A Semantic SLAM Model for Autonomous Mobile Robots using Content Based Image Retrieval Techniques

Tan Choon Ling

Submitted in fulfillment of the requirements for the degree of Doctor of Philosophy in Information Technology

2011

School of Information Technology

Monash University

Acknowledgements

Firstly, I would like to thank my supervisors, Dr. Simon Egerton and Dr. Velappa Ganapathy, for their help and guidance through the entire period of my doctoral studies. I would not have been able to complete this thesis without the support both of you have given me over the years.

I would also like to thank my girlfriend, Kim, and my parents for their patience, support and understanding throughout the many days and nights I have spent working on my research. I promise I'll find a real job after this.

Many thanks and appreciation goes out to Wan Fariza Fauzi, Bhawani Selvaretnam, Anushia Inthiran, Radi Jarrar, Marc Cheong, and David Hong. You've travelled a long and treacherous road through academia with me, and my mental condition would be questionable at best without all the lunchtime breaks and gossip seasons.

To the school administration staff, Serena Liew, Jowey Lim, Siri, Vernessa, Shirlene Wong, and Sock Wee. Thank you for putting up with all the unanswered e-mails.

Finally, I also must mention my dear friends, Terry Saw, Fernanda Faria, Kennedy Michael, Gracie Low, Morna Sulaiman, Fernanda Andrade, Sandie Lee,

Zahir Danial, Grace Tiou, and Teresa Chian. Thank you for reminding me that there's actually a life worth living outside of academia.

Copyright Notices

Notice 1

Notice 2

Publication List

The research within this thesis has also appeared in the following publications:

1. Tan, C.L., Egerton, S. and Ganapathy, V. (2010), "A Semantic SLAM Model for Autonomous Mobile Robots using Content Based Image Retrieval Techniques: A Performance Analysis" in Australian Journal of Intelligent Information Processing Systems (AJIIPS), pp. 30 – 35.

2. Tan, C.L., Egerton, S.J. and Ganapathy, V. (2009), " A Semantic SLAM Model for Autonomous Mobile Robots using Content Based Image Retrieval Techniques", 16th International Conference on neural information processing (ICONIP), 1-5 December, Bangkok, Thailand.

3. Tan, C.L. & Egerton, S. & Ganapathy, V. (2009), "A New Approach to SLAM for Autonomous Mobile Robots Using CBIR Techniques", Proceedings of the 1st Malaysian Joint Conference on Artificial Intelligence (MJCAI), 14–16 July 2009, Kuala Lumpur, Malaysia.

Abstract

Localization and environmental mapping are two fundamental functions for an autonomous mobile robot. This thesis develops a new framework that allows a robot with a vision sensor to simultaneously achieve both of these functions. The novel approach attempts to interpret video images for their meaning, generating a map and localization data from these meanings. The experiments show promising results for this new approach.

If robots are to perform robustly with no *a priori* knowledge of their environment then they must have the ability to perform Simultaneous Localization and Mapping (SLAM), whereby the robot incrementally builds a map of the environment it is navigating while simultaneously keeping track of its location within the built map. SLAM might be considered a solved problem. There is certainly a large body of literature, discussed in this thesis, which provides decent models for building robust solutions. However, most, if not all, of the current state of the art SLAM techniques rely on solutions with a tight loop of detecting and tracking low-level features to update the robots current pose, or location. We argue these methods are brittle and do not offer general purpose solutions to the problem. This thesis takes a cognitive approach to the subject and develops a new SLAM model based on extracting semantic information from the robot's sensor data.

In this thesis, we develop a new SLAM framework which analyses video streams for semantic content. We do this with inspiration from the Content Based Image Retrieval (CBIR) research area. We use the well-established Tamura texture features to decompose the video stream into a grid of lexemes (or recognized categories) which we then use to construct grammatical sentences. These sentences form place descriptions and are used for constructing the environmental map and localization. In contrast to engineered methods, our framework does not return precise location information as we argue it is enough to know roughly where one is located.

We have implemented a proof of concept model and tested within both indoor and outdoor environments. The results show that our model can construct useful semantic descriptions from the video stream and use these descriptions to implement SLAM. Although the derived semantic descriptions are fairly coarse (based on the limitations of the Tamura texture features), the technique could be refined by adopting a richer set of the feature vectors, however, we leave this as future work.

Table of Contents

Table of Figures

Chapter 1

Introduction

Simultaneous Localization And Mapping (SLAM) is the problem of determining if an autonomous mobile vehicle or robot - equipped with one or more low-level sensors - is able to start in an unknown area of an unexplored environment, and then proceed to incrementally build a map of the current environment while simultaneously using this information to accurately deduce its location in relation to other landmarks within the environment (Dissanayake *et al.*, 2001).

The SLAM problem has its roots from the 1970's, where mobile robots on the moon and Mars were required to autonomously formulate their own methods of navigating around hostile environments (Moravec, 1980). During this time period, the operation of SLAM as applied to mobile robots was useful in implementing obstacle avoidance and path planning.

The direct origin of SLAM comes from (Smith, Self & Cheeseman, 1990) where the issue of representing spatial information during the application of robotics is brought up. However, the term SLAM was not actually used in this paper (which preferred to favor the term *stochastic maps*, instead), and was initially coined by (Leonard & Durrant-Whyte, 1991; Leonard & Durrant-Whyte, 1992)

Leonard & Durrant-Whyte summarizes the problem of SLAM into answering three main questions: “where am I?”, “where am I going?” and “how should I get there?”. The first question is related to *localization*: determining what location in the current environment one is, given what one can currently see and what information one has been given previously. The second and third questions revolve around setting a goal and planning a path to achieve that goal, which are related to *navigation* and *mapping*.

The SLAM problem is considered to be hard due to three issues that are yet to be solved: uncertainty, scaling, and the relationship between localization and mapping. Uncertainty is caused due to the fact that in the area of robotics, observations made within an environment by external sensors can be imprecise (for example, due to signal noise or features simply being out of sensor range) (Goldman, 1994) (Stentz, 1994). Furthermore, traversing the environment itself can also be imprecise (due to issues like wheel slippages), which further compounds the issue of uncertainty in an already uncertain pose when a small error in localization can be compounded over time (by internal sensors that measure the current location via wheel position) and thus, create wholly inaccurate maps (Sims, 2004). While some research has been made in order to make more precise estimates, this issue is still yet to be considered trivial.

The second issue, scaling, involves the amount of computational power required to perform SLAM, which grows with the number of environmental features mapped. This is clearly a huge issue, considering that a mobile robot has a limited amount of resources available at hand.

The third issue pertaining to SLAM, which is the relationship between localization and mapping, is related to uncertainty. (Siegwart & Nourbakhsh, 2004) states that if a mobile robot performs the task of localization based on uncertain observations, the estimate of its current pose will itself be uncertain. In the same vein, maps that are created will become inaccurate if the process of mapping is based on an uncertain robot pose. This causes SLAM to be akin to that of the chicken-and-egg problem: in order to accurate estimate its location, a robot needs to accurately estimate surrounding landmarks, and in order to build accurate maps, the robot needs to know its current position accurately. Unfortunately, due to the nature of the SLAM problem, a mobile robot would initially have neither and would have to begin the process of SLAM with a blank slate.

The importance of solving the problem of SLAM would be of great benefit as doing so would truly ensure that a mobile robot would no longer require artificial infrastructures or *a priori* knowledge of the environment in order to simultaneously navigate and build maps – it would therefore, truly become autonomous. A solution to the SLAM problem would prove to be highly beneficial in various real-world applications where absolute position or highly accurate maps are unavailable, which include: autonomous planetary exploration, autonomous air and sub-seaborne vehicles, and autonomous all-terrain vehicles in tasks such as mining and construction (Dissanayake *et al.*, 2001).

It should be noted that not all SLAM algorithms place the same emphasis on both sides of the localization – mapping spectrum, resulting in different behavior, or even purposes between different SLAM algorithms.

1.1 Traditional SLAM methods

In the paper by (Riisgaard & Blas, 2005), the problem of SLAM is stated to consist of multiple smaller processes, which are: landmark extraction, data association, state estimation, state update, and landmark update.

Figure 1.1 (shown in the next page) demonstrates the relationship between the various stages involved in the process of SLAM. As relying solely on robot odometry (i.e. dead reckoning) would produce unreliable results due to reasons stated earlier, data will have to be streamed from some low-level sensor (i.e. ultrasound, sonar, laser scanners, cameras, etc.) in order to interpret the surrounding area of the environment. For this to be accomplished, some common type of landmark, or feature, which is native to the environment, is extracted from the sensor input. Such features can be either low-level like lines and circles, or high-level, such as chairs and trees.

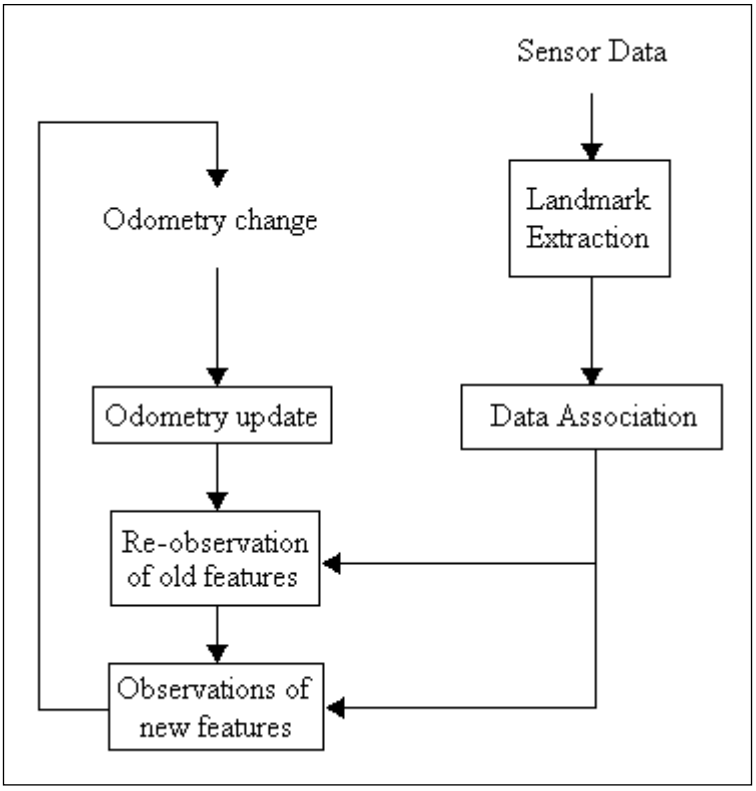
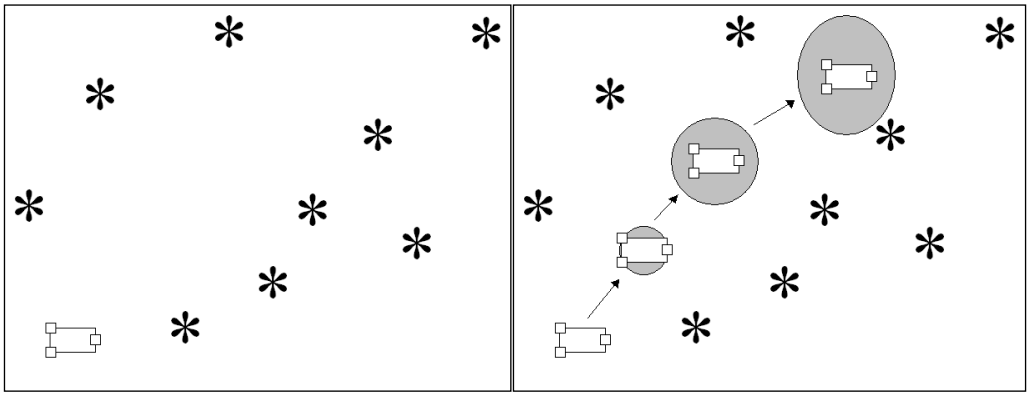


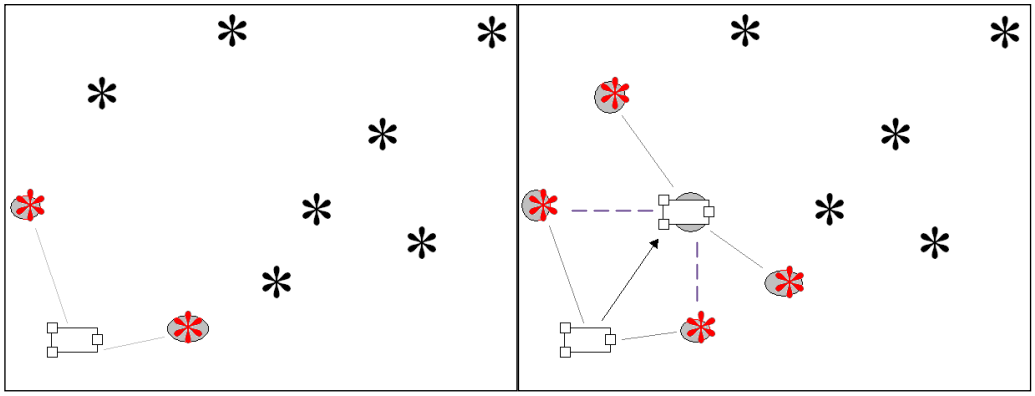
Figure 1.1 Overview of the SLAM process

After feature extraction is performed, data association is carried out in order to determine if two features observed in different points in time correspond to one and the same object in the real world (or not). Accuracy during this stage is crucial when conducting two range scans in close proximity; or when visiting a previously traveled area in the environment (Jensfelt *et al.*, 2006) (Hahnel *et al.*, 2003) (Gutmann & Konolige, 1999). The results from observing new and/or old features are used to update their estimated location, which is then used to determine, and update the change in odometry/pose/state of the mobile robot. Figures 1.2(a) and 1.2(b) demonstrate the problem of navigating through "dead reckoning", while Figures 1.2(c) – 1.2(e) show the sequence of events that revolve around the process of SLAM.



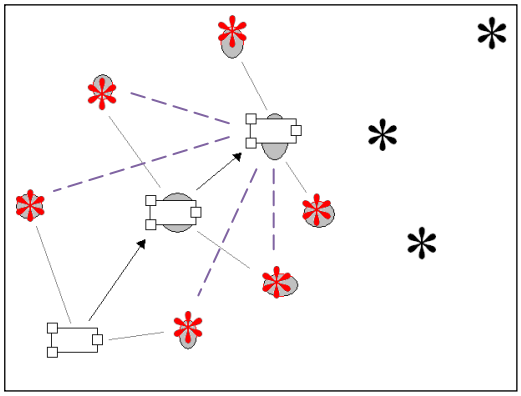
(a) (b)

Navigation with “dead reckoning”, note that the shaded circles in Figure 1.2(b) show the possible area in which the robot is actually located, which increases over time.



(c) At robot’s initial position, two features are observed.

(d) Due to re-observing the first two features (dashed lines), the estimates for both robot and feature are improved (note the smaller shaded circles).



(e) Once again, re-observing the first four features results in improved estimates for both robots and all features.

Figure 1.2 The basic concept of SLAM

The afore-mentioned approach to conducting SLAM is generally known as a *feature-based* approach. One specific feature-based approach used in order to exploit information to continuously make new (close) estimates of features and the robot itself, is to implement what is known as an Extended Kalman Filter (EKF) to construct a stochastic map which keeps track of the spatial relationships between the various features and the pose of the robot (Borges & Aldon, 2002).

To perform localization, the robot observes any features in range (both old and new), and the map is updated in order to reflect the measured locations of all observed features as well as the robot poses. As shown in Figure 1.2(c) – (e), this has the advantage of producing accurate measurements in the stochastic map, while also accounting for false returns, drop-outs and ambiguities with data association (Zunino, 2002). The main issue with such an implementation is that it is susceptible to the *scaling problem*: increased computational requirements as the number of tracked features increase.

1.2 Other Main Approaches

Two decades of research efforts into SLAM have also yielded other main approaches in tackling the problem. Two more general approaches have been put forward by Zunino (2002): grid-based, and topological approaches.

The *grid-based approach* is a technique in which the environment is represented by a grid of cells, with each cell containing the probability of being

occupied by an obstacle or object in its relative position within the real-world: a cell with the value of 0 is definitely known to be free, while a cell with the value of 1 is definitely known to be occupied by some obstacle or object. Through this method, a map of the environment can be generated, which is known as a *certainty grid*, or an *occupancy grid* (Elfes, 1989) (Elinas & Little, 2007) (Grisetti, Stachniss & Burgard, 2007).

In order to perform localization, the robot compared the currently obtained certainty grid with a set of previously constructed ones, and the pose with the maximum correlation between both the new and old map will be considered as the new pose estimate. Once this is done, the new and old maps are merged to increase the accuracy of the certainty grid. (Siegwart & Nourbakhsh, 2004) illustrate how a feature-based map differs from a grid-based one is shown in Figure 1.3.

The main advantages to implementing grid-based approaches in SLAM are that it is easy to understand and implement, as well as being able to naturally extend to higher dimensions if required. However, due to storage of the certainty grids over time, the main disadvantage with such an approach is naturally, high storage and computational requirements (Zelinsky, 1992), quite similar to that faced by feature-based approaches.

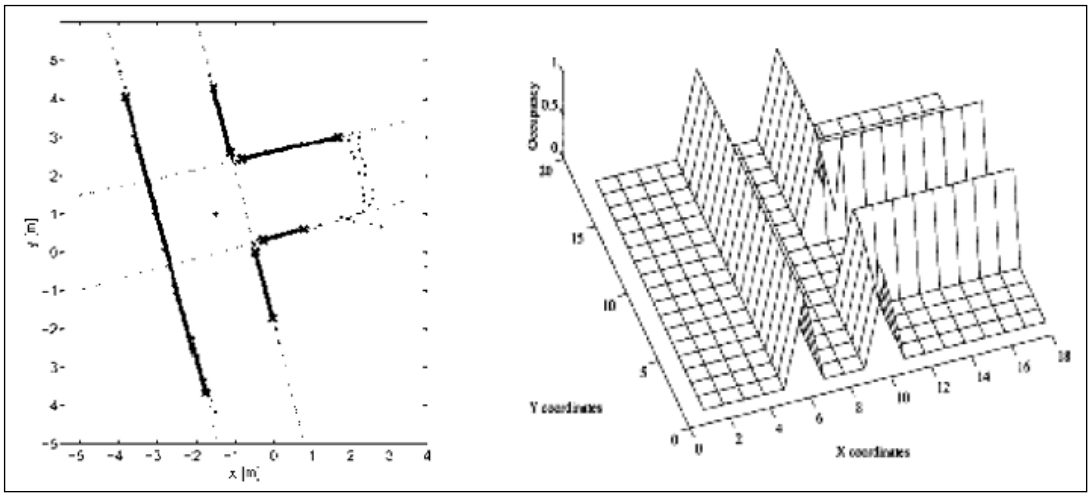


Figure 1.3. A feature-based map (left) compared to a grid-based one (right), both representing the same hallway.

Instead of maintaining a precise metric map like those seen in Figure 1.3, topological approaches to SLAM constructs a graph-like representation of the environment, which consists of nodes connected by arcs: nodes reflect "significant areas" within the environment, while arcs represent the sequence of actions connecting these areas together. Such an implementation works well for navigation in small and simple environments, but has yet to be shown to do so for large and complex environments.

Further information on feature-based, grid-based and topological approaches to the SLAM problem can be obtained by works from (Ni & Dellaert, 2010), (Strasdat, Montiel & Davison, 2010), (Duckett, Marsland & Shapiro, 2000), (Zunino, 2002) and (Thrun, 2002) while (Frese, 2006) provides a more in-depth insight into the background history of SLAM as well as a performance summary of some of the established methods developed for this particular estimation problem.

1.3 A Semantic Approach to SLAM

One common detail behind the three previous approaches to SLAM, is that they have all dealt with the problem on a geometric level (i.e. Room A of size B at location C, or obstacle D at location E), which essentially, is a purely engineered approach towards the SLAM problem. This research effort will put its focus upon a different idea of approaching SLAM, which is more biological in nature – a semantic approach.

The term "semantic" refers to the meaning of words, which in essence, is a form of knowledge representation. Robots that implement a form of mapping semantically would no longer be restricted to just knowing that a room of a certain size is at a certain location, but would have the benefit on knowing that a certain area of the current environment is a living room, or that a certain corridor is more crowded in the daytime compared to nighttime. A robot possessing such knowledge can then be said to possess *semantic information* (Galindo *et al.*, 2005), and would possess an understanding of the environment on a semantic level. This is significantly different compared to a geometric level of understanding as there are many different levels of abstraction that can be afforded to robots that implement a semantic approach to handling SLAM (Oberlander *et al.*, 2008). One such example is that a single location can possess multiple descriptions allocated to it (i.e. the instruction "go to the kitchen" can be the same as "go to the red room" or "go to the room with a stove and a fridge").

Research into semantic approaches is fairly new, though several significant bodies of work have already been identified (Nuchter *et al.*, 2003) (Dellaert & Bruemmer, 2004) (Rottmann *et al.*, 2005) (Galindo *et al.*, 2005) (Persson *et al.*, 2007) (Wolf & Sukhatme, 2008) and selected techniques will be further discussed upon in a later chapter. As research into semantic mapping is still fairly new, there has yet to be a consensus as to which type of low-level sensor input is considered to be the "standard" (Currently, a specific SLAM application or environment would dictate the appropriate set of sensors). However, there have already been efforts to fuse information from multiple low-level sensors, most notably from Thrun (2004).

Relevant research has also been made into conducting semantic mapping with cameras (Persson *et al.*, 2007) (Flint *et al.*, 2010), laser scanners (Nuchter *et al.*, 2003) (Ekvall, Jensfelt & Kragic, 2006), as well as a combination of a sonar ring, laser and color camera (Galindo *et al.*, 2005).

The concept of a semantic approach to SLAM has the potential of changing the way that SLAM has been conducted, as opposed to the other approaches that have been discussed thus far. Instead of just representing the environment purely on a geometric level, semantic mapping is capable of representing captured data on multiple levels of abstraction, and convey that meaning to human operators in a more natural form (Dellaert & Bruemmer, 2004). Because of this advantage, we choose to determine if the semantic SLAM model further described in later chapters is one such possible method.

1.4 Image Retrieval as Applied to SLAM

As a semantic form of SLAM relies upon analyzing captured data related to a particular environment, it is necessary to determine if current image retrieval techniques, as used in large image databases, be successfully transferred to the domain of real-time robotics for the purposes of implementing a more ‘natural’ SLAM model.

Image retrieval is the process of searching through a vast amount of entries that is stored within a database, and retrieving a set of images based on some form of search criteria. While many image retrieval systems are in existence, they can be classified into one of two frameworks: text-based and content/feature-based (Lakdashti, Moin & Badie, 2008) (Deselaers, Keysers & Ney, 2008). Text-based image retrieval is a concept where textual descriptions (i.e. also known as keywords or labels) are attached, or associated with each image within a database. When a user poses a textual query, the system returns the relevant images that possess similar keywords as those from the query.

One disadvantage towards a text-based approach to image retrieval is that human intervention consists of a large portion of the system; a large amount of human labor is required to connect textual descriptions with image entries. Also, describing an image is highly subjective to the human mind and can lead to inaccurate or misleading labels.

The approach of Content-Based Image Retrieval (CBIR) attempts to overcome such disadvantages. In CBIR, images are classified according to some aspect related to their visual content, such as color, texture, hue and contrast (Jhanwar *et al.*, 2004). Conventional methods then represent these features as a vector or histogram that is associated with a particular image such as in (Swain & Ballard, 1991) (Pass & Zabih, 1999) (Berens, Finlayson & Qiu, 2000) (Siggelkow, 2002) (Jonsgård, 2005) . Another contrast of this approach compared to text-based image retrieval is that the user queries themselves are images. One or more example images are submitted to the system instead of providing keywords, which the system will analyze and output images from the database which are similar, based on various distance measures. This allows the possibility of manual human labor to be completely excluded from the image retrieval process and thus, rendering the process to be complete autonomous.

One issue with CBIR is that training usually needs to be carried out (either fully or semi-automated) for a certain number of iterations before any queries can be submitted. This training phase is a common requirement of CBIR as proper categorization/classification of image content is an important factor towards an effective CBIR technique (Chou & Cheng, 2006). Another issue is known as the *semantic gap* (Chiu, Lin & Yang, 2003) (Lakdashti, Moin & Badie, 2008), which is based on the premise that the low-level visual details obtained from CBIR aren't able to relate well towards the high-level descriptions that humans are used to identifying and describing images. For example, assuming a user provides example photos of a green cup to the system; while it might be able to extract all images which contain green (a low-level feature) as the dominant color from the database, not all of them are guaranteed to have cups (a high-level concept).

Despite the potential drawbacks, it has been determined that CBIR methods are the most suitable for implementation with the proposed SLAM method. The autonomous nature of CBIR is crucial as the proposed SLAM model is also intended to be independent from manual human intervention. In addition, while CBIR does not directly understand the content of images (as how object recognition methods do), it is possible that relationships between detected feature vectors of images can be determined, thus adding an additional semantic layer on top of the CBIR process. This will be described in greater details in later chapters.

1.5 Research Question and Objectives

Currently, efforts into implementing SLAM specifically with semantic mapping methods are still in it's infancy as compared with other traditional SLAM methods. As a result, there are still many issues related to semantic mapping that have yet to be resolved. The primary research question that is related to semantic SLAM can be phrased as such:

> "Can we apply CBIR techniques to the domain of SLAM to generate a robust SLAM model?"

To answer this question in a satisfactory manner; the following research objectives have been devised:

1. To develop a real-time semantic SLAM model, by exploring the methods currently used for image retrieval as used in large image databases.
2. In addition to exploring current image retrieval techniques, semantic approaches in scene analysis will be researched upon.

1.6 Overview of Chapters

This section will serve as an overview of the remaining chapters that are in this thesis, by providing a brief description of the content within each chapter.

Chapter 2 reviews several pieces of literature that is related to the fields of SLAM and CBIR. The content of these conference papers and journal articles are discussed, where their strengths and weaknesses are highlighted.

Chapter 3 discusses the methodology of the current research effort, where a semantic SLAM model is shown, and the various stages within this model is explained.

Chapter 4 demonstrates a specific implementation of the semantic SLAM model that is introduced in Chapter 3. Explanations for each and every step in the current implementation are provided, with examples shown where necessary.

Chapters 5 to 7 are devoted towards discussing the various experiments that were conducted on the semantic SLAM model during the course of the research

effort. Chapter 5 is focused on determining if (1) image patches are correctly categorized, and (2) reconstruction of images from categorized patches is correctly performed. The experiments in Chapter 6 demonstrate the process of creating the semantic descriptors that define each image, and Chapter 7 analyses the accuracy of similarity scores generated throughout several indoor and outdoor video streams.

Lastly, Chapter 8 acts as a conclusion for this thesis and also summarizes the outcome for each chapter as well as the performance of the implemented model. Suggestions for future work subsequent to the submission of this thesis are also discussed here.

Chapter 2

Literature Review

In this chapter, a review on various papers related to this research will be provided. In addition, a summation of the fields that are relevant to this research effort will also be discussed. This discussion will cover domains that include semantic mapping and image processing. The fundamental concepts of these topics will be explained in order to facilitate a better understanding of how these topics are inter-related and integral to the process of SLAM through mapping semantically, and what the contributions of this research effort will be.

2.1 Mapping Semantically

As stated previously, while not as prevalent as traditional methods, the idea of mapping semantically has been proposed before by (Dellaert & Bruemmer, 2004), where goal setting can be on a semantic level (for example, "go to the red room") which traditional SLAM methods are incapable of handling, yet. However, it should also be understood that the probabilistic understanding of an environment has still not yet proven to be foolproof (i.e. a robot can interpret an entity in a certain environment to have an 80% chance to be a chair with dimension X or 20% it is a sofa with dimension Y). Therefore, the proposed model within this paper requires human intervention to provide further information or action when such a situation creates

indecision on the part of the autonomous robot, which is clearly unacceptable in the context of SLAM.

One possible method in overcoming this issue is detailed by (Nuchter *et al.*, 2003) who states that the architectural structure of any environment follows a standard convention, either from tradition or utility. This knowledge would provide general attributes of any one domain (i.e. plane walls, ceilings and floors) which can be used to reconstruct the environment (in this case, an indoor environment). While this implies that different domains would require different knowledge, another possible variation upon this suggestion similar to the research of (Galindo *et al.*, 2005) would be to implement existing knowledge in order to clarify the identity of an "ambiguous" entity. For example, in a room with a fridge and a stove (which have already been identified), it would be natural to assume that the unidentified object would be a chair rather than a sofa. Another useful model (albeit simple) proposed in the paper is the implementation of a semantic net, which demonstrates the relationship between objects within a certain environment/room.

2.1.1. Multi-Hierarchical Semantic Maps

While it has been suggested so far that semantic mapping belongs within a distinct category compared to more traditional methods of SLAM, (Galindo *et al.*, 2005) suggests that it is possible for the knowledge provided by both geometric and semantic levels to be integrated in order to assist in the process of SLAM. The semantic perspective would be used for goal-setting (i.e. "go to the bedroom", which

is a *room* which contains a *bed*), while the spatial perspective will be used for navigation (i.e. go 2 meters along this corridor, turn right 90°, etc. etc.).

This paper proposes a model which maintains two different hierarchies in order to represent the current environment: spatial, and conceptual, where basic links between the two hierarchies are established through anchoring. This can be seen below in Figure 2.1.

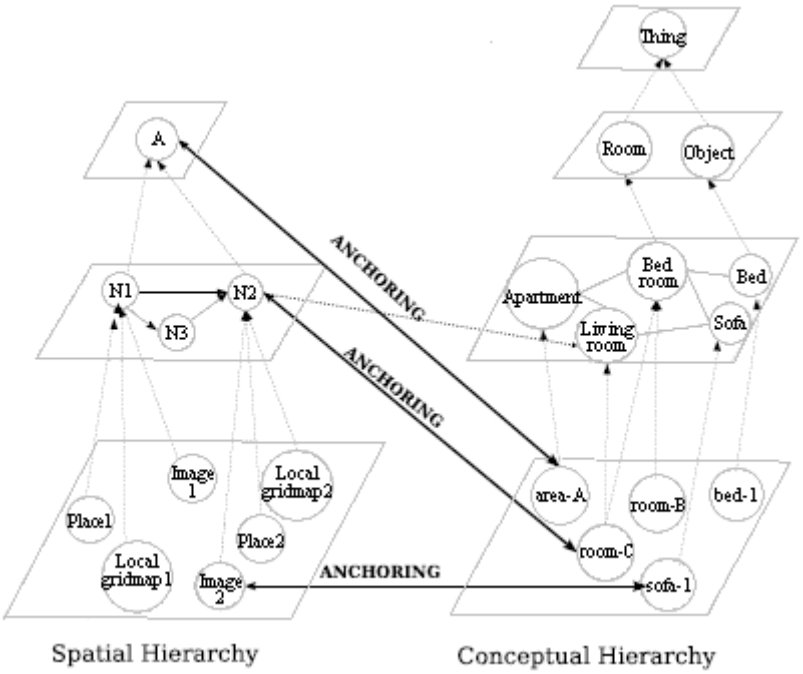


Figure 2.1. The Spatial and Conceptual Hierarchies.

Construction of the Spatial Hierarchy is done by building an occupancy grid of the surrounding area, which is then filtered through fuzzy mathematical morphology, and then segmented through a technique known as water-shedding. The result is an extracted topology of large open spaces that can be anchored to room names. This process can be seen in Figure 2.2.

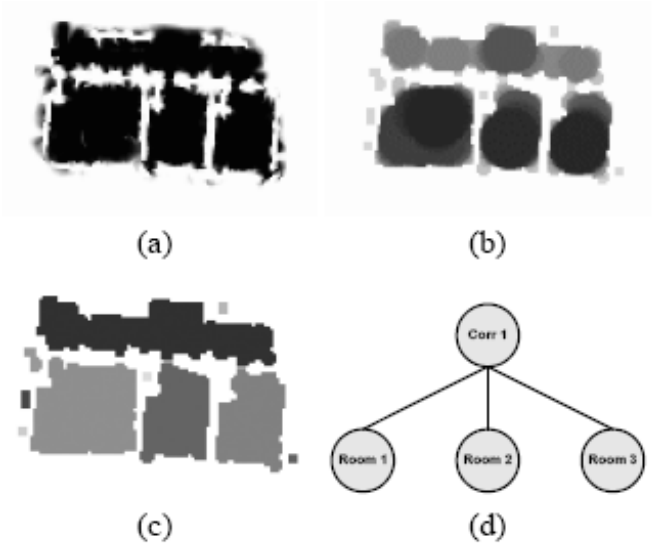


Figure 2.2. (a) Occupancy grid; (b) Fuzzy morphological opening; (c) Watershed segmentation; (d) Extracted topology

The Conceptual Hierarchy, developed by a knowledge representation system known as *NeoClassic* (Borgida *et al.*, 1989), models semantic information available within the environment. This hierarchy functions similar to the parent-child concept of object-oriented programming languages; all conceptual objects have a common ancestor named *Thing*. The next level (Level 2) consists of two general "child" categories within the context of the current domain named *Objects* and *Rooms*. Level 1 has more specific categories, such as *Stove* and *Bed* (which are children of *Objects*) or *Bedroom* and *Kitchen* (children of Rooms). The last level, Level 0, has individual instances of the concepts from Level 1, such as *room-2* and *sofa-b*. On each level, each conceptual category can possess either a vertical link pointing to the higher level (to indicate an "is-a" relationship) or a horizontal link pointing to other categories on

the same level (to indicate a "has-a" relationship), or both. This can be seen in Figure 2.3 which demonstrates the structure of Level 1 on the Conceptual Hierarchy.

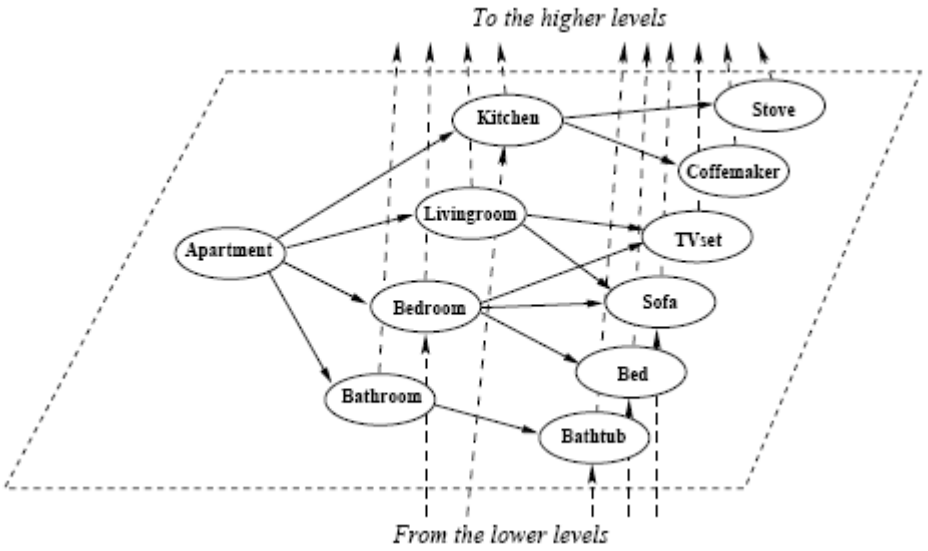


Figure 2.3. Level 1 of the Conceptual Hierarchy

The process of anchoring then follows as this is necessary in order to ensure that the correspondence between sensor data (i.e. categories within the Spatial Hierarchy) and semantic symbols (i.e. categories within the Conceptual Hierarchy) truly refer to the same physical objects or rooms, as shown previously in Figure 2.1. The proposed inference system as a result of anchoring proves to be interesting as it allows a robot to traverse to the location of an object which was not previously observed or identified. For example, a robot receiving the goal of "go to the TV set" would infer that a TV set would be located either within a bedroom or a living room, which the robot would traverse to. However, this would only succeed if such information was known beforehand, which presents the issue of determining how much *a priori* knowledge a robot should possess before it can be considered sufficient. The issue of indecision would also arise if more than one room possesses a

TV set: while the process of goal-setting can be revised to be more specific, the eventual indecision resulting from ambiguity should not be taken lightly.

The issue of ambiguity arises again when it comes to the model's ability to self-diagnose localization errors. The model indicates that the detection of relevant objects that belong to certain rooms would ensure that the robot would be able to constantly perform localization at all times (i.e. picking it up from a bedroom and putting it in a kitchen will not fool it, as it would be able to recognize various kitchen appliances that are in the room). However, the paper does not address the issues of multiple rooms that possess a uniform structure. A house may not have more than 1 kitchen, but an office environment would contain many office cubicles that are of a similar nature.

Finally, the vision system for this model has not yet shown the capability to identify real-world objects; the conducted tests have only used simple shapes of various colors, which would then be identified as furniture. It would be interesting to note how the process of identifying real-world objects (or the ambiguity of it) would have an effect upon the performance and results of the model.

2.1.2. Probabilistic Semantic Mapping for Building/Nature Detection

While previous papers have dealt with mapping semantically in an indoor environment, (Persson *et al.*, 2007) has shown that mapping with a semantic approach

is possible in an outdoor environment as well. It is also from this paper which indicates that 360° vision is possible with commercially available products, which provides the inspiration for this research effort. This omni-directional image is then unwrapped, and then flattened in order for feature extraction to occur. Figure 2.4 shows the structure for all three types of images.

Figure 2.4 An omni-directional image (top), an unwrapped version of the same image (middle), and flat images extracted from the unwrapped image (bottom)

Feature extraction is then performed in order to detect artificial man-made structures. In this paper, three different groups of features are detected, which are: edges, combinations of edges (i.e. corners and rectangles), and grey level clusters (i.e. large homogenous regions within an image). Training with AdaBoost (a machine learning algorithm by (Freund & Schapire, 1999) is then conducted in order to

maximize accuracy with building and non-building detection, as well as decrease the likelihood of false positives/negatives from occurring.

Following this, the local grid map – which is probabilistic representation of a sector in the current occupancy grid that is currently visible by the robot – is updated where probabilities P_i (class | VS^T, α_i) are assigned to all of the *n* objects in view of the mobile robot according to:

$$P_i(class|\mathrm{VS}^T, \alpha_i) = \frac{1}{2} + \frac{\alpha_i}{\theta}(P(class|\mathrm{VS}^T) - \frac{1}{2}) \qquad (2.1)$$

where:

α_i = horizontal covering angles $\alpha_1, \alpha_2, \ldots, \alpha_{i=n}$ of all objects within the current sector.

θ = sector opening angle, and

P(class|VS^T) = conditional probability that a view is *class* when the sensor classification at time T is also *class*.

The global semantic map is then updated with the local map. Each grid cell (x,) is assigned a probability of being occupied after *T* number of sensor updates with $P(occ_{x,y} | s^1, s^2, \ldots, s^T)$, which can also be computed as:

$$P(\mathrm{occ}_{x,y}|s^{1:T}) = 1 - \left(1 + \frac{P(\mathrm{occ}_{x,y}|s^{1:T-1})}{1-P(\mathrm{occ}_{x,y}|s^{1:T-1})} \frac{P(\mathrm{occ}_{x,y}|s^T)}{1-P(\mathrm{occ}_{x,y}|s^T)}\right)^{-1} \qquad (2.2)$$

Since the sensor update s^T is the output VS^T from the sensor at time T, and the grid cells are assigned probabilistic values on whether they belong to *class*, the update formula above for each x-y coordinate can be rewritten to:

$$P(class|\mathrm{VS}^{1:T}) = 1 - \left(1 + \frac{P(class|\mathrm{VS}^{1:T-1})}{1-P(class|\mathrm{VS}^{1:T-1})}\frac{P(class|\mathrm{VS}^{T})}{1-P(class|\mathrm{VS}^{T})}\right)^{-1} \quad (2.3)$$

This will result cells within the global map to contain one of three different classes:

$$\begin{array}{ll} \text{Building} & \text{if } P(\text{class}|\mathrm{VS}^{1:T}) > 0.5 \\ \text{Unknown} & \text{if } P(\text{class}|\mathrm{VS}^{1:T}) = 0.5 \\ \text{Non-building} & \text{if } P(\text{class}|\mathrm{VS}^{1:T}) < 0.5 \end{array} \quad (2.4)$$

All cells within the global map will be initialized to possess a value of 0.5 and will be incrementally updated to one of the other two classes as the robot traverses the environment.

The classification of environmental entities into 2 classes has been shown to produce maps similar to those of traditional methods. However, while the paper has stated that positioning (i.e. localization) issues can be handled with either SLAM or GPS, only the GPS method has been used. The results shown in the paper also do not indicate the time taken to obtain and process visual data, which is crucial when implementing a real-time SLAM algorithm; while the paper indicates that the

occupancy grid is either supplied or built by the robot, it does not explicitly show its effects (if any) on the performance of the proposed algorithm.

2.2 Image Processing

In order for a SLAM algorithm to be considered real-time, incoming visual data must be processed at a fast rate (at least as fast as the robot is traversing from one point to another) such as those demonstrated by (Tieu & Viola, 2000) (Davison, 2005) (Rosten & Drummond, 2006) and (Konolige *et al.*, 2006) where a very rapid scanning of images is demonstrated. While experiments have been made on a database of 3000+ images, the authors have made the claim of possibly scanning a million images a second. This is an important feature to note for this current research effort, as while the robot would start with an initially empty set of images within its database, this set will eventually grow to a set of extremely large proportions as the robot traverses more and more unknown portions of the environment.

2.2.1 Boosting Image Retrieval

In this paper from (Tieu & Viola, 2000), highly selective visual features are automatically generated that respond to a very selective group on image within an image database. The process of generating such features begins by extracting a feature map for each type of simple feature, such as those seen in Figure 2.5.

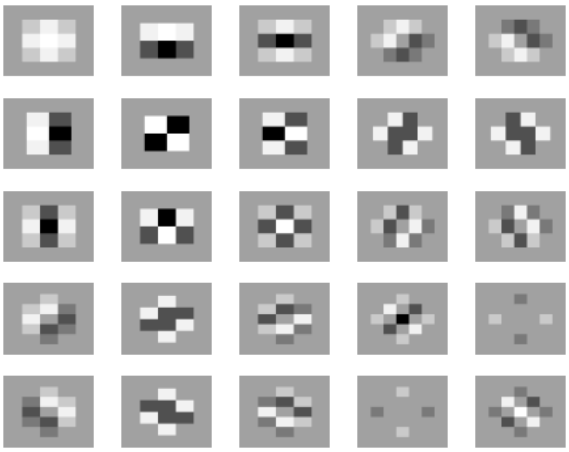


Figure 2.5 A sample of 25 primitive feature maps

After being rectified and down-sampled by a factor of 2, each of these 25 feature maps are then used as input for extraction of the same set of features, thus yielding 625 (25 * 25) feature maps. This is done again in order to yield 13,625 feature maps (625 * 25). 46,875 feature maps are then obtained by filtering over the red, green and blue color channels. Each feature map is finally summed to yield a single feature value.

Each level of processing as detailed previously is done in order to discover arrangements of features on a previous level. For example, a second order feature might be sensitive towards horizontal features arranged diagonally, which can be visible as a staircase pattern. Such features are computed from an image from the following equation:

$$g_{i,j,k,c} = \sum_{pixels} M_{i,j,k,c} \tag{2.5}$$

where $M_{i,j,k,c}$ is the feature map that is connected with primitive filters *i*, *j*, and *k*, and *c* is the color channel. $M_{i,j,k,c}$ is also defined by the following:

$$\begin{aligned} M_{i,j,k} &= \downarrow_2 (|f_k \otimes M_{i,j}|) \\ M_{i,j} &= \downarrow_2 (|f_j \otimes M_i|) \\ M_i &= \downarrow_2 (|f_i \otimes X|) \end{aligned} \tag{2.6}$$

where $\otimes$ is the image, f is a primitive filter, and $\downarrow 2$ is the down-sampling operation by a factor of 2.

This paper argues that such features do indeed reflect upon the structure of images, which is supported by the argument that with an image database of 3000 images, the number of such detected features are sparse where the kurtosis (or "peakedness") of the data distribution curve is 8 on an average and can reach as high as 120 for certain features (In comparison, the standard Gaussian data distribution curve has a kurtosis of 3). With this data, it is reasoned that this form of distribution curve is highly unusual, and therefore, very meaningful as well.

Once again, AdaBoost is used to run the proposed learning algorithm for 20 iterations in order to yield a strong classifier that is dependent upon 20 features. The proposed learning algorithm is described in the following steps:

- For example images $(x_1, y_1), \ldots (x_n, y_n)$, each and every $y_i = 0,1$ for negative and positive examples respectively.
- Initialize the value for weights, $w_{1,\,i} = 1 / (2m), 1 / (2l)$ for each $y_i = 0, 1$ respectively, where m and l are the number of negative and positive examples, respectively.

- For t = 1, … T:
  - Train one hypothesis h_j for each feature j with w_t, with an error, $\epsilon j = \Pr_i^{w_t}[h_j(x_i) \neq y_i]$
  - Select $h_t(\cdot) = h_k(\cdot)$, so that $\forall j \neq k, \epsilon_k < \epsilon_j$ (i.e. select the hypothesis with the lowest error). Set $\epsilon_t = \epsilon_k$.
  - Set $w_{t+1,i} = w_{t,i}\beta_t^{1-e_i}$ where $e_i = 0, 1$ for each example image x_i that is classified correctly or incorrectly, respectively, and $\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$. Normalize $w_{t+1,i} \leftarrow \frac{w_{t+1,i}}{\sum_{j=1}^{n} w_{t+1,j}}$ so that w_{t+1} is a distribution.
- The final resulting hypothesis is:

$$h(x) = \sum_{t=1}^{T} \alpha_t h_t(x) \geq \frac{1}{2} \sum_{t=1}^{T} \alpha_t \qquad (2.7)$$

where $\alpha_t = \log(1 / \beta_t)$.

Testing of the proposed algorithm is performed with five different classes of images, divided into sunsets, lakes, waterfalls, fields, and mountains, where each class consists of 100 images. Figure 2.6 indicates the average precision and recall for each of the five classes.

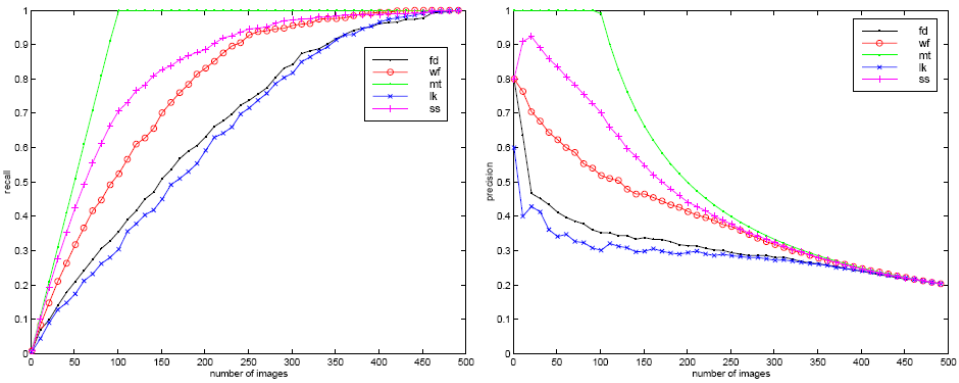


Figure 2.6 Average recall (left) and average precision (right) for all five classes of natural images

The results of this paper shown previously have indicated that as the number of images in the database increases, recall increases while precision decreases. This establishes that as recall increases, precision decreases. While one might argue that this is merely a natural outcome due to the immense size of the database, the results obtained, which will be discussed later, appears to prove otherwise.

Another issue pertaining to this proposed algorithm is that it is initially a weak classifier, which is required to undergo training (i.e. solving a sequence of learning problems) in order to produce a strong classifier that is actually a weighted combination of weak classifiers. Training of the classifier is required to be conducted each time a query is produced to the algorithm, which would most likely prove to be detrimental to the performance of real-time SLAM.

2.2.2 Image Retrieval Based on Weighted Features

(Liu *et al.*, 2007) proposes an interesting idea where different weights are assigned towards deciding the importance of detected low-level features. These weight values are adapted according to an automatically define feature statistic, which is capable of reflecting the visual difference between images with starkly different pre-dominating low-level features. This is based on the premise that the desired images requested by a user should have similar low-level features as the query image. For example, when a user submits a close-up image of a tree trunk, the proposed

system will disregard from returning images which do not have a similar emphasis (such as a blank wall or ceiling). The proposed algorithm consists of several main steps, which are:

1. Dividing images: Each image is divided into sub-blocks of 4 * 4 pixels with each sub-block possessing a six dimensional feature vector, f(k,l) = [f_1(k,l), f_2(k,l), f_3(k,l), f_4(k,l), f_5(k,l), f_6(k,l)] that contains color and texture information of its respective sub-bloc where (k, l) is the position of the image pixel. The first three dimensions f_1(k, l) – f_3(k, l) represent the average values of Hue, Saturation and Value, respectively, and are defined by the following equations:

$$f_1(k,l) = \frac{1}{16}\sum_{i=0}^{3}\sum_{j=0}^{3} x_H(k+i,l+j)$$
$$f_2(k,l) = \frac{1}{16}\sum_{i=0}^{3}\sum_{j=0}^{3} x_S(k+i,l+j) \qquad (2.8)$$
$$f_3(k,l) = \frac{1}{16}\sum_{i=0}^{3}\sum_{j=0}^{3} x_V(k+i,l+j)$$

The latter three dimensions consists of high frequency data for each sub-block that is defined during wavelet transformation.

2. Wavelet transformation: A Harr wavelet transform operation is performed on the Value component of each sub-block, which yields four 2 * 2 pixel transform coefficients, presenting the LL, HL, LH, and HH frequency bands, as shown below in Figure 2.7.

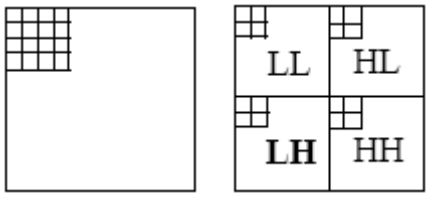


Figure 2.7 The Harr wavelet transform operation

The average value for the LH frequency band is represented by $f_4(k,l)$ according to the following definition:

$$f_4(k,l) = [\frac{1}{4}\sum_{i=0}^{1}\sum_{j=0}^{1} c_1^{\,2}(k+i,l+j)]^{\frac{1}{2}} \tag{2.9}$$

where $c_1 = \{c_{k,\,j},\ c_{k,\,j+1},\ c_{k+1,\,j},\ c_{k+1,\,j+1}\}$. Following this, $f_5(k, l)$ and $f_6(k, l)$ are defined as:

$$f_5(k,l) = [\frac{1}{4}\sum_{i=0}^{1}\sum_{j=0}^{1} c_2^{\,2}(k+i,l+j)]^{\frac{1}{2}}$$
$$f_6(k,l) = [\frac{1}{4}\sum_{i=0}^{1}\sum_{j=0}^{1} c_3^{\,2}(k+i,l+j)]^{\frac{1}{2}} \tag{2.10}$$

3. Feature clustering: Each image is segmented into 10 clustering areas according to the ISODATA clustering method (a classification method similar to k-means, with the difference being that ISODATA allows for different number of clusters, while the k-means algorithm requires *a priori* knowledge on the number of clusters) and is defined as $\{r_i,\ i = 1, 2, \ldots, 10\}$. The ISODATA clustering algorithm itself consists of three portions:

i. Initial clustering: Select K number of feature vectors from image feature database as the initial clustering centers (in this case, K = 10), where:

$$C_n = X_n,\ n = 1, 2, \ldots, K \tag{2.11}$$

ii. Prototype assigning: Each vector Xq is assigned to the nearest cluster center, where:

$$D(X_m, C_k) = \min(D(X_q, C_k), q = 1,2,3\cdots Q)$$
$$X_m \in clust[k] \tag{2.12}$$

iii. Refresh clustering center: Each center of mass is designated as the new clustering center. Repeat step ii. until the clustering centers are stable:

$$\boldsymbol{C_k = \frac{1}{count(k)} \sum_{clust[q] \in k} X_q \qquad q = 1,2,\cdots Q; k = 1,2,\cdots K} \tag{2.13}$$

where clust[q] is the cluster number.

4. Calculate feature statistics: In this paper, all supplied images are 256 * 256 pixels, which are then divided into 16 sub-blocks Z_1, Z_2, .., Z_{16}. If the image consists of only pure textures, then the proportion of a clustering area r_i falling into sub-block Z_j is 1 / 16, and is defined as:

$$p_{i,j} = \frac{\#r_{i,j}}{\sum_{j=1}^{16} \#r_{i,j}} \tag{2.14}$$

where $\#r_{i,j}$ is where the clustering area r_i belongs to sub-block Z_j, and for each r_i, $\sum_{j=1}^{16} p_{i,j} = 1$. The featur'e st'atistic for each r_i is defined by the following:

$$T_i = \sum_{j=1}^{16} \frac{(p_{i,j} - \frac{1}{16})^2}{\frac{1}{16}} = \sum_{j=1}^{16} 16(p_{i,j} - \frac{1}{16})^2 \tag{2.15}$$

and the feature statistic for each image is:

$$T = \frac{1}{10}\sum_{i=1}^{10} T_i \qquad (2.16)$$

The value of T reflects the difference between an image that consists solely of textures and one that consists only of pure colors, with a complete texture image possessing a T value of 0. Some examples are shown in Figure 2.8.

Figure 2.8 The feature statistic T for various images

After the T value for all images within the database have been calculated, normalize their values into:

$$T' = \frac{T - T_{\min}}{T_{\max} - T_{\min}} \qquad (2.17)$$

Following this, the value of T' determines which image feature is to have a higher weight. For example, with a T' value of 0, a heavier weight value should be assigned towards texture features, whereas this weight value should be reduced as the T' value increases. The weight values are defined by the following equation:

$$W_1 = 1.0 + ((1 - T') / 2),\ W_2 = 1 - T' \qquad (2.18)$$

where W_1 and W_2 are the weight values for color and texture features, respectively.

This method of image retrieval was tested upon 1550 images, separated into classes of landscapes, cars, animals, buildings, and others. Results pertaining to precision (but not recall) are compared to other methods, and are shown in Figure 2.9.

Feature / Result / Sample	Color	Texture	Tradition method	Paper method
Landscape	85.5%	66.7%	81.7%	92.8%
Animal	84.3%	56.4%	85.9%	88.6%
Car	71.5%	42.6%	77.2%	87.9%
building	72.6%	42.7%	73.1%	89.4%
Others	85.1%	46.8%	82.6%	89.1%

Figure 2.9 Precision results based on different features

However, the results from this paper are questionable, as the description of the proposed model initially states that four low-level features are considered, but only two appear to play a role within the actual description of the model as well as in determining results. The result also contains details on precision, but not on recall, and does not provide the time required to return relevant images to the user.

The model proposed by (Lakdashti, Moin & Badie, 2008) appears to be well-suited for adaptation to SLAM as feature extraction is initially performed from the query image. Also, training of the algorithm is automated, and can be run after a robot performs a short initial run of SLAM within an environment, and possibly even prior to that. In conducted experiments, the feature vector related to each image was a 3-D histogram of RGB colors and Tamura texture features, which could possibly prove to be useful in the theoretical framework of this current research effort. This will be

explained in further detail in a later chapter. The results also prove to be quite interesting as they are more favorable compared to that of (Tieu & Viola, 2000); while the results appear to have the same trend, precision begins to increase alongside with recall at an early stage. It has also been suggested (though not proven) that with the implementation of more sophisticated features, the results will be significantly better as well.

2.3 Chapter Summary

The focus of this chapter is on discussing any pre-existing method of conducting SLAM through a semantic approach. As semantic SLAM methods are not as well developed or established compared to traditional methods, published papers related to semantic SLAM techniques are relatively few. Therefore, many opportunities exist towards the refinement and development of this particular field. To further this end, existing methods of knowledge representation that may prove to be beneficial towards conceiving a viable SLAM model are discussed.

As CBIR is intended to be part of the proposed SLAM model, selected papers within the field of image processing are reviewed as well. In this portion of the chapter, papers related to (low-level) feature maps and feature extraction are discussed in order to determine the feasibility of classifying images according to different feature vectors.. The advantages and disadvantages of each reviewed method are also discussed in order to determine which of these are deemed to be beneficial towards a semantic form of SLAM.

Chapter 3

Methodology

In the previous chapter, the fundamental concepts of SLAM were discussed. This knowledge serves as a basis for the semantic SLAM algorithm that has been developed for this research effort. This chapter will provide an explanation regarding the various processes that constitute the entire algorithm.

3.1 The Semantic SLAM Pipeline

A pipeline that serves as a basis for the semantic SLAM model is shown in Figure 3.1. This model consists of four separate stages that have the goals of (1) feature extraction, (2) classification and storage, (3) semantic analysis, and (4) location resolving. A description of each stage will be provided beginning on the next page.

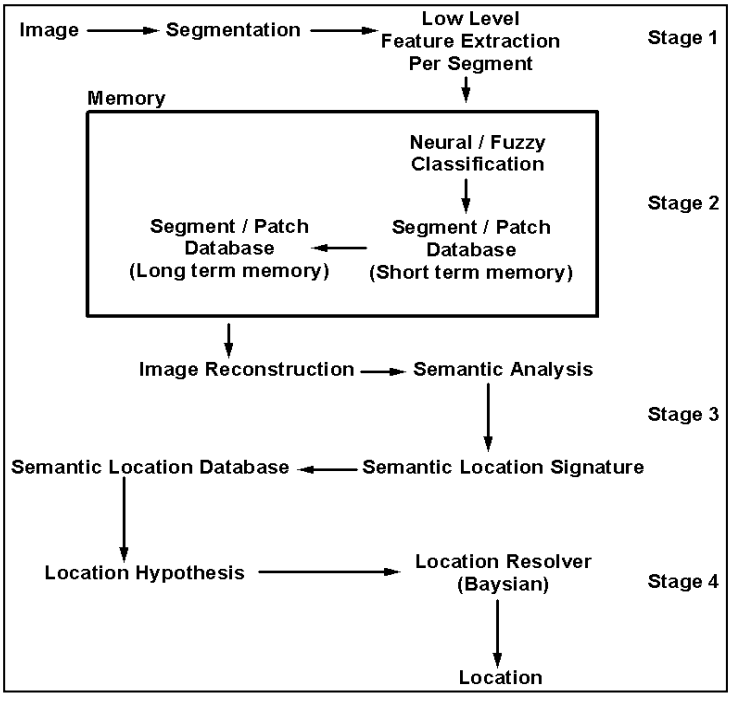


Figure 3.1 The various stages within the semantic SLAM pipeline

3.1.1 Feature Extraction

While entire images can be analyzed, retrieved and categorized according to some metadata or feature vector associated with it, this method is too generalized to be able to gather semantic information for any particular image. Therefore, images are to be divided into a certain number of segments, or patches, before proceeding further with low-level feature extraction for each patch.

The specific feature vector that will serve as an identifier for each patch (or patch signature) is dependent on the sensor input being used, as well as the specific method of image retrieval that is being implemented. These patch signatures collectively form a single image signature, which is similar in concept to shape signatures used in (Zhang & Lu, 2001). This feature vector will then be submitted as input into the next stage.

3.1.2 Classification and Storage

After the process of constructing patch signatures is completed, they are then presented to a fuzzy/neural network classifier, or some other appropriate self-organizing network, similar in concept to that in (Li, Shi & Luo, 2007). This is an important factor to consider, as the classifier should not be limited to a fixed number of classes, but grow accordingly instead as the variance in signatures demands an increasing number of classes.

Each class is determined by a <patch signature, patch> pairing, which are required during image reconstruction in the next stage. Note that a graphical, visual reconstruction of images is not necessary at this development stage, as the proposed semantic analysis would work equally as well on class labels generated from the self-organizing neural network. Such an action may still prove to be useful in later developments in order to aid in enhancing semantic descriptions (i.e. using object recognition or further CBIR techniques).

Thus, the classifier acts as the short-term memory portion of this stage, while <patch signature, patch> pairs being copied into long-term memory whenever a semantic description uses that pairing. However, it is possible for the classifier to generate classes that will never be used while a robot traverses through any particular environment.

3.1.3 Semantic Analysis

Once patch signatures have been classified into various groups, each and every captured image is to be reconstructed through the use of only the unique patch signatures that have been stored within the database. Following this, reconstituted images are then analyzed in order to determine the semantic context that is present. For example, each cluster of patches that are within the same classification is given labels denoting the semantic relationship with other patch clusters as well as its size and current location within the entire image. One example of this is shown in Figure

3.2, where the patch cluster of C1 possesses the relationship of being to the left of cluster C2.

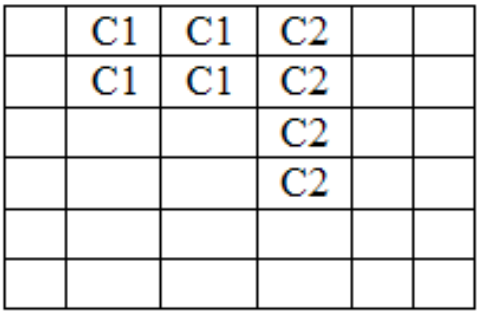


Figure 3.2 Relative locations between patch clusters

Therefore, the cluster of C1 is to be annotated with the relationship labels of:

$$\text{Right}(C1) = C2$$

$$\text{Location}(C1) = \text{top left}$$

In order to provide a certain degree of flexibility in regards to relationships between clusters, we attach high-level descriptors to them, rather than noting the specific patch indices that constitute a cluster. For instance, if cluster C1 were to shift left, the labels denoting the relationship between C1 and C2 would still hold, whereas that might not necessarily be true if specific patch indices were recorded.

The aggregate of all labels related to one cluster is considered to be the semantic location signature for that cluster. There also exists the possibility of detecting high-level features that are present within the captured images such as object recognition (Rottmann *et al.*, 2005) to further contribute towards the amount of semantic data present within each semantic location signature.

The semantic descriptions for each and every cluster (and hence, the entire reconstructed image) is then entered into a semantic descriptor database. This database is responsible for comparing existing descriptors against their incoming counterparts to determine if a match is possible. (Daoutis, Coradeschi & Loutfi, 2009) is one example where a semantic reason system is implemented with Research Cyc (a general knowledge base and inference engine).

3.1.4 Location Resolving

As a result of the previous stage, a set of location hypothesis (i.e. a set of locations each labeled with a probability) is generated. Such sets are generally caused by noise and perceptual aliasing (a situation where different locations appear similar to each other). Therefore, this final stage of resolving which location the robot is currently at is required, as simply selecting the location with the highest probability is naïve.

Bayesian reasoning opens up the possibility of determining the most likely location the robot is currently at by considering where the robot assumes its current location is, along with a new set of location hypothesis. However, a common problem that arises from this is method is the issue of location, or orientation independence. As an example, it can be difficult to recognize that a previous location has been revisited as it was viewed from a significantly different angle. One solution towards this problem is to implement a panoramic image sensor, which can be

financially prohibitive and not easily available. Therefore, we attempt to overcome the aforementioned issue through another method, where locations are semantically related to each other by tracking either unique patches, or a unique ordering of patches.

3.2 Available Sensors

As stated earlier in Chapter 1, a wide range of sensors are available for mobile robots to utilize during the implementation of SLAM. Sensors perform various measurements that are both internal and external to the mobile robot, such as measuring the internal temperature of a robot's electronics or determine a robot's global position (through the use of omni-directional images, as previously mentioned in Section 2.1.2). These sensors can consist of ultrasound, sonar, infra-red/laser scanners and many more. However, only a certain class of sensors would prove to be useful during SLAM, this class is known as *exteroceptive sensors*. Exteroceptive sensors measure information that is external to the robot (as opposed to *proprioceptive sensors*) such as distance measurement, sound amplitude and light intensity (Siegwart & Nourbakhsh, 2004).

Even accounting for only exteroceptive sensors, providing an in-depth detail for each and every single sensor available would require more space than this report can accommodate. Therefore, explanations will be provided for two of the most common exteroceptive sensors used in the field of SLAM research: laser rangefinders and cameras.

Laser rangefinders are capable of performing distance measurement through a technique known as time-of-flight. This behaves very similarly to how sonar works, as the sensor transmits a laser beam to illuminate a target while a receiver that is on the same axis as the transmitted laser beam detects the reflected laser beam that bounces off the target. By determining the time it takes for a laser beam to hit a target and reflect back to the robot, an estimate of the range between the robot and the target can be calculated. While some researchers employ the use of a laser scanner instead, the fundamental principle of operation remains unchanged. Research efforts that have used laser rangefinders include (Gutmann & Konolige, 1999) (Newman, Cole & Ho, 2006) (Thrun *et al.*, 2004) and (Weingarten, 2006).

There are several issues that prevent the laser rangefinder from being considered as the only sensor of choice for SLAM, however. While a laser rangefinder is fast and accurate, it is not a vision sensor and hence, unable to detect certain types of semantic information, such as color and texture. In addition, while the beam from a laser rangefinder sweeps across various obstacles in the area, it is only capable of providing information limited to a plane. Therefore, any obstacles that exist above or below this plane will remain undetected by the sensor (i.e. a shoebox placed on top of a refrigerator). Finally, due to the very nature of light reflection, surfaces with a high degree of specular reflectance would reflect the project laser beam away (Zheng, Fukagawa & Abe, 1995).

Vision sensors (like cameras), avoid such drawbacks, which allows them to be considered as another viable sensor input. As the amount of sensory information

recorded by cameras is similar to that of human vision, a large body of data is obtained (especially in artificial environments) which can aid in interpreting the surrounding area on both a high-level and low-level perspective (Royer *et al.*, 2005). Examples of detectable low-level information are color and texture, while high-level information includes spatial relationship between objects. Some of the most recent SLAM models have implemented stereo cameras (Se, Lowe & Little, 2002) (Dailey & Parnichkun, 2005) (Konolige *et al.*, 2006) (Elinas & Little, 2007) (Santini & Rucci, 2007) as well as monocular cameras (Davison, 2003) (Mouragnon *et al.*, 2006) (Smith, Reid & Davison, 2006) (Jensfelt *et al.*, 2006) (Royer *et al.*, 2007).

There are also reasons why cameras still aren't considered to be vastly superior compared to other sensor types. Due to vast amount of information available in any single image, the amount of computing power required to handle and process them is also much higher compared to other sensors. Another issue related to vision sensors is their sensitivity to light. Similar to human sight, the darker the surrounding, the harder it is to obtain any form of information - a completely dark room would render a vision-equipped robot helpless. While it is possible to overcome such situations with additional equipment such as night-vision or infra-red sensors, it comes with a higher cost in terms of price, space and computational power. In addition, (Dailey & Parnichkun, 2005) states that multiple cameras are required in order to obtain triangulation from various perspectives and/or require *a priori* knowledge of the environment. However, research into monocular SLAM has already provided results with great potential, as earlier stated.

3.3 Cameras as a Sensor Input

In order to implement a semantic approach to conducting SLAM, information that is related to both spatial and non-spatial aspects of the environment will be required. This not only affords different levels of abstraction, as described earlier, but also disqualifies sensors that are not capable of providing the required amounts of data necessary in the first place. It would be impossible to order a mobile robot equipped with a laser rangefinder to "go to the red room" when it is incapable of understanding the term "red". Therefore, cameras will be required as the sensor of choice to gather the amount of environmental data needed for a "semantic SLAM".

Incidentally, there are other benefits in selecting cameras as the sensor input of choice. Most of them are a result of technological advancements that render cameras as a more attractive choice for SLAM when compared to 20 years ago. Such features include: cheaper cost, smaller size, the capacity for greater processing power and increased quality (i.e. such as the transition from capturing black and white video to color and the availability of higher resolutions). In addition, several attempts have been made to improve estimation of camera poses (Fitzgibbon & Zisserman, 1998) (Triggs *et al.*, 2000), allowing for greater accuracy when performing localization.

3.4 Overview of the Current Implementation

An overview of the implementation based on the model previously described in Section 3.1 is shown in Figure 3.3. While specific details of the implementation

regarding this model will be explained in the next chapter, some technical statements will be provided here as well, in order to provide a logical and cohesive justification for some of the methods used within the model.

The overall objective of this implementation is to determine the degree of similarity between images of a traversed environment by comparing their semantic context. For this to be done, each image must first be segmented into individual, equally-sized patches. Each 640 * 480 pixel-sized image is to be divided into patches of size 80 * 96, thereby resulting in 40 patches per image. The number and size of the patches is not chosen arbitrarily as the *bestblk* function from the Image Processing Toolbox of MATLAB dictates that the aforementioned patch size is considered to be optimum for image processing.

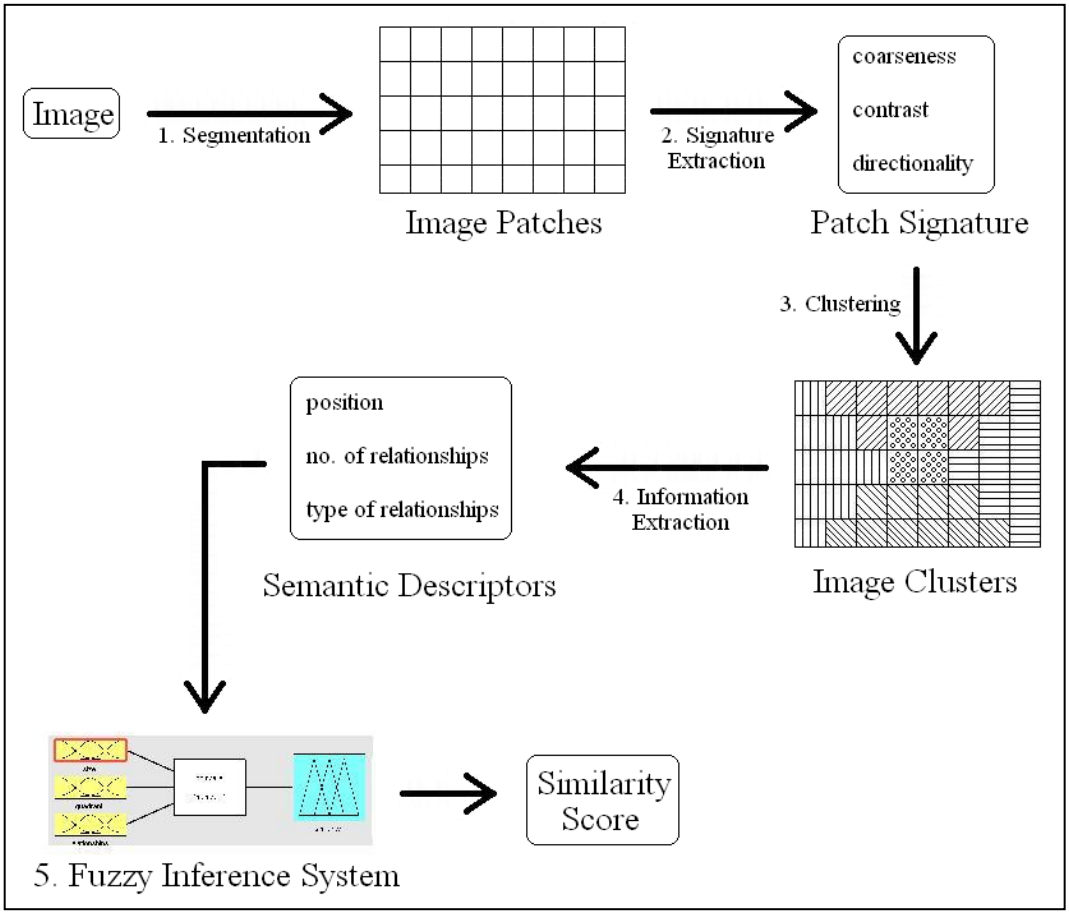


Figure 3.3 The various stages of the current implementation

Following that, the Tamura Texture features (Tamura, Mori & Yamawaki, 1978) – which consists of coarseness, contrast and directionality – for each patch are extracted in order to form a signature that is unique to each patch. This is performed in a similar manner to that in (Lakdashti, Moin & Badie, 2008).

The selection of these features to be tracked is made due to the reason that much research has already been made in implementing CBIR techniques through various color feature vectors (Long, Zhang & Feng, 2003). Also, (Chiu, Lin & Yang, 2003) has already applied Tamura Texture features towards the CBIR domain and (Shi & Tomasi, 1994) (Elinas, Sim & Little, 2006) suggest that good results can be achieved through tracking texture features. Therefore we attempt to determine if satisfactory results can also be achieved within the context of semantic SLAM.

While other Tamura Texture features exist (namely, line-likeness, regularity and roughness), these will not be considered for inclusion within a patch signature, as Tamura, Mori & Yamawaki (1978) state that more effort is required to describe the texture elements that constitute these three elements, and that coarseness, contrast and directionality much more significant in global descriptions of textures. However, (Kulkarni & Verma, 2003) and (Li, Shi & Luo, 2007) have indicated the potential of implementing such features within the context of CBIR. The process of how patch signatures are constructed will be further explained with greater detail within the next chapter.

The patch signatures will then be classified into several clusters through the use of an Adaptive Resonance Theory (ART) neural network that is first developed by

(Carpenter, Grossberg & Rosen, 1991) and described in further detail by (Huang, Georgiopoulos & Heileman, 1995). Through this method, several different patch signatures within an image that deliver the same semantic context should be clustered into the same group. This concept can be seen in Figure 3.4.

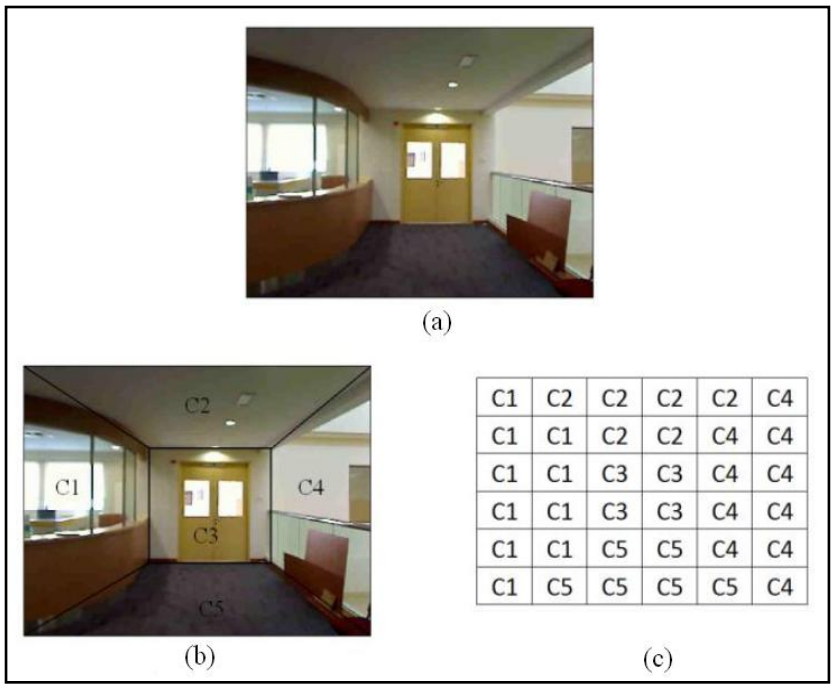


Figure 3.4 The concept of clusters, where an image in (a) possesses 5 areas with a semantic context in (b), resulting in each patch to be assigned to a related cluster in (c).

As an example, consider the image in Fig. 3.4(a), which has 5 potential semantic areas of interest that can be treated as classes. These areas are divided within the regions as shown in Fig. 3.4(b) where C1 to C5 are the counter, ceiling, door, wall/railing, and floor, respectively. Therefore, the reconstituted counterpart to this image would be as seen in Fig. 3.4(c) where several patches constitute a single cluster.

The ART neural network was chosen to be implemented over other fuzzy/neural network classifier as this particular neural network is a self-organizing

network. This is an important factor to consider as the variance of the patch signatures can demand an increasing growth in the number of classes, which factors out the implementation and usage of classifiers that operate with a static, limited amount of classes.

In order to obtain the semantic context of each image, the semantic information pertaining to their related group of image clusters is then extracted. Each image cluster is able to yield: (1) their relative position within the image, (2) the number of relationships they have with other clusters, and (3) the directional type of said relationships. These forms of semantic information take on a general scope rather than record precise, absolute measurements. For example, we record a particular cluster as considered to be within the upper-left portion of an image, rather than at specific x-y coordinates. This is because during comparison between the semantic information of multiple images, small changes within an absolute scale will result in a large amount of dissimilarity, as opposed to a more general scale. Consider also the following scenario shown in Figure 3.5.

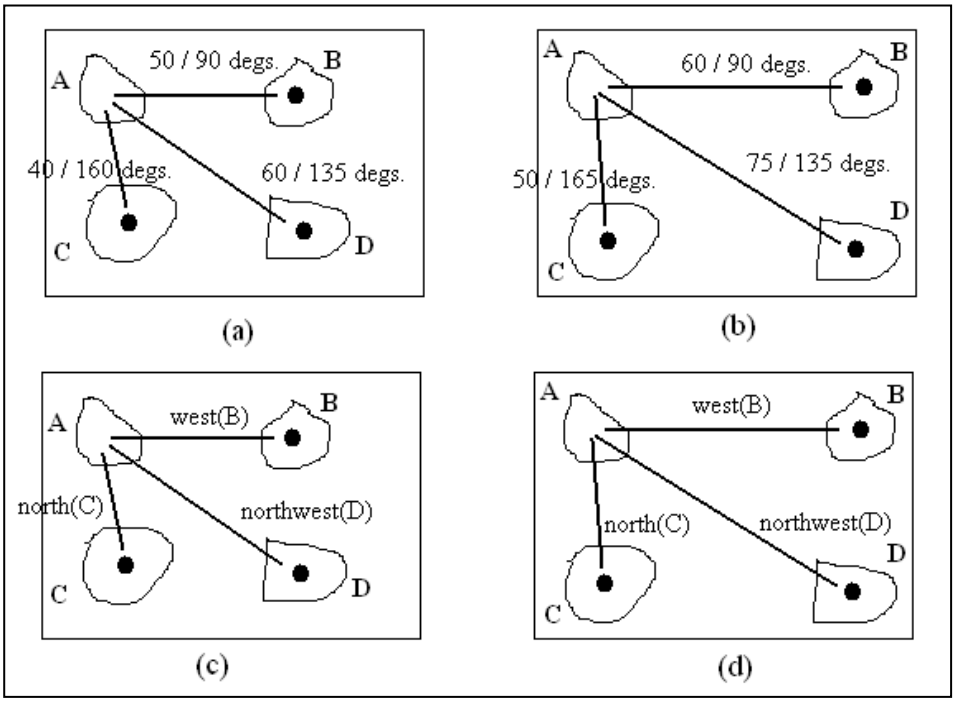


Figure 3.5 Situations where relationship links between clusters are considered to be dissimilar due to a small change in measurement ((a) and (b)), and where the relationships are still similar due to a more generalized scope ((c) and (d)).

Assuming that a camera has traversed a certain portion of the environment and captures two separate images at the same angle, thus resulting in both possessing the same semantic context, but with slightly dissimilar relationships between image clusters, when measured with an absolute scale (Figure 3.5a and 3.5b) where the distance and angle of each relationship is recorded. However, because images are to be categorized according to their semantic context and not measurements of relationships, this is considered to be undesirable as the relationship measurements can continue to increase in magnitude while the camera continues to traverse and capture images within an environment that always possesses the same semantic context. Therefore, this will result in a decreasing similarity score when such a trend is not warranted.

In order to circumvent this situation, we measure relationships on a topological level – similar in concept to that of (Schwering, 2007), which implements a more formalized method of specifying spatial relationships – where distance measurements are completely omitted and angles are generalized into descriptors based on the 8 main cardinal directions (Figure 3.5c and 3.5d). Through this method of generalization, relationships are still correctly maintained even while the topological locations of clusters experience minor changes through a sequence of captured images.

Once the semantic descriptors of every image cluster is extracted, they are then passed as input parameters into a pre-defined Fuzzy Inference System (FIS) which evaluates the differences between all three composite values that are related to

two separate images, thus resulting in a score denoting the degree of similarity between the two images.

The entire process that was previously described is iterated continuously as images of the immediate environment are captured and submitted as input. This can be seen in Figure 3.6 where sequentially captured images are compared with an initial reference image. This process of comparing semantic descriptors of images will result in a series of similarity scores that should demonstrate a downward trend as the camera traverses further away from the initial starting point as time goes by. Once a similarity score is registered below a certain threshold, the previous image that caused this phenomenon will then be selected as the new reference image for analysis and comparison against further images.

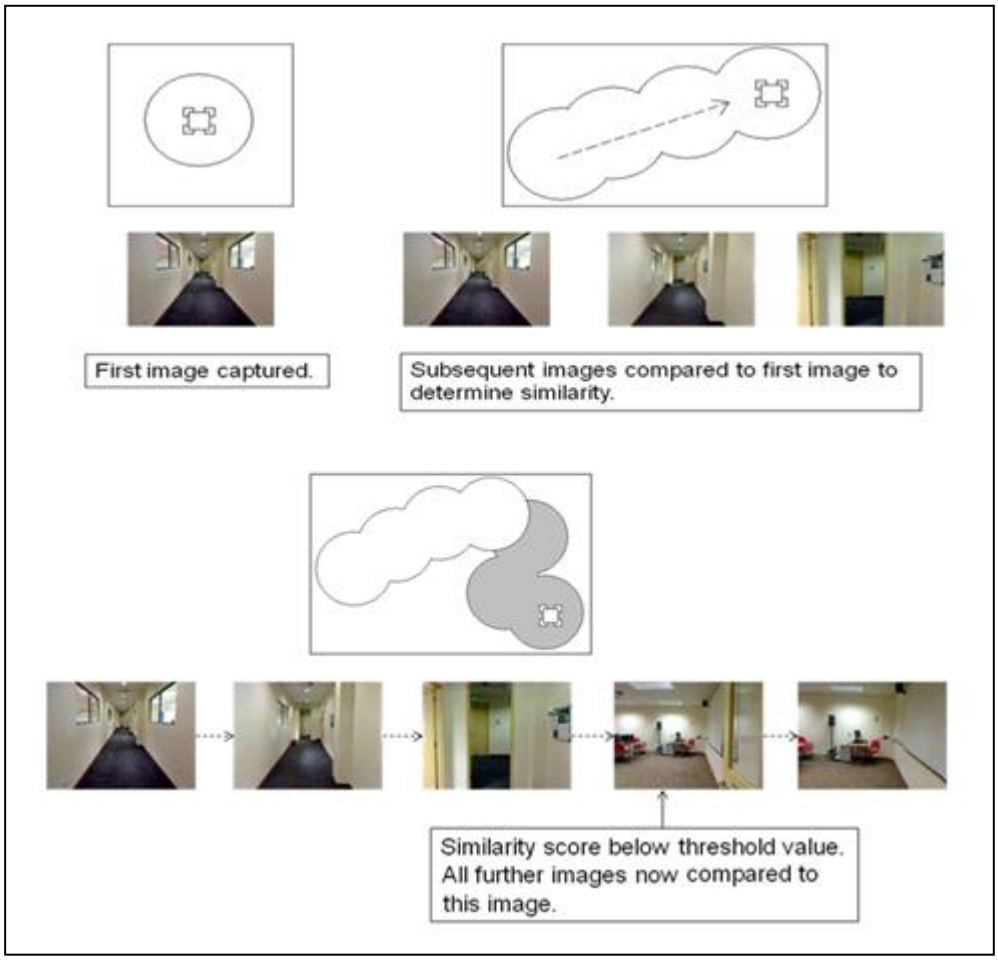


Figure 3.6 Determining the reference image during comparison of image

3.5 Chapter Summary

The first portion of this chapter introduced a 4-stage concept on conducting the operation of semantic SLAM. These stages are focused on the tasks of feature extraction, classification and storage, semantic analysis, and location resolving, with each stage consisting of several composite operations. This is followed with an overview of the high-level sensors most suitable for implementation towards semantic SLAM, with an emphasis on cameras as the sensor input of choice.

An implementation of the 4-stage semantic SLAM concept is then described, in which the Tamura Texture features of coarseness, contrast and directionality constitute as a feature vector that are intended to serve as a method of classification (through an Adaptive Resonance Theory (ART) algorithm) for each of the 40 patches that are part of an image frame. Semantic information (i.e. size, location and relationships) that is determined from groups of such classified patches act as a unique identifier for their related image frame. 2 sets of such data (each representing a different image frame) are then submitted as input into a Fuzzy Inference System (FIS) in order to calculate the degree of similarity between them and generate an appropriate score value.

Chapter 4

Implementation

During the course of this research effort, a prototype system was designed in order to implement the semantic form of SLAM that was previously discussed in earlier chapters. The purpose of this chapter is to explain the methods of this implementation. The programming language used to develop the prototype Guided User Interface (GUI) that runs the implementation is MATLAB (version R2007b) running on the Windows XP operating system. Images were captured through the use of a 2-megapixel Logitech QuickCam Pro for Notebooks.

4.1 Overall Structure of Program Code

The flow of data within the prototype system is shown in Figure 4.1, which denotes the sequence of events that are expected to occur between the various entities involved. Note that the file named "PreRecordedGUI.m" mainly serves as a GUI interface and wrapper for the *PreRecorded()* function, where most of the actual operations and processes related to the system are being handled.

The following sub-sections will provide a detailed explanation regarding each process within the system. While the complete code is available in the CD included

with this thesis, certain portions of code are highlighted or summarized in order to facilitate an easier understanding of the implementation at hand.

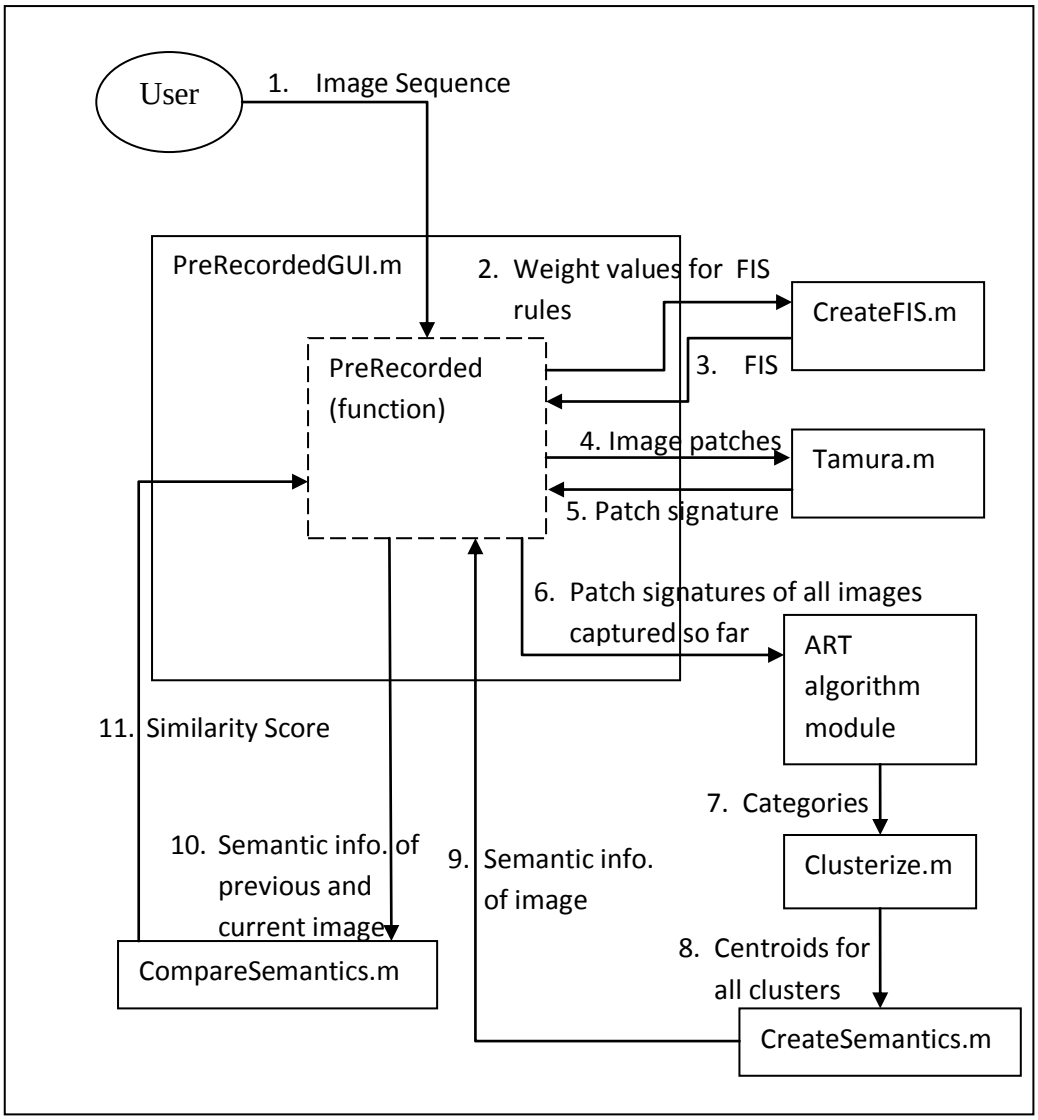


Figure 4.1 The flow of data through the prototype system

It should be noted that this implementation assumes that images are captured beforehand and regarded as a sequential form of input into the system.

4.1.1 The PreRecorded Function

As importing captured images from files is trivial, an explanation is considered to be unnecessary. Therefore, we begin our explanation of the prototype system with the *PreRecorded()* function, with the code for the entire model available within the supplied CD.

First, a fuzzy inference system is created by the CreateFIS m-file with several default settings (that can be modified later) with a weight value of 1. This is followed by extracting patches of size 80 * 96 from each of the images. The extraction of each patch is done in order to obtain the Tamura Texture feature set consisting of coarseness, contrast, and directionality by passing the patch to the Tamura m-file. These features, or patch signatures, are then concatenated into a matrix variable containing all patch signatures captured so far.

Once all 40 patch signatures for a particular image are extracted, they are concatenated again into a cell array containing all signatures for all images previously captured. This is done in order to determine the number of categories that exist within the data contained in the cell array by passing it as input into the ART algorithm module. This process is important in determining groups of the same categories that are in each image (more details regarding this process are provided later, in sub-Section 4.1.5).

Following this, the semantic information of an image is constructed by determining the topological location of each cluster as well as how one cluster is related to the others. A score is generated by obtaining the semantic information of the current image and a previously referenced image, and processing them through the fuzzy inference system that was created at the beginning of the function. Once a particular comparison results in a score below a certain threshold value, the current image is then considered to be the new reference image. This can be seen more clearly in the code segment shown in the *PreRecorded()* function, where the value of the variable *previous* is determined either when then currently image, *i*, is the first image to be analyzed, or when the generated score from the *CompareSemantics*() function is below the threshold value of 5.

4.1.2 The Fuzzy Inference System

The Fuzzy Inference System (FIS) is created solely through the use of functions provided by the Fuzzy Logic Toolbox within MATLAB as its performance has been reviewed favorably by (Hall & Hathaway, 1996). In order to accommodate the 3 semantic features of cluster size, location (expressed as quadrants in the code) and relationships, 3 different input variables were created, each with 2 membership functions denoting the similarity and difference values (in terms of percentage) of each input variable. Both membership functions are trapezoidal in shape and cover a range of values from 0 to 100, where a higher value denotes a higher degree of difference for the related feature between images and vice versa. Figure 4.2 shows how the 2 membership functions of each input variables are organized.

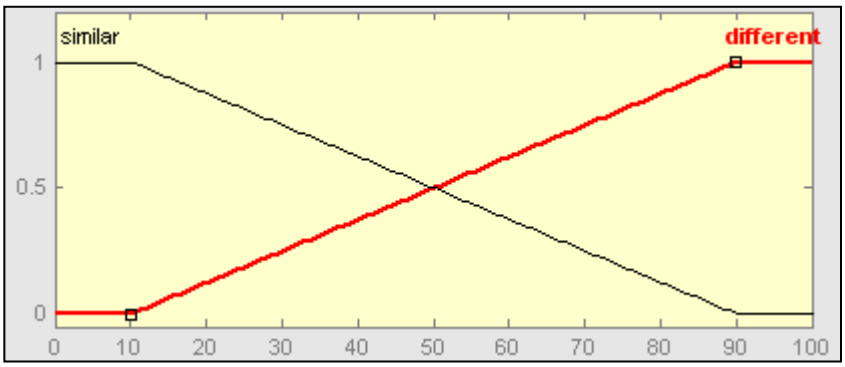


Figure 4.2 The membership functions for any particular input variable in the Fuzzy Inference System

Naturally, an output variable also needs to be defined within the FIS for it to generate any sort of similarity measure. In this case, the similarity measure is also determined by 2 trapezoidal-shaped membership functions, but with a different distribution spread. This is done because the threshold score of 5 (as shown in Figure 4.3) should serve as a boundary determinant to ensure that neither membership function would have a higher influence over the other.

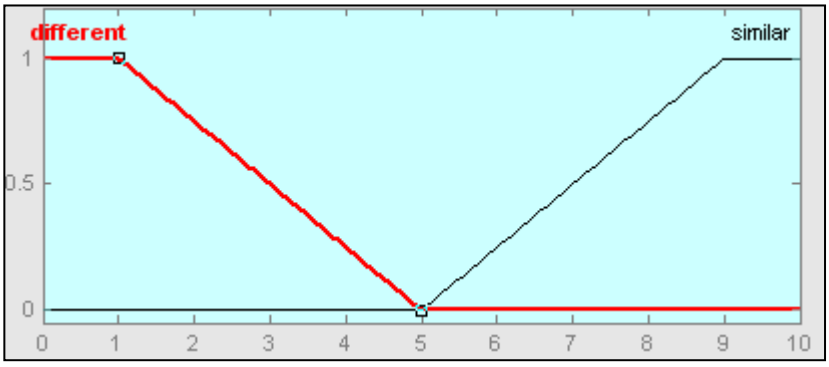


Figure 4.3 The membership functions for the output variable in the Fuzzy Inference System

While the range of values covered by the output variable is from 0 to 10, the minimum and maximum values that can possibly be generated are 1.69 and 8.31, respectively. This is due to the nature of the FIS to choose the centroid of a certain

area bounded by the membership functions (and the implemented rules as well) as the final value. Higher values indicate a higher level of similarity.

Following the creation of the input and output variables, the rules for the system are created, with 3 rules determining the degree of similarity between features for each input variable, and another 3 rules to determine the differences. The weights for each of the rules are determined by the 3 MATLAB variables, *sizeWeight*, *quadrantWeight*, and *relationshipWeight* as shown in the *CreateFIS()* function, where the *addrule()* function adds the rules defined in the variable *ruleList* into the FIS. As described in the previous sub-section, these variables have a default value of 1 when the comparison of images is processed for the first time, but can be set to any value by the user in subsequent operations.

Once the rules have been added to the FIS, they will have the graphical appearance as shown in Figure 4.4, where rules 1 to 3 contribute towards demonstrating a difference between the current image and its previous counterpart (which results in lowering the similarity score), and rules 4 to 6 contribute towards a similarity between them (and hence, raising the similarity score). The relationship between the feature values, membership functions and variables are also demonstrated.

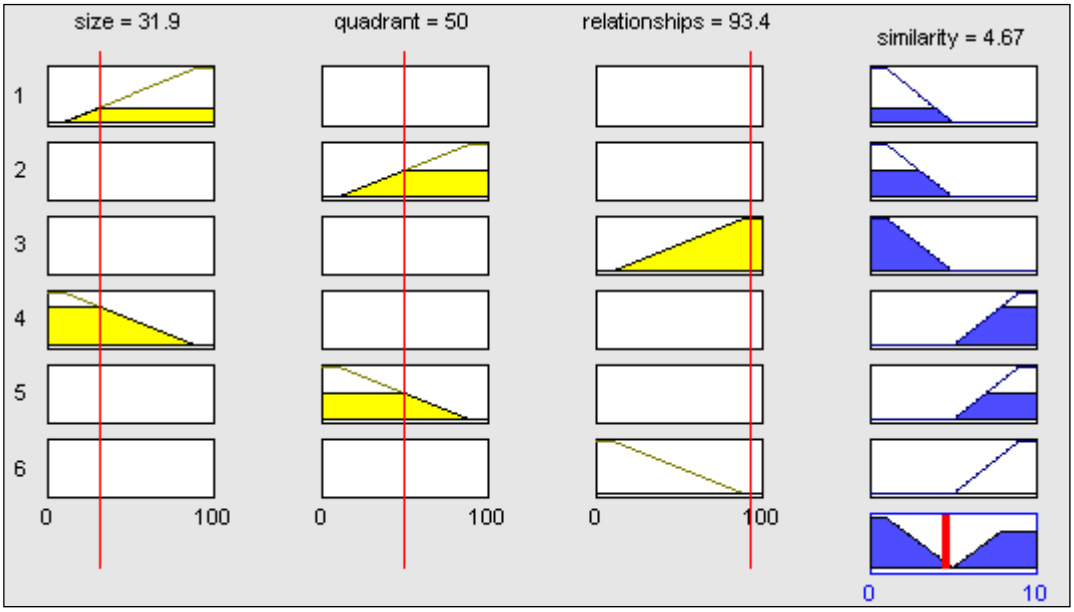


Figure 4.4 The rules of the FIS

4.1.3 Tamura Texture Features and Image Signatures

At this time of writing, the MATLAB environment does not implement the extraction processes for the features of coarseness, contrast and directionality as described in (Tamura, Mori & Yamawaki, 1978). Therefore, this operation is implemented by passing each patch data into the Tamura.m file, which was obtained from an online source at <http://en.pudn.com/downloads123/sourcecode/graph/texture_mapping/detail522203_en.html> and shown in the following page.

4.1.3.1 Coarseness

In order to measure coarseness within any particular patch, the following steps are conducted.

Step 1: For each pixel point (x, y) within the image patch, calculate the sum total over neighbourhoods where the sizes are powers of two (i.e. 1 * 1, 2 * 2, ..., 32 * 32). The sum total over the size 2^k neighbourhood at point (x, y) is

$$Sum_{k,v}(x, y) = \sum_{i=x-2^{k-1}+1}^{x+2^{k-1}} \sum_{j=y}^{y+2^k-1} f(i,j) \quad (4.1)$$

for the vertical orientation, v, and

$$Sum_{k,h}(x, y) = \sum_{i=x-2^k+1}^{x} \sum_{j=y-2^{k-1}}^{y+2^{k-1}-1} f(i,j) \quad (4.2)$$

for the horizontal orientation, h, where f(i, j) is the gray-level pixel value at (x, y).

Step 2: For each and every pixel point of each value of k, calculate the differences between the sum total corresponding to pairs of non-overlapping, equal sized neighbourhoods in both vertical and horizontal orientations, where:

$$E_{k,v}(x, y) = (Sum_{k,v}(x, y) - Sum_{k,v}(x, y - 2^k)) / 2^{2k} \quad (4.3)$$

for the vertical orientation, v, and

$$E_{k,h}(x, y) = (Sum_{k,h}(x, y) - Sum_{k,h}(x + 2^k, y)) / 2^{2k} \quad (4.4)$$

for the horizontal orientation, h.

Step 3: For each and every pixel point, analyse the output values as a result from the previous step, and select the best size, k, that gives the highest value:

$$S_{best}(x, y) = 2^k \quad (4.5)$$

where k maximizes E in either orientation:

$$E_{max} = \max (E_{1,h}, E_{1,v}, E_{2,h}, E_{2,v} ..., E_{5,h}, E_{5,v}) \quad (4.6)$$

Step 4: Finally, calculate the coarseness measure of the entire image patch by averaging all S_{best} values:

$$F_{crs} = \frac{1}{m*n} \sum_{i=0}^{m} \sum_{j=0}^{n} S_{best}(i, j) \quad (4.7)$$

where m and n are the width and height of the image patch, respectively.

4.1.3.2 Contrast

The method of measuring the value of contrast for any particular image patch is also explained in (Tamura, Mori & Yamawaki, 1978) where the following steps are implemented:

Step 1: For each image patch, obtain a histogram of gray-level differences in order to, calculate the average gray-level value:

$$Avg_{image} = \text{sum}\left(\sum_{i=1}^{256} bin(i) * \frac{count(i)}{m*n}\right) \quad (4.8)$$

where bin(i) is the ith bin value with a histogram count of count(i), and m and n are the width and height of the image patch, respectively.

Step 2: Calculate the fourth moment about the mean, μ4 with the following equation:

$$\mu 4 = \text{sum}\left(\left(\sum_{i=1}^{256} bin(i) - Avg_{image}\right)^4 * \frac{count(i)}{m*n}\right) \quad (4.9)$$

Step 3: Measure the amount of polarization by defining the kurtosis, α_4 as

$$\alpha_4 = \mu_4 / \sigma^4 \quad (4.10)$$

where σ^2 is the variance, which is calculated as

$$\sigma^2 = \text{sum}\left(\left(\sum_{i=1}^{256} bin(i) - Avg_{image}\right)^2 * \frac{count(i)}{m*n}\right) \quad (4.11)$$

Step 4: Finally, combine σ and α_4 to obtain the contrast measure with the equation

$$Fcon = \sigma / \alpha_4^{0.25} \quad (4.12)$$

4.1.3.2 Directionality

(Tamura, Mori & Yamawaki, 1978) also provides the method of calculating the feature of directionality which we implement as explained in the following steps:

Step 1: declare the following two 3 * 3 operators to aid in calculating the horizontal and vertical differences, ΔH and ΔV respectively, where

$$OP_H = \begin{matrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{matrix} \qquad OP_V = \begin{matrix} 1 & 1 & 1 \\ 0 & 0 & 0 \\ -1 & -1 & -1 \end{matrix} \quad (4.13)$$

Step 2: Calculate ΔH and ΔV for each gray-level pixel point *f*(*i*, *j*) in the image patch, where each 3 * 3 size neighbourhood with f(i, j) in the centre is multiplied with either OP_H, or OP_V, depending on the orientation, and then taking the sum total. These calculations for ΔH and ΔV can be summarized as:

$$\Delta H(x, y) = sum(\sum_{i=x-1}^{x+1} \sum_{j=y-1}^{y+1} f(i, j) * OP_H) \quad (4.14)$$

$$\Delta V(x, y) = sum(\sum_{i=x-1}^{x+1} \sum_{j=y-1}^{y+1} f(i, j) * OP_V) \quad (4.15)$$

Step 3: From ΔH and ΔV, we are able to obtain a magnitude ΔG, where

$$\Delta G = (|\Delta H| + |\Delta V|) / 2 \tag{4.16}$$

and the local edge direction for each pixel point, θ(i, j), where

$$\theta(i, j) = \begin{cases} 0 & \text{iff } \Delta_H(i, j) = 0 \text{ and } \Delta_V(i, j) = 0 \\ \pi & \text{iff } \Delta_H(i, j) = 0 \text{ and } \Delta_V(i, j) > 0 \\ \tan^{-1}(\Delta_V(i, j) / \Delta_H(i, j)) & \end{cases} \tag{4.17}$$

Step 4: Obtain the histogram HD by quantizing θ over bin values that range from 0 to π, as shown by an example in Figure 3. Following that, we apply a threshold process on to HD where any bin with a count value < 0.01 will have its respective count value be reset to 0. This is done in order to filter out the counting of unreliable directions that cannot be considered as edge points.

Step 5: The directionality is finally determined by calculating the sharpness of the peaks in HD. This is done by summing the peaks across the entire histogram, thus:

$$F_{dir} = F_{dir} + \sum_{m=1}^{length\,(HD)} (\phi_m - \phi_p * 0.0001)^2 * H_D(m) \tag{4.18}$$

where ϕ_m is the value of the m^{th} bin, ϕ_p is the bin value of the highest count (peak) within the histogram, and $H_D(m)$ is the count value of the m^{th} bin.

Once extraction of all 3 texture features is completed, they are then concatenated into a single variable that acts as the feature vector, as shown in Figure 4.2 previously. Over the course of 40 extraction-concatenate operations (1 operation per patch), this will result in the creation of an image signature.

4.1.4 The Adaptive Resonance Theory Module

The Adaptive Resonance Theory (ART) module consists of several files that were obtained online from <http://www.mathworks.com/matlabcentral/fileexchange/4306-fuzzy-art-and-fuzzy-artmap-neural-networks>. While the archive also contains files for an ARTMAP implementation, these were not used as an ARTMAP is regarded as a supervised neural network, and therefore, neither applicable nor suitable for the current research. Though the ART algorithm module consists of 9 m-files, only 3 of the files need to be directly implemented in order to run the algorithm. As our implementation of the ART module is similar to the example provided within the ARTExample.m file, our explanations regarding the individual components with the module will be based upon our implementation.

First, all the patch signature values extracted so far (that are contained within the variable *imageTTCellArray*) are passed into the *ART_Complement_Code()* function where their complement code of any particular value, x is calculated to be $1-x$, and interleaved row-wise with the original data. For example, if an array of values is considered to have the following values:

$$\text{array}_{\text{input}} = \begin{bmatrix} 0.4 & 0.2 & 0.6 \\ 0.1 & 0.3 & 0.3 \\ 0.7 & 0.5 & 0.9 \end{bmatrix}$$

Then, the complement-coded form of the array will take the form of:

$$\text{array}_{\text{complement}} = \begin{bmatrix} 0.4 & 0.2 & 0.6 \\ 0.6 & 0.8 & 0.4 \\ 0.1 & 0.3 & 0.3 \\ 0.9 & 0.7 & 0.7 \\ 0.7 & 0.5 & 0.9 \\ 0.3 & 0.5 & 0.1 \end{bmatrix}$$

Due to the nature of this function requiring all array values to be of 1 or less, each and every value contained within *imageTTCellArray* will have their floating points shifted to the left through the use of the function *calibrateinput()* until all values are equal to or less than 1 before being passed into *ART_Complement_Code()*.

Then next step is to create an untrained ART network, based on the size of the variable *ccInput* (which contains the complement-coded values) as well as a vigilance value that possesses a range of values from 0 to 1. Vigilance is a factor within the ART algorithm that influences the scope of a particular category to "recruit" any one [coarseness, contrast, directionality] tuple. In general, over any particular date set, a higher vigilance value would result in more classes or categories, while a lower vigilance would create fewer categories. While the original default value for the

vigilance is set to 0.75, this still did not result in an optimal number of generated categories. Therefore, this value has been substituted with 0.885, although this can be changed by the user as well, in order to observe any change in results.

Once the network has been created and stored within the variable *net*, this network is then trained to determine the number of categories that exist within the data set contained in *ccInput* by calling the function named *ART_Learn()*. This function returns a trained network (in the form of variable *newNet*) and the categories numbers that have been created during the training process (in the variable *cat{i}*).

Each patch signature in the current (i.e. most recently captured) image – which is stored within the variable *imageTT* – is then categorized by *newNet* through implementating the function ART_*Categorize*(). This function returns a 8 * 5 sized array into *newCat*, which contains the category values of each of the 40 patches within *imageTT*. The source code of the *PreRecorded()* function shows the entire process described previously, from creation of the network, to categorizing an existing batch of patch signatures.

4.1.5 Clustering of Detected Categories

Once the category values for each patch of any particular image are determined, they are then segregated into clusters of similar values, as previously demonstrated in Figure 3.2. In this specific implementation, each patch index is related to a corresponding index location within the variable *checked*, which contains

either the values 1 or 0, denoting whether that specific patch index has already been assigned to a particular cluster. The following steps are performed for each and every patch index:

Step 1: If the element at index *i* has been checked, go to Step 4. If it has not, go to Step 2.

Step 2: Let checked(i) = new cluster number.

Step 3: Add the newly checked index value to a history list variable, *breadcrumbs*.

Step 4: Find all neighbouring index elements that have not yet been checked. If at least one neighbor hasn't been checked, set *i* = any unchecked neighbour's index, and go to Step 1. If all neighbours have been checked, set i = last visited index from *breadcrumbs*, delete that index in *breadcrumbs*, and go to Step 1.

This process continues until there are no more index values available in *breadcrumbs*, thus ensuring that every neighbor is of the same cluster number as the first original index involved in the process detailed above. In this implementation, we consider an index's neighbours to be only those that are immediately adjacent on the "vertical" or "horizontal" axis. Those that are on the "diagonal" axes are not considered to be neighbours, as seen in Figure 4.5, where patches which are classified as Category 1 are considered to be 2 separate and distinct clusters.

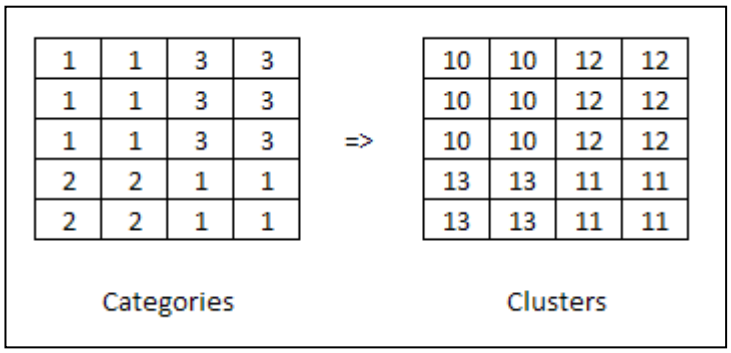


Figure 4.5 Translation of categories to clusters

Following the creation of clusters, the centroid for each cluster is determined. We determine a cluster's centroid through taking the average of its entire member patch's x-y coordinates which can be summarized as:

$$X_{center} = \frac{\sum_{P1}^{Pn}(Xp)}{n} \tag{4.19}$$

$$Y_{center} = \frac{\sum_{P1}^{Pn}(Yp)}{n} \tag{4.20}$$

where X_{center} and Y_{center} are the x and y coordinates for the cluster's centroid, respectively. This method of representing the location of clusters is adequate in almost all cases, except for rare cases where a cluster is ordered in an "irregular" fashion where the centroid is not within an acceptable bounding distance from its related cluster, such as that seen in Figure 4.6. However, as stated before, the circumstances that causes a cluster to be shaped as such are rare and should not impede normal operations of the research implementation.

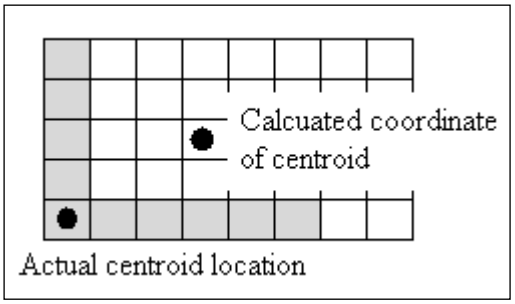


Figure 4.6 The calculated location of a centroid, as opposed to its actual location

Each and every non-trivial cluster and its centroid location are stored in the variables *imageCenter{i}* and *clusters{i}* where *i* is the number of the related image. These variables are necessary in order for the next stage of the implementation to function properly.

4.1.6 Creation of Semantic Information

The purpose of generating clusters from categories is to allow semantic information to be inferred from them through the implementation of the *CreateSemantics()* function. For each and every cluster related to any particular image, the size, location and topographical relationship to all other clusters are recorded. An exception to this rule is applied to clusters which have a size of 1 image patch, as they are considered to be trivial and will not contribute towards a meaningful or significant semantic description of an image scene.

4.1.6.1 Location

Determining the location of a particular cluster is dependent on determining the quadrant in which its centroid is located in. Each and every image is divided into 9 separate and distinct quadrant, which are identified by their coordinates over the x-y axes, which ranges from [1, 1] to [3, 3]. Figure 4.7 below shows the distribution layout of all 9 quadrants over an image of 8 * 5 patches, while Figure 4.8 shows an example of how the location of a cluster is determined.

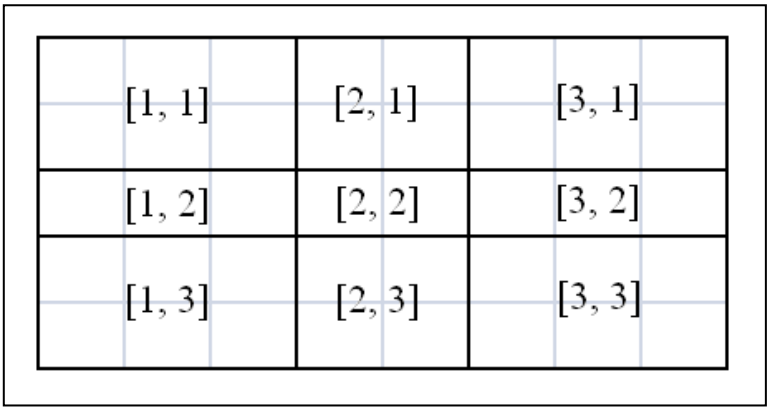


Figure 4.7 The size and identities of all 9 quadrants spread across each image

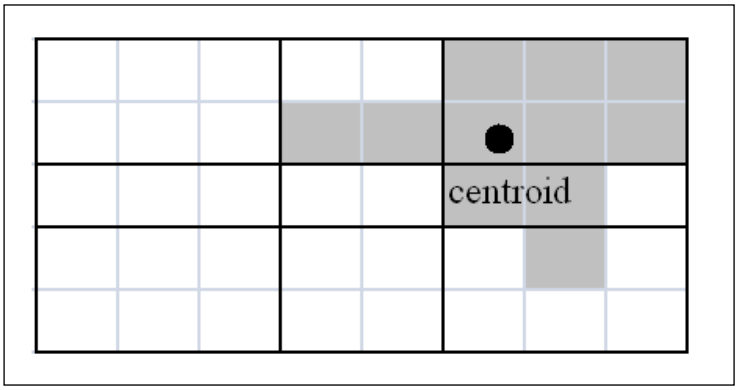


Figure 4.8 This cluster is considered to be in quadrant [3, 1] even though some portions are in other quadrants.

4.1.6.2 Topographical Relationships

Once the quadrant locations of each cluster has been calculated, a record of how each cluster is related to all other clusters is maintained, where high-level descriptors demonstrate the relationship to any particular cluster. An example of how a particular cluster records its relationships with other clusters is shown in Figure 4.9.

relation	left	bottom&left	top	right
relatedTo	2	3	6	8

Figure 4.9 The topographical relationships in which a particular cluster possesses

Whenever a relationship is recorded, its related descriptor describes the relationship that the *current cluster possesses compared to another cluster*. As an example, the owner of the record shown in Figure 4.9 is considered to be on the left from cluster no. 2. A visual example of how this particular owner is related to all other clusters is shown in Figure 4.10.

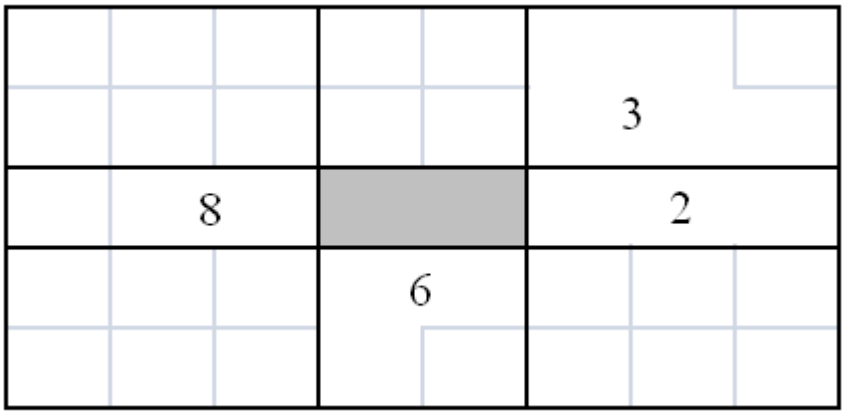


Figure 4.10 A visual example of how a particular cluster (shown in grey) is related to the other clusters as shown in Figure 4.9

Determining which specific high-level descriptor tag to be attached to any particular relationship is dependent on the spatial relationship between the centroid of both clusters, as demonstrated in Figure 4.11, where the candidate descriptor tags for cluster is either bottom, bottom&left, or left, depending on which region the second cluster being compared to is located in. The same method is also applied to determining the other 9 relationships/descriptors as well.

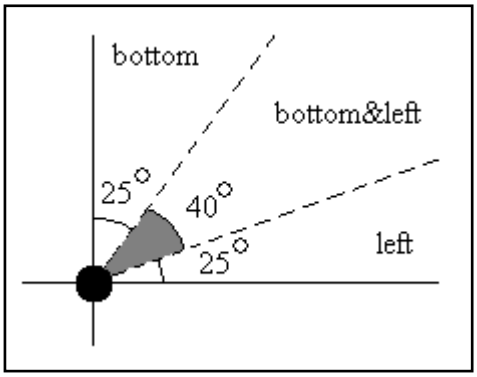


Figure 4.11 Three possible descriptor tags (out of 12 in total) that can be attached to a particular cluster pairing, and the regions which defines them.

4.1.7 Comparison of Semantic Information

Once the semantic information of an image has been completely extracted, it is then compared to its counterpart from the previous image. The differences in size, location and relationship between the two images are each represented as a value between 0 and 100. The greater the value is, the greater the differences for any particular semantic factor.

These three values are then fed as input into the Fuzzy Inference System (FIS), where the fuzzy rules then generate the similarity score of the current image. In the case of the first image, the similarity score generated will always be of the highest value (8.31) as there is no previous image to compare against.

Determining the differences in size is considered to be a trivial operation and is done by adhering to the equation stated below:

$$Size_{diff} = | \min(Size_p, Size_c) / \max(Size_p, Size_c) * 100 | \quad (4.21)$$

where *p* and *c* denote the previous and current image respectively.

4.1.7.1 Comparing Quadrant Locations

Following this, the process of calculating the differences in quadrant locations is performed, which consists of the following steps:

Step 1: Set $Quadrants_L$ to be $\max(Quadrants_p, Quadrants_c)$ and $Quadrants_S$ to be $\min(Quadrants_p, Quadrants_c)$, where $Quadrants_{p/c}$ are arrays containing the centroid locations of the previous and current image clusters, respectively.

Step 2: Calculate the distances between each and every possible centroid pairing:

$$Distances = squareform(pdist(concatenate(Quadrants_S, Quadrants_L)))$$

where the function pdist calculates the Euclidean distance between cluster pairings as a vector, $D(C_1, C_2)$, according to the following formula:

$$D(C_1, C_2) = \sqrt{(C_1 - C_2)\,(C_1 - C_2)'} \quad (4.22)$$

The squareform function is then applied to the calculated distances in order to convert the vector, $D(C_1, C_2)$, into a 2D matrix of values so that element i, j in the matrix, where i < j, corresponds to the Euclidean distance values between centroid i and j within the original data set of concatenate($Quadrants_S$, $Quadrants_L$).

For example, if the arrays of $Quadrants_S$ and $Quadrants_L$ have the following values, where:

$$Quadrants_S = \begin{bmatrix} 1 & 2 \\ 1 & 3 \end{bmatrix} \qquad Quadrants_L = \begin{bmatrix} 2 & 2 \\ 3 & 3 \\ 1 & 3 \end{bmatrix}$$

Applying the pdist() function to concatenate(QuadrantsS, QuadrantsL), will therefore yield a vector with the following values, where :

$$\begin{aligned} pdist(concatenate(Quadrants_S, Quadrants_L)) = [&1.0,\ 1.0,\ 2.2361,\ 1.0, \\ &1.4142,\ 2.0,\ 0,\ 1.4142, \\ &1.4142,\ 2.0] \end{aligned} \tag{4.23}$$

We then apply the *squareform()* function to this vector in order to obtain a more readable format in the form of:

$$\text{Distance}_{SF} = \begin{bmatrix} 0 & 1.0000 & 1.0000 & 2.2361 & 1.0000 \\ 1.0000 & 0 & 1.4142 & 2.0000 & 0 \\ 1.0000 & 1.4142 & 0 & 1.4142 & 1.4142 \\ 2.2361 & 2.0000 & 1.4142 & 0 & 2.0000 \\ 1.0000 & 0 & 1.4142 & 2.0000 & 0 \end{bmatrix} \quad (4.24)$$

Step 3: Obtain only the distance measure values from Distance_{SF} where each cluster from Quadrants_S is compared against Quadrants_L. The values compared between Quadrants_S and Quadrants_L can be represented within Distance_{SF} as shown in Figure 4.12 located on the next page.

The indicated values in Figure 4.12 are then obtained through the following equation, where:

$$\text{Distances} = \text{Distance}_{SF}(1{:}\ \text{size}(\text{Quadrant}_S), \text{size}(\text{Quadrant}_S) + 1{:}\ \text{width}(\text{Distance}_{SF})) \quad (4.25)$$

	Quadrants_S		Quadrants_L		
Quadrants_S	0	1.0000	1.0000	2.2361	1.0000
	1.0000	0	1.4142	2.0000	0
Quadrants_L	1.0000	1.4142	0	1.4142	1.4142
	2.2361	2.0000	1.4142	0	2.0000
	1.0000	0	1.4142	2.0000	0

Figure 4.12 The indicated distance measure values to be extracted. All other values are discarded.

Step 4: For each row *j* within Distances, claim the lowest value in column *k*, where *k* has not been claimed by a previous row. This is done to ensure that every centroid location in $Quadrants_S$ is compared to exactly one (and only one) other counterpart in $Quadrant_L$ with the lowest possible comparison values. As each column *k* is claimed, add *k* to array Claimed. This is to ensure consistency across compared centroids when comparing relationships later on in the next section. Also, each distance calculated is stored within the array *DistanceArray*.

Step 5: Calculate the differences in quadrant location for the *current image* by taking the mean of all differences over the maximum possible distance value (2.8284) and represent it as a percentage, which is carried out by the following formula:

$$Quadrant_{Diff} = \text{mean}(DistanceArray) / 2.8284 * 100; \quad (4.26)$$

4.1.7.1 Comparing Relationships

The process of comparing centroid relationships is performed with the following steps in a manner similar to that in the previous section:

Step 1: Set $Semantics_L$ to be max($Semantics_L$, $Semantics_S$) and $Relationships_S$ to be min($Semantics_L$, $Semantics_S$), where $Semantics_{p/c}$ are vectors containing the semantic information of the previous and current image clusters, respectively.

Step 2: For each value *j* in Claimed, calculate the relationship differences between each k^{th} relationship value in $Semantics_{S,\ j}$ and $Semantics_{L,\ Claimed[j]}$. In order for this to occur, the high-level descriptor tags are converted to angles on a numeric scale according to the 8 cardinal directions as seen in Figure 4.13.

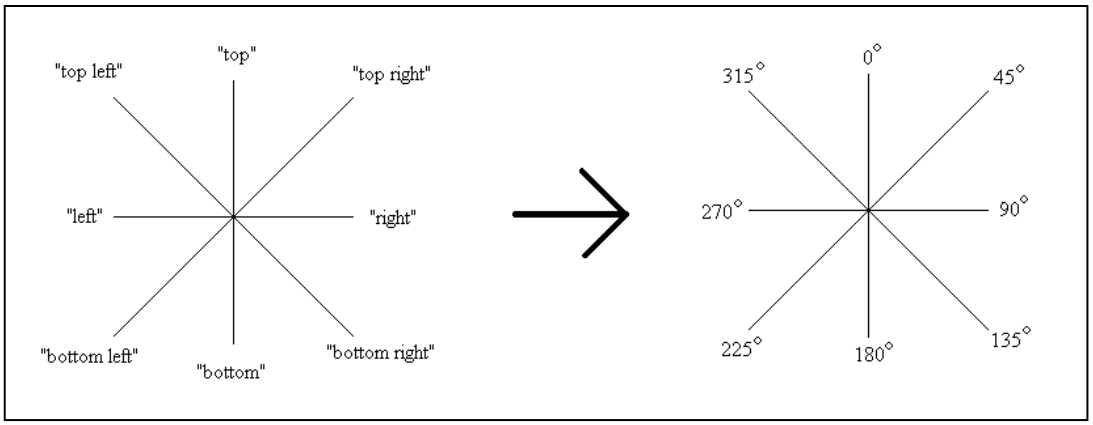


Figure 4.13 Translating high-level descriptor tags into angles (calculated in degrees)

Differences between each relationship are calculated according to the formula on the next page, where:

$$Angle_{Diff} = |\ (Angle_1 - Angle_2 + 180\ \%\ 360) - 180\ | \qquad (4.27)$$

This will result in $Angle_{Diff}$ possessing a value within the range of 0 to 180. Each calculated value of $Angle_{Diff}$ is then stored within the array *RelationshipArray*.

Step 3: Calculate the differences in relationships for the *current image* by taking the mean of all differences over the maximum possible angle difference (180) and represent it as a percentage, which is obtained by using the following formula:

$$\text{Relationship}_{\text{Diff}} = \text{mean}(\textit{RelationshipArray}) / 180 * 100; \quad (4.28)$$

4.1.7.2 Evaluating Differences Through The FIS

Once the values of $\text{Size}_{\text{Diff}}$, $\text{Quadrant}_{\text{Diff}}$ and $\text{Relationship}_{\text{Diff}}$ have been calculated, these three variables are then evaluated with the FIS (previously discussed in Section 4.1.2) through MATLAB's *evalfis()* function, in which the similarity score is obtained though the implementation of the aforementioned function as:

$$\text{similarity} = \text{evalfis}([\text{Size}_{\text{Diff}}, \text{Quadrant}_{\text{Diff}}, \text{Relationship}_{\text{Diff}}], \text{FIS}) \quad (4.29)$$

As stated previously in Section 4.1.2, the similarity score for any particular image ranges from 1.69 to 8.31 as a result of the relationship between the difference values and the membership functions and variables of the FIS.

While the main content of experimental results are demonstrated and explained in the following chapter, we now provide an analysis of how the differences values interact with the FIS in order to produce the final outcome of the similarity score.

Similar to Figure 4.4, the figure below displays the rules which govern how the difference values placed within the membership functions affect the final result.

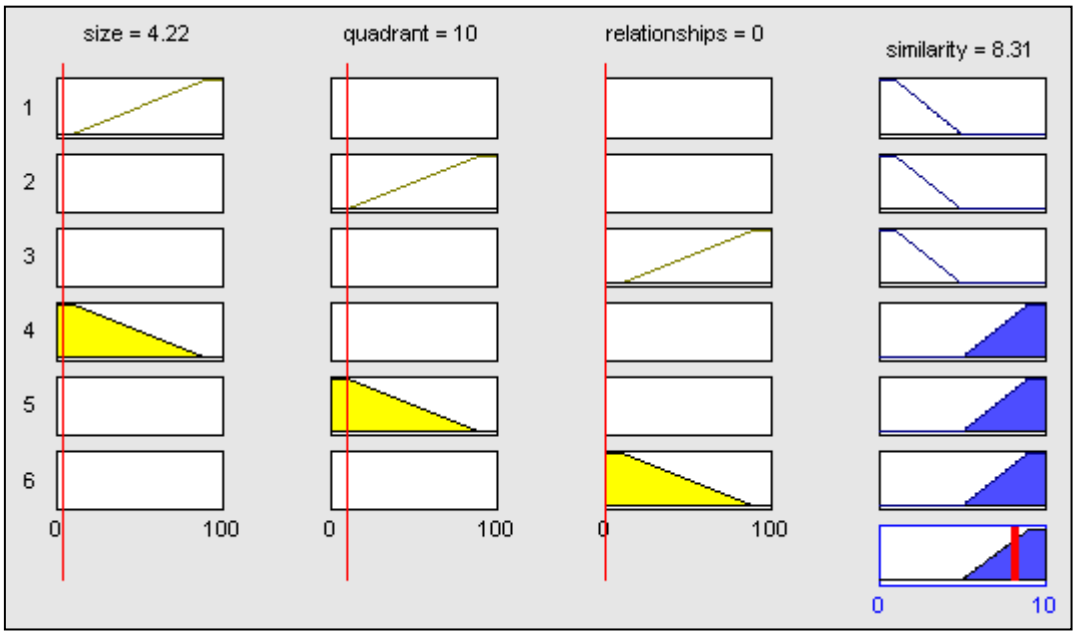


Figure 4.14 The difference values that result in the highest possible similarity score

In the specific case of Figure 4.14, the values of $Size_{Diff}$, $Quadrant_{Diff}$ and $Relationship_{Diff}$ generate a similarity score of 8.31. This is despite the fact that the value of $Quadrant_{Diff}$ has a value of 10, as opposed to the other two difference values of 0. The reason for this phenomenon is due to the membership function defining the similarity between quadrants (in rule 5), where any value than is less than or equal to 10 is still considered to have no difference. This also applies to all other membership functions as well, so an image with a value of 10 for all three difference values will still result in a similarity score of 8.31.

Conversely, all three difference scores will only need to breach the 90% value threshold in order to achieve the lowest similarity score of 1.69, and do not necessarily have to each possess a perfect score of 100% for this to occur.

4.2 Chapter Summary

This chapter focuses upon the details regarding the technical implementation of the proposed semantic SLAM model. An overview of the relationships between the MATLAB file modules that constitute the SLAM model is shown, along with the flow of data that traverses through the model; from the initial video sequence that is submitted as input, to the generated similarity score.

A general description of the operations related to the files modules is then provided. This is then followed by a description of the Fuzzy Inference System (FIS), as well as the creation of its membership functions, rules, and input/output variables. Following this, the focus of discussion is then aimed towards the Tamura Texture features in which the methods of extracting the features of coarseness, contrast and directionality from each and every patch of an image frame are explained.

The process of classifying image patches through the Adaptive Resonance Theory (ART) algorithm is then discussed, detailing the procedure of converting patch signatures into complement-coded values that are accepted as input by a newly created (and untrained) ART network. The clustering of classified image patches is also explained, as it serves as a pre-cursor towards obtaining the semantic information pertaining towards any particular image frame.

The extraction of semantic data from the aforementioned clusters is then explained. In the current model, the extracted data consists of a cluster's location and

size, as well as its topographical relationship with other existing clusters. The degree of similarity between two image frames is determined by comparing their semantic data through the FIS, which generates the final similarity score.

Chapter 5

Patch Categorization and Image Reconstruction

The main purpose of this chapter is to analyze a selection of 20 patches that represent any particular category (due to the extremely large amount of patches involved) and determine if the categorization process results in the patches for each category to contain a particular semantic inference or content. This is done by calculating the mean Tamura Texture feature values for each category, and determining the distance of each patch signature to this mean vector. The smaller the distance, the more indicative that a particular patch is semantically representative of the category that it is in. Any anomalous patches will be further analyzed to provide an explanation regarding the basis in which it was selected to be in that particular category.

To this end, we first introduce the video streams (shot at a resolution of 6480 * 480 in 16-bit color) involved in the experiments that were conducted in the current research effort. Frames from key points of each video stream will be shown, along with the time stamp in which the frame was captured.

All experiments were conducted with a vigilance factor of 0.885, with an equal weight distribution of 1.0 for the input variables of size, quadrant location, and relationship differences.

While the MATLAB environment implements the SLAM model (which was discussed previously in Chapter 4), a third-party video indexing software was utilized to extract the frames from each video stream, at an average of 5 frames per second.

5.1 Patch Categorization

The first video stream consists of a 64 second traversal divided into 322 individual frames. The traversal begins from a corridor environment, through a room at the 23rd second and back into the previous corridor at the 55th second. Several frames of the entire traversal are shown in Figure 5.1, along with the time stamp denoting the moment which is being represented. Key frames from other video streams are also shown in Appendix A and consist of one other indoor environment set (Appendix A.1.), and two outdoor environment sets (Appendix A.2. and A.3.).

Figure 5.1a
0m 0.4s – Beginning of the video stream

Figure 5.1b
0m 10s - Traversal through the corridor

Figure 5.1c
0m 20s - The end of the corridor just before entering the room (to the left)

Figure 5.1d
0m 23s - Entering the room

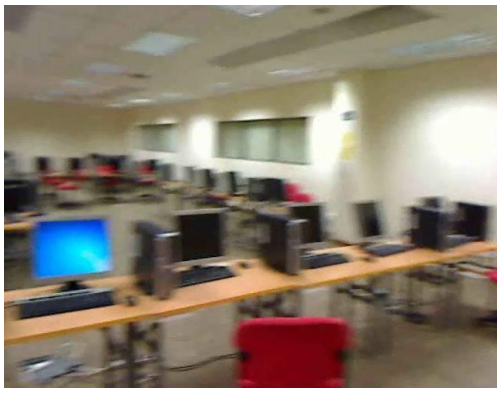


Figure 5.1e
0m 26.00s – General layout of the room

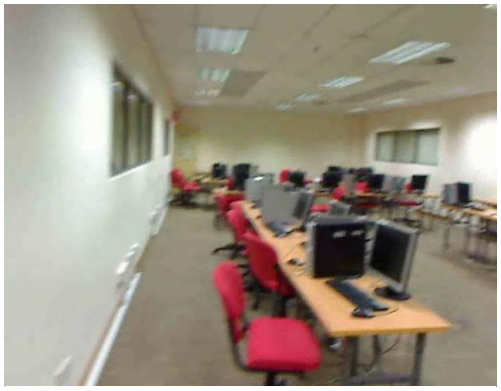


Figure 5.1f
0m 30s – Traversal through the room

Figure 5.1g
0m 40s – Traversal through the room

Figure 5.1h
0m 51s – Facing the end of the room

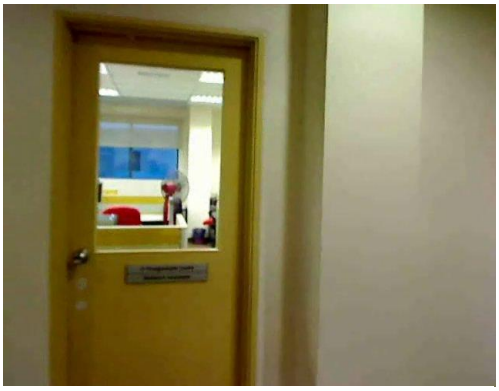


Figure 5.1i
0m 55s – Re-entering the corridor

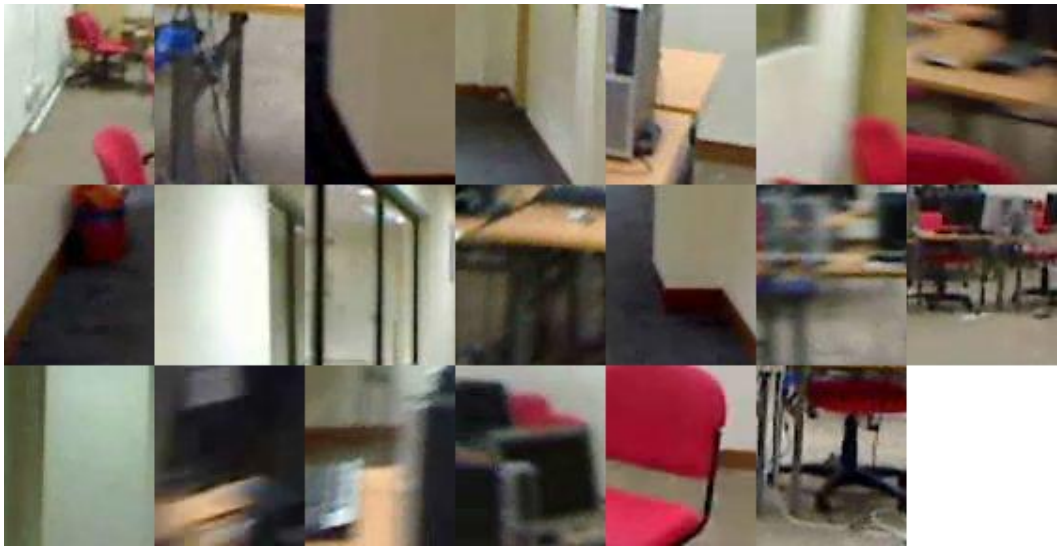
Figure 5.1j
0m 57s – Turning to face the corridor

Figure 5.1 Selected frames (out of 322) from a video stream traversing through a corridor, a room and then back

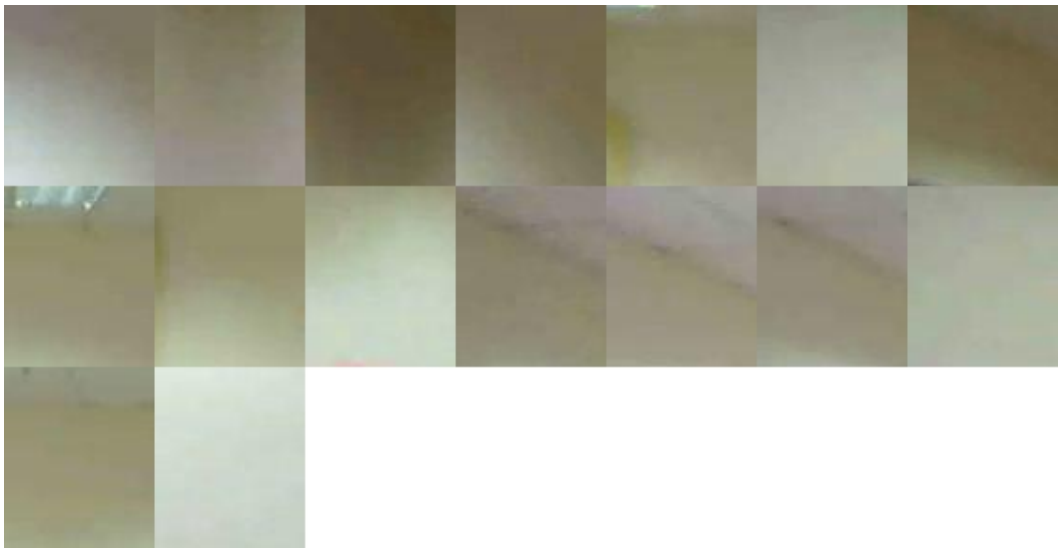
Categorization of the patches contained within the video stream has resulted in the generation of 20 separate categories. The results, which are displayed in Appendix B.1, shows a maximum of 20 patches for each category, ranging from patches that have the least distance to their category's mean patch signature (top left) to those with the largest distance value (bottom right). Categorized patches for the other environment set are also shown in full in Appendix B. Several of the categories from Appendix B.1 are shown in Figure 5.2. and will serve to aid with analyzing the performance of the categorization process.

Corridor Set 1 - Category 1

Corridor Set 1 - Category 6

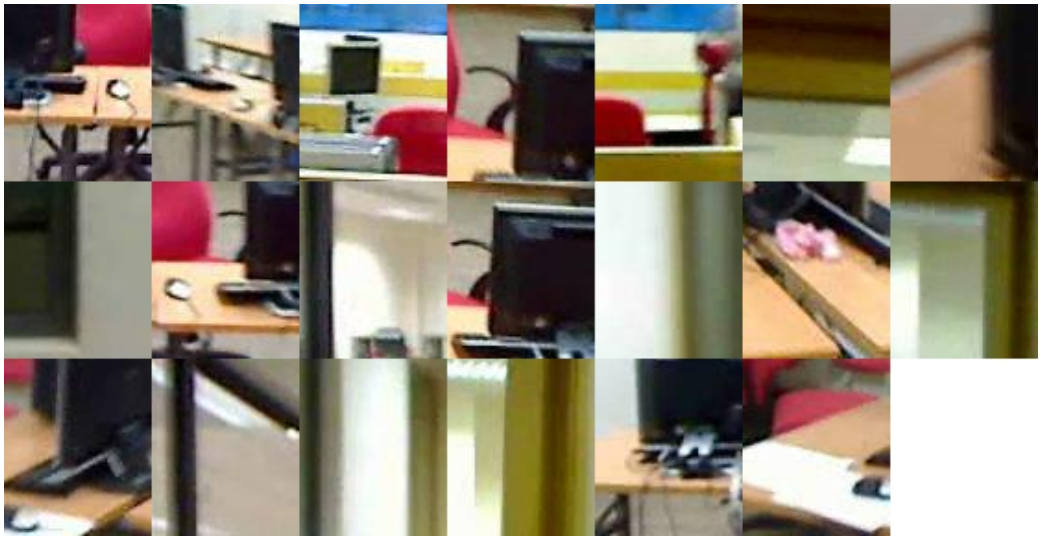


Corridor Set 1 - Category 9

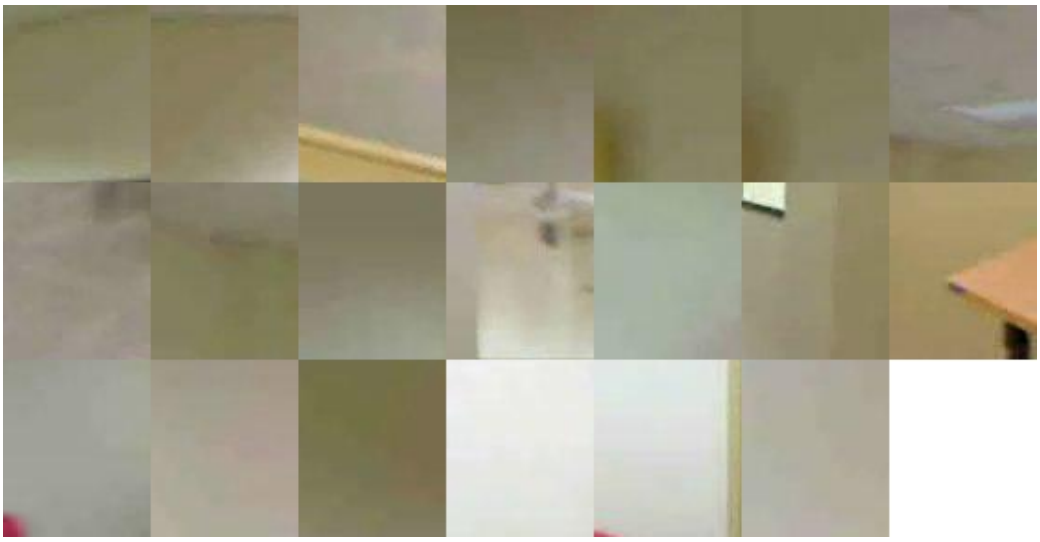


Corridor Set 1 - Category 16

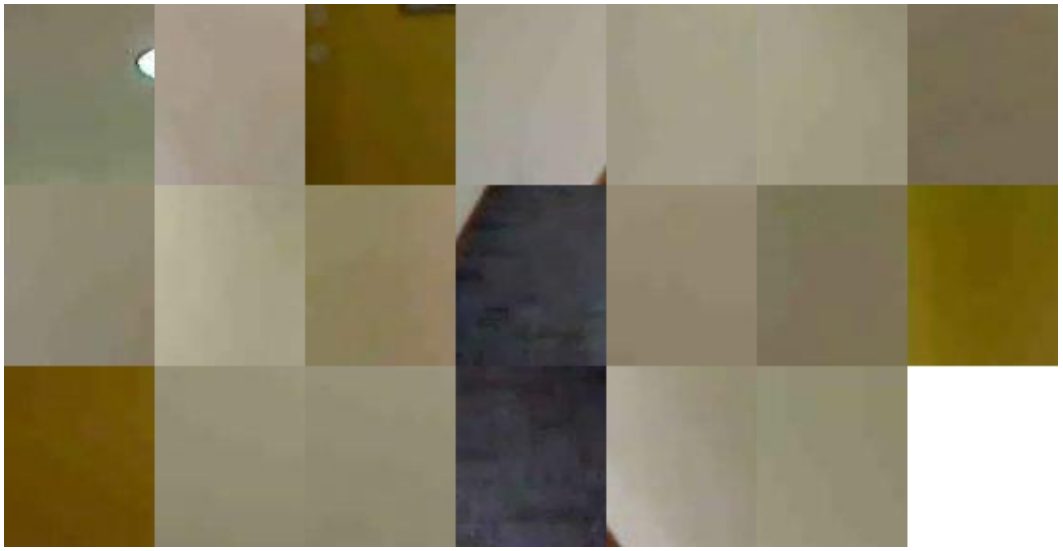
Corridor Set 1 - Category 14



Corridor Set 1 - Category 15



Corridor Set 1 - Category 17



Corridor Set 1 - Category 20

Figure 5.2 Selected categories (out of a total of 20), containing patches from best-fit (top-left) to worst-fit(bottom-right)

A visual inspection of each of the categories shown in Figure 5.2 will demonstrate that while certain categories contain patches with a diverse range of semantic context (examples of these are Categories 1 and 6), there are also categories containing patches with a highly consistent semantic context (such as those from Categories 16 and 20). Another point of interest is where the context of some categories is not restricted to generic indoor features such as walls, floor or ceiling, but specific household items (i.e. categories 14 and 15 consistently have patches that have computers within them).

In order to determine if the "randomness" of categories with no specific semantic context is caused by the patch signatures, their values and distance from the category's mean value will be listed in Figure 5.3, where details for Category 9 are

shown. As a comparison, the details from a more consistent category are shown as well. In this case, Category 16 was chosen for this purpose.

		Categories							
		9				16			
		coarseness	contrast	directionality	distance from mean	coarseness	contrast	directionality	distance from mean
Mean		25.3993	39.2325	0.8743	N/A	26.5412	69.5587	0.8203	N/A
Patch No.	1	25.1589	39.2545	1.0592	0.3041	27.8927	69.668	0.929	1.3603
	2	24.7896	38.0425	0.3388	1.4403	28.2604	69.3859	1.0051	1.7377
	3	26.2089	40.7136	1.8356	1.9424	27.494	67.6696	1.0666	2.13
	4	27.6505	39.7222	0.2673	2.3825	27.8065	72.2427	0.6285	2.9734
	5	26.2974	41.8314	1.0337	2.7543	25.6187	66.0672	0.6574	3.6149
	6	28.3865	39.8697	0.317	3.1048	25.1487	73.1771	0.4041	3.8994
	7	24.3518	35.9133	0.431	3.5087	29.2378	72.7397	1.3235	4.2004
	8	27.1221	35.8097	0.4112	3.8598	30.4729	71.4601	0.5522	4.3755
	9	24.2247	43.2387	0.3877	4.2031	29.5776	72.9241	1.4565	4.5771
	10	22.4987	35.7393	0.096	4.6067	22.5781	66.7339	0.3004	4.8945
	11	26.2802	34.4288	0.1244	4.941	28.0245	64.1672	0.531	5.5993
	12	26.9036	44.33	0.6264	5.3206	31.2948	65.7187	0.75	6.1112
	13	23.6628	33.8531	0.1361	5.7007	28.068	63.4317	1.4144	6.3422
	14	20.9789	43.294	0.6525	6.0071	22.9784	63.5639	0.5532	6.9787
	15	28.6625	33.723	0.5686	6.4106	25.9885	62.4769	0.3133	7.1214
	16	22.3201	33.2448	0.3864	6.7508	27.5115	77.7785	0.9628	8.278
	17	25.8362	46.298	0.2263	7.1086	21.6216	77.0453	1.3859	8.9761
	18	25.8534	46.6773	0.8578	7.4587	26.7388	60.1378	0.506	9.4282
	19	23.4615	31.6361	2.2346	7.9568	27.7581	79.6145	1.5028	10.1521
	20	20.5526	46.4044	0.0312	8.697	28.8391	81.9136	0.9948	12.5679
STD. DEV.		2.2463	4.6753	0.5618	2.3159	2.4784	5.9184	0.3883	2.9969
MIN		20.5526	31.6361	0.0312	0.3041	21.6216	60.1378	0.3004	1.3603
MAX		28.6625	46.6773	2.2346	8.697	31.2948	81.9136	1.5028	12.5679

Figure 5.3 The patch signature values for categories 9 and 16

In Figure 5.3, the features of coarseness and contrast for Category 16 actually have a higher standard deviation compared to Category 9. In fact, only the directionality for Category 16 has a more tightly bound standard deviation value compared to Category 9. This raises the possibility that spread of values for one feature is capable of determining the semantic consistency of a particular group of image patches.

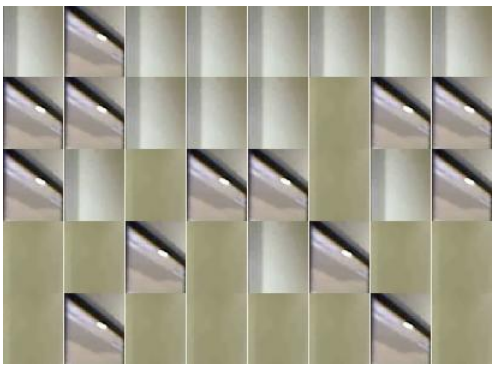
5.2 Image Reconstruction

While the categorization of patches provides an indication if semantic context is being properly segregated by the proposed model, reconstruction of each image frame from the video stream can also be performed to determine if the semantic context for different portions of each frame are correctly interpreted as well. This operation is not integral to the functionality of the proposed model, but would provide some insight as to how the model interprets each image from its own point of view.

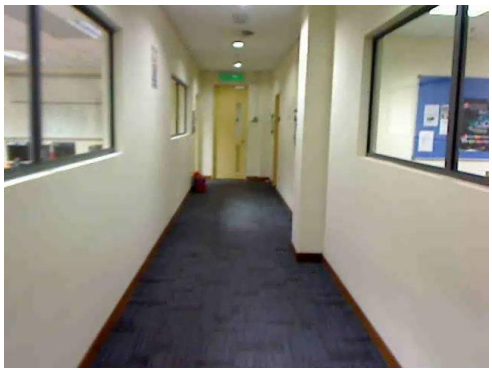
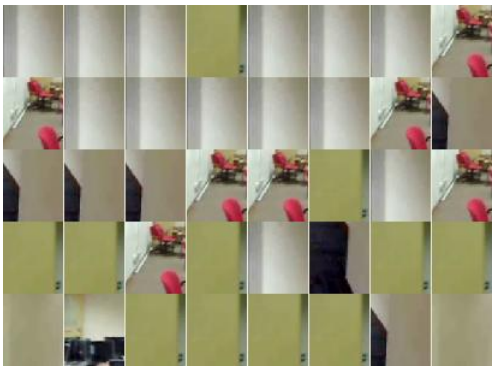
As the category number of each and every patch for any particular image frame is already determined, reconstruction can be performed by selecting the "best-fit" patch from the related category, and replacing the original patch which generated the related category number. This can be seen in Figure 5.4, which contains a comparison between selected reconstructed images and their original counterparts from Figure 5.1. Comparisons between all images for every set are also shown in Appendix C.

Comparisons show that some images do not show a correlation between the original and reconstructed images in a semantic context, such as in Comparisons B, E, and G. However, instances such as those for Comparisons C and F indicate that reconstruction of images with proper semantic context is possible. In addition, the reconstructed images in Comparisons A, D, H and J demonstrate a clear indication of boundary detection between areas with different semantic context, even though the actual context isn't fully accurate.

Another issue upon visual analysis of the comparisons is the issue of consistency. Comparisons A and B do not differ significantly in terms of semantic and visual representation, yet their reconstructed counterparts indicate differing sets of categories.

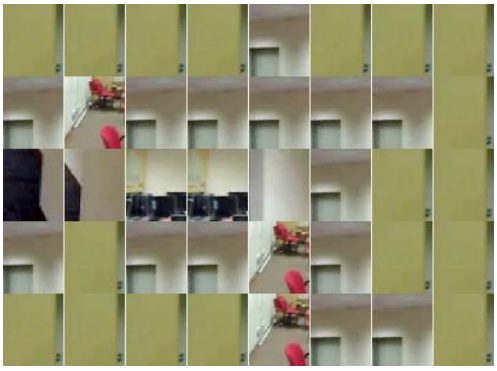



Comparison A
0m 0.4s – Beginning of the video stream

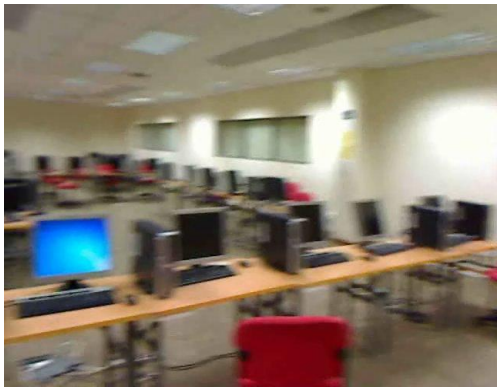



Comparison B
0m 10s - Traversal through the corridor

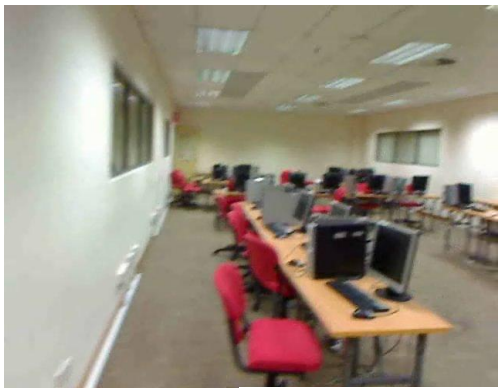
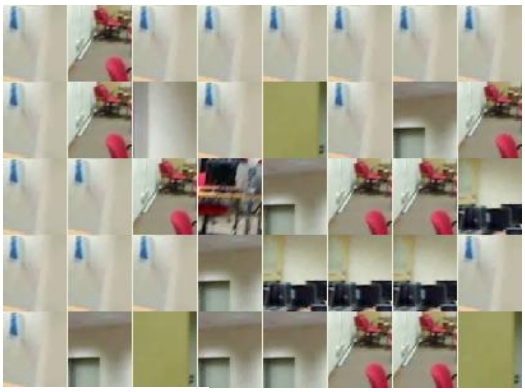

Comparison C
0m 20s - The end of the corridor just before entering the room (to the left)

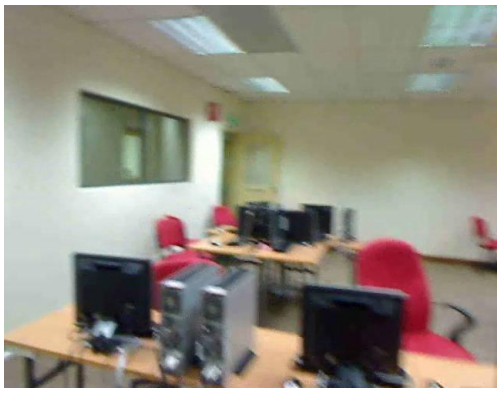
Comparison D
0m 23s - Entering the room



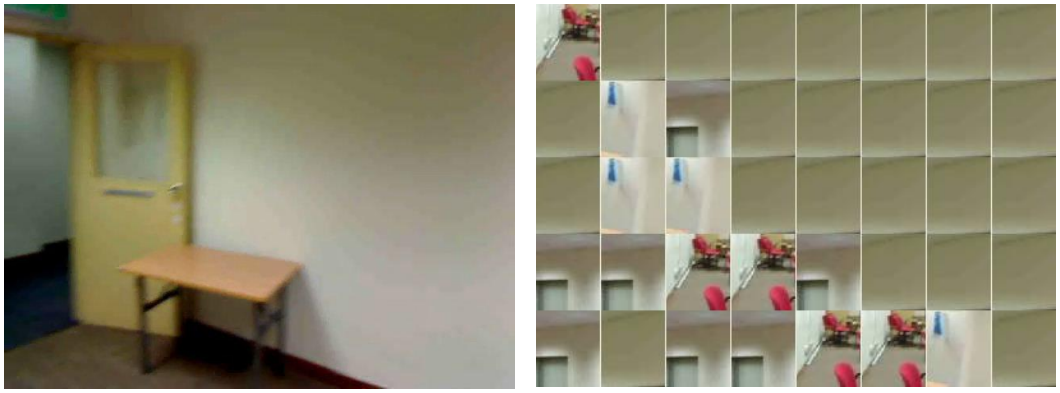

Comparison E
0m 26.00s – General layout of the room

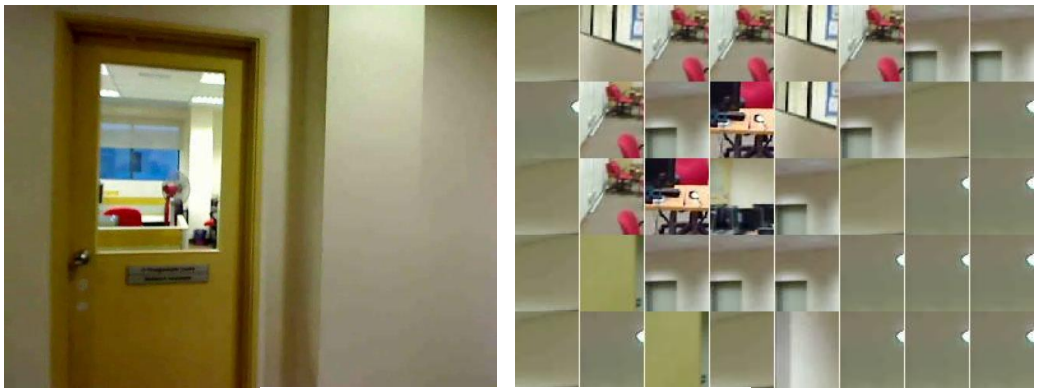
Comparison F
0m 30s – Traversal through the room




Comparison G
0m 40s – Traversal through the room



Comparison H
0m 51s – Facing the end of the room



Comparison I
0m 55s – Re-entering the corridor

Comparison J
0m 57s – Turning to face the corridor

Figure 5.4 Comparisons between original and reconstructed key frames

5.3 Chapter Summary

The focus of this chapter is to determine if the categorization of image patches would result in a consistent semantic context, if the patches were to be viewed according to the categories they are classified as. From the results shown in this chapter and Appendix B, the potential for conducting such a categorization process with the required amount of consistency is evident, but not at a satisfactory level yet as categories still exist with differing semantic contexts.

The categorization of image patches serves as a basis for image reconstruction, which presents a visual representation of how the proposed model interprets each image frame. As the image reconstruction process is highly dependent on patch categorization, it is expected that improved performance on the latter would result in similar improvements in the former. However, as of this moment, accurate image reconstruction from a semantic aspect is noticeable, but requires further refinement for greater reliability.

Chapter 6

Definition of Semantics

While the previous chapter allows us to inspect the visual interpretation of the proposed model towards entire image frames, not every categorized patch will contribute towards the semantic description of any particular frame. As previously described in Chapter 4.1.6, clusters with a size of 1 are considered to be trivial and will not contribute towards forming the semantic descriptors of size, location and relationships that is associated with each and every cluster.

The purpose of the current chapter is to analyze this operation to determine if the generated descriptors are sufficient to provide adequate semantic context towards the processed image frames. In order to provide a certain measure of consistency and to aid in comprehension of the overall process, the results shown within this chapter will be based upon the set of images shown in the previous chapter.

6.1 Cluster Filtering

Figure 6.1 demonstrates the process where clusters are implicitly generated by patch categorization, and the subsequent operation of discarding trivial clusters.

Semantic consistency between the original image frame (Figure 6.1a) and the set of generated clusters (Figure 6.1d) is crucial, as the clusters are directly involved in determining the information contained within the semantic descriptors which serve as the semantic counterpart to each image frame.

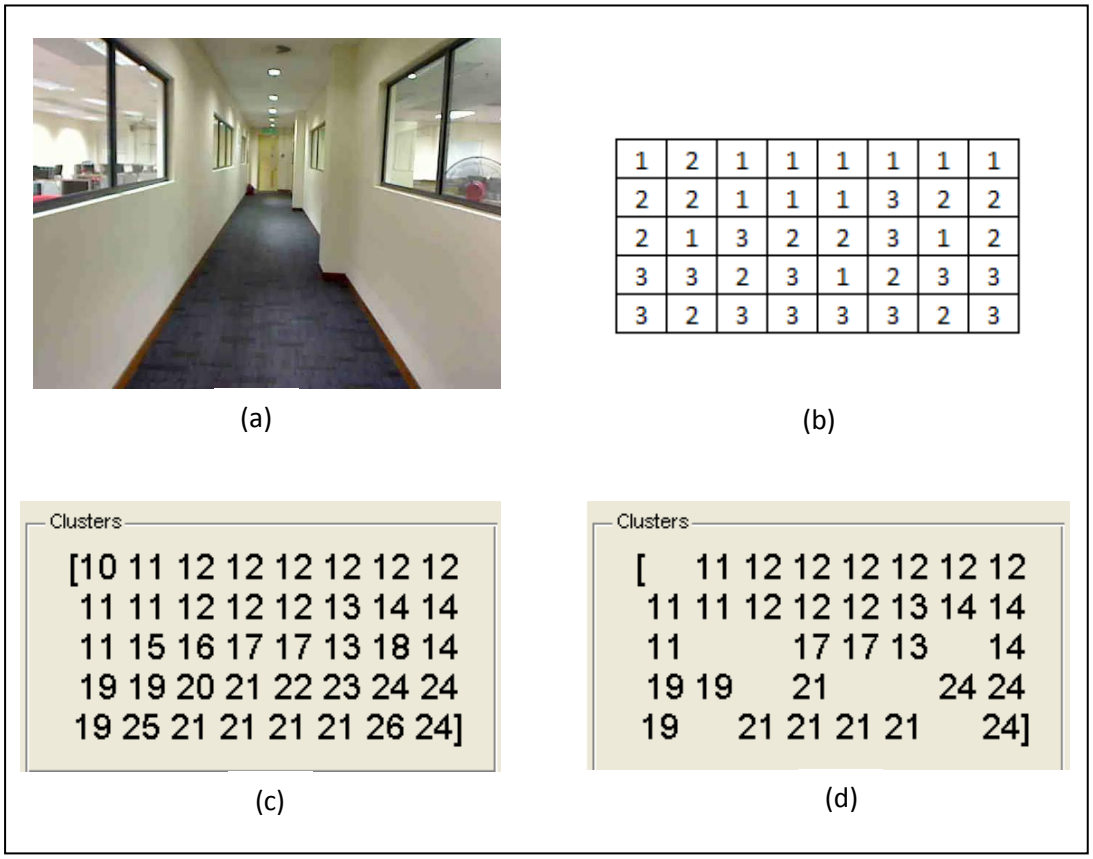


Figure 6.1 The filtering process of non-trivial clusters where (a) is the original image frame, (b) is the set of categorized patches, (c) are the generated clusters, and (d) is where trivial clusters of size 1 are discarded

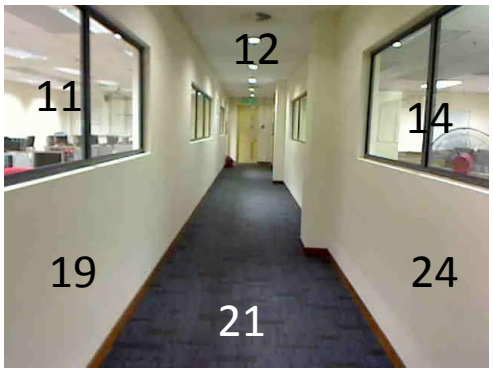
Figure 6.2 will display a comparison between the key frames shown in Figure 5.1 and their associated layout of clusters for the first corridor video stream, while Appendix D contains the comparisons for the other video streams. In order to provide greater ease during comparison, any cluster that possesses a visually explicit semantic context will have its identification number annotated on the original key frame. It should also be noted that the scope of cluster numbers are limited to their associated

images. Therefore, the semantic context for any particular cluster number can differ across separate image frames.

The results shown in Figure 6.2 provide an indication of the generated clusters' feasibility in representing the semantic context of any particular key frame. While the number of visually explicit clusters serve as an indication in regards to the current model's effectiveness, the amount of coverage each cluster possesses in relation to its semantic counterpart is equally crucial.

From Figure 6.2, Comparisons A, D, G, H, and J contain clusters with the most ideal representation of semantic context, as it is trivial to determine the image portion being represented by a particular cluster. In addition, the relative size and location of each cluster as compared to its associated key frame gives an adequate representation of the entire image as a whole.

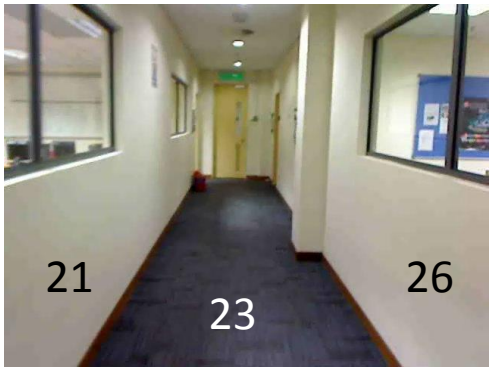
Comparisons C, E, and I show the least amount of explicit correlation between the distribution of clusters and semantic context of their respective images. In the case of comparisons C and I, it is likely that the vertical directionality – which is unique to these images – is the cause for the lack of specificity in the layout of clusters.



Clusters

```
[   11 12 12 12 12 12 12
 11 11 12 12 12 13 14 14
 11       17 17 13    14
 19 19    21       24 24
 19    21 21 21 21    24]
```

Comparison A
0m 0.4s – Beginning of the video stream



Clusters

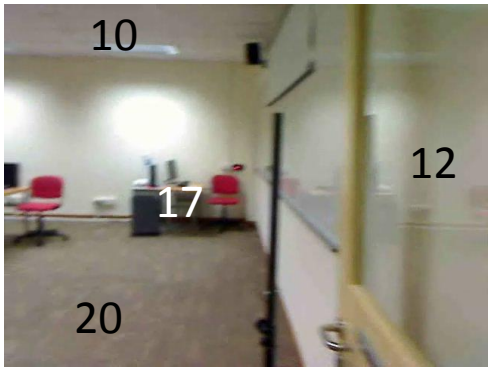
```
[10 10 10    10 10 10
    10 10 10 10 10
 16 16 16 17 17
 21 21    23       26 26
       23 23 23 23      ]
```

Comparison B
0m 10s - Traversal through the corridor

Clusters

```
[10 11 12 12          12
 10 11 16 12 12    12 12
 11 11 16 12 12 12 12 12
 11 18 16 12 12 19 12 12
 11 18 16    12 19 12 12]
```

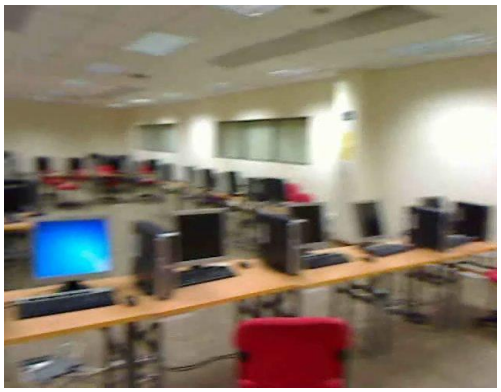
Comparison C
0m 20s - The end of the corridor just before entering the room (to the left)



Clusters

```
[10 10 10 10 11 12 12 12
       11 11 11 11 11 12
       17 17    11 12 12
    20 21 21 22 11 12 12
 20 20 20 20 22 11 11 12]
```

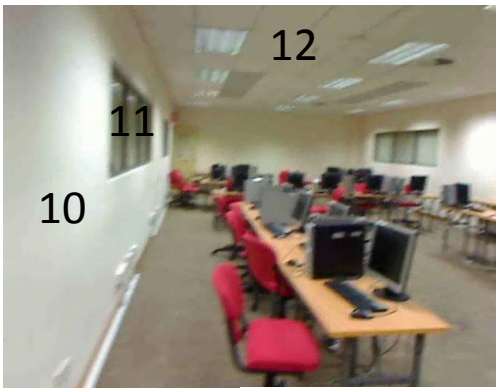
Comparison D
0m 23s - Entering the room



Clusters

```
[10 10 10 10 10
 14 14 15       18 18
       15 22 22 18 18
    25 25 22 18 18 18 26
 27 27 27 27          26]
```

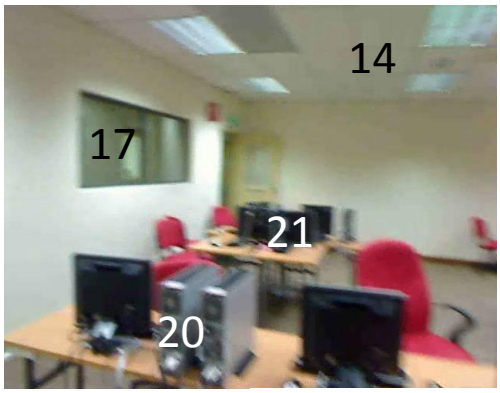
Comparison E
0m 26.00s – General layout of the room



Clusters

```
[10 11 12 12 12 12 12 12
 10 11    12    12
 10 10          20 20
 10 10 10 22 23 23 23
 10       22 22 27 27   ]
```

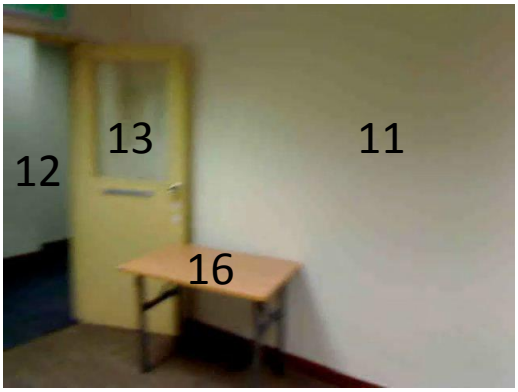
Comparison F
0m 30s – Traversal through the room



Clusters

```
[          14 14
 17 17 17  14 14 14 14
 19 20 20 21 21
 19    20 20 20 20 20
 20 20 20 20          ]
```

Comparison G
0m 40s – Traversal through the room



Clusters

```
[   11 11 11 11 11 11 11
 12 13    11 11 11 11 11
 12 13 13 11 11 11 11 11
 15 15 16 16    11 11 11
 15    19 19 20 20    11]
```

Comparison H
0m 51s – Facing the end of the room

Clusters

```
[      12 12 13     15 15
    17         13        22
 23 17         26     22 22
 23     26 26 26 22 22 22
 23                22 22 22]
```

Comparison I
0m 55s – Re-entering the corridor

Clusters

```
[          13 14     16 16
 17 17     13 14 14 16 16
           14 14 16 16
 23 23     16 16 16 16 16
 23 23 23     16 16 16 16]
```

Comparison J
0m 57s – Turning to face the corridor

Figure 6.2 Comparisons between original key frames and generated clusters

6.2 Semantic Descriptors

Following the process of discarding clusters of size 1, the remaining clusters are used to generate the semantic descriptors that are intended to serve as a unique identifier for each frame within the video stream. To serve as an example, the image in Comparison A within Figure 6.2 would generate the following descriptors:

Clusters		*11*	*12*	*13*	*14*	*17*	*19*	*21*	*24*
Size		*4*	*9*	*2*	*3*	*2*	*3*	*5*	*3*
Quadrant		*1*	*2*	*3*	*3*	*5*	*7*	*8*	*9*
Relationships	*11*		***L***	***L***	***L***	***T & L***	***T***	***T & L***	***T & L***
	12	***R***		***T & L***	***T & L***	***T***	***T & R***	***T***	***T & L***
	13	***R***	***B & R***		***L***	***R***	***T & R***	***T & R***	***T & L***
	14	***R***	***B & R***	***R***		***R***	***T & R***	***T & R***	***T***
	17	***B & R***	***B***	***L***	***L***		***T & R***	***T***	***T & L***
	19	***B***	***B & L***	***B & L***	***B & L***	***B & L***		***L***	***L***
	21	***B & R***	***B***	***B & L***	***B & L***	***B***	***R***		***L***
	24	***B & R***	***B & R***	***B & R***	***B***	***B & R***	***R***	***R***	

L = Left R = Right T = Top B = Bottom

Figure 6.3 An example of a set of semantic descriptors associated with an image frame

The generation of semantic descriptors as shown in Figure 6.3 provides a high-level description of the various areas of distinct semantic context for a particular image. While it is possible for several image frames to possess a similar relationship table, the risk of ambiguity is offset by the quadrant location recorded for each particular cluster, providing further detail towards the extent of the relationship between clusters.

6.3 Chapter Summary

Following the process of image reconstruction through patch categorization, the purpose of this chapter is to determine if the creation of non-trivial patch clusters

(i.e. size > 1) will result in semantic consistency with their related image frames. The method in which this is performed is similar to the previous chapter, in which visual comparisons on clusters are made against the local semantic context of individual frames to determine if any explicit correlation between the two is present.

From the results shown in this chapter and in Appendix D, indoor environments have a more visually explicit relationship towards their image frames as compared to outdoor environments. However, much improvement can be made towards making the observation of this relationship much more consistent throughout all images, regardless of the surrounding environment.

Chapter 7

The Semantic Location Resolver

Once the process of generating semantic descriptors for each captured image frame is completed, they are then compared to their counterparts associated with a referenced image frame previously described in Section 4.1.7. The objective of this chapter is to determine if the Fuzzy Inference System (FIS) within the proposed model is capable at deducing a change in semantic context as the environment within a video stream transitions from one area to another.

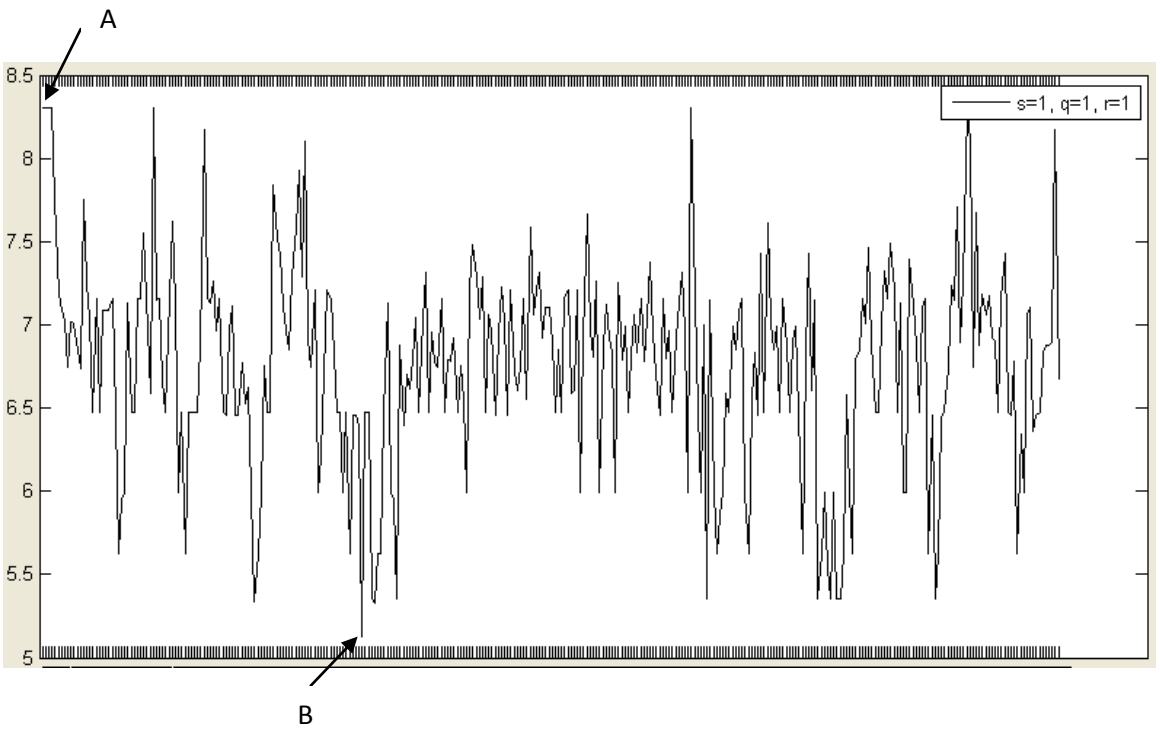
In order to determine the consistency of the proposed model, the results of 4 separate video streams (2 indoor and 2 outdoor environments) will be analyzed, while any notable observations will be discussed upon. Key frames from the first indoor environment can be found in chapter 5.1, while the others are listed in Appendix A.

7.1 First Corridor Video Stream

The 3 input variables of the FIS that determine the similarity of size, location and relationships of clusters have an initial value of 1, while the threshold value for the similarity score has been set to a value of 5, and a vigilance value of 0.885. Once processed by the proposed model, the first corridor video stream – with a length of 64 seconds divided into 322 image frames – generates the similarity scores as shown in

the graph as Figure 7.1. In order to assist in interpreting the results, annotations will be made, showing the frame in which a particular score is recorded.

The first observation to be made is that none of the image frames recorded a score lower than the threshold score value of 5. The global minimum occurs at event B (which is visually analogous to Comparison C in Figure 6.2), and is semantically relevant to the change in environmental context from a corridor to a room setting. However, the reference image was not updated to reflect this change as the similarity score generated at this point (5.127) did not pass below the threshold value.



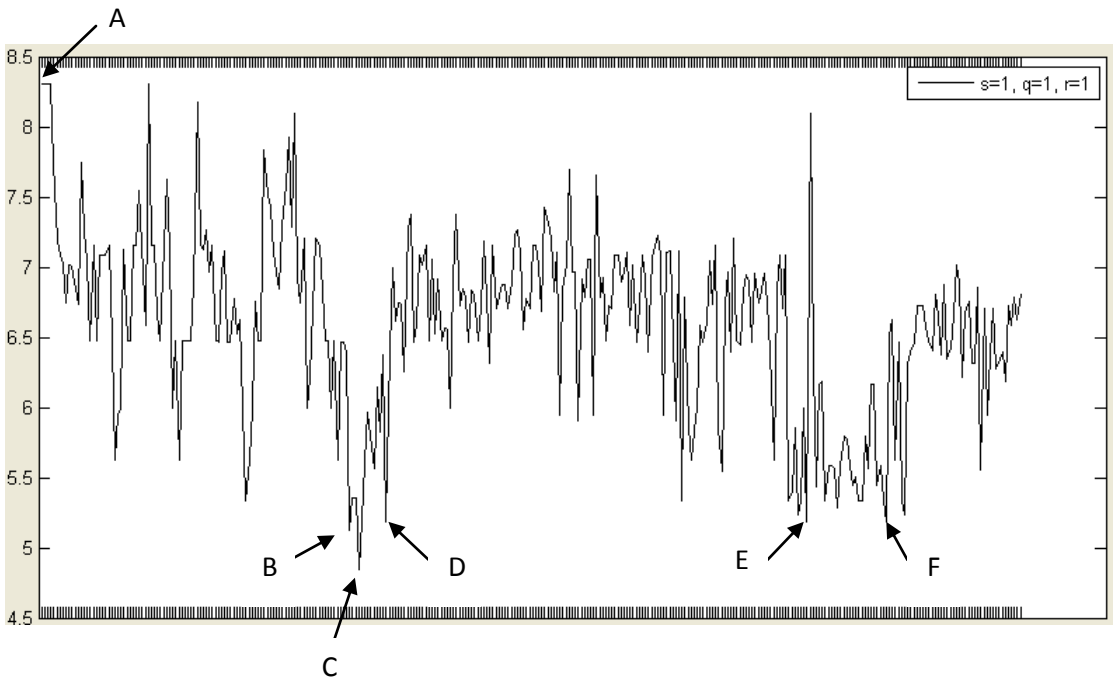
A – Video stream submitted as input. Reference image set to image 1.
B – Camera faces end of first corridor. Similarity score value of 5.127.

Figure 7.1 The similarity scores for each of the 322 frames from the first corridor set

Because of this, all image frame following event B are still compared to image 1, and do not serve as an accurate or semantically relevant comparison. Following

this, the experiment was conducted once again with the threshold score increased to a value of 5.2 to determine if improvements can be made towards the generated results, which are shown in Figure 7.2 below.

This second iteration of the experiment yields some interesting results. Examination of the images which have yielded scores below the threshold value show that for this particular video stream, changes from one semantic context to another within the local environment are able to be detected by the proposed model. From this particular set of results, 3 notable observations have been identified:



A – Video stream submitted as input. Reference image set to image 1.
B – Camera faces end of first corridor. Similarity score value of 5.127.
C – Camera faces end of first corridor. Similarity score value of 4.842.
D – Camera faces room. Score value of 5.181.
E – Camera faces wall within room. Score value of 5.181.
F – Camera within first corridor again. Score value of 5.181.

Reference images updated to current image at each annotated event.

Figure 7.2 Results for the first corridor set, with a modified threshold score value of 5.2.

Observation 1: Between (and including) events B and D, their generated similarity score values were lower than the threshold value at 3 different occasions. While these events occur within a time span short enough to be considered trivial (14 frames or 2.4 seconds), an ideal operation would have a sufficiently low similarity score only at event D. In order to aid determining the cause for this phenomenon, Figure 7.3 displays the image and clusters associated with events B to D.

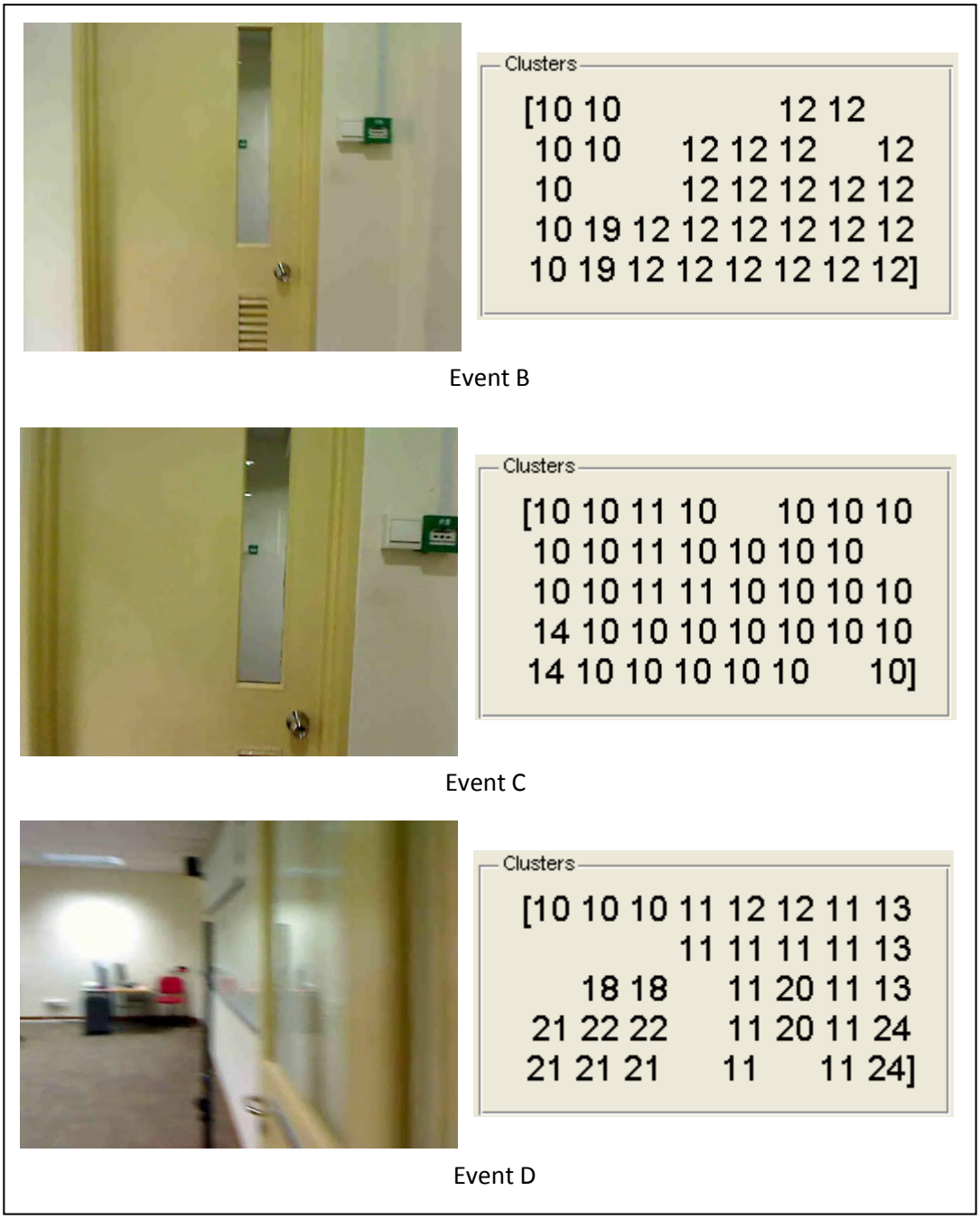


Figure 7.3 Image frames and significant clusters associated with events B to D.

The generation of a low score value for event B is easily explained, as the semantic areas of interest of the related image has a uniform structure, as opposed to the generally symmetrical nature of the reference image within the corridor.

As the images associated with event B and C are very similar, a more detailed inspection in regards to their semantic clusters is required to derive an acceptable explanation for the low similarity score generated by event C.

Upon inspection of the significant semantic clusters for events B and C, it becomes clear as to the reason event C scored lower than the threshold value. The two clusters are very dissimilar in terms of size, location and relationships, and therefore, cause an extremely low similarity value to occur. This observation serves as reinforcement to a statement previously made in chapter 6, where the process of representing semantic context though clusters needs to be further refined in order to prevent occurrences like this particular one from arising.

The similarity score for event D has a similar explanation to that given for event B, in which a change in semantic context would result in a change in the overall structure and distribution of the related clusters (as this is the intended method in which the proposed model operates upon). However, in the case of event D, it would also be preferable if the image could be better represented by clusters with a more visually identifiable layout.

Observation 2: Ideally, the image associated with event E should not have scored lower than the threshold value. Once again, this is similar to event C, where two or

more fairly similar (from a visual and semantic aspect) images possess different cluster distributions, as shown in Figure 7.4.

Clusters

[10 10 10 10
10 10 10 10 10 10 10
10 10 10 10 10 10 10 10
17 10 10 10 10 10
17 17 17 20 20 20]

Before Event E (time = 0m 50.0s, score = 5.991)

Clusters

[10 10 10 10 10 11 11
10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10
13 13 10 10 10 14 14
13 13 13 17 17 17 17]

Event E (time = 0m 50.2s, score = 5.181)

Figure 7.4 The cluster distribution and score values for event E and the image frame immediately preceding it.

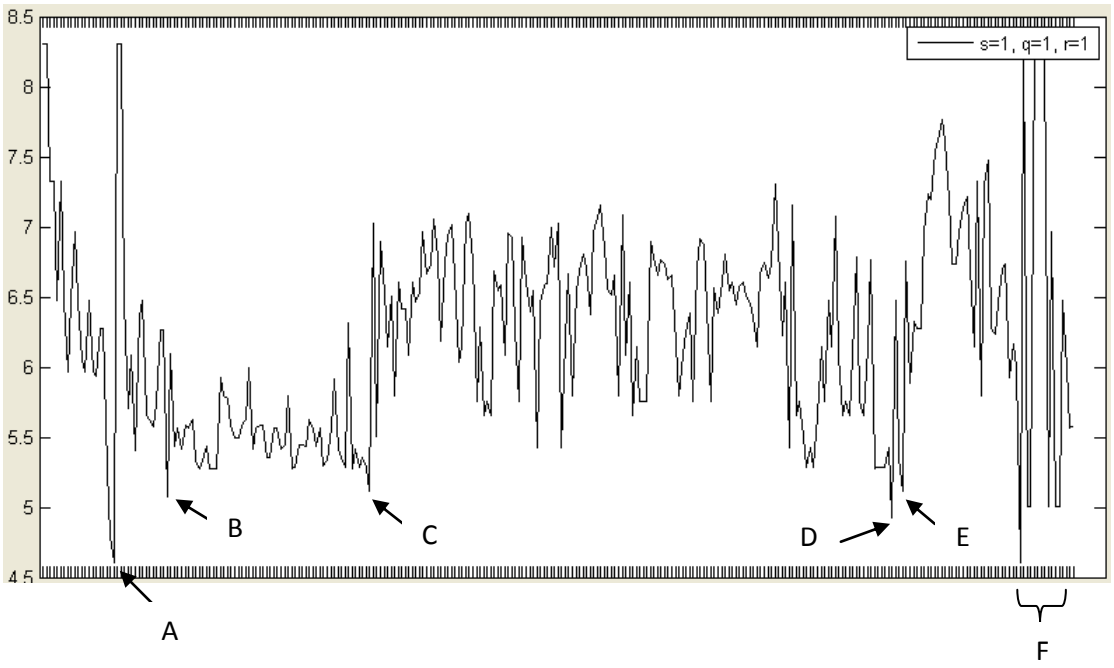
While the dissimilarity between the clusters in Figure 7.4 is not as pronounced as that between event B and C, the presence of two additional clusters (11 and 14) in addition to other differences in event E is sufficient to lower the similarity score below the threshold value. While better cluster representation should be emphasized again, the roles of the 3 similarity factors (size, location and relationships) should be further investigated due to the close similarity between the clusters of event E and its preceding image.

Observation 3: Event F is an ideal example of the proposed model at work, where the detection of a change in semantic context is a singular event within an observable time frame (approx. 5.2 secs since event E), and at the correct location (when the camera is transitioning from the room to the corridor). Figure 7.5 shows the moment in time where this event occurs.

Figure 7.5 The image frame for event F

7.2 Second Corridor Video Stream

The second video stream (shown in Appendix A.1) is a shorter, but more complex indoor environment, traversing 3 corridors and 2 different room environments, and lasting for 58 seconds divided into 291 frames. Using the same parameter values as the previous experiment (with a threshold value of 5.2), Figure 7.6 on the next page contains the resulting similarity scores that have been generated.



A – Camera traverses first corridor. Score value of 4.6.
B – Camera traverses first corridor. Score value of 5.074
C – Camera reaches end of first corridor. Score value of 5.114.
D – Camera turns after end of third corridor. Score value of 4.922.
E – Camera turns after end of third corridor. Score value of 5.111.
F – 6 low score values when the camera turns left and right in the lounge area.

Figure 7.6 Results for the second corridor set.

The results from Figure 7.6 are not as encouraging compared to the previous corridor set. Out of the 6 events noted, only events D and E registered low scores at the proper moment in the video stream. Event C is an observation that has been made previously on the first corridor set, while events A, B and F should not have occurred under ideal circumstances.

Events A and B occur under an interesting premise: where the semantic context of both images are identical (corridor), but the differences in semantic *content* are somewhat dissimilar. Figure 7.7 provides a visual example of this statement.

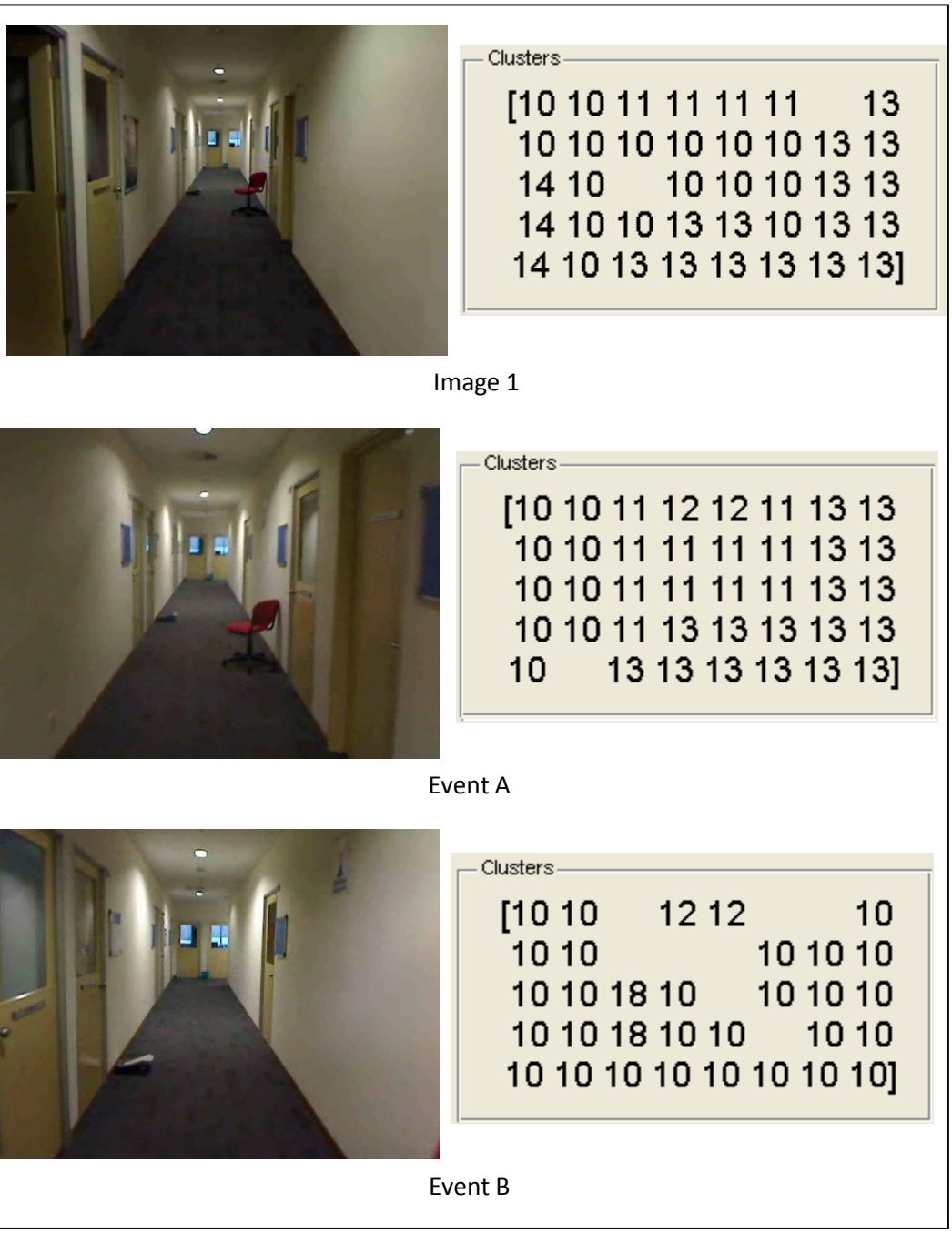
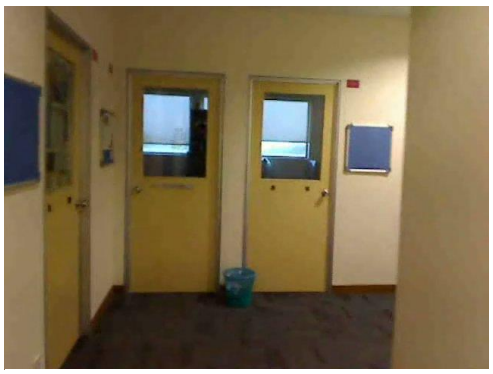


Figure 7.7 Image frames and significant clusters associated with image 1 to event B.

While all three image frames in Figure 7.7 are of the first corridor set, the image from event A contains a blank wall on the leftmost portion, whereas the opposite is true of image 1 and event B. Therefore, event A is incorrectly classified as the beginning of a different environment.

The case of event C is similar to that of event B for the first corridor set, where a door is encountered at the end of a corridor and thus, triggering a low score below the threshold value. However, in this case, the corridor further extends to the right rather than transition into a room with a different semantic context, as in the first corridor set, and therefore event C should have not occurred under ideal conditions. Figure 7.8 contains the image frames of event C as well as the preceding and subsequent environmental surroundings.



Clusters

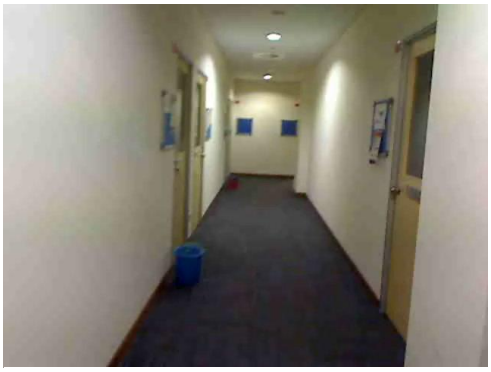
```
[   11 11 11 11 11    13
    11    16 17 17 17 13
    11 11 16 16 19 19 13
       11 16 11 19 19 13
       11 11 11 11 19 13]
```

Time = 0m 16s

Clusters

```
[10 10 11 11 11 11    13
 10 10 10 10 10 10 13 13
 14 10    10 10 10 13 13
 14 10 10 13 13 10 13 13
 14 10 13 13 13 13 13 13]
```

Event C (time = 0m 18.6s)



Clusters

[10 11 11 13 13
10 10 18 19 19 20
10 10 10 18 19 19 20
10 10 23 23 19 20
10 10 23 23 23 20]

Time = 0m 20.267s

Figure 7.8 Image frames and clusters of event C and its preceding and subsequent time moments

The time frame in which events D and E occur (shown in Figure 7.9) is also similar to events B to D from the first corridor set, where multiple instances of low scoring values are calculated in a short period of time. In this case, the time difference between events and E is 0.6 seconds.

Clusters

[10 10 10 10 10 10 10 10
10 10 10 10 10 10 11 12
10 10 10 10 10 10 11 12
10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10]

Event D

Clusters

[12 12 12 12 13
16 17 13
16 16 17
16 16 16 16 16 16
16 16 16 16 16]

Event E

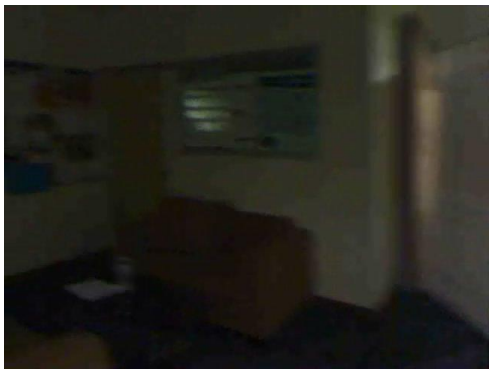
Figure 7.9 Image frames and clusters associated with events D and E.

Event F is composed of 6 separate instances where – over a time period of 2.2 seconds – the calculated similarity score is below the threshold value. The image frames and clusters for these instances are shown in Figure 7.10.

Clusters

[10 10 10 10 10 10 10
10 10 10 10 10 10 10
10 10 10 10 10 10
10 10 10 10 10 10 10 10
10 10 10 10 10 10]

Time = 0m 55.2s, Score = 4.6



Clusters

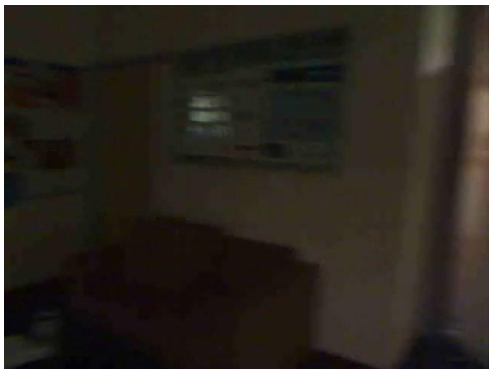
[10 10 10 10 10 10 11 10
10 10 10 10 10 10 11 10
10 10 10 10 10 10 11 10
10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10]

Time = 0m 55.6s, Score = 5

Clusters

[10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10
10 10 10 10 10 10 10
10 10 10 10 10 10 10]

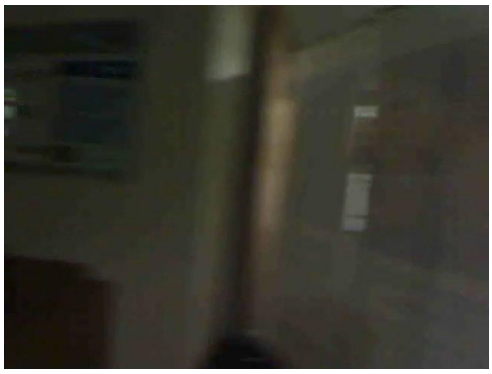
Time = 0m 55.8s, Score = 5



Clusters

[10 10 10 10 10 10 10
10 10 10 10 10 10 10 12
10 10 10 10 10 10 10 12
10 10 10 10 10 10 10 13
10 10 10 10 10 10 13]

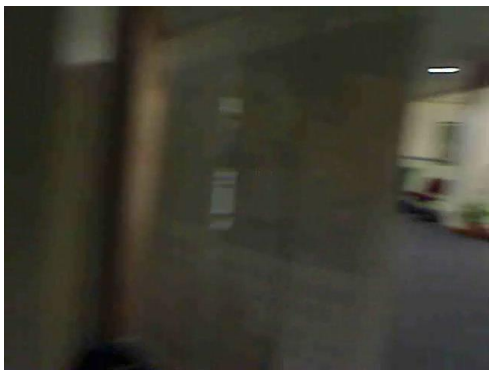
Time = 0m 56.8s, Score = 5



Clusters

[10 10 10 10 10 10 10
10 10 10 10 10 10 10
10 10 10 10 10 10 10
10 10 10 10 10 10 10 10
10 10 10 10 10 10 10]

Time = 0m 57.2s, Score = 5



Clusters

[10 12 10 10 10 10 10
10 10 12 10 10 10 13 14
10 10 10 10 10 10 13 14
10 10 10 10 10 10 10 10
10 10 10 10 10 10 10 10]

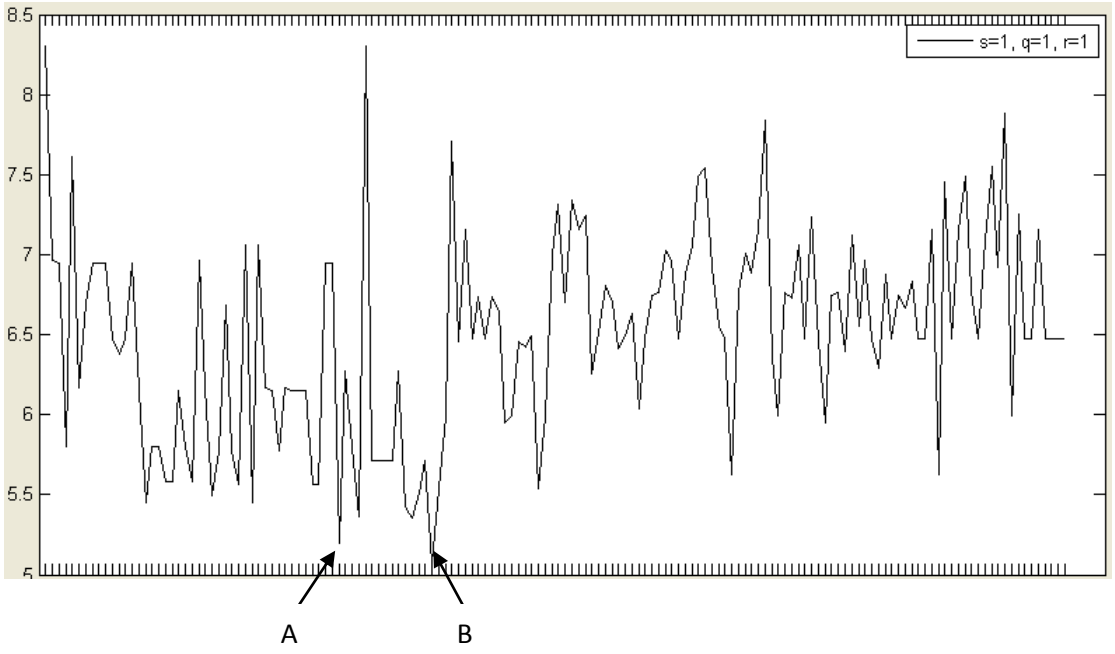
Time = 0m 57.4s, Score = 5

Figure 7.10 The 6 low-scoring instances in Event F

Two possible explanations for the frequent occurrences of low scores present are: unfavorable lighting conditions in the lounge area could serve as one possible reason as this does not occur in other portions of the video stream, which are well-lit. Another possible explanation is due to the constant rotation of the camera within the lounge area, which could also be the cause of Event D as well. However, in order to confirm either of these explanations as a definite cause, more research is required.

7.3 First Outdoor Video Stream

Experiments have also been conducted to determine the viability and potential of the proposed model operating in outdoor environments, which possesses a less defined and structured semantic context. The first outdoor video stream is located within a park and consists of travelling along only one semantic context (that of a pathway). It has a time length of 22.2 seconds divided into 154 individual frames. Key frames from this video stream can be found in Appendix A.2 while the generated similarity scores (with the same settings as previous experiments) can be seen in Figure 7.11.



A – Camera traverses along path. Score value of 5.198.
B – Camera traverses along path. Score value of 5.054.

Figure 7.11 Results for the first outdoor set.

Ideally, the generated scores would not have any value below the threshold value of 5.2. However, from Figure 7.11, two separate instances of low-scoring value occur at Events A and B, which can be inspected visually in Figure 7.12.

Clusters

[10 10 10 10 11 11 10
10 10 10 10 10 10 10 10
13 13 13 13 13 13 13 13
13 13 13 13 13 13 13 13
13 13 13 13 13 13 13 13]

Image 1

Clusters

[10 10 10 10
15 10 10 10 10
15 15 10 10 10 10 10 10
15 15 15 15 15 15 15 15
15 15 15 15 15 15 15 15]

Event A

Clusters

[11 12 15 16
11 11 12 12 12 15 16 16
11 11 12 12 12 12 18 18
19 19 20 20 12 22 22
19 19 20 20 22 22 22 22]

Event B

Figure 7.12 Image frames and clusters of image 1 and events A and B

It is possible for Event A to have been avoided, as its similarity score is extremely close to the threshold (with a difference of 0.002), and better semantic

representation though clusters would likely ensure a higher score value. This is true in the case of Event B as well, though the formation of its clusters is more diverse (8 clusters) than those of Event A (2 clusters).

7.4 Second Outdoor Video Stream

The second outdoor video stream consists of the camera traversing along a pathway for approximately 11.6 seconds (58 frames), then moving on to grass for the remaining 10.6 seconds (53 frames) for a total of 22.2 seconds (111 frames). Similarity scores for this video stream are shown in Figure 7.13.

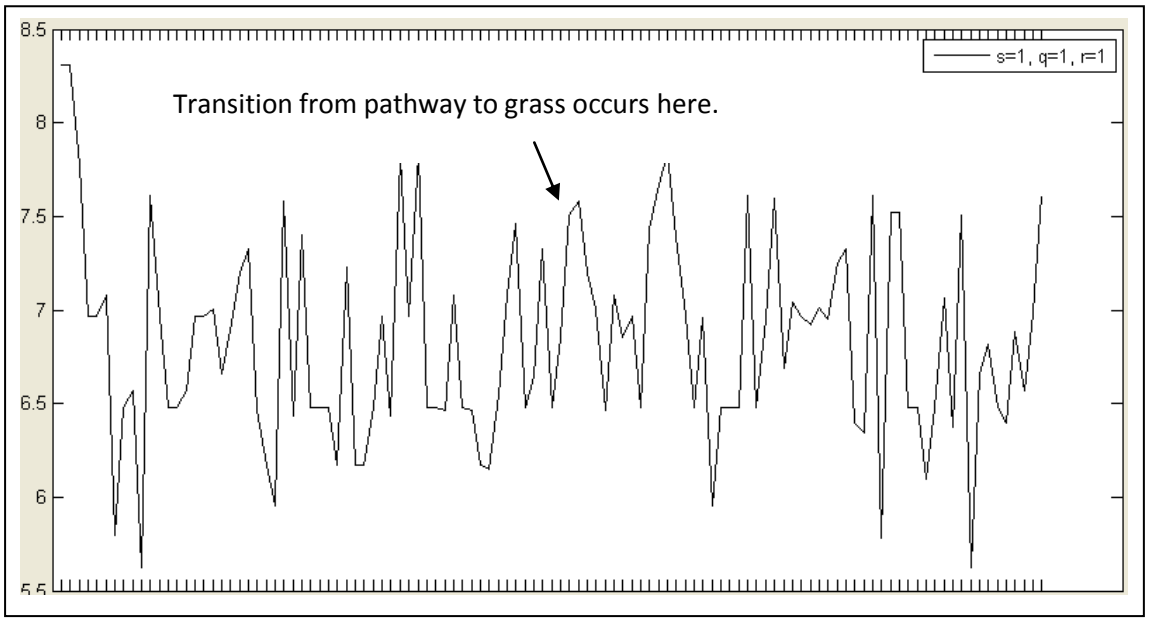
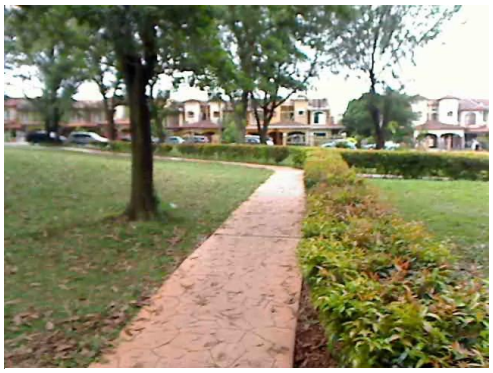


Figure 7.13 Results for the second outdoor set.

Figure 7.13 indicates that the proposed model does not register any change in semantic context during the entire video stream. However, during the moment of image 58 (as indicated), the similarity score obtained a value of 7.503 when compared

to image 1 as the reference image. A visual inspection of both these images and their related clusters is shown in Figure 7.14.



Clusters

[12 12 12 12 12
14 14 14 12 12 15 15 16
17 17 14 17 12 16 16
17 17 17 17 17 16 16
17 17 17 17 16 16 16 16]

Image 1

Clusters

[10 10 12
17 17 12 12 12 12
20 20 20 20 21 21
20 20 20 20 20 20 20 20
20 20 20 24 24 20 20 20]

Image 58

Figure 7.14 Image frames and clusters of image 1 and 58

Figure 7.14 demonstrates a potential issue regarding the proposed model, where the dissimilarity between the distributions of both image clusters does not justify the high value of the generated similarity score. Therefore, the fuzzy rule set currently implemented should be further refined in order to overcome such an issue where a low score should be triggered.

7.5 Chapter Summary

In summation, the proposed model does hold some promise when being implemented within indoor environments, where the semantic content is highly structured and strictly defined. However, the model does not perform as well when experiments were conducted within outdoor environments. Due to the results obtained from this chapter, two issues will have to be resolved before the model can be fully implemented with satisfactory results:

(1) Better representation of the semantic content within an image frame is needed. Visual inspection of clusters shown in this chapter indicate that improvements in this aspect are necessary, are certain frames have been observed to have been improperly represented due to incorrect/inconsistent distribution of clusters.

(2) To determine the extent to which the fuzzy rules influence the overall scoring system. This is to ensure that explicitly different cluster distributions are not assigned a high similarity score (as shown in the second outdoor video stream). If necessary, investigate other methods of representing relationships between clusters (i.e. semantic networks).

In order to make further contributions towards the general knowledge and application of semantic forms of SLAM, resolving these issues should be considered as future work that are proposed to be carried out subsequent to the submission of this thesis.

Chapter 8

Conclusion

A summary of each of the chapters will now be provided, describing their intent and contribution made towards this body of work. Chapter 1 introduces and describes the concept of SLAM. The main approaches of conducting SLAM were discussed and the idea of a semantic form of SLAM was put forth. A research question and research objectives were established to provide a framework into determining the potential and feasibility of conducting real-time semantic SLAM through CBIR.

Chapter 2 provides details upon specific real-time implementations of semantic SLAM that are related towards the proposed model. Image retrieval methods are also discussed in this chapter. Chapter 3 outlines and discusses the various stages of the proposed semantic SLAM model, where images from a video stream are to be represented by clusters of patches. The nature of these clusters is determined by the semantic content of the images, and serves as a unique identifier for their associated image frame. These identifiers are then compared against each other to determine if their respective images are considered to be in an area with the same semantic context.

Technical details on the currently implemented model are discussed in Chapter 4. An Adaptive Resonance Theory (ART) algorithm was selected as the classifier of

choice to categorize patches, while a Fuzzy Inference System (FIS) was chosen in order to calculate the similarity between image frames. Within the FIS, 3 input variables (associated with the size, location and relationships of clusters) with 2 membership functions per variable are used to determine the value of similarity.

Chapters 5 to 7 discuss the various experiments that were undertaken to determine the performance of the proposed model at different stages of the semantic SLAM pipeline. Due to the visually-focused nature of the current research effort, key frames from certain moments within the video streams are shown and compared against the categorization and clustering process.

8.1 Main Contributions

The research effort described within this thesis possesses several contributions towards the field of semantic SLAM, which include:

(1) A 4-stage model designed to conduct a semantic for of SLAM was introduced, along with a specific implementation involving the usage of an ART algorithm and a fuzzy logic ruleset.

(2) Demonstrating the feasibility of implementing three Tamura Texture features within the proposed model. The experimental results shown in Chapters 5 to 7 are derived from the involvement of the Tamura Texture features at every stage of the proposed model.

(3) Determining the capability of the proposed model to accurately categorize image patches into various categories that have a semantic context that is visually explicit when inspected with the human eye.

(4) Demonstrating the capability of the proposed model in detecting a transition between areas that possess different semantic context.

8.2 Summary of Performance

The performance of the proposed model is intended to answer the research question posed in Chapter 1, which is: "Can we apply CBIR techniques to the domain of SLAM to generate a robust SLAM model?"

From the results obtained from the experiments, it can be observed that the proposed model has the potential of conducting the process of semantic SLAM within an indoor environment. In order for this to occur, the following actions are required in order to amend the liabilities that have been observed within the system:

(1) Investigation of a more suitable method of representing the semantic content of a particular image. Experiments have shown that the implementation of Tamura Texture features to represent a particular patch within an image is capable of outlining the boundaries between local areas of semantic context. However, a

significant number of cases appear in which the categorization of patches do not accurately correspond to their semantic context, especially in outdoor environments.

(2) Determine an ideal rule set within the FIS to ensure that the scoring system is consistent during comparison of images. Currently, the distribution of similarity scores is erratic, even though the threshold score is generally crossed at the right time moment; the comparison between several continuous image frames can yield scores with significantly different scores even though the semantic context generally remains the same.

As the image comparison process partially depends on how patches are categorized, the act of solving the first issue can also possibly result in solving the second as well.

8.3 Future Work

In order to resolve the issues that were discussed previously, several courses of action are required to take place that are to be considered future work that is to commence after the submission of this thesis:

(1) Determine an appropriate method of patch representation. 3 texture features are clearly insufficient in order to adequately separate the semantic content of an image, and therefore, the inclusion of other features to act as a patch signature needs to be explored.

(2) Investigate the degree of influence in which each of the three cluster features (size, location and relationships) affect the similarity score. All the experiments previously discussed have operated under the condition where all three cluster features possess equal weights. Determining an "ideal" combination that results in similarity scores that correlate with the change in semantic context will require a significant amount of time, due to the large amount of possible weight value combinations.

Reference List

1. Berens, J., Finlayson, G. D. & Qiu, G. (2000). "Image indexing using compressed colour histograms", IEE Proceedings - Vision, Image and Signal Processing, vol. 147, issue 4, pp. 349 - 355.

2. Borges, G. A. & Aldon, M. J. (2002). "A Decoupled Approach for Simultaneous Stochastic Mapping and Mobile Robot Localization" in *2002 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2002)*, vol. 1, pp. 558 - 563.

3. Borgida, A., Brachman, R., McGuinness, D. & Resnick, L. (1989). "CLASSIC: A Structural Data Model For Objects" in *Proceedings of the 1989 ACM SIGMOD International Conference on Management of Data*, pp. 58 - 67.

4. Carpenter, G. A., Grossberg, S. & Rosen, D. B. (1991). "Fuzzy ART: Fast Stable Learning and Categorization of Analog Patterns by an Adaptive Resonance System", Neural Networks, vol. 4, issue 6, pp. 759 - 771, Oxford, UK.

5. Chiu, C. Y., Lin, H. C. & Yang, S. N. (2003). "A Fuzzy Logic CBIR System" in *12th IEEE International Conference on Fuzzy Systems, 2003.(FUZZ '03)*, vol. 2, pp. 1171 - 1176.

6. Chou, T. C. & Cheng, S. C. (2006). "Design and Implementation of a Semantic Image Classification and Retrieval of Organizational Memory Information Systems Using Analytical Hierarchy Process", Omega, vol. 34, issue 2, pp. 125 - 134.

7. Dailey, M. N. & Parnichkun, M. (2005). "Landmark-based Simultaneous Localization and Mapping with Stereo Vision" in *Proc.Of the 2005 Asian Conference on Industrial Automation and Robotics (ACIAR '05)*, pp. F - 15.

8. Daoutis, M., Coradeschi, S. & Loutfi, A. (2009). "Integrating Common Sense in Physically Embedded Intelligent Systems" in *Proceedings of the 5th International Conference on Intelligent Environments*, vol. 2, pp. 212 - 219.

9. Davison, A. (2003). "Real-Time Simultaneous Localization and Mapping with a Single Camera" in *Proceedings of the Ninth International Conference on Computer Vision ICCV'03*, pp. 1403 - 1410, Nice, France.

10. Davison, A. (2005). "Active Search for Real Time Vision" in *Tenth IEEE International Conference on Computer Vision (ICCV 2005)*, vol. 1, pp. 66 - 73.

11. Dellaert, F. & Bruemmer, D. (2004). "Semantic SLAM for Collaborative Cognitive Workspaces" presented at the AAAI Fall Symposium Series, Arlington, VA, USA.

12. Deselaers, T., Keysers, D. & Ney, K. (2008). "Features for Image Retrieval: An Experimental Comparison", Information Retrieval, vol. 11, issue 2, pp. 77 - 107.

13. Dissanayake, M. W. M. G., Newman, P., Durrant-Whyte, H. F. & Csorba, M. (2001). "A Solution to the Simultaneous Localization and Map Building (SLAM) Problem", IEEE Transactions on Robotics and Automation, vol. 17, issue 3, pp. 229 - 241.

14. Duckett, T., Marsland, S. & Shapiro, J. (2000). "Learning Globally Consistent Maps by Relaxation" in *Proceedings of the 2000 IEEE International Conference on Robots and Automation (ICRA '00)*, vol. 4, pp. 3841 - 3846.

15. Ekvall, S., Jensfelt, P. & Kragic, D. (2006). "Integrating Active Mobile Robots Object Recognition and SLAM in Natural Environments" in *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2006)*, pp. 5792 - 5797.

16. Elfes, A. (1989). "Using Occupancy Grids for Mobile Robot Perception and Navigation", Computer, vol. 22, issue 6, pp. 46 - 57.

17. Elinas, P. & Little, J. J. (2007). "Stereo Vision SLAM: Near Real-Time Learning of 3D Point-Landmark and 2D Occupancy-Grid Maps Using Particle Filters" in *2007 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2007)*, to be published.

18. Elinas, P., Sim, R. & Little, J. J. (2006). "σSLAM: Stereo Vision SLAM Using the Rao-Blackwellised Particle Filter and a Novel Mixture Proposal Distribution" in *Proceedings of the 2006 IEEE International Conference on Robotics and Automation (ICRA 2006)*, pp. 1564 - 1570.

19. Fitzgibbon, A. & Zisserman, A. (1998). "Camera Recovery for Closed or Open Image Sequences" in *Proceedings of the Fifth European Conf.on Computer Vision*, vol. 1, pp. 311 - 326, Freiburg, Germany.

20. Flint, A., Mei, C., Reid, I. & Murray, D. (2010). "Growing Semantically Meaningful Models for Visual SLAM" in *2010 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 467 - 474.

21. Frese, U. (2006). "A Discussion of Simultaneous Localization and Mapping", Autonomous Robots, vol. 20, pp. 25 - 42.

22. Freund, Y. & Schapire, R. E. (1999). "A Short Introduction to Boosting", Journal of Japanese Society for Artificial Intelligence, vol. 14(5), pp. 771 - 780.

23. Galindo, C., Saffiotti, A., Coradeschi, S., Buschka, P., Fernandez-Madrigal, J. & Gonzalez, J. (2005). "Multi-hierarchical Semantic Maps For Mobile Robotics" in *Proc.of the IEEE/RSJ Intl.Conf.on Intelligent Robots and Systems (IROS 2005)*, pp. 3492 - 3497, Edmonton, CA, USA.

24. Goldman, J. A. (1994). "Path Planning Problems and Solutions" in *Proceedings of the IEEE 1994 National Aerospace and Electronics Conference*, vol. 1, pp. 105 - 108, Dayton, OH, USA.

25. Grisetti, G., Stachniss, C. & Burgard, W. (2007). "Improved Techniques for Grid Mapping with Rao-Blackwellized Particle Filters", IEEE Transactions on Robotics, vol. 23, issue 1, pp. 34 - 46.

26. Gutmann, J. & Konolige, K. (1999). "Incremental Mapping of Large Cyclic Environments", IEEE Int.Symp.on Computational Intelligence in Robotics and Automation (CIRA 1999), pp. 318 - 325.

27. Hahnel, D., Thrun, S., Wegbreit, B. & Burgard, W. (2003). "Towards Lazy Data Association in SLAM" in *Proceedings of the 11th International Symposium of Robotics Research (ISRR'03)*, Sienna, Italy.

28. Hall, L. & Hathaway, R. (1996). "Fuzzy Systems Toolbox, Fuzzy Logic Toolbox [Software Review]", IEEE Transactions on Fuzzy Systems, vol. 4, issue 1, pp. 82 - 85.

29. Huang, J., Georgiopoulos, M. & Heileman, G. L. (1995). "Fuzzy ART Properties", Neural Networks, vol. 8, issue 2, pp. 203 - 213, Oxford, UK.

30. Jensfelt, P., Kragic, D., Folkesson, J. & Bjorkman, M. (2006). "A Framework for Vision Based Bearing Only 3D SLAM" in *Proc.2006 IEEE International Conference on Robotics and Automation (ICRA 2006)*, vol. 1, pp. 1944 - 1950.

31. Jhanwar, N., Chaudhuri, S., Seetharaman, G. & Zavidovique, B. (2004). "Content Based Image Retrieval Using Motif Cooccurrence Matrix", Image and Vision Computing, vol. 22, issue 14, pp. 1211 - 1220.

32. Jonsgård, O. A. F. (2005). *Improvements on Colour Histogram-based CBIR*, Masters thesis, Gjøvik University College, Norway.

33. Konolige, K., Agrawal, M., Bolles, R. C., Cowan, C., Fischler, M. & Gerkey, B. (2006). "Outdoor Mapping and Navigation using Stereo Vision" in *Proc.Of Intl.Symp.on Experimental Robotics (ISER)*, Rio de Janeiro, Brazil.

34. Kulkarni, S. & Verma, B. (2003). "Fuzzy Logic based Texture Queries for CBIR" in *Fifth International Conference on Computational Intelligence and Multimedia Applications (ICCIMA'03)*, pp. 223 - 228.

35. Lakdashti, A., Moin, M. S. & Badie, K. (2008). "Semantic-Based Image Retrieval: A Fuzzy Modeling Approach", IEEE/ACS International Conference on Computer Systems and Applications (AICCSA 2008), pp. 575 - 581, Doha, Qatar.

36. Leonard, J. J. & Durrant-Whyte, H. F. (1991). "Mobile Robot Localization by Tracking Geometric Beacons", IEEE Transactions on Robotics and Automation, vol. 7, issue 3.

37. Leonard, J. J. & Durrant-Whyte, H. F. (1992). *Directed Sonar Sensing for Mobile Robot Navigation*, Kluwert Accademic Publisher,Boston, USA.

38. Li, Q., Shi, Z. & Luo, S. (2007). "A Neural Network Approach for Bridging the Semantic Gap in Texture Image Retrieval" in *Proceedings of International Joint Conference on Neural Networks*, pp. 581 - 585, Orlando, Florida.

39. Liu, P., Jia, K., Wang, Z. & Lv, Z. (2007). "A New And Effective Image Retrieval Method Based On Combined Features", Fourth International Conference on Image and Graphics (ICIG 2007), pp. 786 - 790, Sichuan, China.

40. Long, F., Zhang, H., and Feng, D. D. 2003, 'Fundamentals of Content-based Image Retrieval' in Feng, D., *Multimedia Information Retrieval and Management*, Springer, Berlin.

41. Moravec, H. P. (1980). *Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover*, PhD thesis, Carnegie-Mellon University.

42. Mouragnon, E., Lhuillier, M., Dhome, M., Dekeyser, F. & Sayd, P. (2006). "Monocular Vision Based SLAM for Mobile Robots" in *Proceedings of the 18th International Conference on Pattern Recognition*, vol. 3, pp. 1027 - 1031.

43. Newman, P., Cole, D. & Ho, K. (2006). "Outdoor SLAM using Visual Appearance and Laser Ranging" in *Proc.2006 IEEE International Conference on Robotics and Automation (ICRA 2006)*, pp. 1180 - 1187.

44. Ni, K. & Dellaert, F. (2010). "Multi-Level Submap Based SLAM Using Nested Dissection" in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2010)*, pp. 2558 - 2565.

45. Nuchter, A., Surmann, H., Lingemann, K. & Hertzberg, J. (2003). "Semantic Scene Analysis Of Scanned 3D Indoor Environments" in *Proceedings of the 8th International Fall Workshop Vision, Modeling, and Visualization 2003 (VMV '03)*, pp. 215 - 222, Munich, Germany.

46. Oberlander, J., Uhl, K., Zollner, J. M. & Dillmann, R. (2008). "A Region-based SLAM Algorithm Capturing Metric, Topological, and Semantic Properties" in *IEEE International Conference on Robotics and Automation, 2008 (ICRA 2008)*, pp. 1886 - 1891.

47. Pass, G. & Zabih, R. (1999). "Comparing Images Using Joint Histograms", Multimedia Systems, vol. 7, issue 3, pp. 234 - 240.

48. Persson, M., Duckett, T., Valgren, C. & Lilienthal, A. (2007). "Probabilistic Semantic Mapping With A Virtual Sensor for Building Nature Detection" in *Proc.IEEE Int.Symp.on Computational Intelligence in Robotics and Automation (CIRA 2007)*, pp. 236 - 242.

49. Riisgaard, S. & Blas, M. R. (2005). "SLAM for Dummies: A Tutorial Approach to Simultaneous Localization and Mapping", MIT OpenCourseWare, Aeronautics and Astronautics, viewed 30 May 2008, <http://www.core.org.cn/NR/rdonlyres/Aeronautics-and-Astronautics/16-412JSpring-2005/9D8DB59F-24EC-4B75-BA7A-F0916BAB2440/0/1aslam_blas_repo.pdf>.

50. Rosten, E. & Drummond, T. (2006). "Machine Learning for High-Speed Corner Detection" in *Proceedings of the Ninth European Conference on Computer Vision (ECCV06)*, vol. 1, pp. 430 - 443, Graz, Austria.

51. Rottmann, A., Mozoz, O. M., Stachniss, C. & Burgard, W. (2005). "Semantic Place Classification of Indoor Environments with Mobile Robots using Boosting" in *Proc.Nat.Conf.Artif.Intell.(AAAI), 2005*, pp. 1306 - 1311.

52. Royer, E., Bom, J., Dhome, M., Thuilot, B., Lhuilier, M. & Marmoiton, F. (2005). "Outdoor Autonomous Navigation Using Monocular Vision" in *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2005)*, pp. 1253 - 1258.

53. Royer, E., Lhuilier, M., Dhome, M. & Lavest, J. (2007). "Monocular Vision for Mobile Robot Localization and Autonomous Navigation", International Journal of Computer Vision, vol. 74, issue 3, pp. 237 - 260, Hingham, MA, USA.

54. Santini, F. & Rucci, M. (2007). "Active Estimation of Distance in a Robotic System that Replicates Human Eye Movement", *Robotics and Autonomous Systems*, vol. 5, issue 2, North-Holland Publishing Co., Amsterdam, The Netherlands, pp. 107 - 121.

55. Schwering, A. (2007). "Evaluation of a Semantic Similarity Measure for Natural Language Spatial Relations" in *5th conference on spatial information theory (COSIT07)*, vol. 4736, pp. 116 - 132, Melbourne, Australia.

56. Se, S., Lowe, D. & Little, J. (2002). "Mobile Robot Localization and Mapping with Uncertainty using Scale-Invariant Visual Landmarks", International Journal of Robotics Research, vol. 21, pp. 735 - 758.

57. Shi, J. & Tomasi, C. (1994). "Good Features to Track", IEEE Conference on Computer Vision and Pattern Recognition (CVPR '94), pp. 593 - 600, Seattle, Washington, USA.

58. Siegwart, R. & Nourbakhsh, I. R. (2004). *Introduction to Autonomous Mobile Robots*, The MIT Press,Cambridge, Massachusetts, USA, London, England.

59. Siggelkow, S. (2002). *Feature Histograms for Content-Based Image Retrieval*, PhD. thesis, Albert-Ludwigs-Universitat, Freiburg.

60. Sims, R. (2004). *On Visual Maps and Their Automatic Construction*, PhD thesis, McGill University in Montreal.

61. Smith, P., Reid, I. & Davison, A. (2006). "Real-Time Monocular SLAM with Straight Lines" in *British Machine Vision Conference*, vol. 1, pp. 17 - 26.

62. Smith, R., Self, M. & Cheeseman, P. (1990). "Estimating Uncertain Spatial Relationships in Robotics", *Autonomous Robot Vehicles*, Springer-Verlag, New York, NY, USA, pp. 167 - 193.

63. Stentz, A. (1994). "Optimal and Efficient Path Planning for Partially-Known Environments" in *1994 IEEE International Conference on Robotics and Automation*, vol. 4, pp. 3310 - 3317, San Diego, CA, USA.

64. Strasdat, H., Montiel, J. & Davison, A. (2010). "Real-time Monocular SLAM: Why Filter?" in *Proceedings of the 2010 IEEE International Conference on Robotics and Automation (ICRA 2010)*, pp. 2657 - 2664.

65. Swain, M. J. & Ballard, D. H. (1991). "Color Indexing", International Journal of Computer Vision, vol. 7, issue 1, pp. 11 - 32.

66. Tamura, H., Mori, S. & Yamawaki, T. (1978). "Textural Features Corresponding to Visual Perception", IEEE Transactions on Systems, Man and Cybernetics, vol. 8, issue 6, pp. 460 - 473.

67. Thrun, S. 2002, 'Robotic Mapping: A Survey' in *Exploring Artificial Intelligence in the New Milenium*, Morgan Kaufmann, San Francisco, CA, USA.

68. Thrun, S., Thayer, S., Whittaker, W., Baker, C., Burgard, W., Ferguson, D., Hahnel, D., Montemerlo, D., Morris, A., Omohundro, Z., Reverte, C. & Whittaker, W. (2004). "Autonomous Exploration and Mapping of Abandoned Mines", IEEE Robotics and Automation Magazine, vol. 11, issue 4, pp. 79 - 91.

69. Tieu, K. & Viola, P. (2000). "Boosting Image Retrieval" in *Proc.2000 IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2000)*, pp. 228 - 235.

70. Triggs, B., McLauchlan, P., Hartley, R., and Fitzgibbon, A. 2000, 'Bundle Adjustment - A Modern Synthesis' in *Proceedings of the International Workshop on Vision Algorithms*, Springer-Verlag,

71. Weingarten, J. (2006). *Feature-based 3D SLAM*, PhD thesis, Swiss Federal Institute of Technology in Lausanne.

72. Wolf, D. F. & Sukhatme, G. S. (2008). "Semantic Mapping Using Mobile Robots", IEEE Transactions on Robotics, vol. 24, issue 2, pp. 245 - 258.

73. Zelinsky, A. (1992). "A Mobile Robot Navigation Exploration Algorithm", IEEE Transactions on Robotics and Automation, vol. 8, issue 6, pp. 707 - 717.

74. Zhang, D. & Lu, G. (2001). "A Comparative Study on Shape Retrieval Using Fourier Descriptors with Different Shape Signatures" in *Proc.of International Conference on Intelligent Multimedia and Distance Education (ICIMADE01)*, pp. 1 - 9, Fargo, ND, USA.

75. Zheng, J. Y., Fukagawa, Y. & Abe, N. (1995). "Shape and Model from Specular Motion" in *Proceedings of the Fifth International Conference on Computer Vision ICCV'95*, pp. 72 - 79, Cambridge, MA, USA.

76. Zunino, G. (2002). *Simultaneous Localization and Mapping for Navigation in Realistic Environment*, Licentiate Thesis thesis, Royal Institute of Technology, Stockholm.