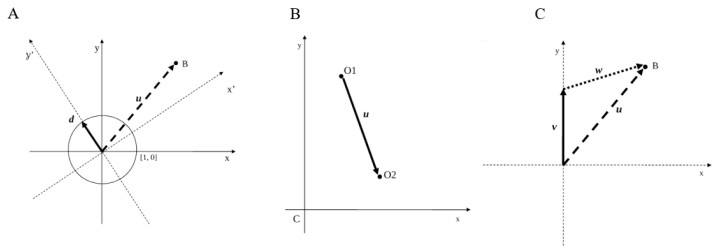
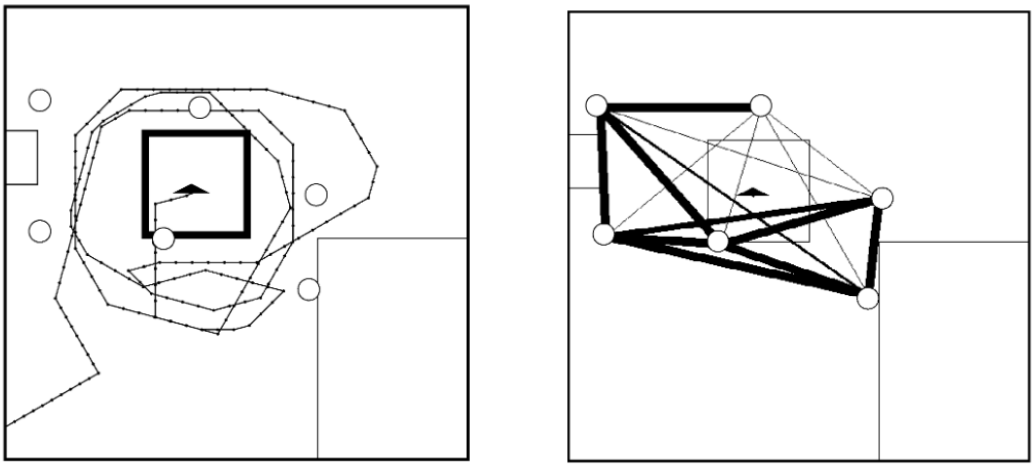


Figure 2.9: A shows the use of an egocentric frame. In this frame, axis x and y indicate the direction of the walls; axis *x*' and *y*' corresponds one's heading direction, vector *d*. It also computes the vector *u* with respect to the object B. B shows the use of an allocentric frame. It only computes the vector between objects such as vector *u*, which indicates the spatial relation between two objects, O_1 and O_2. C shows the update process as moving along the vector *v*. Finally, vector *u* will be updated to vector *w* for depicting the egocentric relation between oneself and object B. (Reproduced from Figures 7, 8 and 9, Brom et al., 2012)



(a) (b)

Figure 2.10: An example of a computed allocentric map using Brom et al.'s model. (a) shows the travelled trajectory (lines) in the test environment with 6 objects (empty circles). (b) shows the computed allocentric map with allocentric vectors (black solid lines). (Reproduced from Figures 16 and 20, Brom et al., 2012)

In Figure 2.10, (b) shows the result of computed allocentric map using Brom et al.'s model. It only contains the spatial relationship between objects with different weights. Thick lines indicate the allocentric vectors with high weight. Thin lines indicate those allocentric vectors with low weight.

Importantly, in their work, they also discussed that errors causing disorientation exists in nature, and applied gaussian errors as configuration errors at perception level in their system instead of building an error free system. Although they successfully implemented the system with cognitive features and applying both sub-systems, transient and enduring, makes this approach different from other robotic approaches, their model still relies on the accurate coordinate information to depict its surrounding environment and to point out the positions of remembered objects.

2.3 Geometry of local environment

Cheng (1986) first reported an experiment whereby rats ignored visual landmarks and orient using the geometry of the room. Subsequently, many derivative experiments were conducted that show that various species (Hermer & Spelke, 1994; Vallortigara, Zanforlin & Pasti, 1990; Sovrano, Bisazza & Vallortigara. 2007; Kelly, Spetch & Heth, 1998; Learmonth et al, 2001; Lee et al., 2013) could also orient using local geometry. In this section, I discuss some of these works to provide some background of this important research. Section 2.2.1 reviews some empirical evidence that supports the idea. Section 2.2.2 reviews some studies that discuss what geometry means. Section 2.2.3 reviews some studies investigating how geometry is encoded. Section 2.2.4 reviews some studies investigating the use of geometry in more complex environments.

2.3.1 Empirical evidence on the use of geometry to re-orient

In Cheng's seminar paper, hungry rats view the location of food in a rectangular chamber with multiple visual landmarks (Figure 2.11). The rectangular box (Figure 2.11A) has both distinguishable featural cues and geometric cues from the enclosed boundary. They were then disoriented and how they subsequently search for food was observed. In the test arena with only geometric boundary (Figure 2.11B), rat searched

the rotationally equivalent position more often than the target position. In the test arena without geometric boundary (Figure 2.11C), it was discovered that rats ignore the distinctive landmarks and still search for the food both at the correct location and at the symmetrical location. Based on this study, Cheng then proposed that rats might have a geometry module for orienting itself.

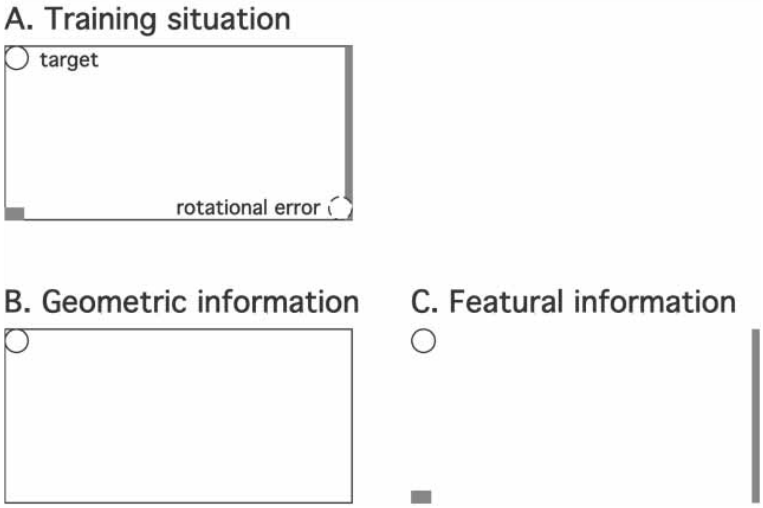


Figure 2.11: Example of test of reorientation system. (A) Animals have the task of relocating the unmarked target. The arena has an overall shape with geometric shape as well as a wall with different colour (the thicker wall) a distinguishable object (lower left corner). Animals were trained in this arena. (B) animals were tested in this arena (rotated 180°) only with geometric boundary. (C) the test arena (rotated 180°) contains all distinguishable featural cues but no enclosure boundary. (Reproduced from Figure 1, Cheng, 2005)

Hermer and Spelke (1994) conducted experiments to test if young children would be able to derive geometric information from an enclosed space and use it for navigation. Initially, children were asked to find hidden toys in both all white room and a room with a blue wall as featural cue. In white room, children searched correct corner and geometrically equivalent corner equally often. But, in the room with a blue wall, although children searched geometrically appropriate corners, they did not search the correct corner more than its rotational equivalent and did not search the two appropriately coloured corners more than the other corners. This result suggests that children might

have encoded the spatial relationship using geometric cues prior to featural cues.

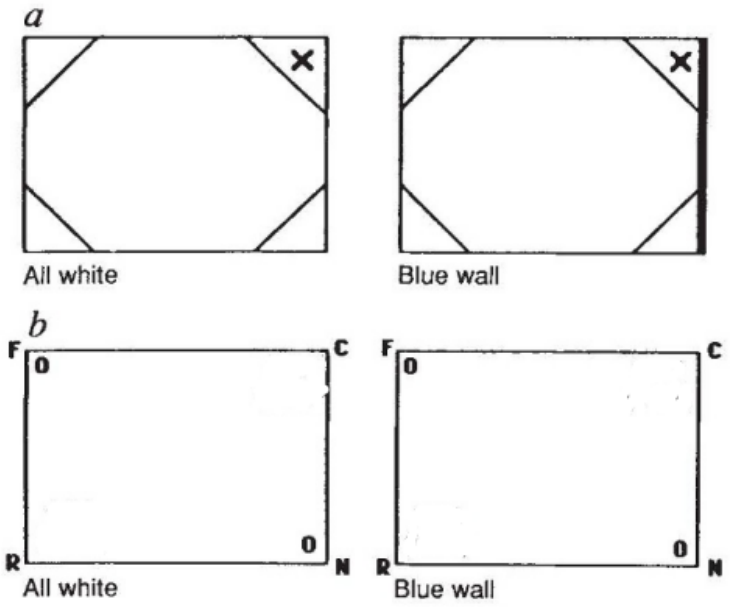


Figure 2.12: Four experiments were conducted to test the ability of reorientation of young children using geometric information. The first environment had 90-cm height walls, which are perceived as boundary hindering behaviours. The second environment was using four columns instead of walls. The third environment had 30-cm height of surfaces, which was not barrier to participants' view and behaviours. The fourth environment was outlined the shape on the floor using tapes. (Reproduced from Figure 1, Lee & Spelke, 2008)

Sovrano et al. (2007) trained fish to reorient to find a corner in a rectangular tank with a distinctive featural cue and then tested fish after displacement of the feature on another adjacent wall. In a large enclosure, the fish chose the two corners with the feature, and also tended to choose the one that maintained the correct arrangement of the feature cue with respect to geometric sense. In contrast, in a small enclosure, fish chose both the two corners with the features and the corner, without any feature, that maintained the correct metric arrangement of the walls with respect to geometric sense. Figure 2.13 shows the experimental arena used in Sovrano et al.'s work. The top left shows the sketch of the geometrical information available in a rectangular arena. The target corner (solid circle) stands in the same geometric relation to the shape of the

environment as its rotational equivalent. Metric information (length of the surfaces) together with geometric sense suffices to distinguish between corner A/C and B/D, but not distinguish between A and C (or between B and D). Top right shows that the featural cue is added on the wall, then each pair of geometrically equivalent corners becomes distinguishable. Bottom shows cues can be perceived at corner A.

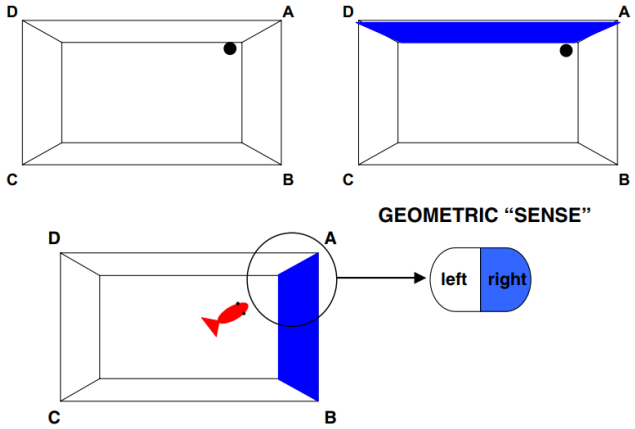


Figure 2.13: Redtail splitfin fish was trained and tested in a rectangle-shape environment with non-geometric features. (Reproduced from Figure 1, Sovrano et al., 2007).

Kelly and Bischof (2005) conducted experiments to test whether human adults use geometric cue to search for hidden target and compared this ability with that of using a featural cues. In Figure 2.14, geometric-only image shows the space does not have any featural cues, and there are four configural black patches, which are near the corners. In images b, c and d, the size of the shape composed of 4 black response patches differs from that in the geometric control image. In image e, the shape composed of response patches was rotated. All participants trained using geometric control image and tested using the other 4 images. All of these images are illustrated from the same point in the room. In the experiments, participants were trained using geometric control image (Figure 2.14a) and testing using other 4 images (Figure 2.14b-e). The result was that the subjects continued to respond to the geometrically correct corners regarding to the geometry of boundary, no matter how the configural geometry was manipulated, which

implied that human adults showed strong reliance on the boundary geometry over the configural geometry.

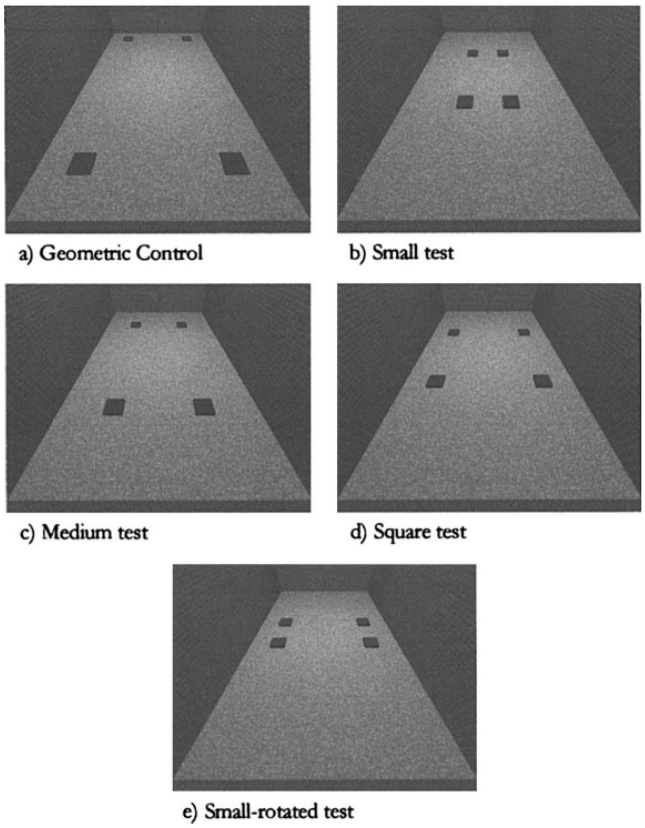


Figure 2.14: Images used in experiments for testing reorienting ability of human adults. (Reproduced from Figure 2, Kelly & Bischof, 2005)

Kelly et al. (2008) employed immersive virtual reality (VR) to test the reorienting capability of human using geometric cues. As shown in Figure 2.15, participants were tested in four types of shapes of the rooms, ∞-fold (circular), 4-fold (square), 2-fold (rectangular) and 1-fold (trapezoidal). They allowed participants to walk continuously in all different rooms and observed the errors made. On each trail, participants attempted to remember the location of a target post while walking to a sequence of other posts. At the end of the walked path, participants attempted to point to the location of the target post from among the 12 possible post locations. Participants in the circle room

become disoriented increasingly with increasing path segments, which is different from those moving in square, rectangular and trapezoidal rooms. Environmental geometry effectively compensates the limitations of path integration.

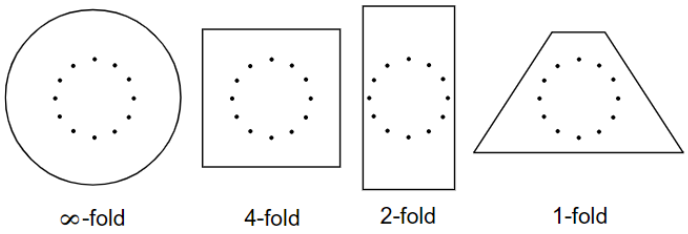


Figure 2.15: Human participants have been tested their orientation ability in an enclosed space with different shape. Filled circles represent the locations of posts used in the tasks (Reproduced from Figure 1, Kelly et al., 2008)

Table 1 provides a summary of some work done in this area (for a more detailed summary, see Cheng et al., 2013). These empirical studies, from animals to human, thus support that many species inherit the ability of using geometric information to navigate.

Table 1

Experiments about reorienting tasks using geometric cues by different species

Species	**Task and arena**	**Researchers**
Rats	- Searching food in typical enclosed rectangular arena	- Cheng,1986 - Margules & Gallistel, 1988 - Cheng & Newcombe, 2005
Fishes	- Searching food in enclosed rectangular arena	- Sovrano, Bisazza, & Vallortigara,2002 - Sovrano, Bisazza & Vallortigara.2007

Toddlers	- Searching toys in enclosed rectangular arena	- Hermer & Spelke, 1994 - Learmonth, Nadel, & Newcombe, 2002 - Gouteux, Vauclair, & Thinus-Blanc,2001 - Hermer-Vazquez,Moffet, & Munkholm, 2001
Adult human	- Reorienting in Virtual environment in VR - 2-D images of rectangle arena - Reorienting in typical enclosedrectangular arena	- Kelly et al., 2008 - Kelly & Bischof, 2005 - Hermer-Vazquez et al., 1999 - Gouteux & Spelke, 2001 - Gouteux,Vauclair, & Thinus-Blanc, 2001 - Gouteux, Vauclair,& Thinus-Blanc, 2001
Chicks	- Searching food in rectangular arena	- Vallortigara, Zanforlin & Pasti,1990 - Vallortigara, Pagni & Sovrano, 2004 - Vallortigara et al., 2004
Pigeons	- Searching food in enclosed rectangular arena	- Kelly, Spetch, Heth, 1998

Rhesus monkeys	- Searching food in enclosed rectangular arena	- Gouteux, Thinus-Blanc, Vauclair, 2001
Adult human	- Reorienting in enclosed trapezoid environment	- Kelly et al., 2008

2.3.2 On the nature of geometric information

When it comes to geometry, the general definition of geometry is mathematically explained as the relationship of points and lines, regarding to the Euclidean coordinate system. However, in nature, geometry can be used to characterise the location of objects such as geometrically arranged isolated objects. How do geometric measurements of length, angle, distance and direction guide navigation? It has long been theorized that these measurements apply primarily to the internal encoding of proprioceptive cues to track one's movement. (Descartes, 1637/2011). Moreover, in nature, geometry also could be an abstracted representation of the boundary of a space with simple orientation such as indoor environment. In this section, I present some empirical research that investigated what geometric information is used for re-orientation.

Lee and Spelke (2008) found that only elevation of boundary will be used as geometric information for reorientation. In their experiments, 4-year-old children were asked to reorient to the correct location in different types of geometric environments with different heights of walls. However, participants could only use geometric information to reorient to the correct location in the first and third environments (as shown in Figure 2.16), where the elevation was added as walls. Children could derive geometric information from elevation boundary, but failed when the same geometric information draw on the floor. Elevation defines a space for one because it forms a barrier to one's behaviour and perception; one thus has a sense of this local space and its shape of the boundary. Therefore, in nature, the geometry would be defined by a kind of

boundary that barriers one's movement and perception, which could be composed of walls, isolated objects, or even trees and rivers.

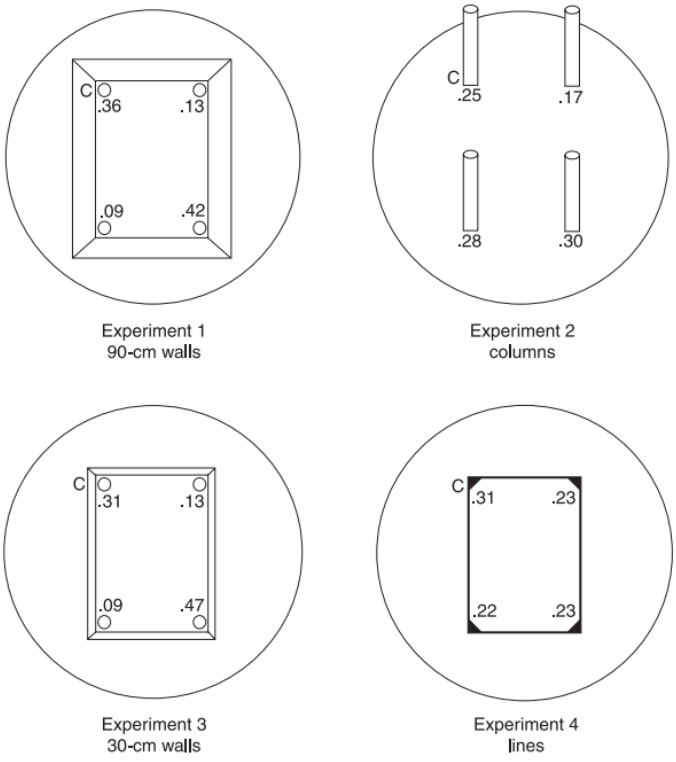
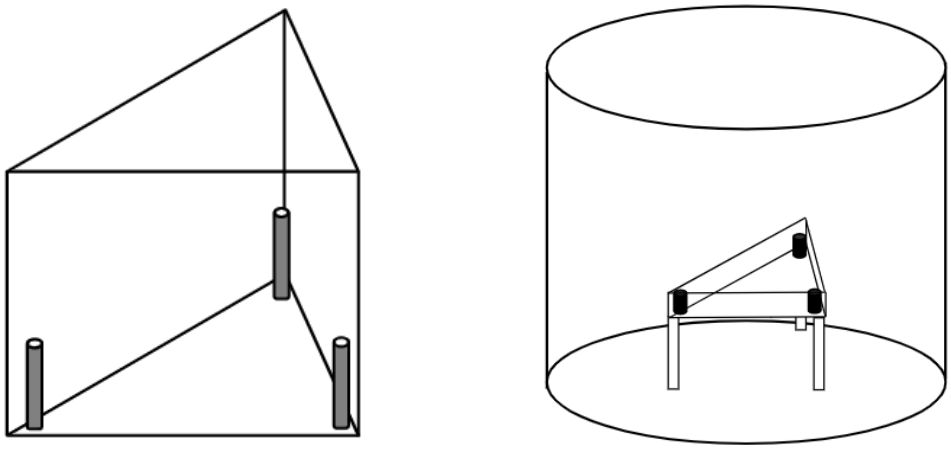


Figure 2.16: Four experiments were conducted to test the ability of reorientation of young children using geometric information. The first environment had 90-cm height walls, which are perceived as boundary hindering behaviours. The second environment was using four columns instead of walls. The third environment had 30-cm height of surfaces, which was not barrier to participants' view and behaviours. The fourth environment was outlined the shape on the floor using tapes. (Reproduced from Figure 1, Lee & Spelke, 2008)

Huttenlocher and Vasilyeva (2003) found that young children were able to find hidden toys from both inside and outside of a triangle space. Children were trained inside a triangular space (Figure 2.17a) and they were able to achieve the required task from outside of the triangular space (Figure 2.17b). Their experiments thus show that children were able to solve geometry tasks when views change radically, and they do not need to be bounded by the view. This suggests that the use of geometry is different from the view-matching approach because geometric cues need to be derived from the

surrounding space rather than from matching views. Technically, view-based matching is to match pixel by pixel between memory and views. Geometric information is thus enduring and is view-independent (Buckely et al., 2019a, 2019b).



(a) (b)

Figure 2.17: Left environment was to test reorientation ability of toddlers inside it in first experiment. Right environment was triangular box in a round enclosure, which was to test toddlers outside of the triangular box. (Reproduced from Figures 1 and 2, Huttenlocher & Vasilyeva, 2003)

Wystrach and Beugenon's work (2009) showed that both the untrained rats and trained rats were only searching current corners and geometric equivalent corners rather than exploring the whole arena. They showed that instead of exploring the whole environment, rats directly derived the geometric information from partial information of the space for spatial tasks. In other word, it is not necessary to perceive the whole space, but partial information would be enough for deriving the helpful geometric cues for reorienting.

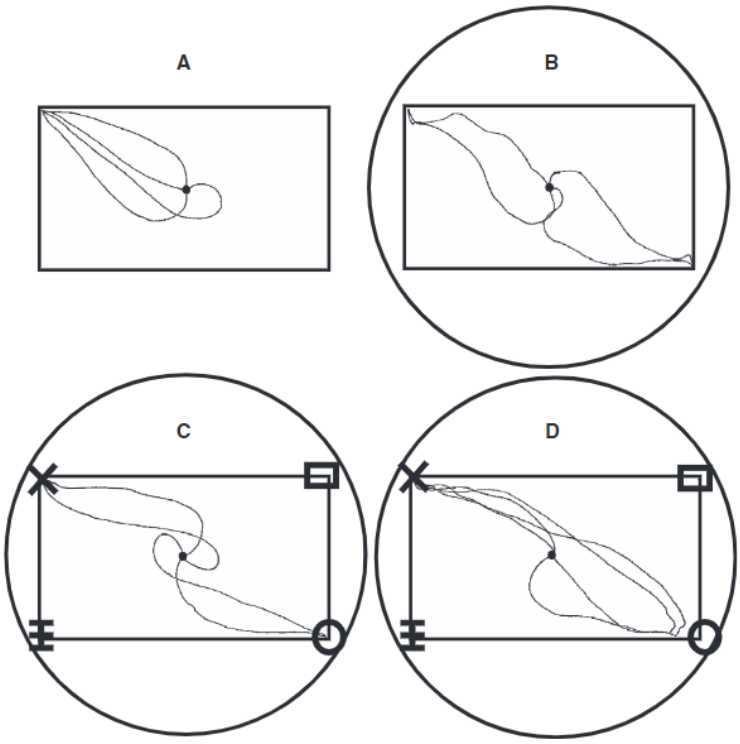


Figure 2.18: The trajectories of subjects in the different experimental environments. (Reproduced from Figure 3, Wystrach & Beugenon, 2009)

2.3.3 Ways of encoding geometry

For spatial behaviours, the relative length of boundaries of environments has been studied to test if this kind of information will influence one's orientation ability. For instance, a rectangle-shaped space contains long and short sides, which is distinguishable. Note that many of these experiments utilized small, simple, regular shaped, enclosed environments. In other words, the boundary of these test environments directly provides one with clear geometric information. In this section, I present some empirical work that studies how geometric information can be computed.

Gibson and Kelly (2007) found that rats could use the geometric arrangement of discrete objects to find hidden water. Figure 2.19 shows the schema of the experimental arena. In their experiments, rats were presented with an array of four objects with unique

features arranged to form a rectangle. The rats were released from the start box. After releasing, each rat had a fixed amount of time to locate water that was hidden inside the object C. Then, in control tests, rats were tested in the arena with geometry condition (geometric cues remained only) and diagonal condition (featural cues remained only) with rotated array in each trail so that studying which cues rate would use. The results showed that rats searched the correct location and the geometrically equivalent location more often than others.

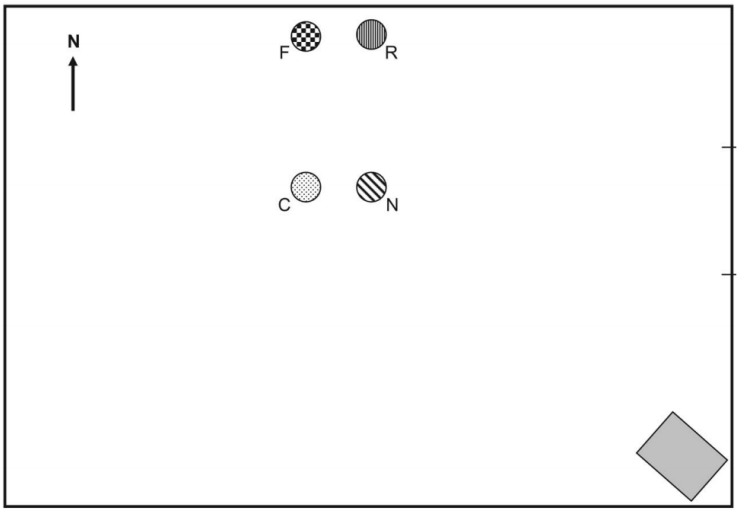


Figure 2.19: Schema of experimental arena. Rats were released from bottom right box. Four objects were placed in rectangle shape. C indicates the correct location; F indicates far location; R indicates geometrically equivalent location and N indicates near location. (Reproduced from Figure 1, Gibson & Kelly, 2007)

Yaski and Eilam (2008) examined how isolated arranged objects impacted rat's exploratory behaviour. They conducted two different test environments, a round and a square (Figure 2.20, top two rows). Figure 2.20 shows isolated objects were placed in the environments 2-5, which composed geometric arrangement such as a square. Bottom two rows in Figure 2.20 illustrate that, during a single session of test, the trajectories of a representative rat were illustrated in the round and square environments.

In the arenas without placing any objects inside (Figure 2.20 array 1), both circular and square, rats mainly explored along the boundary of space. However, if the space

was not empty and objects were placed inside, it clearly influenced rat's behaviour. To be specific, the trajectories (Figure 2.20 array 2-4) around objects became dense, especially when objects placed closer. Namely, it implied that rats spent more time around those objects and used these objects to guide them in the space.

Interestingly, if only 3 rotated objects placed inside the test environments, rats kept spending time around them to navigate. These results reflected that rats could use geometric arrange objects as cues to navigate in the environment because all these objects were not distinguishable objects, it was impossible that rats used them as landmarks to navigate. Instead, geometric arrangement of those isolate objects supported rats to produce such behaviours.

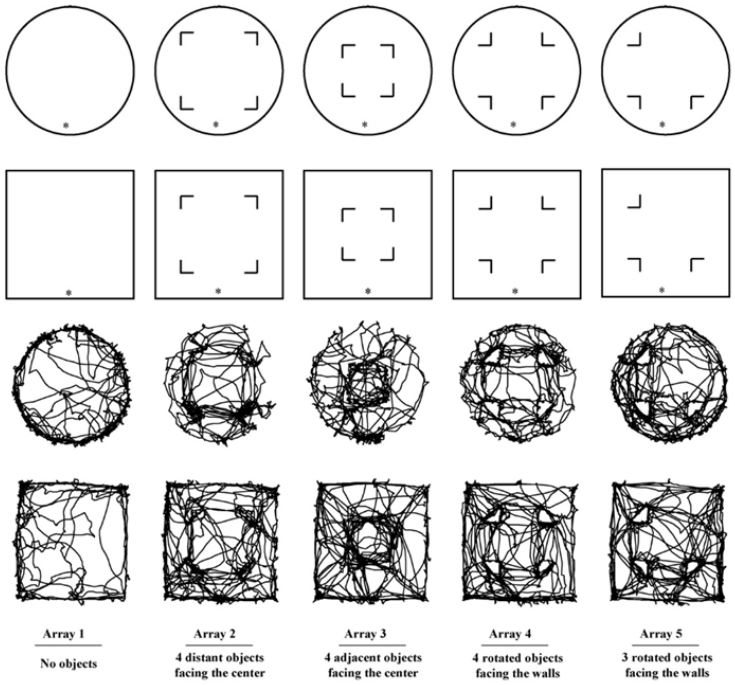


Figure 2.20: Experiments conducted in Yaski and Eilam'work for observing how isolated arranged objects impact rat's behaviour. Top two rows illustrate the test environments and another two illustrate the trajectories produced by rats in arenas. (Reproduced from Figures 1 and 2, Yaski & Eilam, 2008)

Nevertheless, this result may be debated that rats' locomotor behaviour had been impacted because of the placed objects but not the geometric arrangement of them. In

other words, no matter how these objects are placed geometrically or not, it will always impact rat's behaviours. Yaski and her colleagues (2011) findings pinpointed what rats used in earlier experiments was geometric cues but landmarks. In Figure 2.21, they designed two different test environments, grid layout and irregular layout spaces, which were comparing to human's urban environments. Isolated objects were placed regularly in grid layout environment and isolated objects were arranged irregularly in irregular layout environment. Then, it was observed whether these two different arrangements of objects in space impact rat's behaviours. The result, Figure 2.21 b, showed that when rats travelled in grid layout environment, they stayed around isolated objects. However, rats spent more time near the boundary when travelling in irregular layout environment. This implied that rats could encode a list of objects to geometric information for further navigation.

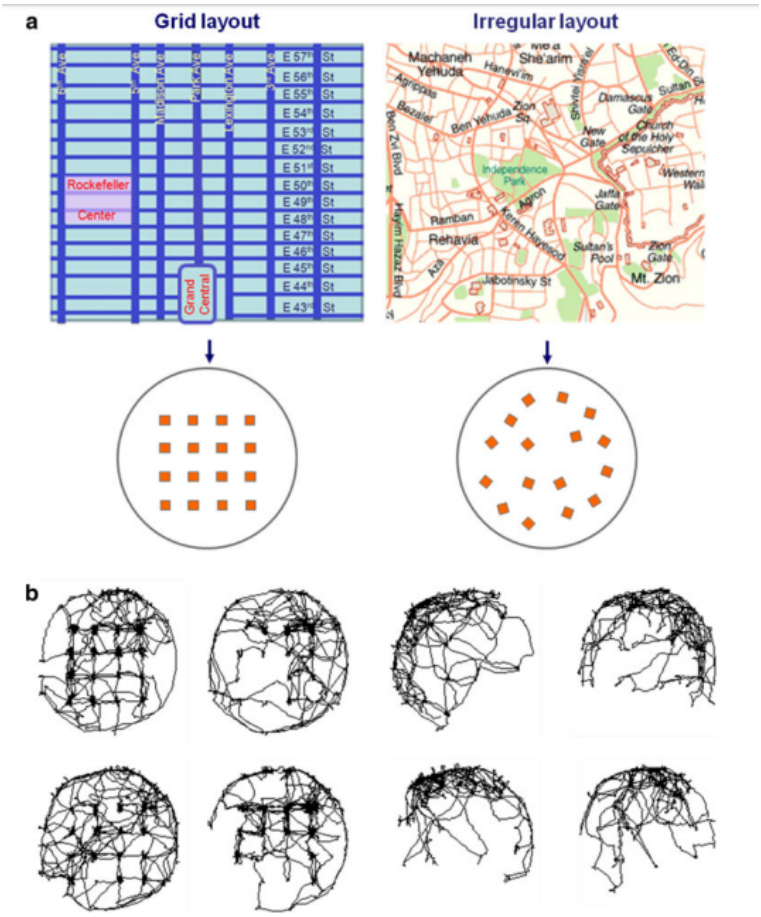


Figure 2.21: Inspired from human's living environment, grid layout and irregular layout environments were designed to test rat's spatial behaviours. a) shows that objects were arranged regular regarding to grid layout space. In an irregular layout space, objects were placed irregularly. b) illustrates 8 trajectories, 4 in grid layout space and 4 in irregular layout space respectively. (Reproduced from Figure 1, Yaski, Juval & David, 2011)

Shusterman, Lee and Spelke (2008) conducted experiments with young children and investigated if they could use the geometrically arranged objects to find the target. To be specific, children learned the arrangement of the three boxes from a 2D map and then were asked to locate the target box behind them in the room. In Figure 2.22, it illustrates 3 different shapes of arrangement of the boxes. In the result, it showed that children could find the target objects in all three different arrangements no matter these objects are egocentrically or allometrically placed in the room. Namely, this result

provided evidence that young children could use geometric information to reorient in space.

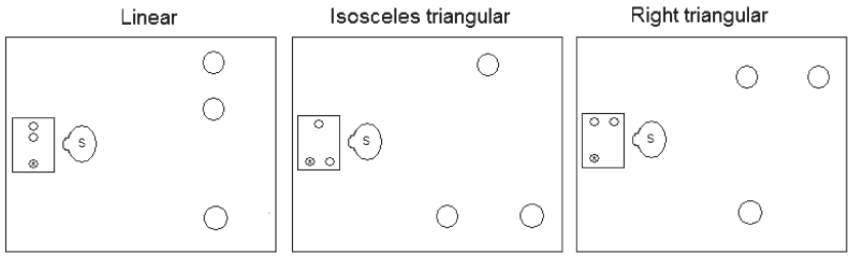


Figure 2.22: Schema of experimental room. The boxes were arranged in three different ways, linear, isosceles triangular and right triangular. S indicates the position where the children started, left rectangle indicates the 2D map with the arrangement of these boxes. Circles on the right side of the room are these boxes. (Reproduced form Figure 1, Shusterman, Lee & Spelke, 2008)

Different from exploring a space, in this one, children learnt the geometric arrangement of objects from overhead viewpoint – the map, which would be easier to observe the overall geometric shape of the arrangement of the boxes. Although, it provided the evidence of that young children were able to derive the geometric shape of the spatial arrangement of isolated objects and find the target.

Hupbach and Nadel (2005) argued geometric information contains distance and direction but not angle or length because in their experiments, human toddlers were searching randomly in the rhombus arena and failed to find hidden objects. In other words, the result of their experiment suggested that the shape of geometry is not the most important issue. However, Lee et al. (2012) replicated the same experiments and made extension based upon their experiments, which only proved that geometric information is existing but also proved that children could derive the geometric description of isolated surfaces and use it to navigate. Lee and his colleagues conducted 8 experiments for testing children's use of geometry for reorientation and obtained different results from that of Hupback and Nadel's experiments.

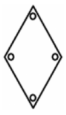
Table 2 shows all 8 different experiments conducted by Lee et al (2012). The first and

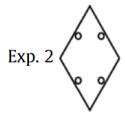
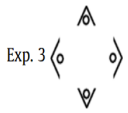
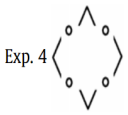
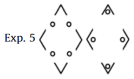
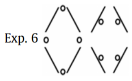
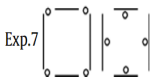
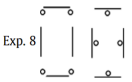
second experiments were replicate of Hupbach and Nadel's work. Other experiments were extended based on the replication. From experiments 3 to 8, all those experiments were designed using isolated boundaries to test if the children could still use geometric process to finish the tasks. Interestingly, in rhombic environment, if the overall shape of the environment is manipulated and slightly irregular (experiments 3 and 4), children failed half of the tests using neither distinguishable landmarks nor corners, which implied that deriving the geometric cue from irregular structural environment is harder than doing it from regular structural environment. Furthermore, in experiments 5 and 6, it proved that using geometric shape of the boundary is prior to using distinguishable corners. In the last 2 experiments, children failed to reorient in a squire environment but succeeded to do it in rectangle space and both square and rectangle space were using isolated surfaces.

This series of experiments suggested that children prefer to derive geometric cue from the boundary to reorient rather than to use distinguishable information such as the corners of the space and landmarks.

Table 2

Eight experiments for testing whether children navigate using geometric information

Experiments	**Proportion of geometric searches**	**Success?**
Exp. 1 	0.73	Yes

	0.70	Yes
	0.56	No
	0.54	No
	0.45	No
	0.69	Yes
	0.45	No
	0.69	Yes

Note. Reprinted from "Navigation as a source of geometric knowledge: Young children's use of length, angle, distance, and direction in a reorientation task" by S.A. Lee,

V.A. Sovrano and E.S. Spelke 2012, *Journal of Cognition*, 123(1), p.148. Copyright 2011 by Elsevier B.V.

From the results above, geometric arrangement of isolated objects or surfaces can provide one with good orientation as well even though such a geometric cue is not so obvious as geometric closured boundary. This would require a more complex process to derive geometric cue to do so.

2.3.4 Geometry for complex environment

Exploring in a complex environment is completely different from doing the same in a simple regular shaped environment because the former does not provide a straight and clear geometric boundary for one to use for navigation. Lee and Spelke (2010) argued that geometric process fails to convey orientation in large-scale environments because they argued that the core geometric navigation systems was more limited. Specifically, in their work, they pointed out that the animals encode the shapes of the extended surfaces that form the border of the traversable layout, but they fail to encode the shape of surface markings or landmark objects. Moreover, young children were able to reorient by using the geometric shape of a rectangular room but failed to do so by using columns or black lines drawn on floor. All these failures were because the core geometric system was limited to capture the sense of geometry.

Sutton (2009) argued that the representation of geometry would be an abstract description, especially in natural environment. In her work, she firstly argued that the laboratory studies of geometry might not be valid examples because the environmental cues presented both in nature and office-like environment are quite different. Then, she pointed that forming the macroscopic shape of a natural environment such as using the position of river and trees can help animals correct their orientation. Namely,

geometry process still plays an essential role when exploring in large-scale and complex environment in nature. However, it might require a more complicate process for deriving such a kind of information.

Dafna, Yaski and David (2011) argued whether the spatial behaviour would be affected by global or local geometry and conducted relevant experiments using rats. Rats in the first group were exposed repeatedly in the same square environment and rats in the second group were exposed to the space with geometric changes. In Figure 2.23, it illustrates the conducted experiments and recorded locomotion of each rat in different test arena. From the group 1, the paths indicate that rats explored the square space freely, which implies that all 5 rats could encode the geometric shape – square for navigation otherwise they might explore the space following walls. From group 2, the results show that once the whole environment becomes more complex, it would be hard to encode the whole space as a simple square for further usage. Therefore, in the second group, rats explored more around the changed corners (paths appeared around the changes more than the rest of the space). It implied that rats might encode these local geometric cues for navigation if the whole environment was complex for deriving the overall geometric shape.

Earlier, we have reviewed that Yaski et al. (2011) compared the environment with regular arrangement of objects with the environment with irregular arrangement of objects. If Yaski et al. compared that test arena to a complex environment, that result may imply that rats could use local cues deriving from irregularly arranged objects to navigate (Figure 2.21b), which was why those trajectories were displayed.

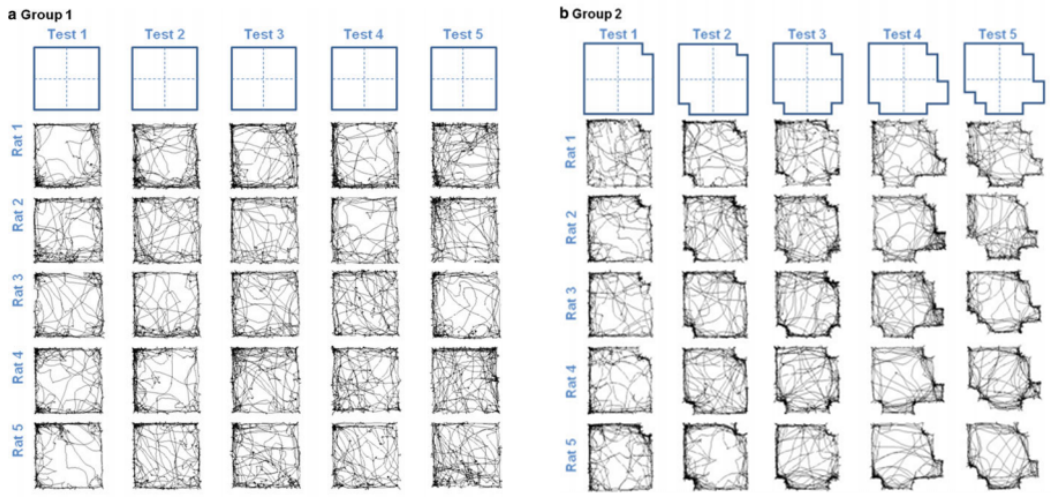


Figure 2.23: Spatial behaviour in arenas with two different groups of rats. Group 1 were trained and tested in the square arena and group 2 was in a more complex environment with gradually changed local geometry. For each group, 5 different rats were tested in from test 1 to 5. Lines in the figures indicated the path of locomotion of rats. (Reproduced from Figure 1, Dafna, Yaski & David, 2011)

2.4 Conclusion

In this chapter, some recent work on cognitive mapping, both empirical and computational, have been reviewed to show the diverse research and interests in cognitive mapping. However, it is worth nothing that few, if any, computational models have attempted to explain how cognitive mapping works. They focus instead on creating a working model of mapping based on some findings in empirical research. This is the key difference between these models and Y&H's model. One empirical research that forms the key part to extending Y&H's model is the idea of computing geometry of local environments for re-orienting oneself. Empirical data, as always, remains inconclusive despite raising many interesting questions. In chapter 4, using Albot1, I investigate how Y&H's model can be extended incorporating the idea of geometry to overcome the key problem identified for this research.

Chapter 3

How Albot1 uses its maps to return home

In this chapter, nine experiments were conducted to gain insights into Albot1's behavior as it attempts to find its way home. The key questions to be addressed here are: (1) Can Albot1 return home using its cognitive map and if so, what does it tell us about cognitive mapping, and (2) can it return home using short-cuts?

These nine experiments are as follows. Section 3.1 presents two experiments that investigate how Albot1 could re-trace home using its route map. Recall that a route map is an ever-growing list of overlapping local maps. Using it to return home, Albot1 is empowered to generate a trace map – a global map of all the points where each local map is computed. However, Albot1 failed to return home due to sensor errors and the nature of the route map. The approach is thus of limited use, only for species exploring a short distance outside its home.

Albot1 is then empowered with an improved algorithm that uses a simplified trace map, thereby allowing it to re-trace home following key points in the route. Section 3.2 presents two experiments using routes without loops and two experiments using routes with loops. Albot1 could return home unless the trace map itself is distorted. It was

observed that the use of a simplified trace map allows one to detect short-cuts and this idea was tested in three experiments in section 3.3. Again, Albot1 was able to return home using short-cuts but only when the trace map itself is not distorted. Section 3.4 concludes with a summary of the key results obtained.

3.1 Albot1 re-tracing home using its route map

As in Y&H, a route map is represented as a list of tuples $(V_i, R^i_{x,y})$, where V_i is the view remembered and $R^i_{x,y}$ is the coordinates of the robot's viewing position in the previous local map, V_{i1}. From this, one could generate a global map of all the points where a local map is created in the route map. This would be the list consisting of all $R^i_{x,y}$. Such a global map is referred to as a trace map and represented as a list $\{R^m_{x,y}, R^{m-1}_{x,y}, \ldots . R^1_{x,y}\}$.

To return home, Albot1 creates a trace map from its route map, turns around 180 degrees, moves towards $R^{m-1}_{x,y}$, turns towards $R^{m-2}_{x,y}$, moves towards it and so on until it reaches $R^1_{x,y}$. The pseudo-code for this algorithm is as shown below:

Algorithm 1 : Simple retrace

Input: trace-map = {$R^m_{x,y}, R^{m-1}_{x,y}, \ldots, R^1_{x,y}$}; R_O – Albot1's current orientation
Output:Albot1 executes its moves.

1: $R_O \leftarrow$ turn(180^o) ▷ Albot1 turns around
2: **Until** trace-map has only 1 element **do**;
3: $R_O \leftarrow$ Turn-towards(R_O, trace-map)
4: Move-to-if-possible(trace-map) ▷ take the co-ordinates of the 1st two points
5: Pop (trace-map) ▷ get the next robot position
6: **end until**

Note that turn-towards re-orients Albot1 towards the next point in the trace map on its way home and move-to-if-possible checks that there is no obstruction between the two points of travel. If there is, Albot1 will simply abandon execution.

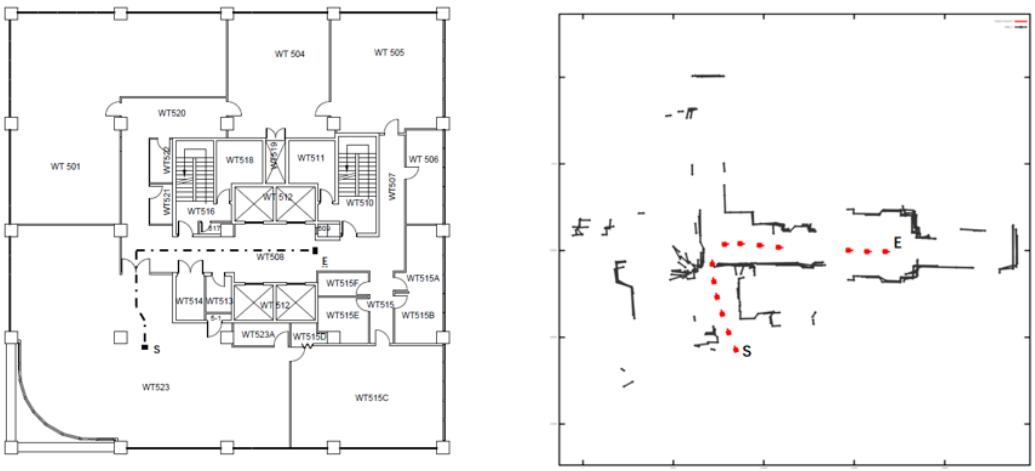
Two experiments are conducted using Albot1 empowered with the above algorithm.

3.1.1 Experiment 1

Figure 3.1a shows the route of the first experiment through some offices in a building. The route consists of 24 views and 13 of which were used to create its route map. The distance traversed is approximately 14.5 meters. Figure 3.1b shows the trace map generated (13 red dots). The trace map thus consists of a series of points that traced the route experienced. To make the display more meaningful, these points will be displayed either together with each local map superimposed (as in Figure 3.1b) or in an egocentric map.

Table 3.1: Experiment 1 parameters

Information	**Parameters**
1. Distance	14.5m
2. Views Collected	24.
3. Views in Route map	13.

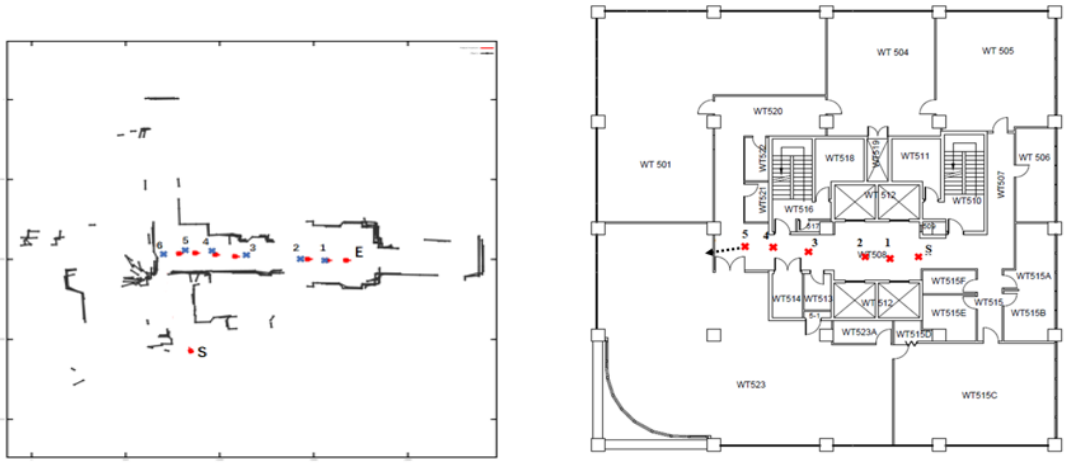


(a) (b)

Figure 3.1: Experiment 1: (a) The environment and the route, $S \rightarrow E$. (b) Albot1's trace map (12 red dots) displayed with each local map superimposed.

The result of the first experiment is as shown in Figure 3.2. Albot1 failed to return

home; after re-tracing the first 6 steps, it stopped because it is going to crash through a wall.



(a) (b)

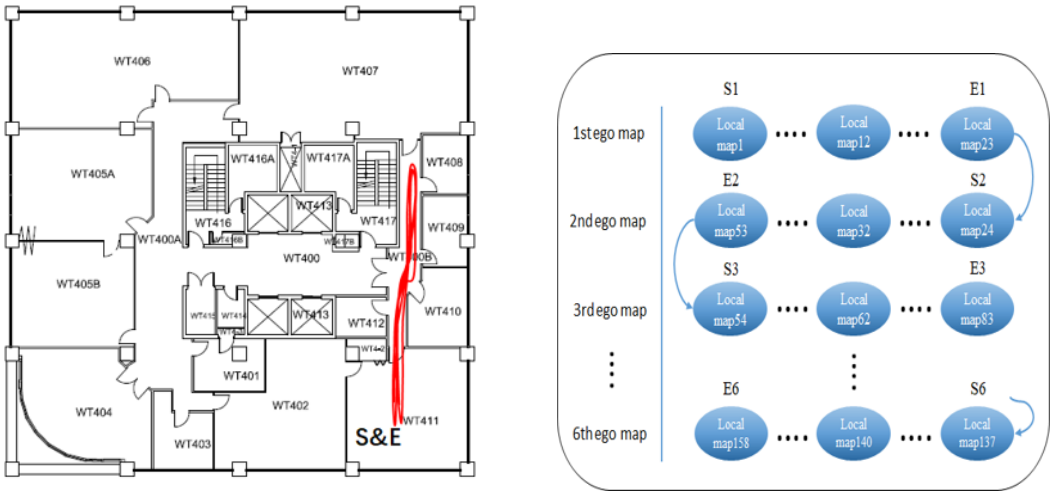
Figure 3.2: Result of Experiment 1: Albot1 failed to return home. The red dots show Albot1's re-tracing home.

3.1.2 Experiment 2

Figure 3.3a shows Albot1 moved up and down a corridor leading into a room three times. Since Albot1 does not remove any local maps experienced, its route map is a list of 158 local maps instead of about 20 for each trip down the corridor (see Figure 3.3b).

Table 3.2: Experiment 2 parameters

Information	Parameters
1. Distance	108.20m
2. Views Collected	158.
3. Views in Route map	143.

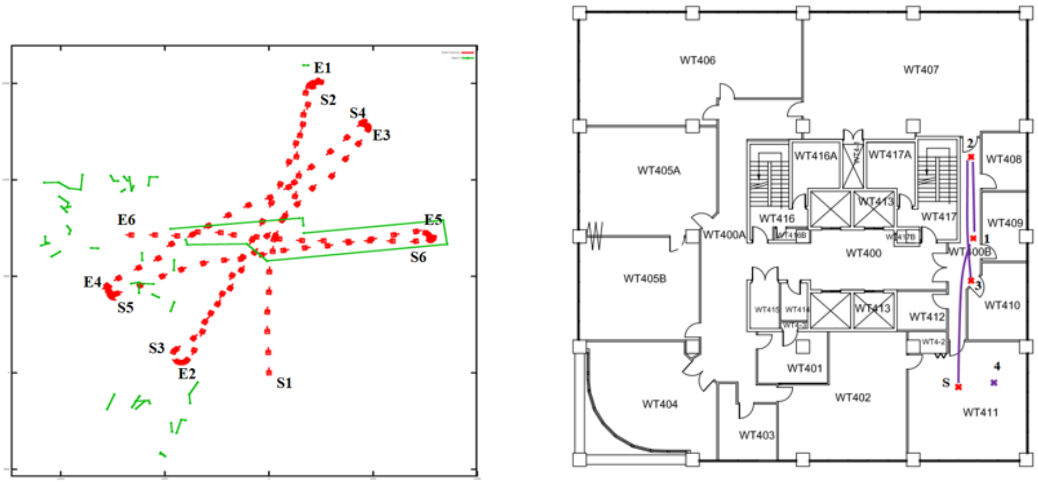


(a) (b)

Figure 3.3: Experiment 2: (a) The environment and path, and (b) the route map.

Figure 3.4a shows the trace map generated. For each trip from one end of the corridor to the other, it generates a useful map of the corridor. However, as it turns around (for example, reaching E1, turn around and start as S2) and move down the corridor, it generates another useful map of the corridor which is distorted from the earlier map. In the egocentric map, the earlier map would be deleted but no local map

is deleted in the route map. Re-tracing home using a route map would be problematic if there exist loops. Nonetheless, in this example, Albot1 encountered problem when re-tracing E6 → S6/E5 → S5/E4. This is because it thinks that E4 is at point 4 in Figure 3.4b. Interestingly, if there were no sensor error, Albot1 would be observed to repeat the journey, travelling up and down the corridor three times to return home. Such silliness in its behavior is not observed in natural species.



(a) (b)

Figure 3.4: (a) The trace map (starting from S1-E1, S6-E6) displayed with its egocentric map, and (b) the result. Albot1's retrace path is $S \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 4$ it fails to reach 4.

3.1.3 Summary

A straightforward idea of returning home is to follow one's route back. With its route map, one could generate a map of all the points where a local map was computed. Henceforth, Albot1 was empowered with an algorithm to re-trace home tracking these points. Two experiments were conducted. The first is a linear path and the other a circular path. In both experiments, Albot1 failed to return home and, in the second experiment, Albot1 exhibited silliness in its behavior. Both sensor errors and the nature of the route map prevents Albot1 from returning home unless the path is a short one

and has no loops.

To be specific, locomotion of re-tracing will generate errors, which will be accumulated as moving. In both experiment 1 and 2, Albot1 simply retraced every steps in the journey. Because of the existence of error, each step it retraced will be a little way off. As accumulating of those errors, it finally failed to reach a the target position, which may not be the final position but an intermediate one.

3.2 Albot1 re-tracing home using a simplified trace map

Observe that a local map is bounded by one's view and does not correspond to, say, a room. Henceforth, one could have had several local maps while moving across a room. A closer analysis of Albot1's path in Experiment 1 shows that this is the case (see Figure 3.5). Since finding one's way home is not about how one could retrace one's steps faithfully but rather how one could move intelligently through the empty space leading home, these intermediate positions can be ignored. Without being disadvantaged, one could and should head straight towards a point that lies just outside of the current view.

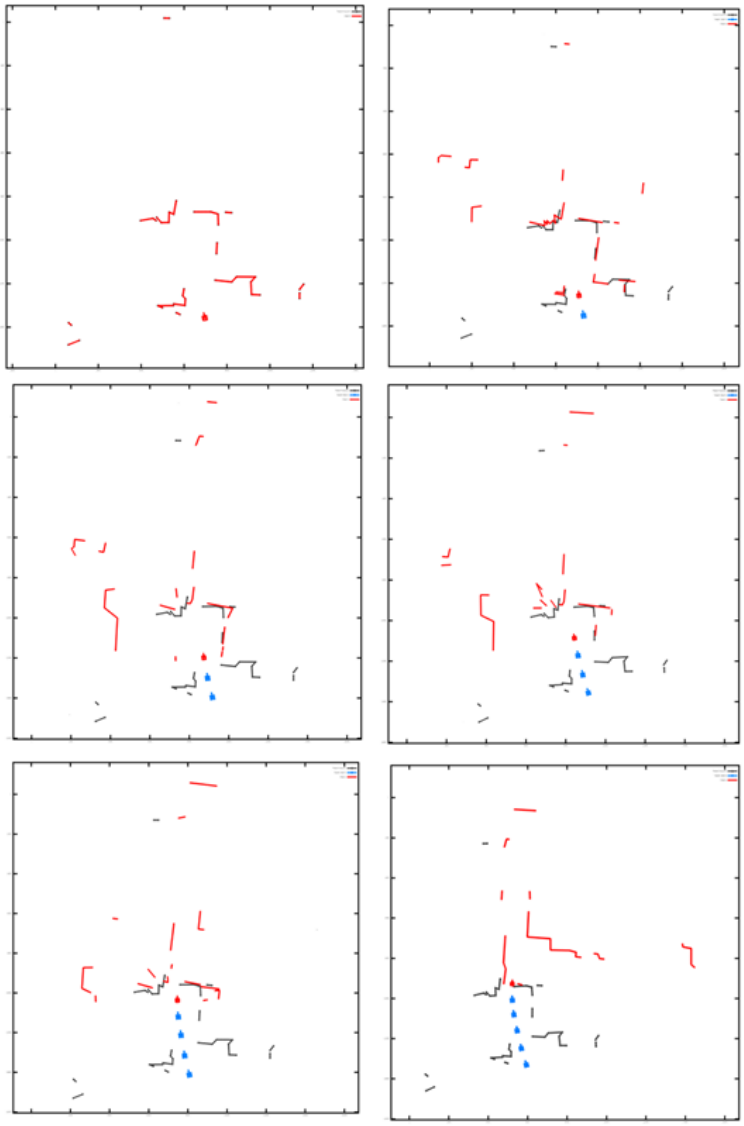
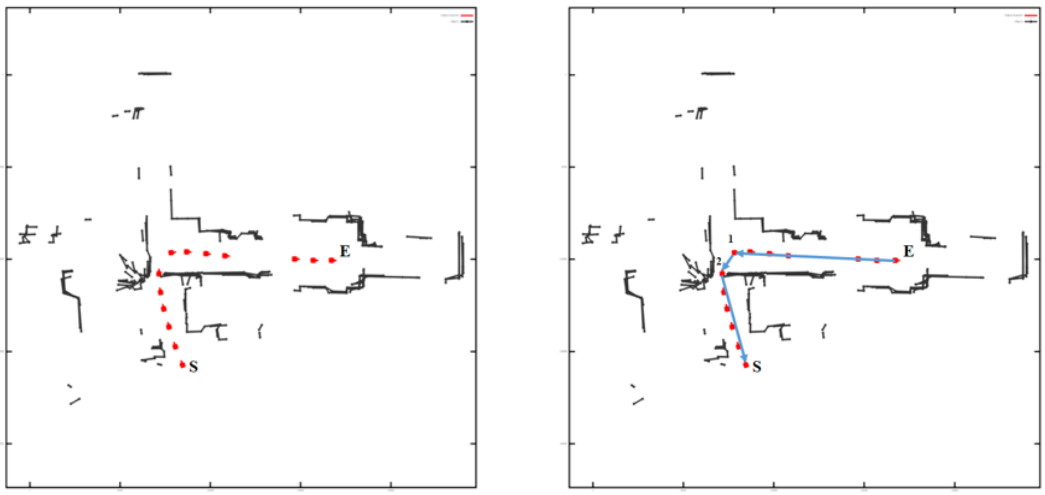


Figure 3.5: Albot1's first 6 local maps computed using the route shown in Figure 3.1: (top left) The first view accepted as the first local map. (the rest) Other local maps computed (in red) and each is superimposed onto the first local map (in black). The blue dots indicate Albot1's position where a view is accepted as a local map.

In the example shown, one heads towards the point where the seventh local map was created. Note that when simplifying the trace map, one repeats the path taken but on re-tracing home, one moves from the end to the start (Figure 3.6). In this case, on re-tracing, one reaches point 1 and then trying to go home by reaching point S. The path is blocked and to know where to turn to at point 1 to go to point S, it is important

to retain point 2 as well. Point 2 tells Albot1 where to turn to before going straight to S.

Figure 3.6 shows the theoretical path re-traced.



(a) (b)

Figure 3.6: (a) Albot1's trace map consisting of 13 red points. (b) Albot1's simplified trace map consisting of only 4 points. Position '2' is regarded as an intermediate point for returning from '1' to 'S' and thus avoiding the obstruction.

The new re-tracing algorithm and the simplify-trace-map algorithm are given below:

Algorithm 2 : retrace using a simplified trace map

Input: trace-map* = {$R^m_{x,y}, R^{m-1}_{x,y}, \ldots, R^1_{x,y}$};; simplified trace-map R_O – Albot1's current orientation

Output:Albot1 executes its moves.

1: $R_O \leftarrow$ turn(180^o) ▷ Albot1 turns around
2: Move-and-check (trace-map*) ▷ see below
3: Pop(trace-map*)
4: Position-at-opening(trace-map*)
5: Pop(trace-map*)
6: **Until** trace-map* = { } **do** ;
7: $R_O \leftarrow$ Turn-towards(R_O , trace-map*)
8: Move-and-check(trace-map*) ▷ see below
9: Pop (trace-map*)
10: Position-at-opening(trace-map*)
11: Pop(trace-map*)
12: **end until**

Using a simplified map, a move between two points could be large. Consequently, we implement move-and-check which, given two points, always move half-way, check if the next half is still in view. If not, shifts slightly to get it in view and then continues to the desired point. This is a simple correction of errors due to motion drift. Position-at-opening is a function that looks for a gap in the direction of the intermediate point and position Albot1 somewhere in the gap. Such approximation is found to be adequate to compensate for sensor errors in locating one's position in the environment.

;; route-map is a list of (V_i, $R_{xi,yi}$) whereby V_i is the view remembered and $R_{xi,yi}$ is the robot position in the previous view, V_{i-1} when V_i is taken.

Algorithm 3 : Simplify route map

Input: route-map = {$(V_1, R_{x1,y1}), (V_2, R_{x2,y2}), \ldots, (V_m, R_{xm,ym})$**} ;; route map**
Output:route-map* = {$R_{x1,y1}, R_{xi,yi}R_{xj,yj}, R_{xk,yk}, \ldots (V_m, R_{xm,ym})$**} ;; simplified route map**

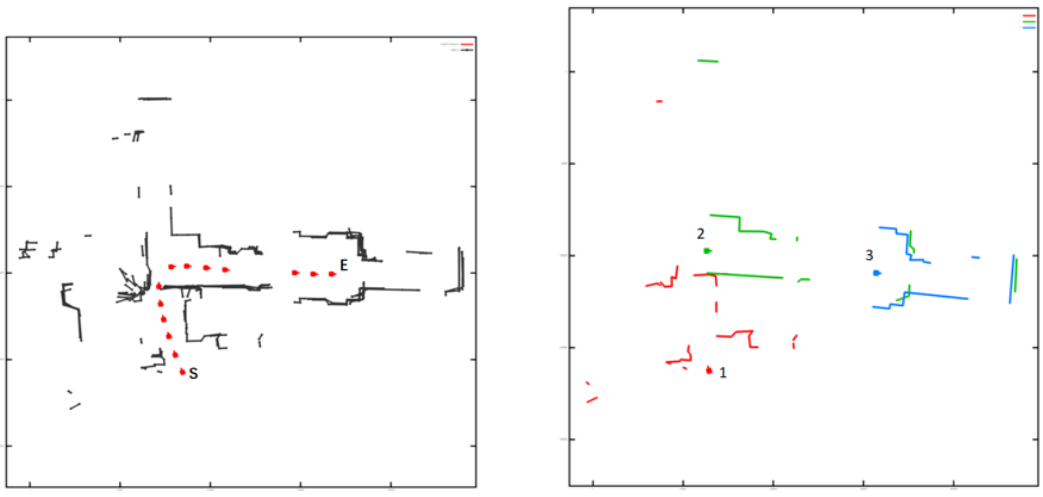
;; initialisation
$RP_x \leftarrow$ Get-robot-position(route-map) ; ▷ return $R_{x,y}$ of the first view in route-map
$V \leftarrow$ Get-robot-view(route-map) ; ▷ Take the first entry
$RP_{next} \leftarrow \{\}$; $RP_{intermediate} \leftarrow \{\}$
Pop(route-map)
Until route-map = { } **do** ;
    $RP_{next} \leftarrow$ Get-robot-position(route-map)
    if is-behind(RP_{next}, RP_x) **then**
    ;; it has turned around, so don't need this local map but the next. Re-initialise
        $RP_x \leftarrow RP_{next}$
        $V \leftarrow$ Get-robot-view (route-map)
        $RP_{next} \leftarrow \{\}$; $RP_{intermediate} \leftarrow \{\}$
    else
        if is-intersect(RP_{next}, V) **then**
        ;; RP_{next} could not be seen from V, start simplifying the map
            route-map* $\leftarrow$ route-map* + RP_x + $RP_{intermediate}$
            $RP_x \leftarrow PR_{next}$
            $V \leftarrow$ Get-robot-view (route-map)
            pop(route-map)
        else

```
                $RP_{intermediate} \leftarrow RP_{next}$
                pop(route-map) ;;                          ▷ ignore this view
            end if
        end if
    end until
    route-map* ← route-map* + $RP_x$
```

Note: Is-intersect is a Boolean function that checks if two points in space intersects with any surfaces in view. If it does, then the robot has moved out of the boundary for that view. Is-behind is a Boolean function that checks if the robot has moved behind the current viewing position.

3.2.1 Experiment 3

The failed attempt to return home in Experiment 1 is repeated using the new algorithm. Figure 3.7 shows the result of simplifying the route map computed in Figure 3.1 and Figure 3.8 shows Albot1 successfully retraced its steps home.



(a) (b)

Figure 3.7: (a) Albot1's trace map. (b) Albot1's simplified trace map with each local map displayed in a unique color (intermediate points are not shown). Albot1 now re-tracing with 4 points rather than 12.

The failed attempt to return home in Experiment 1 is repeated using the new

algorithm. Figure 3.7 shows the result of simplifying the route map computed in Figure 3.1.

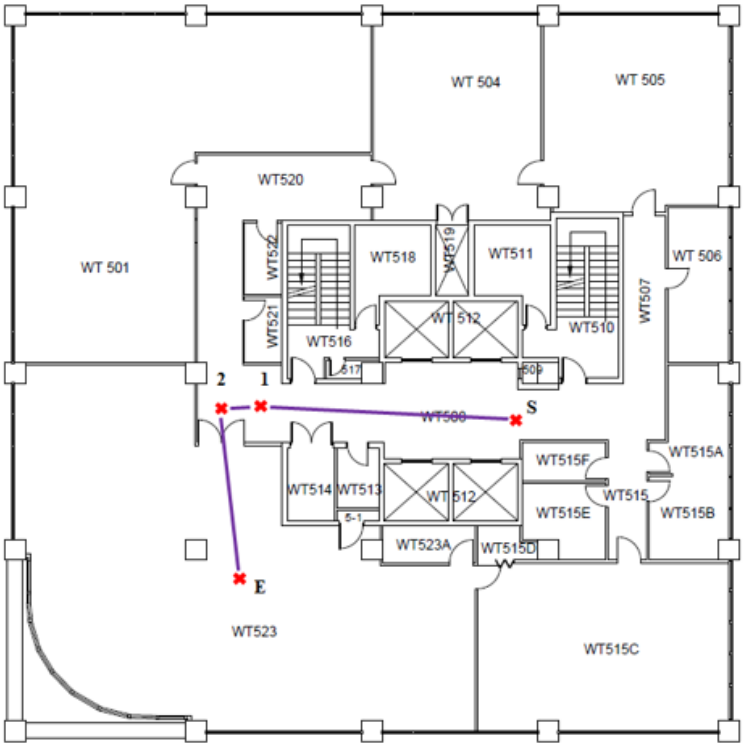


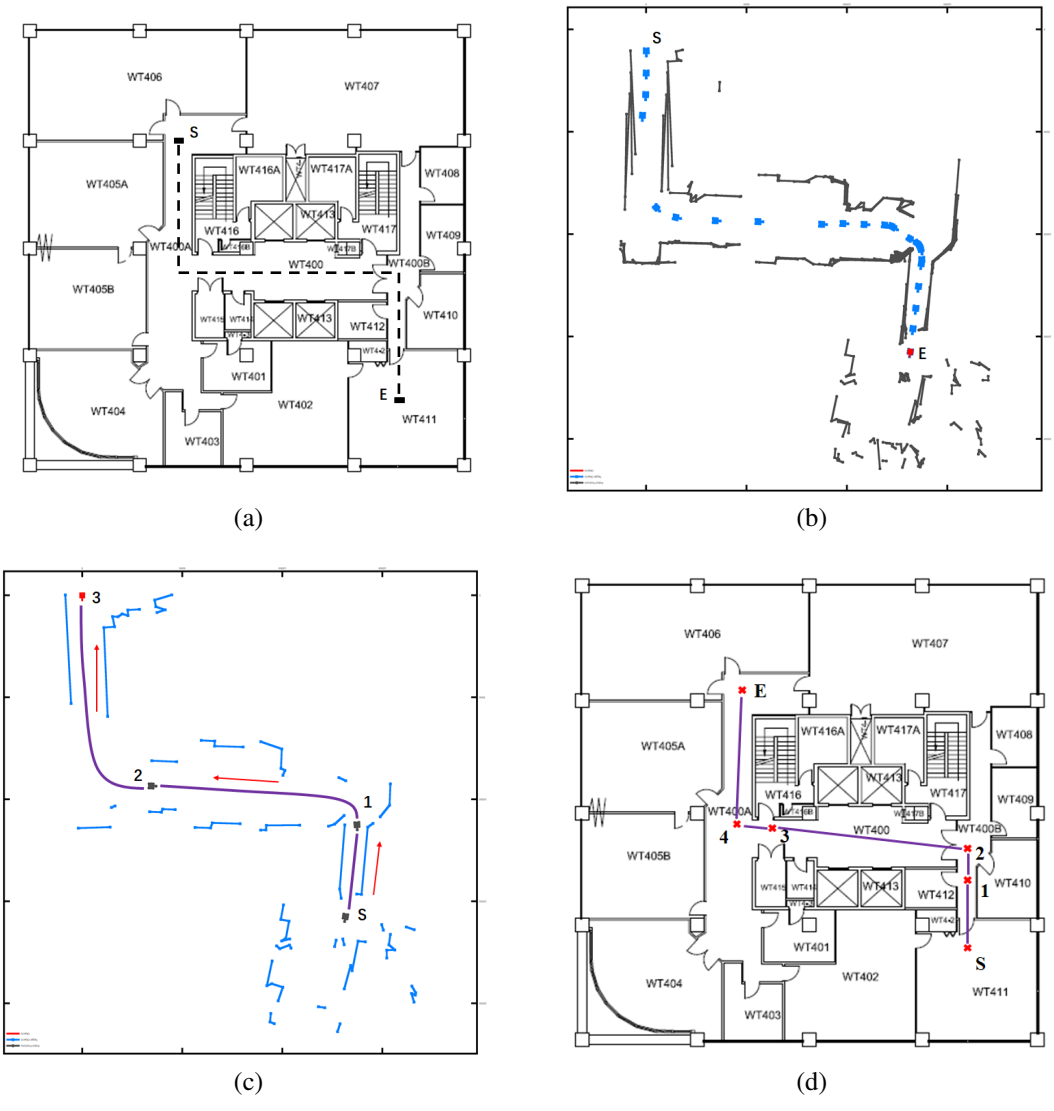
Figure 3.8: Albot1 successfully re-traced its step home.

3.2.2 Experiment 4

Figure 3.9 shows another experiment using the same algorithm but on a longer path. and Albot1 was again able to return home. In this case, Albot1 traveled approx. 33.41m and collected 32 views and remembered 25 of them in its route map.

Table 3.3: Experiment 4 parameters

Information	**Parameters**
1. Distance	33.41m
2. Views Collected	32.
3. Views in Route map	25.



(a) (b) (c) (d)

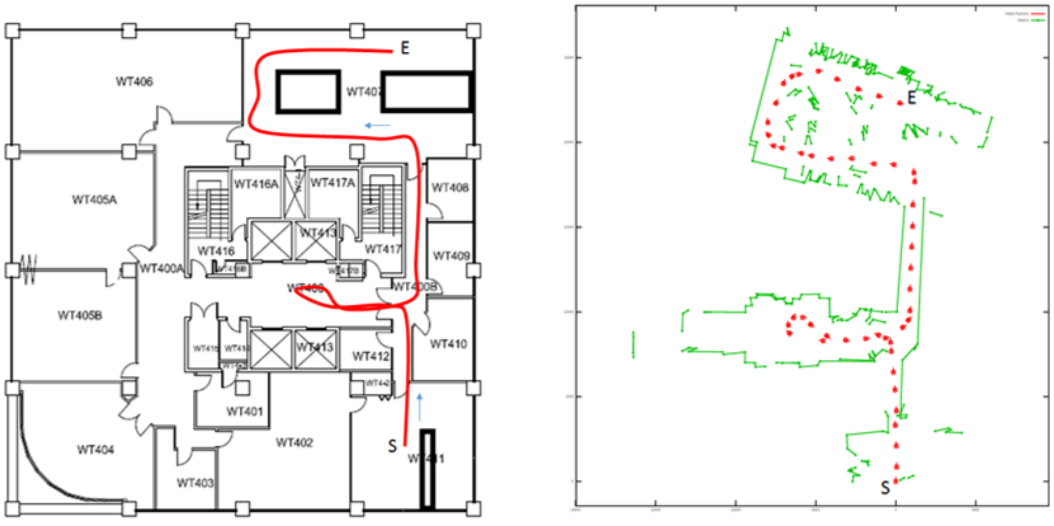
Figure 3.9: (a) The environment and the path taken. (b) The trace map. (c) The simplified trace map (intermediate points not shown). (d) Albot1 successfully returned home.

3.2.3 Experiment 5 – Paths with loops

Recall that our initial algorithm could not handle loops (see Figure 3.3). In this experiment, Albot1 explored an environment using a route with a simple loop (Figure 3.10). Albot1 traveled approx. 68.65m and collected 54 views and remember 49 of them in its route map. Figure 3.11 shows Albot1 successfully returned home albeit exhibiting a silliness as observed earlier. However, it is interesting to note that a single loop does not necessarily distort the map to the extent that one would fail to re-trace home. Nonetheless, it is not good to repeat the loop itself.

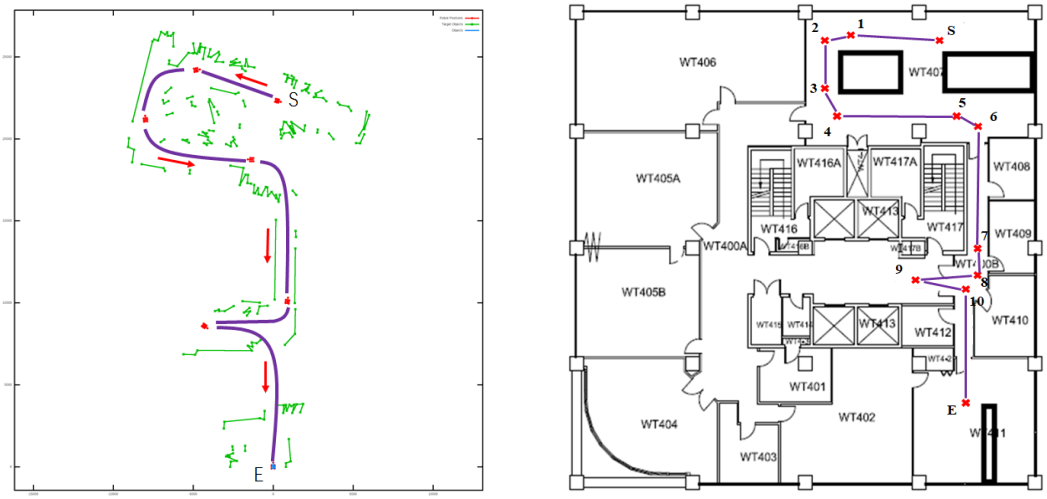
Table 3.4: Experiment 5 parameters

Information	Parameters
1. Distance	68.65m
2. Views Collected	54.
3. Views in Route map	49.



(a) (b)

Figure 3.10: (a) The environment and the path taken. (b) The trace map generated.



(a) (b)

Figure 3.11: (a) The simplified trace map (intermediate points not shown). (b) Albot1 successfully returned home.

3.2.4 Experiment 6 – Paths with loops

Figure 3.12 shows Albot1 exploring an environment that involves more complex looping. Note that this is the same environment and route used in Y&H's experiment 2. Albot1 traveled approx. 130.31m, had 153 views and collected 103 local maps in its route map. Albot1 refreshed its egocentric map three times in this route (at point 6, 9 and 11 in Figure 3.12).

Table 3.5: Experiment 6 parameters

Information	**Parameters**
1. Distance	130.31m
2. Views Collected	153.
3. Views in Route map	103.

Figure 3.13b shows the trace map superimposed with all the local maps. The start of the map (where S is) is now distorted from the end part (where E is) and one could see

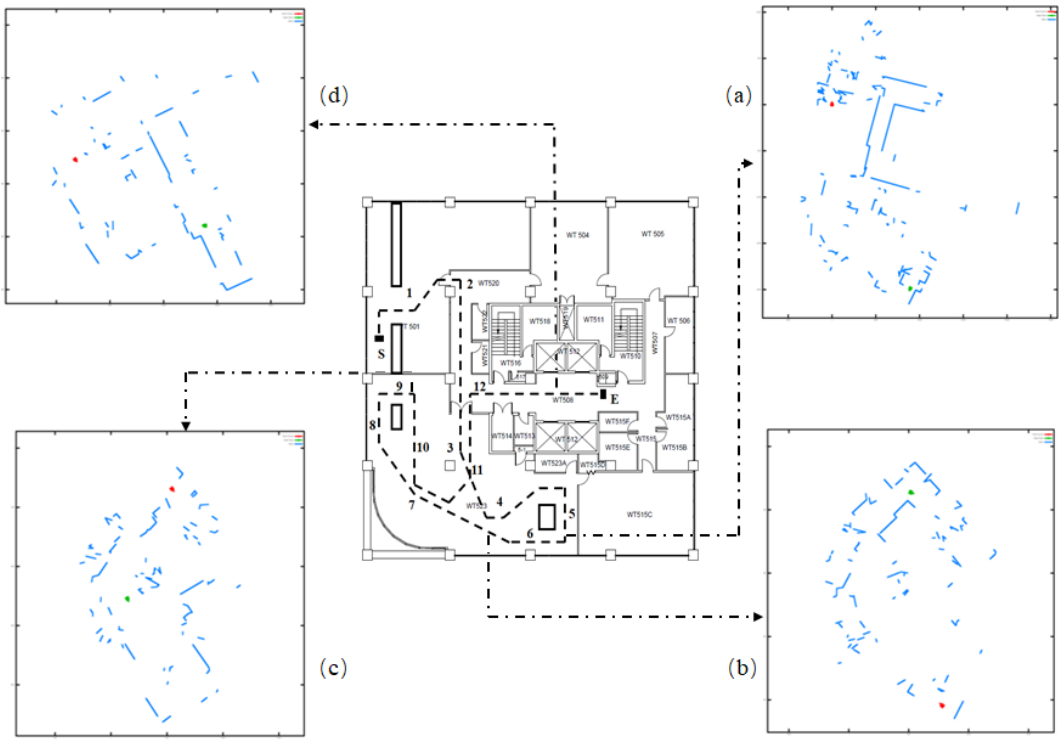


Figure 3.12: The environment and the path. The numbers indicate the travel direction. (a), (b), (c), (d) are four egocentric maps computed while exploring the environment.

one re-tracing through walls. This could also be seen more clearly in the simplified trace map as shown in Figure 3.13c. Re-tracing from point 9 to 10 would not be possible. Figure 3.13d shows Albot1 re-tracing home. At point 12, it is too distorted to continue. Albot1 thus failed to go home.

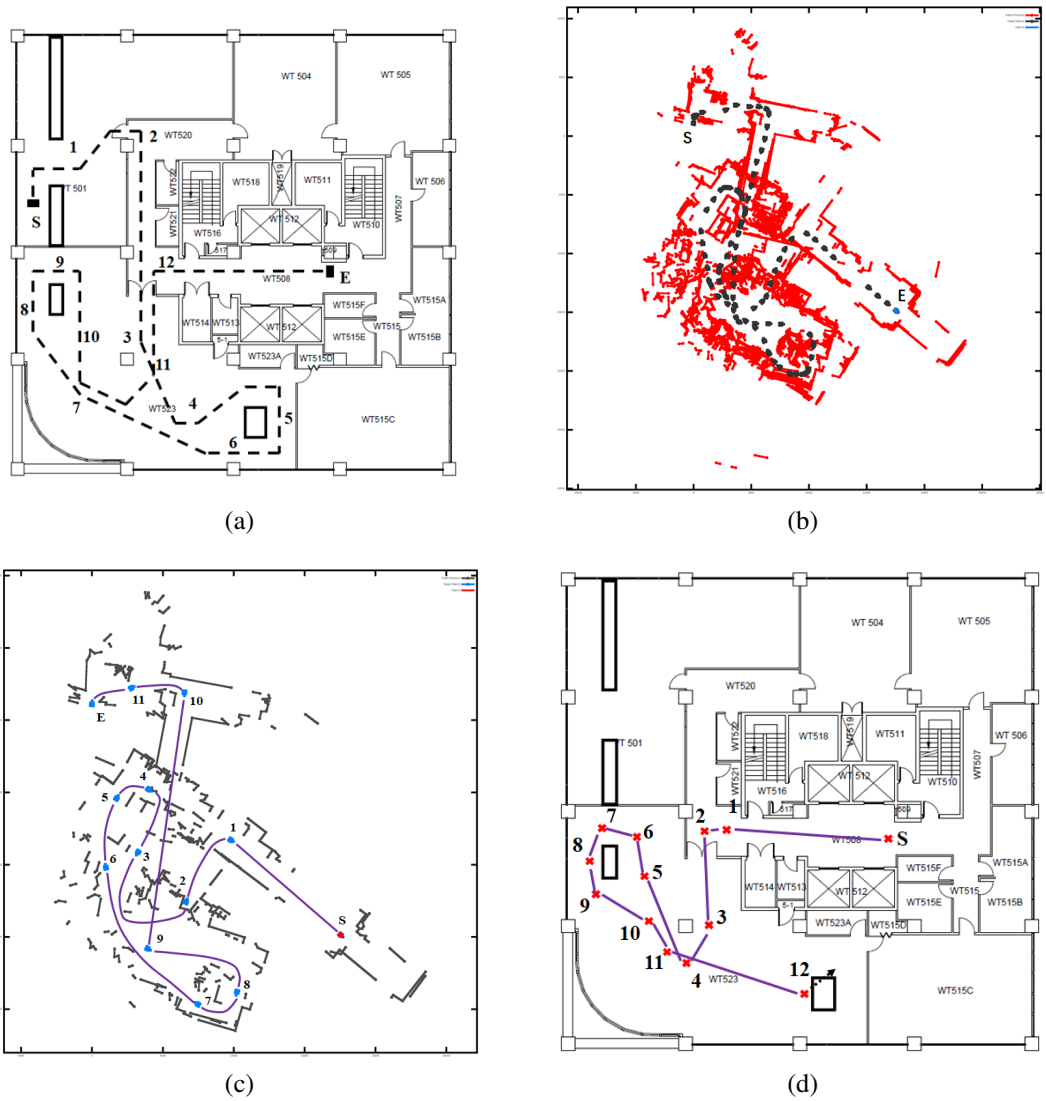


Figure 3.13: (a) The environment and the path. (b) The trace map superimposed with all the local maps (c) The simplified trace map (intermediate points not shown). (d) The result of Albot1 re-tracing home. It failed and stopped at point 12.

3.2.5 Summary

By pre-processing the information in the route map, it is possible to navigate home despite sensor errors. However, there are still limitations. First, one still repeats a loop. Second, having explored a large environment that involves re-visiting parts of it, the trace map could be too distorted to be useful. Third, for a large environment, sensor

errors would still cause one to fail to return home.

In experiments 3-5, the simplified trace map reduces the steps of retracing; it thus reduces the errors occurred during movements. Nonetheless, the simplified trace map could provide good orientation for returning, which is a necessary requirement for using it to return home. In experiment 6, it shows that the simplified trace map has some parts which are too distorted, which is the key reason why Albot1 failed to return home.

3.3 Short-cuts

The use of a trace map to return home is interesting. It generates a global view of the route needed to go home and, in some situations, one could detect intersection points which afford short-cuts. Consequently, Albot1's algorithm to return home is modified to detect intersection points in its trace map and if found, it will attempt to return using short-cuts. The algorithm is presented below:

Algorithm 4 : Finding Intersections of Path segments

Input: trace-map = { $R_{mx,y}$**,** $R_{m-1x,y}$**, ... ,** $R_{1x,y}$**} ;; simplified trace-map**
Output:trace-map* = { $R_{mx,y}$**,** $R_{m-1x,y}$**, ... ,** $R_{1x,y}$**} ;; reduced trace map**

While length(trace-map) > 2 **do**
    $Path_1$ ← Get-first-2-robot-positions(trace-map)
    trace-map* ← pop(trace-map) ;; ▷ the first point is definitely the starting point
    trace-map* ← pop(trace-map) ;; ▷ and pop the second point as well avoid two adjacent paths
    rest-of-route ← route-map ;;
    While length(rest-of-route) > 1 **do**
        $Path_n$ ← (Get-first-2-robot-positions(rest-of-route)
        if intersect($Path_1$, $Path_n$) **then**
            pop(rest-of-route) ;; ▷ remove the first point of $Path_n$
        end if
    end while
end while
trace-map* ← rest-of-route;;
Return trace-map*

3.3.1 Experiment 7 – Possible Short-cuts

In this experiment, Albot1 explored the environment as shown in Figure 3.14. Albot1 traveled approximately 68.22m with 42 views and its route map consists of 36 local maps. Albot1 was able to return using short-cuts.

Table 3.6: Experiment 7 parameters

Information	**Parameters**
1. Distance	68.22m
2. Views Collected	42.
3. Views in Route map	36.

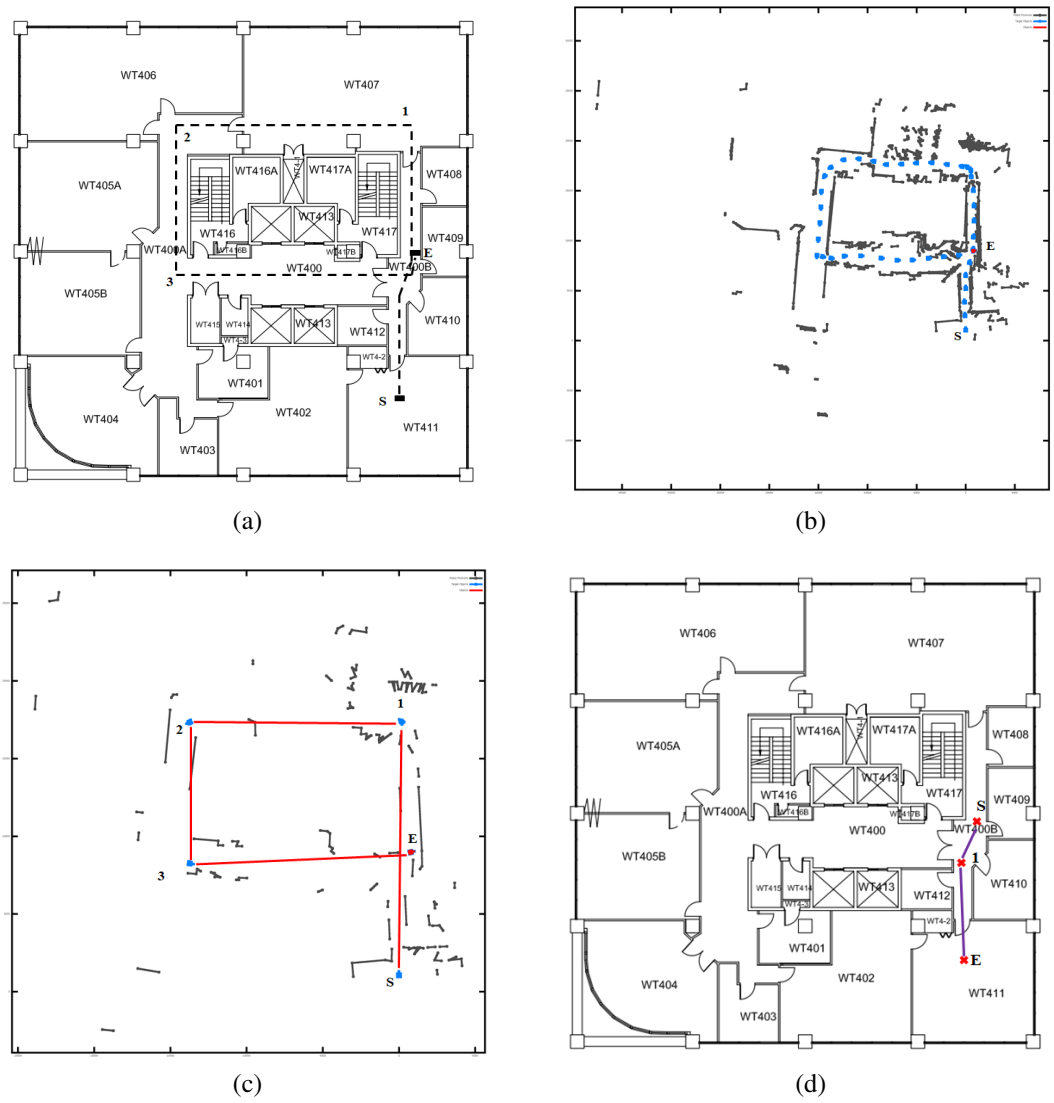


Figure 3.14: (a) The environment and the path (b) The trace map. (c) The simplified trace map with an intersection point detected. (d) Albot1 returned home using the short-cut.

3.3.2 Experiment 8

In this experiment, Albot1 explored the environment as shown in Figure 3.15. Albot1 traveled approximately 74.69m with 62 views and its route map consists of 52 local maps. Again, Albot1 is able to return using short-cuts.

Table 3.7: Experiment 8 parameters

Information	Parameters
1. Distance	74.69m
2. Views Collected	62.
3. Views in Route map	52.

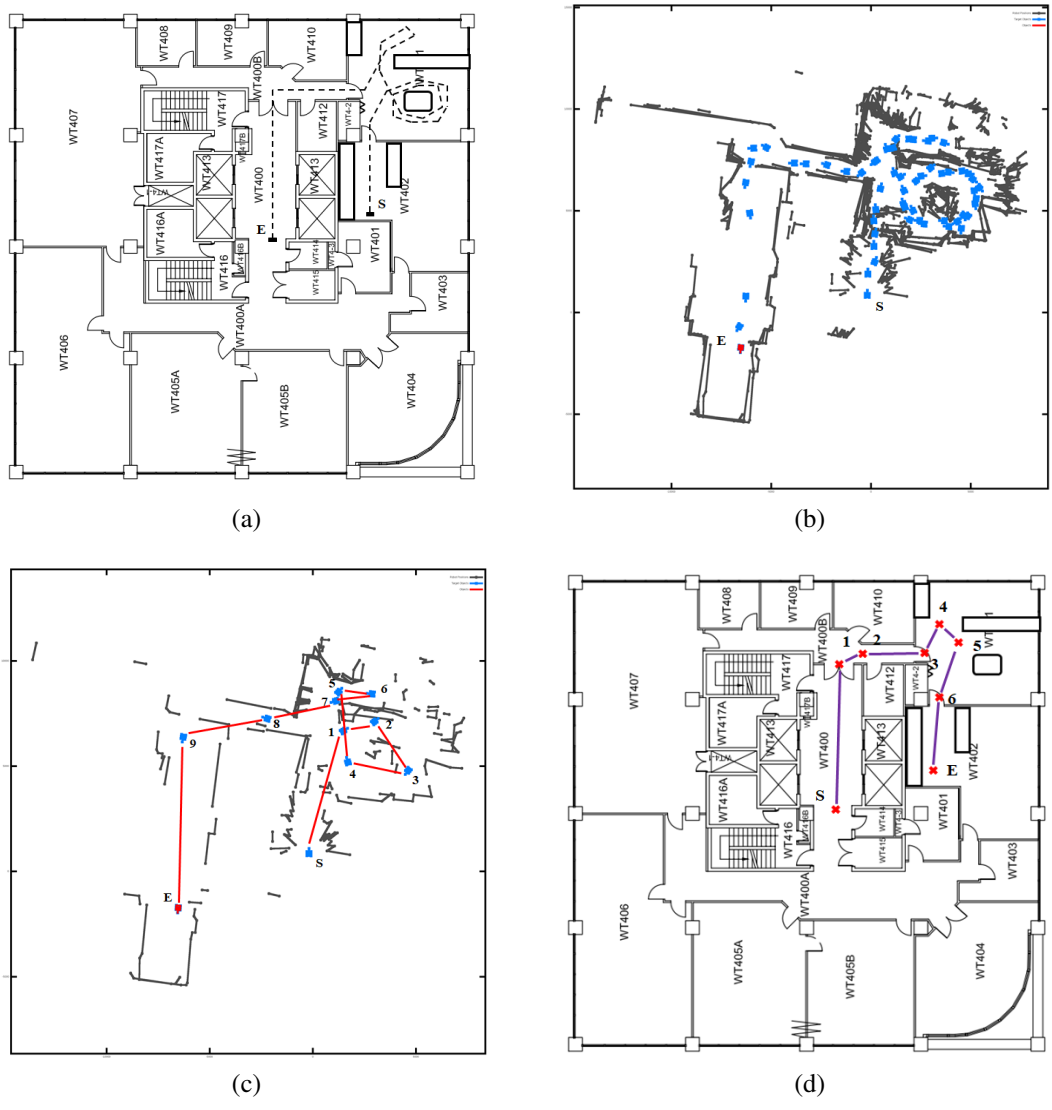


Figure 3.15: (a) The environment and the path taken. (b) The trace map. (c) The simplified trace map. (d) Albot1 returned home using a short-cut.

3.3.3 Experiment 9 – Failed Short-cuts

Two earlier experiments could be inspected to show that Albot1 would fail to detect short-cuts despite their presence.

In Experiment 2, Albot1 could detect the intersection of the paths up and down the corridor but these intersections gave an incorrect short-cut home.

In Experiment 6, the short-cut home is to travel from E to 12 to 2 to S (see Figure 3.13a) but unfortunately the trace map (Figure 3.13c) has been distorted and the short-cut could not be computed.

3.3.4 Summary

Short-cut is possible when using the new re-tracing algorithm. However, the problem remains if the trace map is distorted. As mentioned earlier, the positions stored in the simplified trace map are presented using a global coordinate system. Thus, holding a good orientation in the global coordinate is the most important pre-requisite for finding short-cuts while returning, which is the reason why the two experiments in Experiment 9 failed although intersections are detected in the journey.

3.4 Conclusion

In this chapter, Albot1 is empowered with algorithms to re-trace its steps home using its route map by generating a trace map. Such a map could be more useful if generating it, one takes into account the structure of the environment. Doing so produces a simpler path to re-trace home and that helps reduce errors. Using a trace map, one could detect short-cuts but only with certain path structure.

However, the nature of the route map computed means that the trace map produced could be seriously distorted. For example, this happens when one travels up and down

the corridor a few times or repeat visiting a part of a large environment several times. The trace map would not then correspond meaningfully to the physical environment.

Chapter 4

On computing an enduring map

With Y&H's model of cognitive mapping, one begins with a route map that consists of a list of local maps (i.e. views) visited. The route map is used to generate an egocentric global metric map but, as shown in chapter 3, using it to generate a trace map to re-trace home is problematic. In this chapter, I investigate empowering Albot1 to compute a representation that is more enduring. Section 4.1 describes a straightforward solution to the corridor problem as discussed in Experiment 2 in chapter 3. An intuitive idea is to make a copy of the egocentric map prior to it being refreshed. One then becomes aware that one is in a familiar environment and tracks one's position in it. This suggests that one computes both an egocentric map and an enduring map in tandem whenever the latter is created. Three experiments were conducted. Section 4.2, investigate empowering Albot1 to compute the geometry of a place and use it to re-orient towards exits of a place. Each local map in a simplified trace map suggested in chapter 3 is used to define a place. Two experiments were conducted. Section 4.3 concludes with a summary of the key results obtained.

4.1 Transforming an egocentric map into an enduring map

When local maps are added to the egocentric map without causing any overlap, the egocentric map computed is enduring, in the sense that it has no information loss yet. Hence, prior to deleting information in an egocentric map, one could make a copy of it and use it as an enduring map. However, when an overlap occurs, Albot1, in general, does not know that it has entered a familiar environment except when Albot1 turns around and walks back towards where it came from. In this special case, Albot1 could make a copy of its egocentric map and uses it as an enduring map.

With an enduring map, Albot1 could track its position in it but performs no updating of the enduring map. Constant updating will lead to a complete map which is unlike a cognitive map. This algorithm is as shown below:

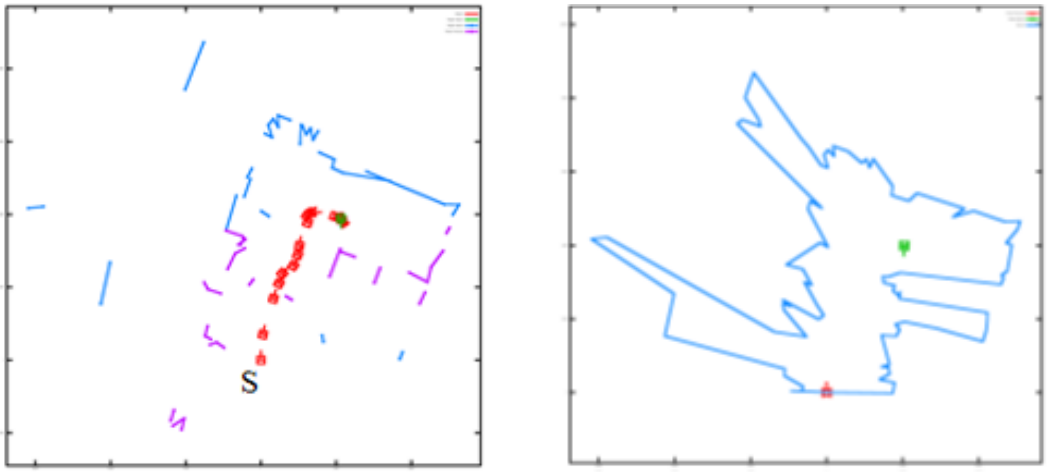
Algorithm 5 : Mapping with an Enduring Map

Input: R_p **= robot's position, ...** R_o **= robot's orientation route map, ...** M_{ego} **= the egocentric map,** V_c **= {Sc1, Sc2, ..., Scn} ; current view of the physical environment**

Output:

1: $M_e \leftarrow M_{ego}$;; ▷ make a copy of the egocentric map as initial enduring map
2: $V_m \leftarrow$ get the robot's view of the enduring map at (R_p, R_o)
3: Match(V_m, V_c) and update R_p and R_o and computes a boundary for M_e
4: Execute next move and get a new V_c, R_p, and R_o.
5: Match(V_m, V_c) and update R_p and R_o
6: Check if Albot1 is still inside the enduring map and if true, go to step 4.

Briefly, Albot1 first makes a copy of its egocentric map (step 1) when it has turned around. It then aligns the map with its current view so as to orient itself in it (step 2). The boundary of the enduring map is then computed (step 3; Figure 4.1b) and as Albot1 continues to move, it tracks its position in the enduring map but performs no updating.

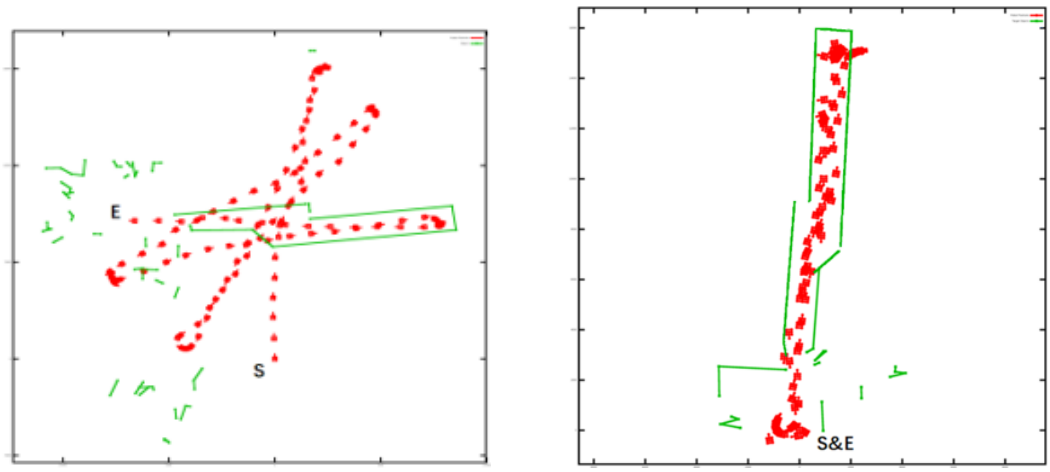


(a) (b)

Figure 4.1: (a) Albot1's egocentric map before turning around. The purple lines indicate the part where the incoming view will cause an overlap (b) Albot1's enduring map which is a copy of the egocentric map shown in (b). Its boundary is computed using the algorithm: `http://www.angusj.com/delphi/clipper.php`.

Figure 4.2 shows the result of an initial test of the algorithm. Albot1 moved up and down the corridor three times as before in Experiment 2 but now maintaining both an egocentric map and an enduring map. Note that each egocentric map computed is positioned differently when plotted within a single reference frame whereas the enduring map is a single map computed.

What happens when Albot1 moves out of its enduring map? Albot1 continues tracking its movements in both maps simultaneously until its egocentric map is refreshed. When that happens, it depends on whether Albot1 has turned around or not. If yes, Albot1 updates its existing enduring map by combining it with the egocentric map prior to being refreshed, thereby creating a much larger enduring map. If not, Albot1 de-activates the enduring process and deletes the existing enduring map. A flowchart showing how the two processes interact is shown in Figure 4.3. The above algorithm is now extended to allow simultaneous tracking in the enduring map and in the egocentric map: Three experiments are conducted to study such a cognitive mapping process.



(a) (b)

Figure 4.2: Experiment 2 repeated: (a) Albot1's egocentric map refreshed 6 times. (b) A single enduring map is computed. Red dots show Albot1's absolute position in the map.

Algorithm 6 : Mapping with an Existing Enduring Map

Input: R_p **= robot's position, ...** R_o **= robot's orientation route map, ...** M_e **= the enduring map, ...** M_{ego} **= the egocentric map,** V_c **= {Sc1, Sc2, ..., Scn} ; current view of the physical environment;**

Output: R_p^* **= tracked robot position**

$V_c \leftarrow$ get the robot's view of the enduring map at (R_p, R_o) ;; ▷

if is-inside(R_p, M_e) **then**;

  $R_p^* \leftarrow$ track-position(M_e, V_c, R_p);; ▷ track its position in enduring map

  Execute next move and get a new V_c, R_p, and R_o.;

  go to step1

else

  if turn-around(M_{ego}) **then**;

    $M_e \leftarrow M_t + M_e$; ▷ merge the two as one

    Execute next move and get a new V_c, R_p, and R_o.

    go to step 1;;

  else

    if is-refreshed(M_{ego}) **then**

    break; ;; ▷ inactivate enduring process

    end if

  end if

end if

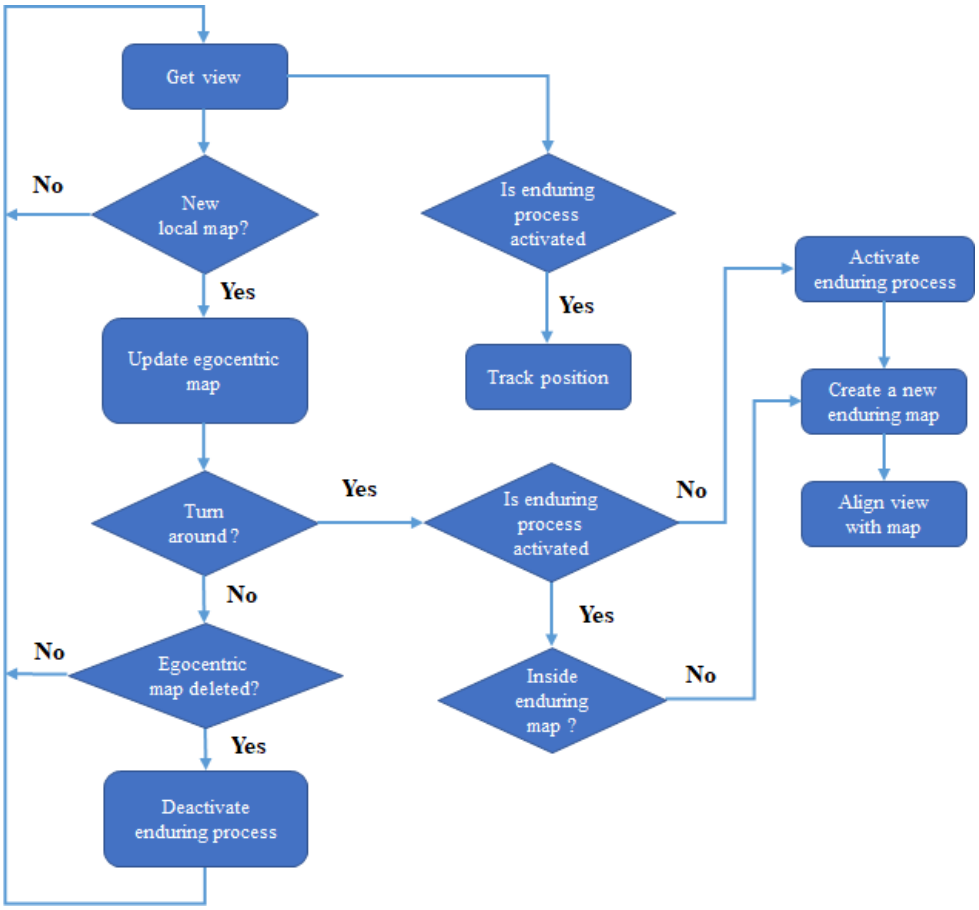


Figure 4.3: A flowchart of Albot1's cognitive mapping that computes both an egocentric map and an enduring map.

4.1.1 Experiment 10

In this experiment, the environment and Albot1's path is shown in Figure 4.4 (same environment as Experiment 5 but with a different path). Albot1 travelled approximately 73.90m with 119 views and collected 105 of them as local maps. It starts from position 'S' and comes back to its original position, now marked 'E'.

Table 4.1: Experiment 10 parameters

Information	**Parameters**
1. Distance	73.90m
2. Views Collected	119.
3. Views in Route map	105.

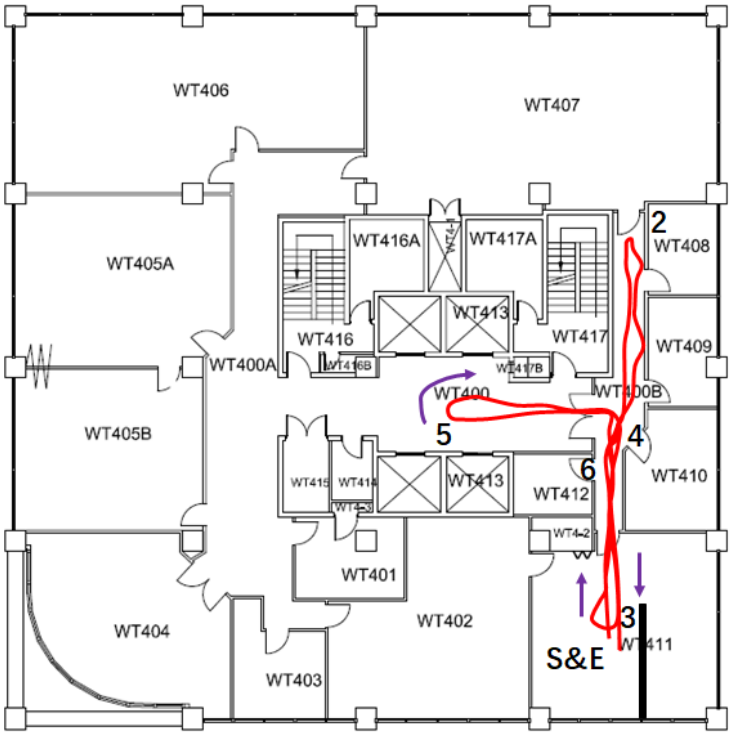
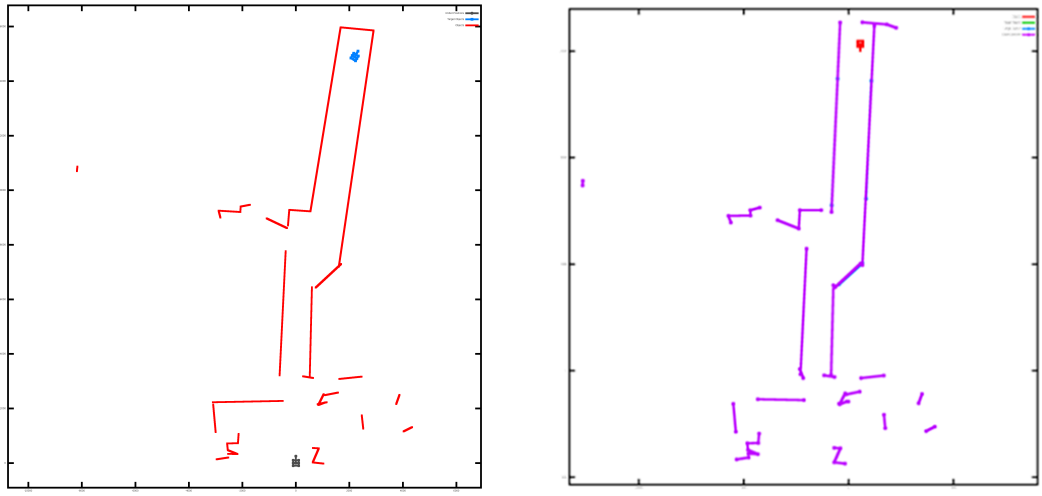


Figure 4.4: Experiment 10: The environment and its path.

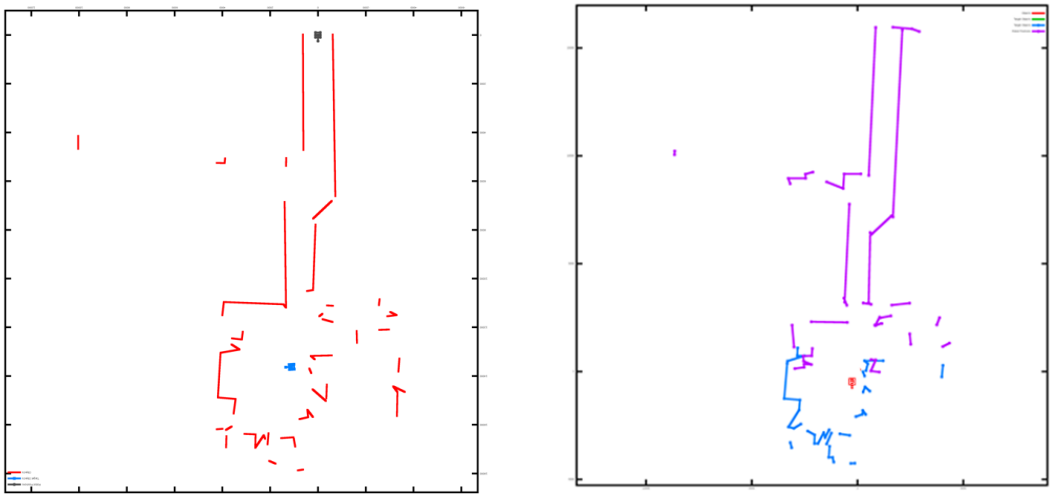
There are three points where a turning around occurs, namely points 2, 3 and 5. At point 2, the first enduring map is computed (Figure 4.5) and at both points 3 and 5,



(a) (b)

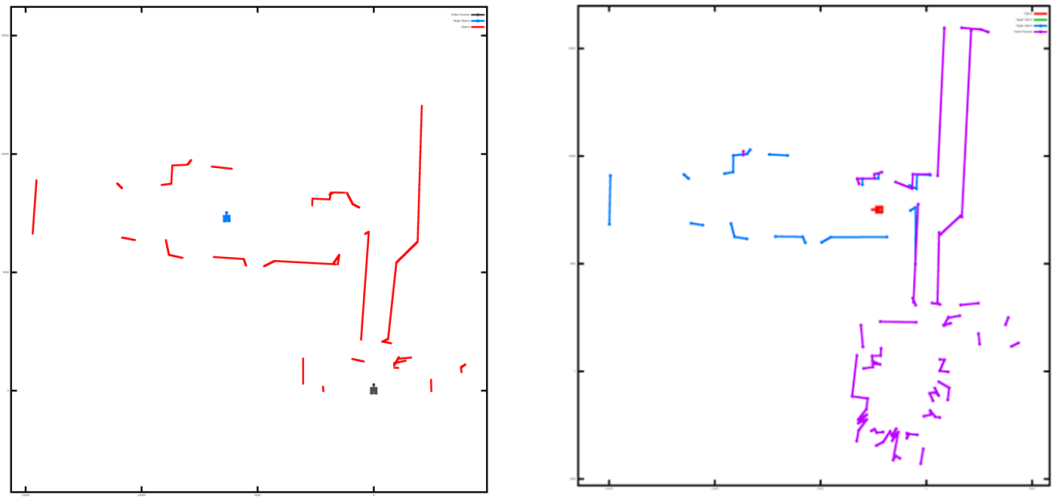
Figure 4.5: (a) The egocentric map computed prior to turning around at the top. (b) The enduring map computed and re-aligned with Albot1's current view looking down the corridor.

Albot1 has moved out of its enduring map and then turned around. This caused a new enduring map to be computed which is a combination of the existing enduring map with the new information in the current egocentric map (Figure 4.6 and 4.7).



(a) (b)

Figure 4.6: (a) The egocentric map computed prior to turning around at the bottom. (b) The enduring map computed by adding the new part of the egocentric map to its existing enduring map, and red point indicates where it exited the enduring map.



(a) (b)

Figure 4.7: (a) The egocentric map computed prior to turning around on the left side of the corridor. (b) The enduring map computed again by adding the new part of the egocentric map to its existing enduring map.

4.1.2 Experiment 11

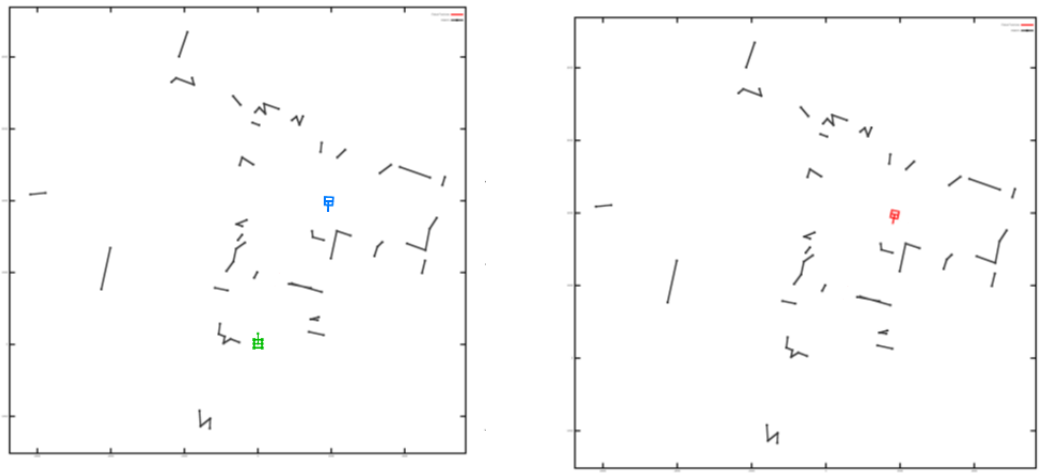
Figure 4.8: The environment and Albot1's path for Experiment 11.

In this experiment, Albot1 explored a much larger environment (Figure 4.8), downloaded from the robotics data set repository (Howard and Roy, 2003). It was the same environment Y&H used in their experiment 1 (Y&H's Figure 9). This environment is interesting because the path causes Albot1 to re-visit some parts of the environment a few times. Albot1 started from one room labelled 'S' and moved to E. Note that the

Table 4.2: Experiment 11 parameters

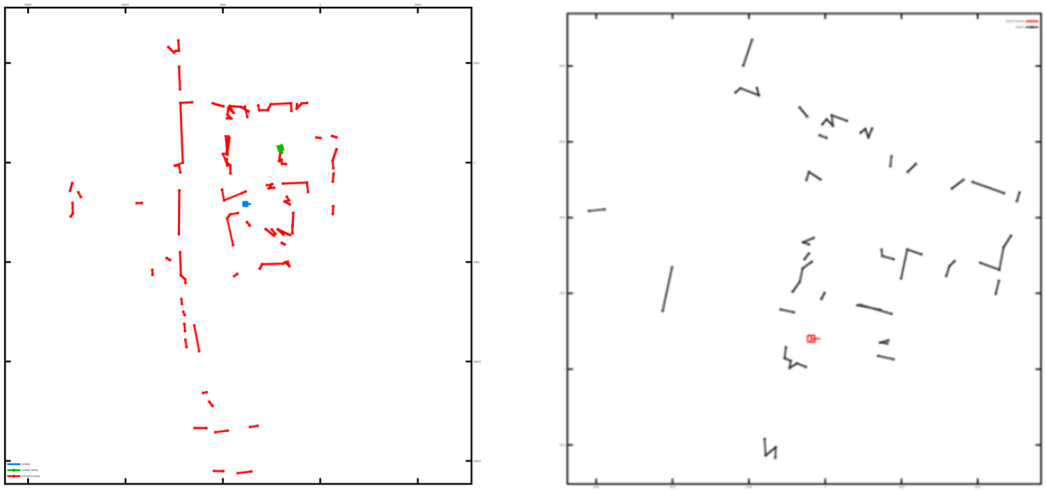
Information	**Parameters**
1. Distance	99.88m
2. Views Collected	184.
3. Views in Route map	162.

complete route of the data set is not used in this experiment. The travelled distance is approx. 99.88m and there are 184 steps and Albot1 remembers 162 local maps. Albot1 started from 'S', all other labelled points indicate where Albot1 made serious deletion of its current egocentric map. At point 1, Albot1 made its first turn around and this caused an enduring map to be computed (Figure 4.9) At point 2, a new egocentric map is computed but Albot1 detected that it has re-entered into a familiar part of the environment (Figure 4.10). It does not create a new enduring map.



(a) (b)

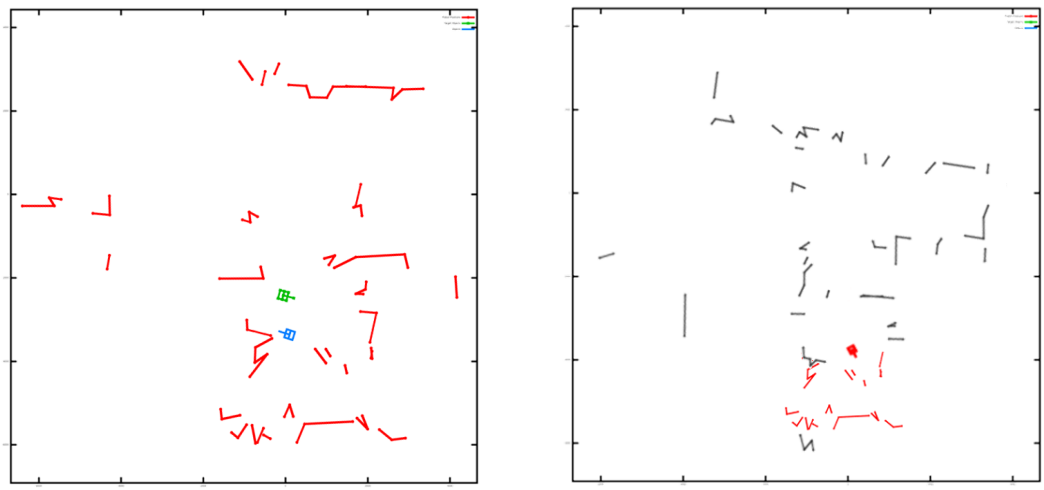
Figure 4.9: (a) The egocentric map computed prior to turning around at point 1. (b) The enduring map computed and re-aligned with Albot1's current view.



(a) (b)

Figure 4.10: (a) A new egocentric map is computed at point 2 but (b) no new enduring map is computed.

At point 3, Albot1 has moved out of its enduring map and then turned around. This time, it creates a new enduring map by adding the new part in the egocentric map to the existing enduring map (Figure 4.11).

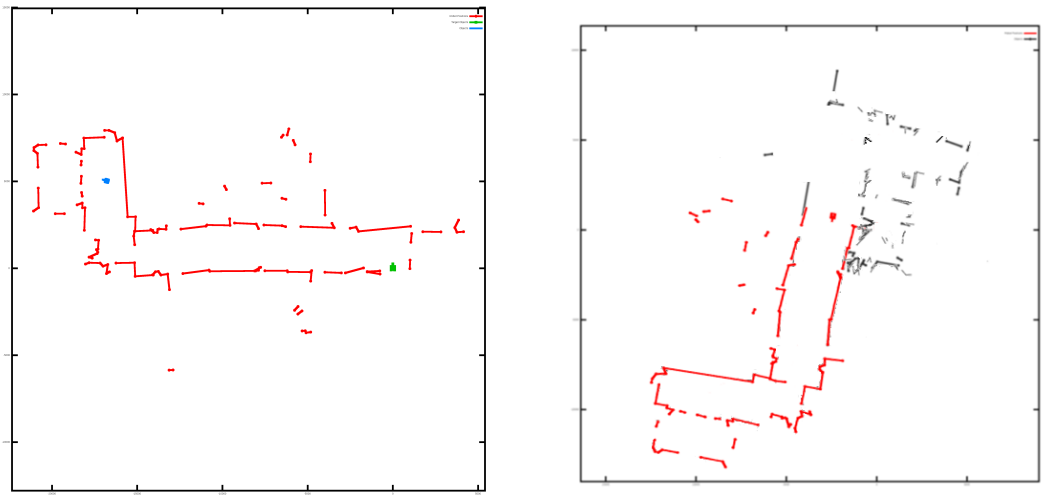


(a) (b)

Figure 4.11: (a) The egocentric map computed prior to turning around at point 1. (b) The enduring map computed and re-aligned with Albot1's current view.

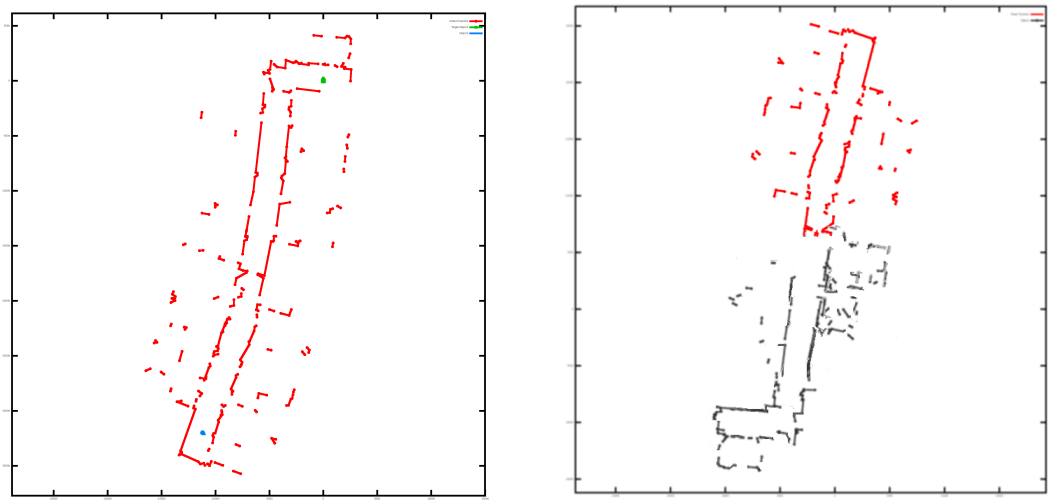
At point 4, Albot1 computed a new egocentric map but detected that it is still in

its enduring map. Thus, no new enduring map is created until it moved to point 5 and turned around. A new enduring map is created which includes the existing enduring map (Figure 4.12).



(a) (b)

Figure 4.12: (a) The egocentric map computed prior to turning around at point 5. (b) The enduring map computed.



(a) (b)

Figure 4.13: (a) The egocentric map computed prior to turning around at point E. However, for simplification, the display shows only the egocentric map computed from point 6 to point E. (b) The enduring map computed.

At point 6, Albot1 moved out of its existing enduring map and then turned around at point E. Again, it triggers computing a new enduring map that combines the whole environment explored (Figure 4.13).

4.1.3 Experiment 12

In this experiment, Albot1 traversed the environment as shown in Figure 4.14a. It started with a small loop which caused an enduring map to be computed (Figure 4.14b) and then a large loop back to its starting position. It traversed approximately 63.77m with 87 views and created 68 local maps in its route map (Figure 4.14c). On returning to its starting position, its incoming local map at point 5 caused an overlap with its egocentric map thereby causing it to delete its enduring map and egocentric map. As such, it could not recognize returning to a part of the environment visited earlier (Figure 14.4d).

Table 4.3: Experiment 12 parameters

Information	**Parameters**
1. Distance	63.77m
2. Views Collected	87.
3. Views in Route map	68.

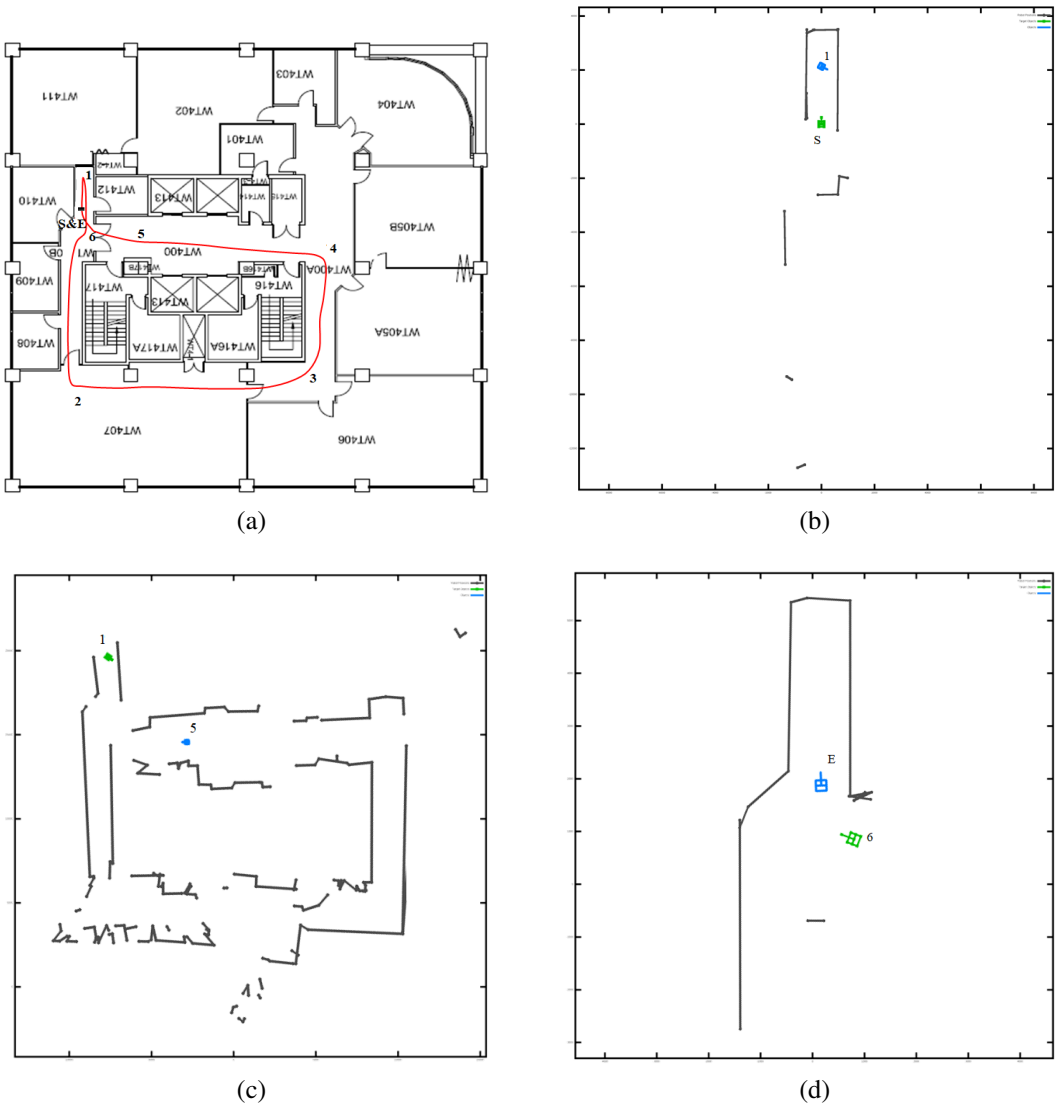


Figure 4.14: (a) The environment and the route traversed, (b) The enduring map computed after turning around at point 1, (c) The egocentric map computed at point 5, and (d) The egocentric map computed at point E.

4.1.4 Summary

A straightforward idea of computing an enduring map is to copy one's egocentric map and use it as an enduring map. Albot1 was empowered with one such algorithm and its behaviour was observed in three experiments. Albot1 successfully utilized its enduring

map to recognise it has returned to a familiar part of the environment. However, it relies on one's turning around and building up a large global map of the environment traversed. If not, and as shown in Experiment 12, it does not recognise returning to a familiar place. Building a large global map is uncharacteristic of cognitive mapping.

Experiments 10-12 successfully show that Albot1 could compute a single enduring map from its route map and egocentric maps using novel algorithms, which holds a good orientation for navigation especially returning home. In the process, Albot1 computes its enduring map by adding regions onto existing map. At the meantime, Albot1 overcomes errors by keep tracking its position with respect to reliable object in the enduring map. Although these experiments show how the transient process interacts with enduring process, but computing a single global map is still an big issue.

4.2 Strategy for reorienting using structural geometry

A roboticist would realise immediately that the previous algorithm could breakdown easily due to sensor errors. Given that one needs to realign one's view to the map and that the map could be expanded and turn into a larger map, such errors could easily distort the map computed. One (the roboticist) could come up with a better algorithm, technically. Similarly, the enduring map computed does not needed to be deleted (see Experiment 12) and one could also devise an algorithm to combine them together. However, doing so would mean that what one is computing is a large map of the environment explored but there is no empirical evidence that a cognitive map is such a map. A better understanding of why an enduring map is computed is needed.

In the earlier approach, turning around triggers the computation of an enduring map. While such an action is indicative that an enduring map is being computed, it is however not the rationale for doing so. What then would cause a species to compute such a map? If computing an enduring map is not about computing a single global metric map, then

what is computed? If we turn to the empirical data, one exciting idea that has been discussed over the past 30 years is the idea of computing the geometric of a place for spatial re-orientation (see review in Chapter 2). This observation is important because it was later tested that many different species utilize the geometry of the room to re-orient themselves too. As such, it could be, like path integration, a basic mechanism used in cognitive mapping. If so, could Albot1 be empowered to do so?

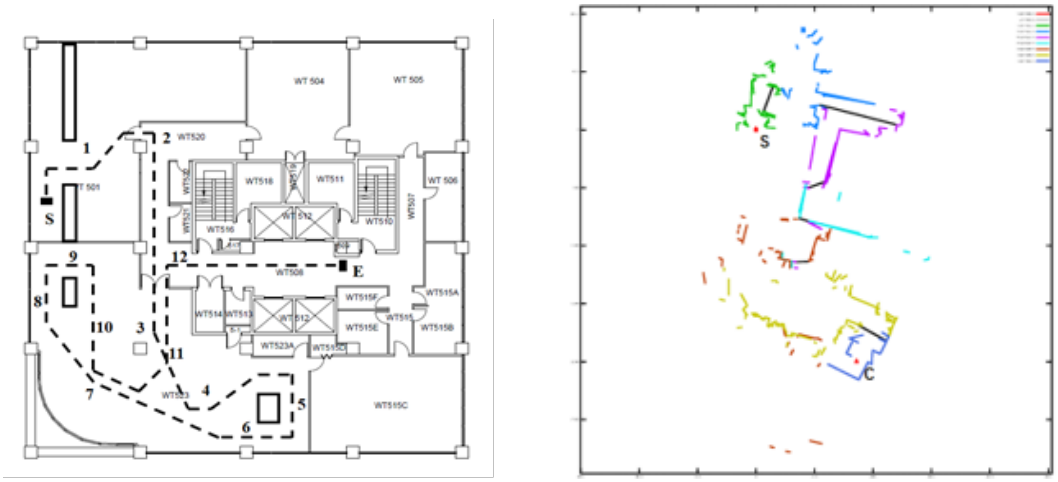
In this section, I studied one such algorithm using an environment and route as shown in Experiment 6 in Chapter 3 (Figure 3.12). This environment is chosen because all earlier attempts to return home in it failed. Then I conduct an experiment using the earlier environment downloaded from the internet (Figure 4.8).

4.2.1 On computing the geometry of a place

To empower Albot1 with an algorithm to utilize the geometry of a place to re-orient itself, one needs to identify a place, compute its geometry and re-orient oneself in it. In empirical experiments on geometry, species were trained mostly to locate food in a room. For Albot1, it will use it to re-orient towards the exits of a place. But, what is a place?

Intuitively, each local map in a route map defines a place visited. However, as noted when using a route map to re-trace home, it is not necessary to consider all local maps in a route map as places visited. This is because some local maps are perceived while still inside a local map (which leads to the computation of a simplified map in Section 4.2). Henceforth, when a local map is identified as a place, one computes its exits immediately and if any of them is crossed, one would have left the place and entered a new one. Like before, local maps computed while in a place would be ignored.

In traversing the environment as shown in Figure 4.15a, Albot1 starts with the first local map at point S. It computes potential exits for it and this local map becomes the



(a) (b)

Figure 4.15: (a) The environment and path as used in Experiment 6 (Figure 3.12a). (b) Seven places (each with a different colour) were computed when Albot1 arrives at point 5 and after computing 37 local maps. Black lines denote exits crossed.

first place visited in its current environment. It then explores the environment, creating several local maps, until it crosses its exit (indicate as a black line in Figure 4.16b). The next local map becomes the second place visited in the environment and its exits are computed. The process is repeated as it continues to explore its environment. At the point where it stops in Figure 4.16b, seven places were computed. But, what is an exit?

At the perceptual level, an exit can be defined as a gap between two surfaces that allows one to move out of a place. Jefferies and Yeap (1999) provided an algorithm for computing such exits in office-like environments. They define it as the shortest edge covering the occluded edges in view. By covering, it is meant that the viewer must cross the exit in order to reach the occluded edge. This algorithm is used to compute the exits for Albot1. Figure 4.16 presents a flowchart for computing places.

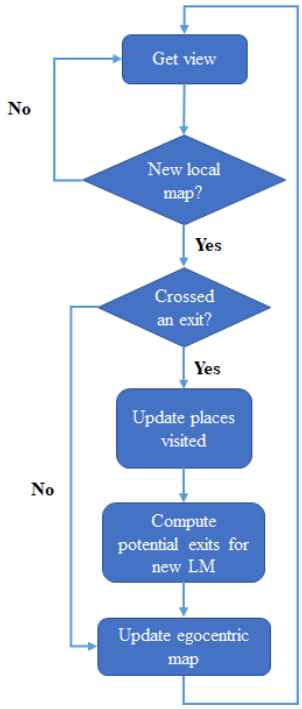


Figure 4.16: A flowchart for computing places visited.

A place is a descriptor: <P#, LMb#, LMe#, exit-crossed> where P# is a number identifying the place, LMb# and LMe#, are the numbers identifying the local maps where the place begins and ends respectively, and exit-crossed map is a pair consisting of the co-ordinates of the exit and the place to which it connects. Figure 4.17 shows how Albot1 computes places as it explores its environment as shown in Figure 4.15. Given that a place and its exists can now be identified in Y&H's mapping process, how could Albot1 compute the geometry of a place and use it to re-orient itself in its environment?

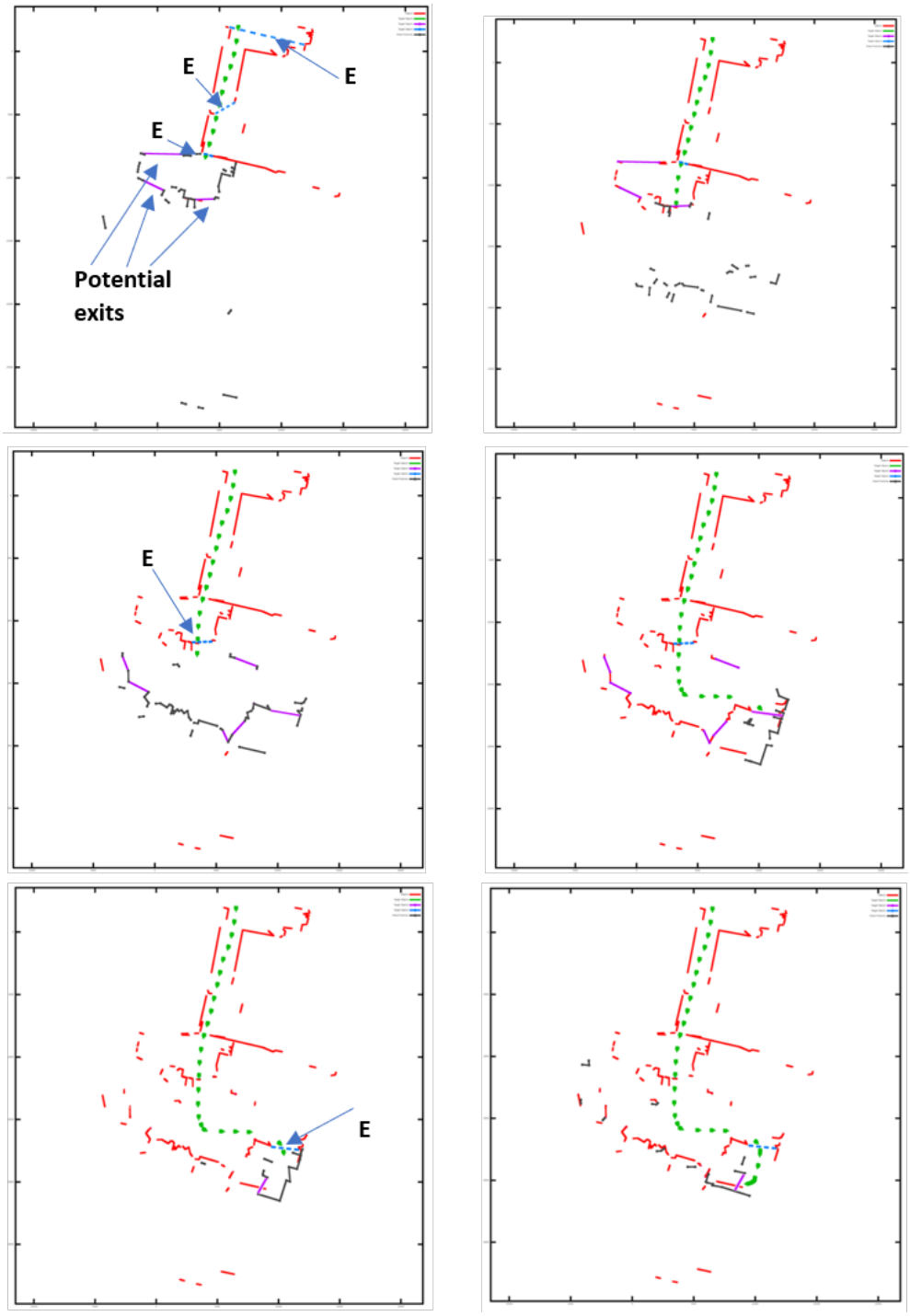
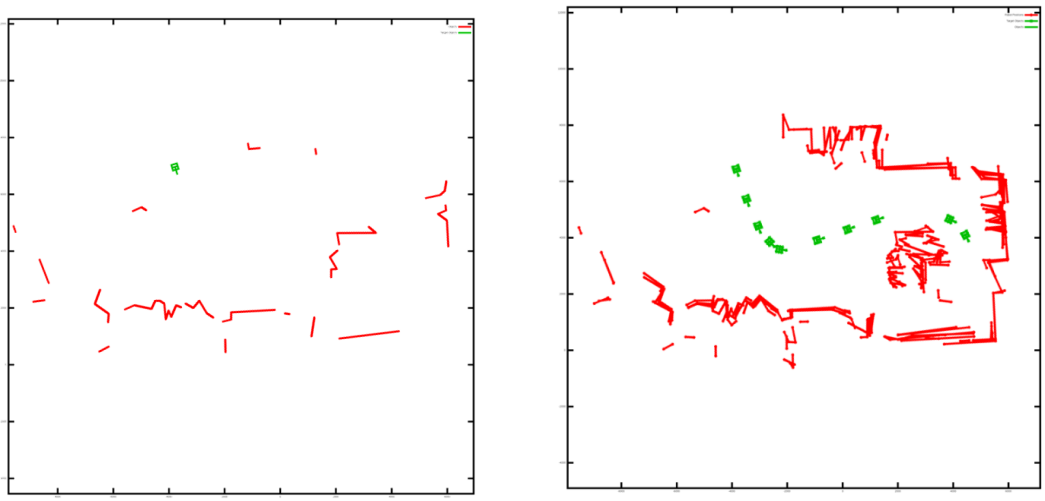


Figure 4.17: Computing places: (Left) Albot1 just crossed an exit (dashed blue line) and thereby identifying the current local map as a place and its potential exits (purple lines) are then computed. (Right) Albot1 moved inside a place, computing local maps as before. Each robot position (green dot) indicates a local map is being computed.

Observe that a critical point in Y&H's mapping process is when its egocentric map is refreshed as a result of an incoming local map overlaying some parts of its egocentric map. Thus, it is argued that it is at this point that computing the geometry of a place might help one to re-orient where one is. In particular, one could compare the geometry of the current place with the previous to check if the two places are indeed the same place. Observe that this is the case when one turns around in the corridor. However, a lesson learned in the previous section is that it is cumbersome to compute the geometry of a room based on its physical shape. Computing the latter turns out to be too detailed and exact and this is problematic given that the egocentric map itself is inexact.

Henceforth, Albot1 is empowered with an algorithm that computes a rectangle as the geometry of a place visited. A rectangle is chosen not just because of its simplicity but also because it provides a spatial reference that corresponds to one's left-right and front-back (ignoring top-bottom). However, a place is first identified by the local map one is in after crossing an exit of the previous place. A single view of a place is inadequate to compute the geometry of a place; the information in it is simply not rich enough (see Figure 4.18a). Henceforth, to compute the geometry of a place, all the local maps computed in a place are displayed as a global map to enable it to compute a better geometric shape of the place (Figure 4.18b).



(a) (b)

Figure 4.18: (a) Albot1 just entered into a new place, from Figure 4.17 (bottom row). (b) overlaying all local maps experienced in this place.

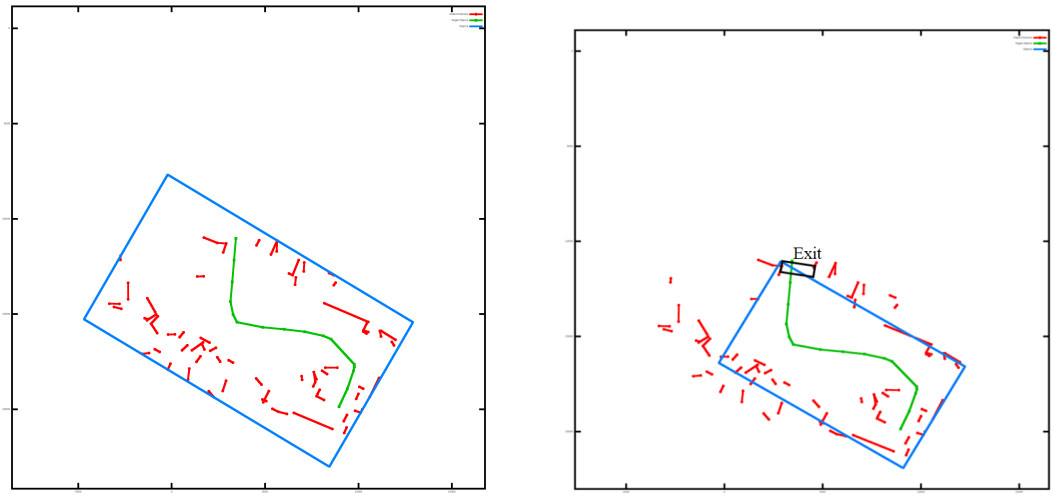
The algorithm for computing the geometry of a place is as shown below. It computes first an outer boundary which is the largest boundary covering the place (Figure 4.19a). It then computes an inner boundary which is the boundary of place (see Figure 4.19b). Note that surfaces deemed to be too small are removed before computing the boundary. The outer boundary gives an indication of the possible extent of the place while the inner boundary gives an indication of the possible extent of the empty space in it. Furthermore, the entrance from which one took to come into this place is also marked on the inner boundary as shown in Figure 4.19b.

Algorithm 7 : Computing geometry of a place

Input: M_{ego} **= {**S_1**,** S_2**, . . . ,** S_n**}; current egocentric map**
Output:*G* = ({S_1**,** S_2**,** S_3**,** S_4**}, {**E_i**}) ;; a four lines plus a list of exits ; . . . *Places* = < LM#, exit-crossed>,**

use LM# to represent local maps stored as places
Until is-refreshed(M_{ego}) == true **do**
    $V_c \leftarrow$ get current view ;; ▷
    $R_c \leftarrow$ get current robot position ;; ▷
    if is-new-local-map(V_c) **then**
        if is-crossed-exit(R_c, R_p, *Exits*) **then**
            Places ← update-places(V_c)
            Exits ← compute-potential-exits(V_c)
            exit-crossed ← update-crossed-exits(R_c, R_p, *Exits*)
        end if
    else
        goto step 1
    end if
    $M_{ego} \leftarrow$ update-egocentric-map(V_c, R_c)
    locomotion execution
    goto step 1
end until
$G_{outer} \leftarrow$ compute-outer-boundary(LM#, exit-crossed) ; ▷ mini bounding box
$G_{inner} \leftarrow$ adjust-and-optimize(G_{outer}) ;;
return G_{inner} and *Places*;

Note that the outer-boundary is computed by taking the minimum and maximum surface points on the x and y axis of the map and creating a line for these four sides by finding the orientation of that line.



(a) (b)

Figure 4.19: Computing the geometry of a place: (a) the outer boundary. (b) the inner boundary. The black rectangle indicates from where Albot1 enters into this place and the green line indicates the path Albot1 took inside this place.

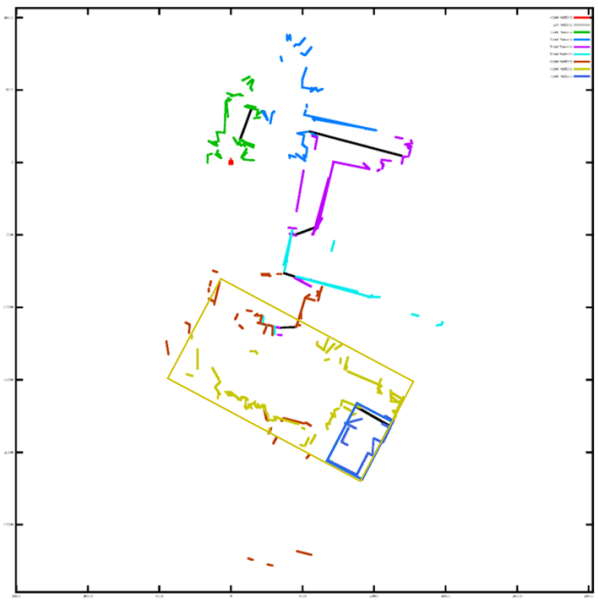
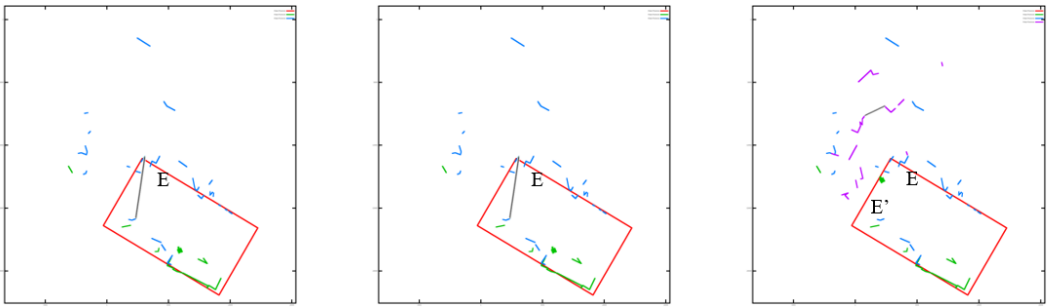


Figure 4.20: Computing the geometry of the last two places (in yellow and blue) allows Albot1 to detect that it is in the same place or not (see Figure 4.16).

When the egocentric map is to be refreshed, the boundary of the current place is computed and compared with the boundary of the previous place. If they overlap significantly, then one is in the same place. In Figure 4.17 (bottom right), when Albot1 explored that place, its egocentric map was refreshed. The boundary of the current place and that of the place just left are computed (Figure 4.20) and found to overlap significantly. Consequently, Albot1 knows that it is in the same place.

Like in previous section, having detected that it is now moving in a familiar place, how does Albot1 utilise its knowledge of the geometry of a place while navigating in its environment? In the previous section, Albot1 tracks its movement in it but this was found to be ineffective. Given that one could track one's position in one's local map, it is both unnecessary and inappropriate to use the geometry of a place to track one's position inside the geometry. Instead, the geometry of the place is overlaid onto the new egocentric map so that (i) one knows that one is in a familiar place and (ii) one knows roughly where lie the exits of this place and to where they lead.

Continue with the journey above where Albot1 has reached the point whereby the egocentric map is refreshed with only the current local map, Figure 4.21a shows Albot1's new egocentric map together with the geometry of the familiar place (red box). Most importantly, Albot1 can now orient towards a known exit in this place (marked E) and know to where it leads.

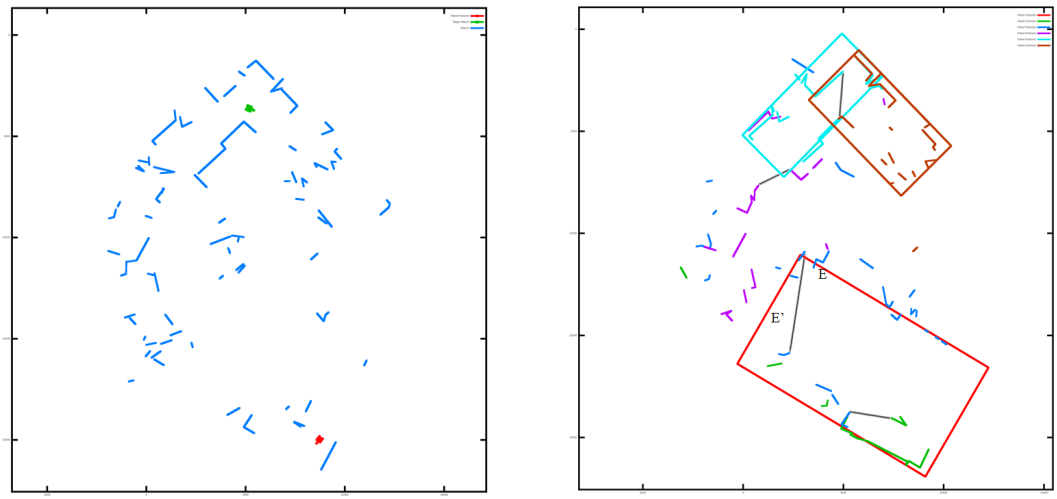


(a) (b) (c)

Figure 4.21: (a) Albot1 refreshes its egocentric map (in green) together with the geometry of the familiar place that it is in (the red rectangle). (b) Albot1 moves into a new place and identify that it is still part of the familiar place that it was in. (c) Albot1 moves into another new local place which is now no longer a part of its familiar place. A new exit, E', now appears on the geometry of the familiar place visited.

As Albot1 continues to explore the environment, it will move in and out of places. For each of them, Albot1 will generate its geometry and check if it is still inside the familiar place or not. Figure 4.21b shows the next place computed which is still part of the place Albot1 visited earlier. Figure 4.21c shows Albot1 moving out of the familiar place and exploring new places. The exit connecting them is marked as E' in Figure 4.21c. It is important to emphasize that the boundary is indicative only. In other words, its exact size and exact positioning of the exits are not important.

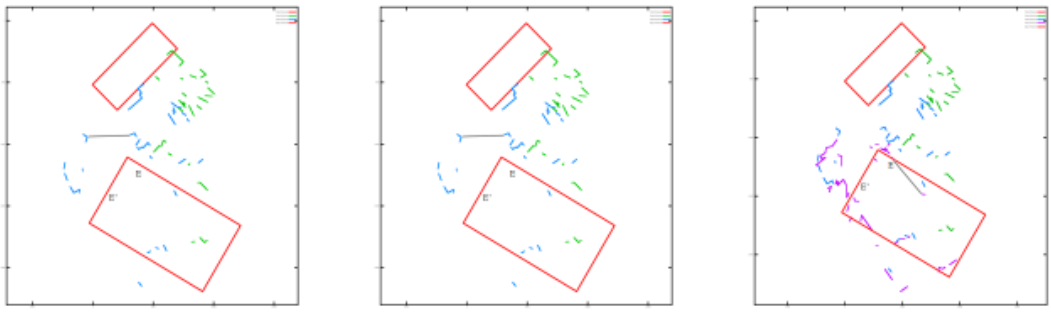
Figure 4.22a shows Albot1 continues exploring the environment and reaches a point where it needs to refresh its egocentric map. Again, it computes the geometry of the last two places visited to check if it is in a familiar place (Figure 4.22b). It is not. It refreshes its egocentric map, deleting everything in it except the current place that it is in and geometries of places it has re-visited (the red rectangle in see Figure 4.23a). The latter allows one to detect if one is re-visiting a familiar part of the environment.



(a) (b)

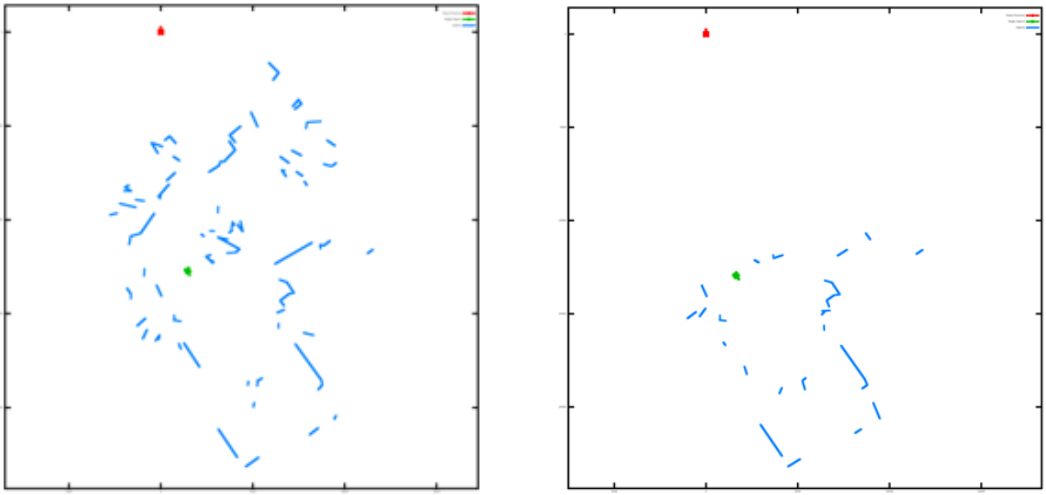
Figure 4.22: (a) Albot1's second egocentric map just prior to the map being refreshed. (b) Albot1's map of places and the geometries of three places visited.

Figure 4.23 shows Albot1 continues exploring its environment and computed three more places. While in the last place (Figure 4.23c), it needs to refresh its egocentric map (Figure 4.24). Again, Albot1 has to check if it is still in the same place by computing the geometry of the last two places visited (Figure 4.25b). It is not but it overlaps with a place visited earlier (Figure 4.25c).



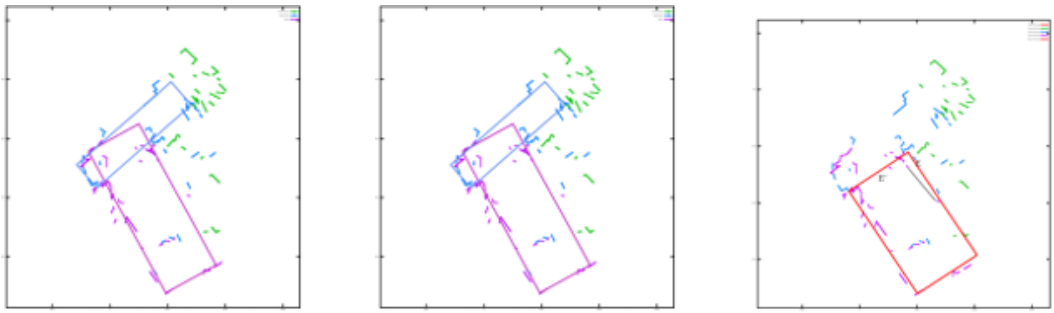
(a) (b) (c)

Figure 4.23: Albot1 continues exploring its environment and created three places, denoted in different colours: (a) green, (b) blue, and (c) purple.



(a) (b)

Figure 4.24: (a) Albot1's third egocentric map. (b) Albot1's refreshed its egocentric map.



(a) (b) (c)

Figure 4.25: (a) Three places were computed in the third egocentric map (in different colours). (b) Geometry of the last two places visited. (c) aligning the geometry of the previous place visited (the red rectangle in Figure 4.22b).

Figure 4.26 shows Albot1 continues exploring its environment. As it crosses the exit for the current place (the black line in Figure 4.26a), it computes the geometry of the new place and checks if it is the same place as before. It is not but this new place contains an exit (E in Figure 4.26a) that leads to a place visited earlier. The geometry of that place is projected on the egocentric map (the purple box in Figure 4.26c). Albot1 then went through its exit and into another place (blue box in Figure 4.26d) eventually arrived at the end of the corridor (Figure 4.27a).

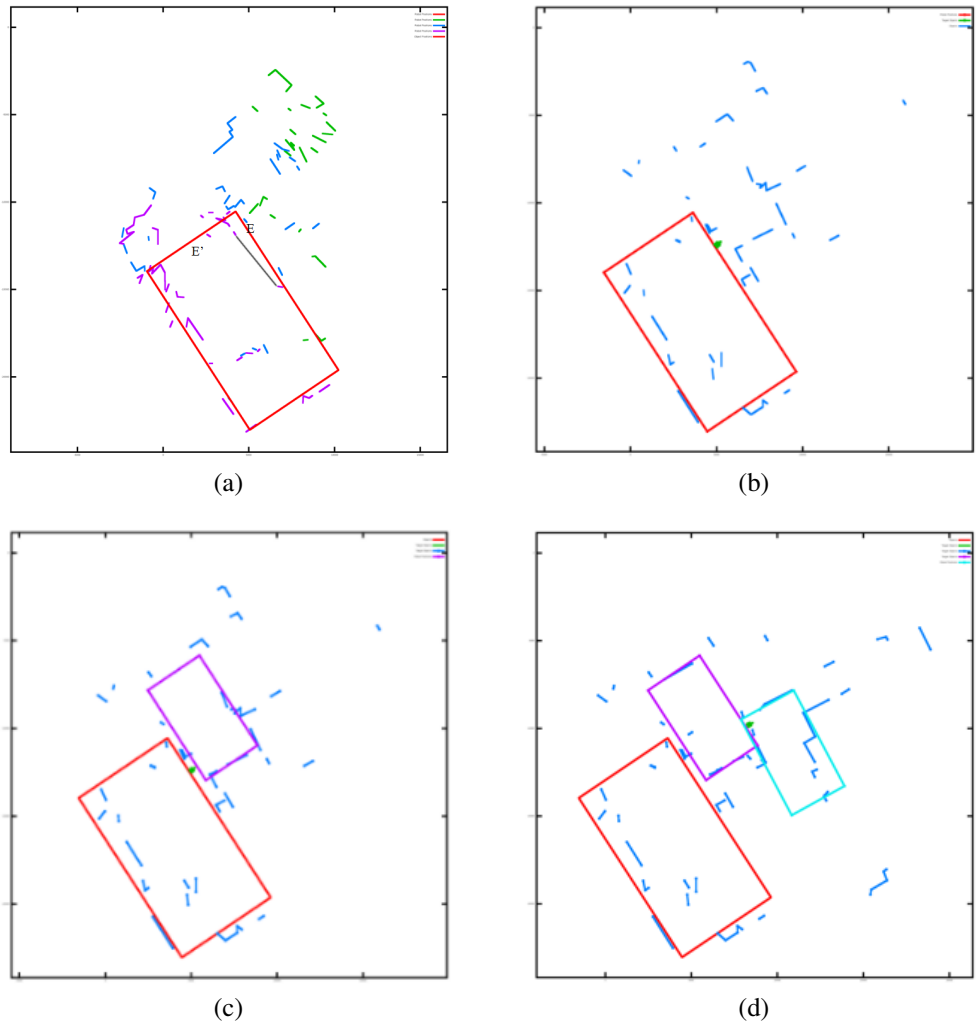
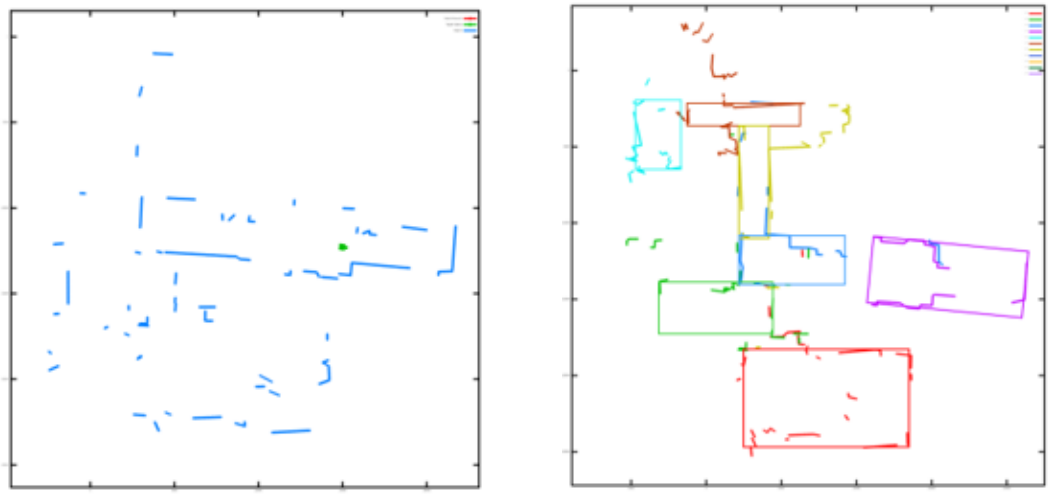


Figure 4.26: Albot1 continues exploring its environment until it reaches the end of the corridor.

Figure 4.27 shows Albot1 turned around and realised that it is in a familiar place (i.e. the corridor). Interestingly, if we were to ask Albot1 to return home, one could envisage an algorithm whereby Albot1finds whether the geometry of its home could be connected to the geometry of the current place, it could return home using short-cuts (Figure 4.27b)!



(a) (b)

Figure 4.27: (a) Albot1's final egocentric map. (b) geometry of all places from home to the places in the current egocentric map

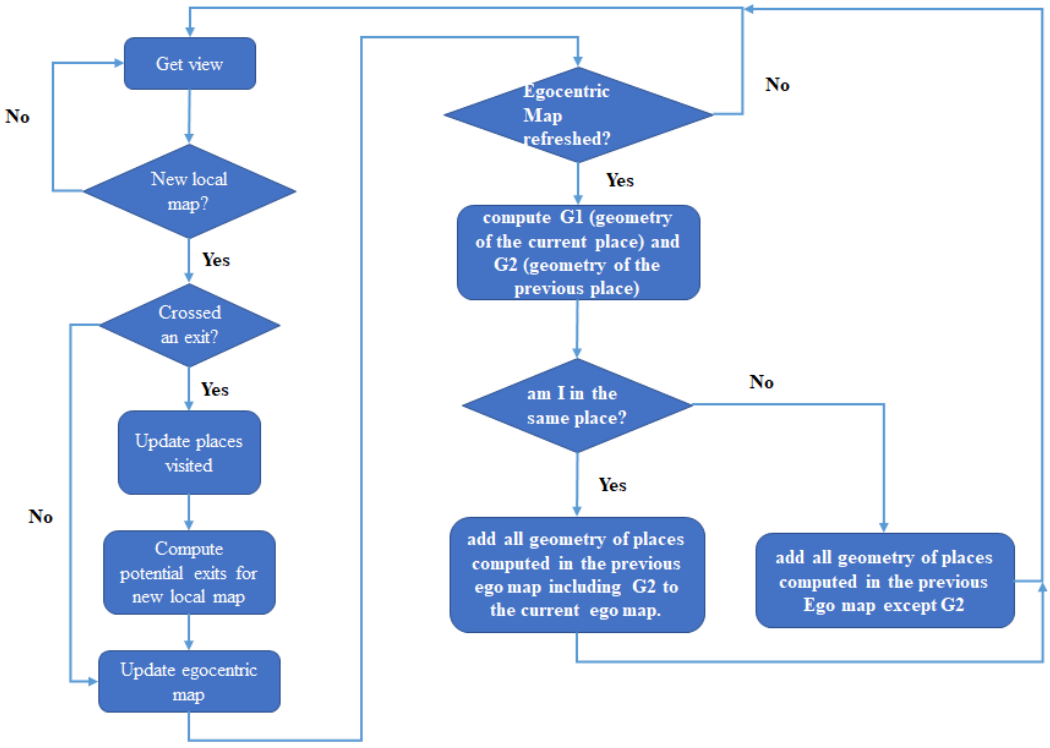


Figure 4.28: shows the flowchart for the new mapping process and algorithm is as shown below.

Algorithm 8 : Computing geometry when exploring

Input: M_{ego} **= {**S_1**,** S_2**, ... ,** S_n**}; current egocentric map; ...** V_c **= {} ; current view; ...** R_c **={} ; current robot position**

Output:*Geometry* = ({*Places#, G#*},M_{ego}**)**

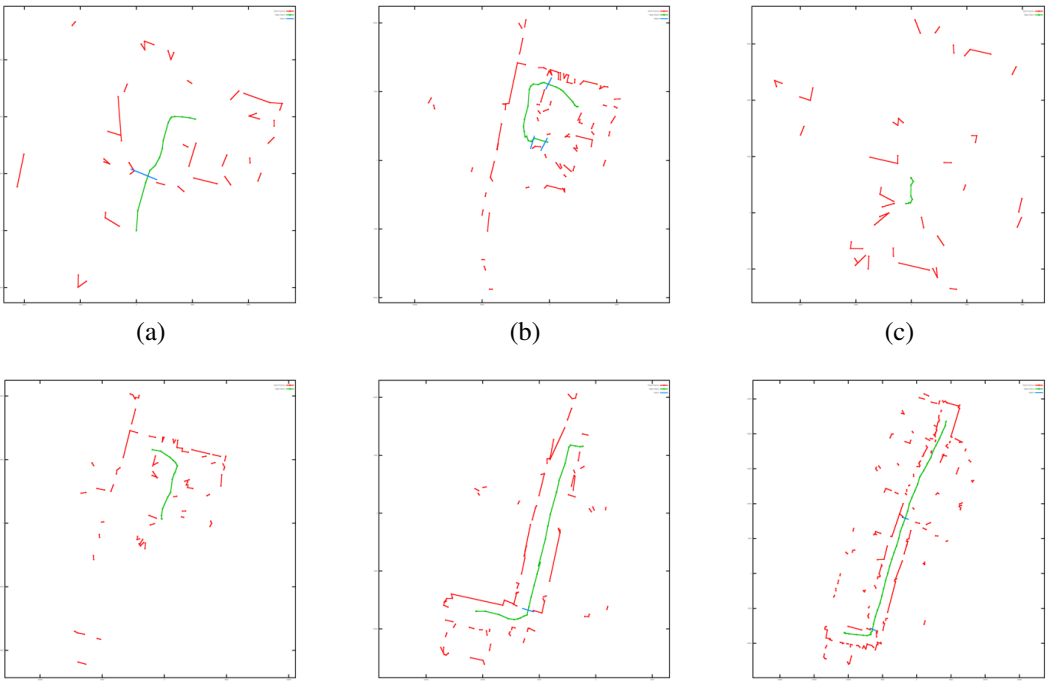
1: Locomotion execution.
2: $V_c \leftarrow$ get current view
3: $R_c \leftarrow$ get current robot position
4: **if** is-new-local-map(V_c) **then**;;
5: **if** is-crossed-exit(R_c, R_p, *Exits*) **then**
6: *Places* $\leftarrow$ update-places(V_c)
7: *Exits* $\leftarrow$ compute-potential-exits(V_c)
8: *exit-crossed* $\leftarrow$ update-crossed-exits(R_c, R_p, *Exits*)
9: **end if**
10: **else**
11: goto step 1
12: **end if**
13: $M_{ego} \leftarrow$ update-egocentric-map(V_c, R_c)
14: **if** is-refreshed(M_{ego}) == true **then**
15: $G_1 \leftarrow$ compute-geometry(*Places[c]*) ;; ▷ geometry of current place
16: $G_2 \leftarrow$ compute- geometry(*Places[p]*) ;; ▷ geometry of previous place
17: **if** is-inside-same-place(G_1, G_2, R_c) **then**
18: **for** all *place* in *Places* **do** ;; ▷
19: *G* $\leftarrow$ compute-geometry(*Places[i]*) ;; ▷ including G_1 and G_2
20: **end for**
21: $M_{ego-new} \leftarrow$ add(*Places#*, *G*)
22: **else**
23: pop(*Places*) ;; ▷ excluding current place
24: **for** all *place* in *Places* **do**
25: *G* $\leftarrow$ compute-geometry(*Places[i]*) ;; ▷ excluding G_2
26: **end for**
27: $M_{ego-previous} \leftarrow M_{ego}$;;
28: $M_{ego-previous} \leftarrow$ add(*Places#*, *G*)
29: end if
30: **end if**
31: goto step 1

4.2.2 Experiment 13

Figure 4.29: The environment and Albot1's path for Experiment 13.

With the successful design and implementation of the necessary algorithms to empower Albot1 to navigate using geometry of places, we repeat the experiment using the downloaded data as shown in Figure 4.8 (reproduced above as Figure 4.29). In this environment, there are two loops: from S → 1 → 2 → 3 → 4 and 4 → 5 → 6. When exploring this environment, Albot1's egocentric map is refreshed six times (Figure 4.30).

Using the geometry of places, Albot1 is able to recognise all familiar places revisited. The key steps are shown in Figure 4.31. Figure 4.31a shows the geometry of the last two places computed in the first egocentric map when Albot1 makes a turn (Figure 4.30a) and it shows that Albot1 is in the same place. Figure 4.31b shows Albot1 refreshes its egocentric map and continues to explore its environment. Figure 4.31c shows Albot1 reaches the point just before it needs to refresh its egocentric map again. The geometry of the last two places computed shows that Albot1 is in a different place. However, it detects that it's geometry overlaps with the geometry of a place in the first egocentric map. Figure 4.31d and 4.31e show that Albot1 detects that it is moving in familiar places. Figure 4.31f shows Albot1 moving down the corridor and into the large room at point 5 of the journey. Here, Albot1 turns around and detects that it is in the same place (Figure 4.31g). It then continues the path back to the top.



(a) (b) (c)

(d) (e) (f)

Figure 4.30: Albot1 computed 6 different egocentric maps for the environment as shown in Figure 4.29. Green lines indicate the traversed paths in each of them.

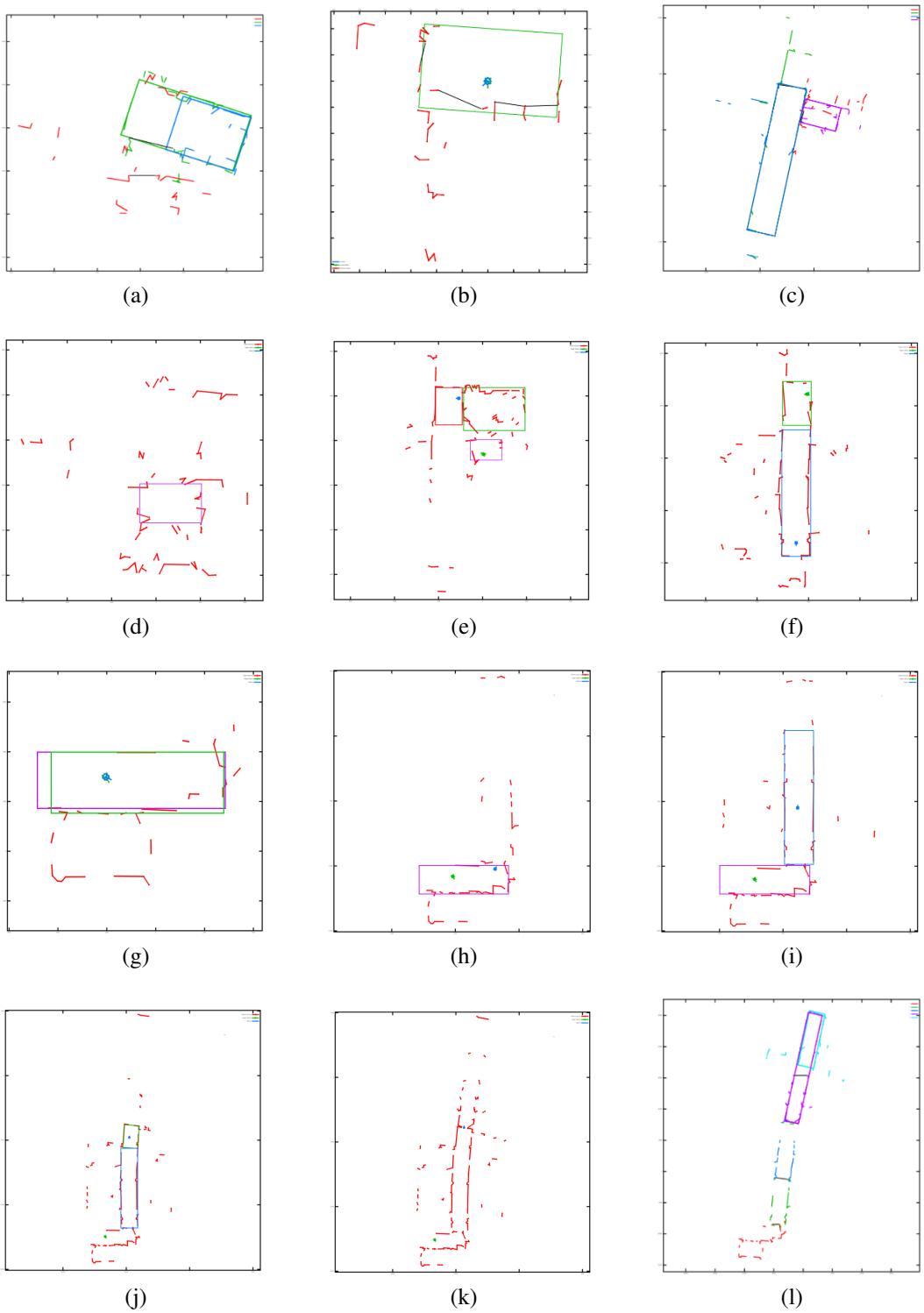


Figure 4.31: Results of exploring the environment in Figure 4.29.

4.2.3 Summary

The idea of computing the geometry of a place to re-orient in it has strong empirical support. Using an earlier environment with a complex path, one such algorithm is developed. The result shows that Albot1 could potentially generate a short-cut to return home. The algorithm is further tested using another large environment and found to perform satisfactory.

Algorithm 7 and 8 significantly explain how local geometry can be computed from the space. As travelling inside the space, one can identify the local space rather than the whole global environment, then it can abstract geometric representation from the local space to achieve spatial task - reorienting, which is very high level spatial reasoning ability. In experiment 13, Albot1 computes a list of local geometries and it is able to use those computed geometries to point a position especially an earlier computed geometry. In other word, this gives Albot1 a sense of where the visited spaces are and once it re-visits them, it will recall the corresponding geometry to help it reorient in it.

4.3 Conclusion

In this chapter, two different approaches to empower Albot1 to compute a more enduring map were investigated. The first approach makes a copy of the egocentric map prior to it being refreshed and the second computes places and their geometry.

Empowering Albot1 with these algorithms lead to questioning the need for an enduring map in cognitive mapping. In the first approach, saving a copy of the egocentric map means that one would use its structural description for, say, as a basis for developing a much richer representation and for performing a variety of tasks in it. For example, one could continuously update it thereby turning it from being a view-dependent representation to a view-independent representation. One could remember the events

that have occurred, the significant actions one might have performed and what interesting objects were in it. With such a rich detail, one would recognize the place easier. However, doing so amount to computing a detailed description of a place and such a process is not straightforward. If one could do so, why not compute such a rich description for the entire environment? This approach would thus lead to a very different approach to computing a cognitive map.

In the second approach, computing the geometry of a place is a straightforward process. It is an abstract representation that makes explicit roughly the whereabouts of the exits of a place. Henceforth, it does not require one to attend to much of the details in the environment. Yet, it provides an elegant solution to the problem identified in Y&H's model of cognitive mapping; it is a straightforward approach and it has strong empirical support.

Chapter 5

Conclusion

Y&H developed an interesting model of cognitive mapping that shows what kind of an inexact and incomplete global map of one's environment can be computed and why. Despite being an interesting cognitive model, it has one shortcoming: its global map does not allow one to realize that it is exploring a familiar part of the environment. This thesis explores possible solutions by empowering Albot1 to use its cognitive map to navigate in its environment. The significance of its findings from the theory standpoint are as follow:

1. The route map is a versatile representation and can be used to generate more than an egocentric map that supports orienting to places just visited. In this study, I have shown that one could use it to generate a trace map and a place map.

2. A trace map can be used to support, albeit in a limited way, finding one's way home and possibly using short-cuts. It thus provides a solution for species who does not need to wander far away from home.

3. A local map could be identified as a bounded place with exits. At this level, it is unnecessary to compute a detailed description of a place. Instead,

the geometry of a place is computed and this enables one to re-orient in it towards exits and food.

4. A place map is a global map of connected places that one has visited. It can be used to identify if two places are in the same physical space and to detect if one is returning to a familiar place or if short-cuts are available.

Based upon the above findings, I have successfully extended Y&H's model of cognitive mapping to overcome its major shortcoming. The extension is considered sound for two reasons. First, it incorporates into the model a mechanism, geometry of a place, that is fundamental to many species. Second, the extension is based on what is computed earlier in the process. Figure 5.1 shows the extended model.

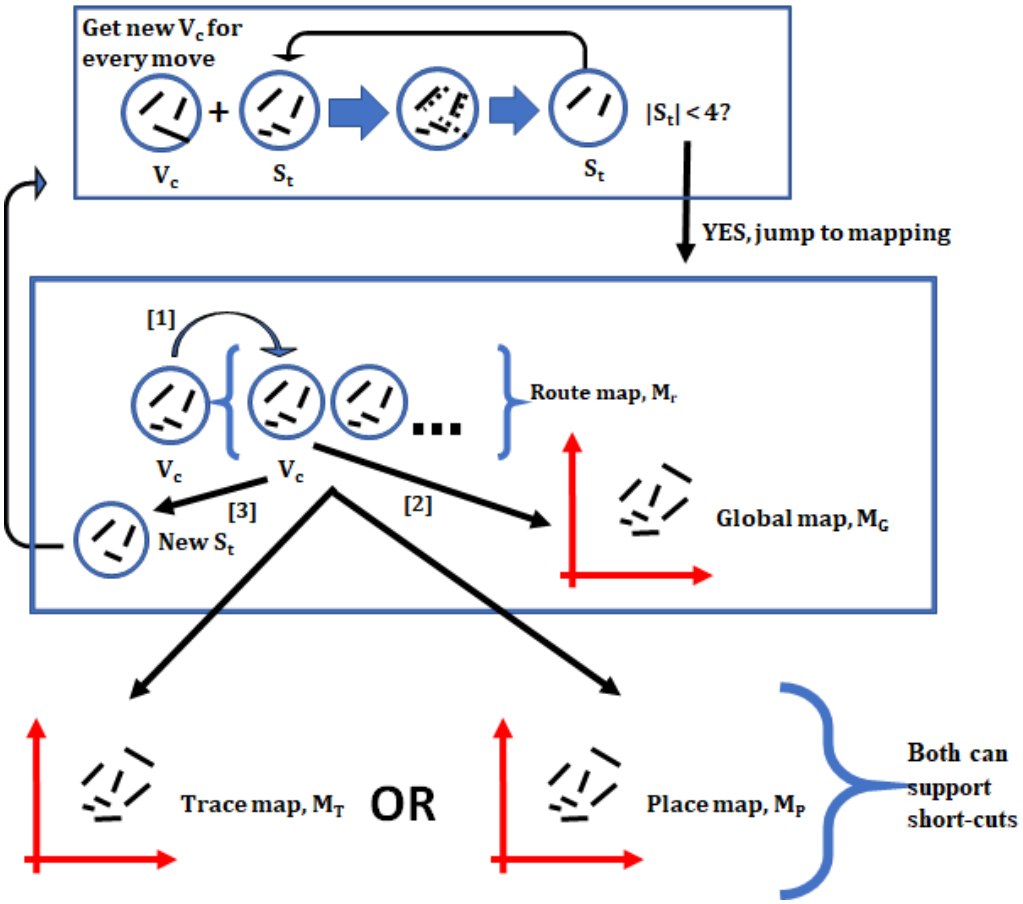


Figure 5.1: Y&H's extended model of cognitive mapping. (top) The perceptual process that delivers a view and tracks surfaces in it. (middle) Y&H's mapping process that computes both a route map and a global egocentric map. (bottom) Extension of the model to compute either a trace map or a place map. Top and middle figures reproduced from Yeap and Hossain, Figure 4.

5.1 Limitations and Future Work

While a solution to the key problem has been found, existing work has three significant limitations from an implementation standpoint. First, more experiments need to be conducted with the final model proposed and in particular, showing how Albot1 uses the geometry of a place to navigate in its environment. The latter would allow a more in-depth study of the interactions between the different key representations in the model

and this would require testing it using several different environments. Second, the use of a rectangular shape of a place is an over simplification although it suffices for the office-like environments used in the current study. Again, testing the model in different environments that require different geometries of a place to be computed could shed light on what kind of geometry is best. Third, given that the model is now complete (in the sense that it could now explore an environment and find its way home), it is timely to test the model with an Albot equipped with different sensors (e.g. vision) or embodiments (e.g. drones).

From a cognitive standpoint, the next step is to unravel the mystery of landmarks use in cognitive mapping and idea of returning home using short-cuts. While the latter has been demonstrated to be possible using the model developed here, a more thorough study of this problem in cognitive mapping of natural species is needed before one could shed light to the mysterious nature of this problem.

References

Benhamou, S. (1996). No evidence for cognitive mapping in rats. *Animal Behaviour*, *52*(1), 201–212.

Bennett, A. T. (1996). Do animals have cognitive maps? *Journal of Experimental Biology*, *199*(1), 219–224.

Ben-Yehoshua, D., Yaski, O. & Eilam, D. (2011). Spatial behavior: the impact of global and local geometry. *Animal cognition*, *14*(3), 341–350.

Brom, C., Vyhnánek, J., Lukavskỳ, J., Waller, D. & Kadlec, R. (2012). A computational model of the allocentric and egocentric spatial memory by means of virtual agents, or how simple virtual agents can help to build complex computational models. *Cognitive Systems Research*, *17*, 1–24.

Brunec, I. K., Moscovitch, M. & Barense, M. D. (2018). Boundaries shape cognitive representations of spaces and events. *Trends in Cognitive Sciences*, *22*(7), 637–650.

Buckley, M. G., Holden, L. J., Spicer, S. G., Smith, A. D. & Haselgrove, M. (2019). Crossing boundaries: Global reorientation following transfer from the inside to the outside of an arena. *Journal of Experimental Psychology: Animal Learning and Cognition*.

Buckley, M. G., Smith, A. D. & Haselgrove, M. (2019). Thinking outside of the box ii: Disrupting the cognitive map. *Cognitive psychology*, *108*, 22–41.

Cheng, K. (2008). Whither geometry? troubles of the geometric module. *Trends in*

cognitive sciences, *12*(9), 355–361.

Cheng, K. & Newcombe, N. S. (2005). Is there a geometric module for spatial orientation? squaring theory and evidence. *Psychonomic bulletin & review*, *12*(1), 1–23.

Chown, E. & Yeap, W. K. (2015). Cognitive robotics. In *International workshop on multi-disciplinary trends in artificial intelligence* (pp. 294–305).

Descartes, R. (2001). *Discourse on method, optics, geometry, and meteorology*. Hackett Publishing.

Dillon, M. R. & Spelke, E. S. (2015). Core geometry in perspective. *Developmental science*, *18*(6), 894–908.

Duffy, S., Huttenlocher, J. & Levine, S. (2005). It is all relative: How young children encode extent. *Journal of Cognition and Development*, *6*(1), 51–63.

Gibson, B. M., Leichtman, M. D., Kung, D. A. & Simpson, M. J. (2007). Use of landmark features and geometry by children and adults during a two-dimensional search task. *Learning and Motivation*, *38*(1), 89–102.

Gouteux, S., Thinus-Blanc, C. & Vauclair, J. (2001). Rhesus monkeys use geometric and nongeometric information during a reorientation task. *Journal of Experimental Psychology: General*, *130*(3), 505.

Hardt, O., Hupbach, A. & Nadel, L. (2009). Factors moderating blocking in human place learning: The role of task instructions. *Learning & Behavior*, *37*(1), 42–59.

Hartley, T. & Burgess, N. (2005). Complementary memory systems: competition, cooperation and compensation. *Trends in Neurosciences*, *28*(4), 169–170.

Hartley, T., Lever, C., Burgess, N. & O'Keefe, J. (2014). Space in the brain: how the hippocampal formation supports spatial cognition. *Philosophical Transactions of the Royal Society B: Biological Sciences*, *369*(1635), 20120510.

Hegarty, M., Montello, D. R., Richardson, A. E., Ishikawa, T. & Lovelace, K. (2006).

Spatial abilities at different scales: Individual differences in aptitude-test performance and spatial-layout learning. *Intelligence*, *34*(2), 151–176.

Hermer, L. & Spelke, E. S. (1994). A geometric process for spatial reorientation in young children. *Nature*, *370*(6484), 57–59.

Hermer-Vazquez, L., Moffet, A. & Munkholm, P. (2001). Language, space, and the development of cognitive flexibility in humans: The case of two spatial memory tasks. *Cognition*, *79*(3), 263–299.

Hermer-Vazquez, L., Spelke, E. S. & Katsnelson, A. S. (1999). Sources of flexibility in human cognition: Dual-task studies of space and language. *Cognitive psychology*, *39*(1), 3–36.

Howard, A. & Roy, N. (2003). Radish: The robotics data set repository. *URL http://radish. sourceforge. net*.

Hupbach, A. & Nadel, L. (2005). Reorientation in a rhombic environment: No evidence for an encapsulated geometric module. *Cognitive Development*, *20*(2), 279–302.

Huttenlocher, J., Hedges, L. V. & Duncan, S. (1991). Categories and particulars: Prototype effects in estimating spatial location. *Psychological review*, *98*(3), 352.

Huttenlocher, J. & Lourenco, S. F. (2007). Coding location in enclosed spaces: is geometry the principle? *Developmental Science*, *10*(6), 741–746.

Keinath, A. T., Julian, J. B., Epstein, R. A. & Muzzio, I. A. (2017). Environmental geometry aligns the hippocampal map during spatial reorientation. *Current Biology*, *27*(3), 309–317.

Kelly, D. M. & Bischof, W. F. (2005). Reorienting in images of a three-dimensional environment. *Journal of Experimental Psychology: Human Perception and Performance*, *31*(6), 1391.

Kelly, D. M., Spetch, M. L. & Heth, C. D. (1998). Pigeons'(columba livia) encoding of geometric and featural properties of a spatial environment. *Journal of Comparative Psychology*, *112*(3), 259.

Kelly, J. W., McNamara, T. P., Bodenheimer, B., Carr, T. H. & Rieser, J. J. (2008). The shape of human navigation: How environmental geometry is used in maintenance of spatial orientation. *Cognition*, *109*(2), 281–286.

Knight, R., Hayman, R., Ginzberg, L. L. & Jeffery, K. (2011). Geometric cues influence head direction cells only weakly in nondisoriented rats. *Journal of Neuroscience*, *31*(44), 15681–15692.

Learmonth, A. E., Nadel, L. & Newcombe, N. S. (2002). Children's use of landmarks: Implications for modularity theory. *Psychological Science*, *13*(4), 337–341.

Learmonth, A. E., Newcombe, N. S. & Huttenlocher, J. (2001). Toddlers' use of metric information and landmarks to reorient. *Journal of experimental child psychology*, *80*(3), 225–244.

Lee, S. A., Sovrano, V. A. & Spelke, E. S. (2012). Navigation as a source of geometric knowledge: Young children's use of length, angle, distance, and direction in a reorientation task. *Cognition*, *123*(1), 144–161.

Lee, S. A. & Spelke, E. S. (2008). Children's use of geometry for reorientation. *Developmental science*, *11*(5), 743–749.

Lee, S. A., Vallortigara, G., Flore, M., Spelke, E. S. & Sovrano, V. A. (2013). Navigation by environmental geometry: the use of zebrafish as a model. *Journal of Experimental Biology*, *216*(19), 3693–3699.

Lew, A. R., Gibbons, B., Murphy, C. & Gavin Bremner, J. (2010). Use of geometry for spatial reorientation in children applies only to symmetric spaces. *Developmental Science*, *13*(3), 490–498.

Lourenco, S. F., Addy, D. & Huttenlocher, J. (2009). Location representation in enclosed spaces: What types of information afford young children an advantage? *Journal of Experimental Child Psychology*, *104*(3), 313–325.

Lynch, K. (1960). *The image of the city* (Vol. 11). MIT press.

Mackintosh, N. J. (2002). Do not ask whether they have a cognitive map, but how they

find their way about. *Psicológica*, *23*(1).

Margules, J. & Gallistel, C. (1988). Heading in the rat: Determination by environmental shape. *Animal Learning & Behavior*, *16*(4), 404–410.

McNamara, T. P. (2002). How are the locations of objects in the environment represented in memory? In *International conference on spatial cognition* (pp. 174–191).

Miller, N. & Shettleworth, S. (2007). An associative model of geometry learning. *Journal of Experimental Psychology: Animal Behavior Processes*, *33*, 191–212.

Mou, W., McNamara, T. P., Valiquette, C. M. & Rump, B. (2004). Allocentric and egocentric updating of spatial memories. *Journal of experimental psychology: Learning, Memory, and Cognition*, *30*(1), 142.

O'keefe, J. & Nadel, L. (1978). *The hippocampus as a cognitive map*. Oxford: Clarendon Press.

Pearce, J. M., Ward-Robinson, J., Good, M., Fussell, C. & Aydin, A. (2001). Influence of a beacon on spatial learning based on the shape of the test environment. *Journal of Experimental Psychology: Animal Behavior Processes*, *27*(4), 329.

Pecchia, T. & Vallortigara, G. (2010). View-based strategy for reorientation by geometry. *Journal of Experimental Biology*, *213*(17), 2987–2996.

Ponticorvo, M. & Miglino, O. (2010). Encoding geometric and non-geometric information: a study with evolved agents. *Animal Cognition*, *13*(1), 157.

Sotelo, M. I., Bingman, V. P. & Muzio, R. N. (2015). Goal orientation by geometric and feature cues: spatial learning in the terrestrial toad rhinella arenarum. *Animal Cognition*, *18*(1), 315–323.

Sovrano, V. A., Bisazza, A. & Vallortigara, G. (2002). Modularity and spatial reorientation in a simple mind: Encoding of geometric and nongeometric properties of a spatial environment by fish. *Cognition*, *85*(2), B51–B59.

Sovrano, V. A., Bisazza, A. & Vallortigara, G. (2007). How fish do geometry in large and in small spaces. *Animal cognition*, *10*(1), 47–54.

Spelke, E., Lee, S. A. & Izard, V. (2010). Beyond core knowledge: Natural geometry. *Cognitive science*, *34*(5), 863–884.

Sutton, J. E. (2009). What is geometric information and how do animals use it? *Behavioural Processes*, *80*(3), 339–343.

Taube, J. S. (2007). The head direction signal: origins and sensory-motor integration. *Annu. Rev. Neurosci.*, *30*, 181–207.

Twyman, A. D., Newcombe, N. S. & Gould, T. J. (2009). Of mice (mus musculus) and toddlers (homo sapiens): evidence for species-general spatial reorientation. *Journal of Comparative Psychology*, *123*(3), 342.

Vallortigara, G., Pagni, P. & Sovrano, V. A. (2004). Separate geometric and non-geometric modules for spatial reorientation: evidence from a lopsided animal brain. *Journal of Cognitive Neuroscience*, *16*(3), 390–400.

Vallortigara, G., Zanforlin, M. & Pasti, G. (1990). Geometric modules in animals' spatial representations: a test with chicks (gallus gallus domesticus). *Journal of Comparative Psychology*, *104*(3), 248.

Wang, R. F. & Spelke, E. S. (2000). Updating egocentric representations in human navigation. *Cognition*, *77*(3), 215–250.

Wang, R. F. & Spelke, E. S. (2002). Human spatial representation: Insights from animals. *Trends in cognitive sciences*, *6*(9), 376–382.

Wystrach, A. & Beugnon, G. (2009). Ants learn geometry and features. *Current Biology*, *19*(1), 61–66.

Yaski, O. & Eilam, D. (2008). How do global and local geometries shape exploratory behavior in rats? *Behavioural brain research*, *187*(2), 334–342.

Yaski, O., Portugali, J. & Eilam, D. (2011). City rats: insight from rat spatial behavior into human cognition in urban environments. *Animal cognition*, *14*(5), 655–663.

Yeap, W. K. (2011a). A computational theory of human perceptual mapping..

Yeap, W. K. (2011b). How albot0 finds its way home: A novel approach to cognitive

mapping using robots. *Topics in cognitive science*, *3*(4), 707–721.

Yeap, W. K. (2014). On egocentric and allocentric maps. In *International conference on spatial cognition* (pp. 62–75).

Yeap, W. K. & Hossain, M. (2019). What is a cognitive map? unravelling its mystery using robots. *Cognitive processing*, *20*(2), 203–225.

Yeap, W. K. & Jefferies, M. E. (1999). Computing a representation of the local environment. *Artificial Intelligence*, *107*(2), 265–301.

Zeng, T. & Si, B. (2017). Cognitive mapping based on conjunctive representations of space and movement. *Frontiers in neurorobotics*, *11*, 61.